Control Design VIs and Functions

June 2008, 370853E-01

Owning Palette: Control Design and Simulation VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Control Design VIs and functions to construct, analyze, and deploy dynamic system models in LabVIEW.

The VIs on these palettes can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Refer to the <u>LabVIEW Control Design User Manual</u> for more information about using the Control Design VIs and functions.

Subpalette	Description
<u>Analytical PID</u> Design VI	Use the VI on this palette to design the gain values for a proportional-integral-derivative (PID) controller.
Dynamic Characteristics <u>VIs</u>	Use the Dynamic Characteristics VIs to calculate properties related to the dynamics of a given system model. Dynamic characteristics include DC gain, stability, norm, root locus, and pole-zero analysis.
<u>Frequency</u> <u>Response VIs</u>	Use the Frequency Response VIs to analyze a system model in the frequency domain.
Implementation VIs and Functions	Use the Implementation VIs and functions to simulate the dynamic response of a discrete system model, deploy a discrete model to a real-time target, implement a discrete Kalman filter, and implement current and predictive observers.
Model Construction VIs	Use the Model Construction VIs to create linear system models and modify the properties of a system model. You also can use the Model Construction VIs to save a system model to a file, read a system model from a file, or obtain a visual representation of a model.
Model Conversion VIs	Use the Model Conversion VIs to convert a system model from one representation to another, from a continuous-time to a discrete-time model, or from a

	discrete-time to a continuous-time model. You also can use the Model Conversion VIs to convert a control design model into a simulation model or a simulation model into a control design model.
<u>Model</u> Information VIs	Use the Model Information VIs to obtain or set parameters, data, and names of a system model. Model information includes properties such as the system delay, system dimensions, sampling time, and names of inputs, outputs, and states.
<u>Model</u> Interconnection VIs	Use the Model Interconnection VIs to perform different types of linear system interconnections. You can build a large system model by connecting smaller system models together.
<u>Model</u> Reduction VIs	Use the Model Reduction VIs to perform a zero-pole cancellation or to reduce the number of states in state- space models. You also can use the Model Reduction VIs to eliminate inputs and outputs that are uncontrollable or unobservable.
<u>Predictive</u> <u>Control VIs</u>	Use the Predictive Control VIs to construct and implement a predictive controller model for a state- space plant.
<u>Solvers VIs</u>	Use the Solvers VIs to compute the solutions to the continuous and discrete algebraic Riccati equations, the continuous and discrete Lyapunov equations, and integrals involving matrix exponentials.
<u>State-Space</u> <u>Model Analysis</u> <u>VIs</u>	Use State-Space Model Analysis VIs to calculate properties of a given state-space model, such as observability, detectability, controllability, stabilizability, similarity transformations, model balance, and system Grammians.
<u>State</u> <u>Feedback</u> <u>Design VIs</u>	Use the State Feedback Design VIs to calculate controller and observer gains for closed-loop state feedback control or to estimate a state-space model. You also can use State Feedback Design VIs to configure and test state-space controllers and state estimators in time domains.

<u>Stochastic</u> Systems VIs	Use the Stochastic Systems VIs to construct, manipulate, and analyze stochastic state-space system models.
<u>Time</u> <u>Response VIs</u>	Use the Time Response VIs to create generic linear simulations and time domain plots for step inputs, impulse inputs, and initial condition responses.

 $\ensuremath{\textcircled{\sc 0}}$ 2004–2008 National Instruments Corporation. All rights reserved.

Control Design Controls and Indicators

Use the Control Design controls indicators to display system models and plots on the front panel window of a VI. Many of the Control Design controls and indicators place corresponding, preconfigured code on the block diagram window of the VI to enable you to build your application more efficiently.

Create Model Controls

Use the Create Model controls to display system models on the front panel window. When you place a Create Model control on the front panel window, the LabVIEW Control Design and Simulation Module places preconfigured code using the <u>Model Construction</u> VIs on the block diagram window of the VI. Wire the specified output to the next Control Design VI or function to continue building your application.

Model Analysis Indicators

Use the Model Analysis indicators to display plots such as Bode, Nyquist, and Nichols plots on the front panel window. When you place a Model Analysis indicator on the front panel window, the Control Design and Simulation Module places preconfigured code using the Dynamic Characteristics VIs, Frequency Response VIs, and Time Response VIs on the block diagram window of the VI. Wire the specified input to the appropriate system model to continue building your application.

Plots Indicators

Use the Plots indicators to display plots such as Bode, Nyquist, and Nichols plots on the front panel window. These indicators do not place preconfigured code on the block diagram window.

Analytical PID Design Plots Indicators

Use the Analytical PID Design Plots indicators to display 3D plots for the **Stable Set Boundary** and **Stable Set Interior Points** outputs of the <u>CD</u> <u>Design PID for Discrete Systems</u> VI. These indicators do not place preconfigured code on the block diagram window.

Limitations of the Control Design Indicators

The following is a list of limitations you might encounter when using the Control Design indicators.

- RT targets do not support Control Design indicators. When creating VIs for an RT target, do not use these indicators. Instead, use a graph indicator, such as the XY graph.
- You can copy and paste the plot indicators only by selecting the terminal on the block diagram.
- The option to show the unique grid or Cartesian grid is available only on the block diagram.

Analytical PID Design VI

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the VI on this palette to design the gain values for a proportional-integral-derivative (PID) controller.

Palette Object	Description
<u>CD</u> <u>Design</u> <u>PID for</u> <u>Discrete</u> <u>Systems</u>	Calculates the proportional gain K_P , integral gain K_I , and derivative gain K_D that stabilize the specified controller model(s). This VI accepts discrete single-input single-output (SISO) models only. However, you can specify more than one SISO model at a time by using the two-dimensional transfer function(s) array control of the <u>CD Construct Transfer</u> <u>Function Model</u> VI.

CD Design PID for Discrete Systems VI

Owning Palette: Analytical PID Design VI

Installed With: Control Design and Simulation Module

Calculates the proportional gain K_P , integral gain K_I , and derivative gain K_D that stabilize the specified controller model(s). This VI accepts discrete single-input single-output (SISO) models only. However, you can specify more than one SISO model at a time by using the two-dimensional **transfer function(s)** array control of the <u>CD Construct Transfer Function Model</u> VI.

This VI is based on algorithms and techniques described in the following resource: Datta, A.; M. T. Ho; and S. P. Bhattacharyya. 2000. Structure and synthesis of PID controllers. London: Springer-Verlag.

•

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Design PID for Discrete Systems (State-Space)



State-Space Model specifies a state-space representation of the controller model for which this VI calculates the PID gains.

- Sampling Specifications specifies the options relating to how this VI divides and samples the common K_3 range. Refer to the <u>Details</u> section for more information about these steps.
 - **Note** If the range between K_3 points is large but the actual stable set of PID gains is very small, LabVIEW might return an error saying the problem is infeasible even though a feasible solution actually exists.
 - **Num K Grid Points** specifies the number of points into which this VI divides the common K_3 range. More values result in a longer execution time but more accurate results.
 - **Num Search Points** specifies the number of tuples of PID gain values for which this VI searches at each K_3 point. More values result in a longer execution time but more accurate results.
- Min Gain and Phase Margins specifies the optional performance constraints on the PID controller model. If you specify a value of 0 for either of these constraints, this VI does not attempt to verify that constraint. If you specify nonzero a value for either parameter, this VI returns PID gain values that satisfy the performance constraint(s).
 - Note Specifying performance constraints significantly increases the execution time of this VI.
 - Gain Margin (dB) specifies the minimum gain margin, in dB, of the PID controller.
 - **Phase Margin (deg)** specifies the minimum phase margin, in degrees, of the PID controller.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Design PID Gains returns the PID gain values that are closest to the centroid, or average, of all intersecting planes. These gain values guarantee simultaneous closed-loop stability for all specified models. If you specified nonzero Min Gain and Phase Margins, these gain values also satisfy those performance constraints.

The **Design PID Gains** tuple is a subset of the **Stable Set Interior Points.** If the **Design PID Gains** is acceptable, you can choose an alternate tuple from the **Stable Set Interior Points** parameter. This parameter is a cluster of three one-dimensional arrays. The n^{th} element of each array corresponds to the K_{P} , K_{I} , or K_{D} value of a single set.

Kp returns the proportional gain of the PID controller model.

DBL

Ki returns the integral gain of the PID controller model.

- **Kd** returns the derivative gain of the PID controller model.
- **Stable Set Boundary** returns the boundary between the stable and unstable PID gain values. All gain values within the stable boundary are guaranteed to be stable for the PID controller.
 - Note This data structure traces the outline of each polygon in succession such that the polygons are connected along one edge of the three-dimensional boundary.
- Stable Set Interior Points returns the set of gain values that lie within the Stable Set Boundary. All gain values within the stable boundary are guaranteed to be stable for the PID controller. Additionally, all gain values are guaranteed to meet any performance criteria you specified by using the Min Gain and Phase Margins parameter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Design PID for Discrete Systems (Transfer Function)



- **Transfer Function Model** specifies a transfer function representation of the controller model for which this VI calculates the PID gains.
- **Sampling Specifications** specifies the options relating to how this VI divides and samples the common K_3 range. Refer to the <u>Details</u> section for more information about these steps.
 - N
 - **Note** If the range between K_3 points is large but the actual stable set of PID gains is very small, LabVIEW might return an error saying the problem is infeasible even though a feasible solution actually exists.
 - **Num K Grid Points** specifies the number of points into which this VI divides the common K_3 range. More values result in a longer execution time but more accurate results.
 - **Num Search Points** specifies the number of tuples of PID gain values for which this VI searches at each K_3 point. More values result in a longer execution time but more accurate results.
- Min Gain and Phase Margins specifies the optional performance constraints on the PID controller model. If you specify a value of 0 for either of these constraints, this VI does not attempt to verify that constraint. If you specify nonzero a value for either parameter, this VI returns PID gain values that satisfy the performance constraint(s).
 - Note Specifying performance constraints significantly increases the execution time of this VI.
 - Gain Margin (dB) specifies the minimum gain margin, in dB, of the PID controller.
 - **Phase Margin (deg)** specifies the minimum phase margin, in degrees, of the PID controller.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Design PID Gains returns the PID gain values that are closest to the centroid, or average, of all intersecting planes. These gain values guarantee simultaneous closed-loop stability for all specified models. If you specified nonzero Min Gain and Phase Margins, these gain values also satisfy those performance constraints.

The **Design PID Gains** tuple is a subset of the **Stable Set Interior Points.** If the **Design PID Gains** is acceptable, you can choose an alternate tuple from the **Stable Set Interior Points** parameter. This parameter is a cluster of three one-dimensional arrays. The n^{th} element of each array corresponds to the K_{P} , K_{I} , or K_{D} value of a single set.

- **Kp** returns the proportional gain of the PID controller model.
- DBL

Ki returns the integral gain of the PID controller model.

- **Kd** returns the derivative gain of the PID controller model.
- **Stable Set Boundary** returns the boundary between the stable and unstable PID gain values. All gain values within the stable boundary are guaranteed to be stable for the PID controller.
 - Note This data structure traces the outline of each polygon in succession such that the polygons are connected along one edge of the three-dimensional boundary.
- Stable Set Interior Points returns the set of gain values that lie within the Stable Set Boundary. All gain values within the stable boundary are guaranteed to be stable for the PID controller. Additionally, all gain values are guaranteed to meet any performance criteria you specified by using the Min Gain and Phase Margins parameter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Design PID for Discrete Systems (Zero-Pole-Gain)



Zero-Pole-Gain Model specifies a zero-pole-gain representation of the controller model for which this VI calculates the PID gains.

- **Sampling Specifications** specifies the options relating to how this VI divides and samples the common K_3 range. Refer to the <u>Details</u> section for more information about these steps.
 - **Note** If the range between K_3 points is large but the actual stable set of PID gains is very small, LabVIEW might return an error saying the problem is infeasible even though a feasible solution actually exists.
 - **Num K Grid Points** specifies the number of points into which this VI divides the common K_3 range. More values result in a longer execution time but more accurate results.
 - **Num Search Points** specifies the number of tuples of PID gain values for which this VI searches at each K_3 point. More values result in a longer execution time but more accurate results.
- Min Gain and Phase Margins specifies the optional performance constraints on the PID controller model. If you specify a value of 0 for either of these constraints, this VI does not attempt to verify that constraint. If you specify nonzero a value for either parameter, this VI returns PID gain values that satisfy the performance constraint(s).
 - Note Specifying performance constraints significantly increases the execution time of this VI.
 - Gain Margin (dB) specifies the minimum gain margin, in dB, of the PID controller.
 - **Phase Margin (deg)** specifies the minimum phase margin, in degrees, of the PID controller.
- error in describes error conditions that occur before this VI or

function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Design PID Gains returns the PID gain values that are closest to the centroid, or average, of all intersecting planes. These gain values guarantee simultaneous closed-loop stability for all specified models. If you specified nonzero Min Gain and Phase Margins, these gain values also satisfy those performance constraints.

The **Design PID Gains** tuple is a subset of the **Stable Set Interior Points.** If the **Design PID Gains** is acceptable, you can choose an alternate tuple from the **Stable Set Interior Points** parameter. This parameter is a cluster of three one-dimensional arrays. The n^{th} element of each array corresponds to the K_{P} , K_{I} , or K_{D} value of a single set.

- **Kp** returns the proportional gain of the PID controller model.
- **Ki** returns the integral gain of the PID controller model.

Kd returns the derivative gain of the PID controller model.

- **Stable Set Boundary** returns the boundary between the stable and unstable PID gain values. All gain values within the stable boundary are guaranteed to be stable for the PID controller.
 - Note This data structure traces the outline of each polygon in succession such that the polygons are connected along one edge of the three-dimensional boundary.
- Stable Set Interior Points returns the set of gain values that lie within the Stable Set Boundary. All gain values within the stable boundary are guaranteed to be stable for the PID controller. Additionally, all gain values are guaranteed to meet any performance criteria you specified by using the Min Gain and Phase Margins parameter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Design PID for Discrete Systems Details

This VI assumes the following system structure:



where

- *z* is the discrete time variable.
- G(z) is any proper transfer function that represents the plant you want to control.
- *C*(*z*) is a PID controller with proportional gain *K*_P, integral gain *K*_I, and derivative gain *K*_D. To search for these values, this VI transforms *K*_P, *K*_I, and *K*_D into *K*₁, *K*₂, and *K*₃ by using the following equations:

$$\boldsymbol{C}(z) = \mathcal{K}_{p} + \mathcal{K}_{I} \frac{T}{2} \cdot \frac{(z+1)}{(z-1)} + \mathcal{K}_{D} \frac{2}{T} \cdot \frac{(z-1)}{(z+1)}$$

where

$$\begin{aligned} \kappa_0 &= -\kappa_p + \frac{\kappa_I T}{2} + \frac{2\kappa_D}{T} \\ \kappa_1 &= \kappa_I T - \frac{4\kappa_D}{T} \\ \kappa_2 &= \kappa_p + \frac{\kappa_I T}{2} + \frac{2\kappa_D}{T} \\ \kappa_3 &= \kappa_2 - \kappa_0 \end{aligned}$$

This VI executes the following steps:

- 1. Calculates a valid K_3 range for each specified model. If this VI cannot calculate a valid K_3 range for each model, the problem is infeasible, and LabVIEW returns an error.
- 2. Calculates a K_3 range that is feasible for all specified models. If this VI cannot calculate this range, the problem is infeasible, and LabVIEW returns an error. The following figure shows an example of steps 1 and 2.



3. Divides this common range into a number of points specified by the **Num K Grid Points** parameter. The following figure shows an example of this step, where *n* is the **Num K Grid Points** parameter.



- 4. Calculates two-dimensional polygons at each K_3 point. Each polygon corresponds to a specified model. The dimensions of these polygons are K_1 on the x-axis and K_2 on the y-axis.
- 5. Calculates the intersection between the polygons at each K_3 point. The result is a set of two-dimensional polygons, known as slices. Each slice is common to each model at each K_3 point. The following figure shows an example of steps 4 and 5.



6. Lines up each slice in three-dimensional space, where K_3 is the third axis. This step approximates a three-dimensional shape that represents the boundary between stable and unstable sets of PID gain values. On this three-dimensional grid, K_1 is the y-axis, K_2 is the z-axis, and K_3 is the x-axis. The following figure shows an example of this step.



- 7. Takes a uniform random sampling of points inside each slice. You specify the total number of points by using the Num Search **Points** parameter. Each point represents a set of K_1 , K_2 , and K_3 values.
 - N
 - **Note** This VI defines a rectangle around the polygon, takes samples of points within the entire rectangle, and keeps only those points that lie inside the slice. The number of discarded points depends on the aspect ratio of the polygon. Because this approach discards points that lie outside the slice, the number of points returned might not match the Num Search Points parameter exactly.

The following figure shows an example of this step.



This VI discards points that lie outside the slice. These discarded points are marked with an X in the previous figure.

- 8. Transforms each K_1 , K_2 , and K_3 value into K_P , K_I , and K_D values.
- 9. This VI returns the set of two-dimensional slices in the Stable Set **Boundary** parameter. The following figure shows an example stable boundary.



All points within the boundary, as shown in the previous figure, are stable.

10. This VI also returns the points within the boundary in the **Stable Set Interior Points** parameter. The following figure shows a sample set of points.



Refer to the following resources for more information about the algorithms and methods this VI uses.

- Keel, L. H., J. I. Rego, and S. P. Bhattacharyya. 2003. A new approach to digital PID controller design. *IEEE Transactions on Automatic Control* 48, no. 4.
- Keel, L.H., and S.P. Bhattacharyya. 2002. Root counting, phase unwrapping, stability and stabilization of discrete time systems. *Linear algebra and its applications* 351–2:501–518.
- Ho, Ming-Tzu, G. J. Silva, A. Datta, and S. P. Bhattacharyya. 2004. Real and complex stabilization: stability and performance. *Proc. Of the 2004 American Control Conference* 5:4126–38.

Dynamic Characteristics VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Dynamic Characteristics VIs to calculate properties related to the dynamics of a given system model. Dynamic characteristics include DC gain, stability, norm, root locus, and pole-zero analysis.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD</u> <u>Covariance</u> <u>Response</u>	Returns the covariance of the outputs and/or states when Gaussian white noise excites the input model. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD</u> Damping Ratio and Natural Frequency	Gives the damping ratios and natural frequencies of the poles of the input system. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD DC</u> <u>Gain</u>	Calculates the DC gain, or the ratio of the outputs to the inputs of a system, after all the transients decay. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD</u> <u>Distribute</u> <u>Delay</u>	Distributes the total delay of a system into input, output, and transport delays such that the transport delays are minimal. This VI allocates the majority of the delays as input or output delays. The remaining delays become transport delays. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Norm</u>	Calculates the infinity-norm and 2-norm of linear time- invariant (LTI) systems. The 2-norm is infinite for unstable systems and for state-space systems whose D matrix is not equal to zero. The data type you wire to the State-Space

	Model input determines the polymorphic instance to use.
<u>CD Pole-</u> Zero Map	Plots the poles and zeros of a system model on an XY graph that represents a complex plane. You can display this data in the CD Pole-Zero S Grid or CD Pole-Zero Z Grid <u>indicator</u> . The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Poles</u>	Returns the model poles. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Root</u> Locus	Plots the Evans plot or trajectory of closed-loop poles of a single-input single-output (SISO) system as the feedback gain varies from zero to infinity. You can display this data in a root locus graph indicator. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD</u> <u>Stability</u>	Determines if the input system is stable, unstable, or marginally stable. The data type you wire to the State- Space Model input determines the polymorphic instance to use.
<u>CD Total</u> Delay	Adds all the delays between each input and output pair and returns the total time delay. The total time delay includes the input, output, and transport delays. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Zeros</u>	Returns the locations of the model zeros. The data type you wire to the State-Space Model input determines the polymorphic instance to use.

CD Covariance Response VI

Owning Palette: Dynamic Characteristics VIs

Installed With: Control Design and Simulation Module

Returns the covariance of the outputs and/or states when Gaussian white noise excites the input model. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Covariance Response (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and additional <u>model information</u> about the system of which this VI determines output covariance.
- **Gaussian White Noise Covariance** is a symmetric, positive semidefinite matrix *N* that specifies the covariance of the noise exciting the system. The dimensions of this parameter also must be consistent with the system model. Refer to the <u>Details</u> section for the definition of this parameter when the model is continuous or discrete.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

Output Covariance is the covariance matrix of the output y of the system when Gaussian white noise with covariance N excites the system.

- **State Covariance** is the covariance matrix **X** of the state vector *x* when Gaussian white noise with covariance *N* excites the system that the **State-Space Model** represents. **State Covariance** is valid only when the model is in state-space form.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Covariance Response (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines output covariance.
- **Gaussian White Noise Covariance** is a symmetric, positive semidefinite matrix *N* that specifies the covariance of the noise exciting the system. The dimensions of this parameter also must be consistent with the system model. Refer to the <u>Details</u> section for the definition of this parameter when the model is continuous or discrete.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

Output Covariance is the covariance matrix of the output *y* of the system when Gaussian white noise with covariance **N** excites the system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Covariance Response (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines output covariance.
- **Gaussian White Noise Covariance** is a symmetric, positive semidefinite matrix *N* that specifies the covariance of the noise exciting the system. The dimensions of this parameter also must be consistent with the system model. Refer to the <u>Details</u> section for the definition of this parameter when the model is continuous or discrete.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

[DBL]

Output Covariance is the covariance matrix of the output *y* of the system when Gaussian white noise with covariance **N** excites the system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Covariance Response Details

The model must be stable for covariance to exist. You can use the <u>CD</u> <u>Stability</u> VI to determine model stability.

Continuous Models

This VI assumes continuous state-space models are of the following form:

```
\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A} \boldsymbol{x}(t) + \boldsymbol{B} \boldsymbol{w}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C} \boldsymbol{x}(t) + \boldsymbol{D} \boldsymbol{w}(t) \end{aligned}
```

For continuous models, the **Gaussian White Noise Covariance** is defined as

 $\boldsymbol{N}\mathrm{d}(t-\mathrm{t})=\mathsf{E}\{\boldsymbol{w}(t):\boldsymbol{w}^\mathsf{T}(\mathrm{t})\}$

The Output Covariance is defined as

```
\mathsf{E}\{\boldsymbol{y}(t) \; . \; \boldsymbol{y}^\mathsf{T}(t)\}
```

The State Covariance is defined as

 $\mathsf{E}\{\boldsymbol{x}(t) \ . \ \boldsymbol{x}^\mathsf{T}(t)\}$

This VI uses the following equations to calculate the outputs:

Output Covariance = CXC^{T} , where *X* is the solution to the continuous Lyapunov equation $AX+XA^{T} = -BNB^{T}$.

Note If *D* does not equal zero, the Output Covariance is infinite.

State Covariance = *X*, which is the solution to the continuous Lyapunov equation.
Discrete Models

This VI assumes discrete state-space models are of the following form:

 $\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{w}(k)$

 $\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{w}(k)$

For discrete models, the $\ensuremath{\textbf{Gaussian}}$ $\ensuremath{\textbf{White Noise Covariance}}$ is defined as

 \boldsymbol{N} d_{kl} = E[$w_k \cdot w^T_l$]

The Output Covariance is defined as

 $\mathsf{E}\{\boldsymbol{y}(k) : \boldsymbol{y}^{\mathsf{T}}(l)\}$

The State Covariance is defined as

 $\mathsf{E}\{\boldsymbol{x}(k) \ . \ \boldsymbol{x}^{\mathsf{T}}(l)\}$

This VI uses the following equation to calculate the outputs:

Output Covariance = $CXC^{T}+DND^{T}$, where *X* is the solution to the discrete Lyapunov equation $AXA^{T}-X = -BNB^{T}$.

State Covariance = *X*, which is the solution to the discrete Lyapunov equation.

where y is the output vector

x is the state vector

A is the state matrix

B is the input matrix

 \boldsymbol{C} is the output matrix

D is the direct transmission matrix

X is the solution to the continuous or discrete Riccati equation

w is the Gaussian white noise

t is continuous time

k is discrete time

d(t - t) is the Dirac delta function

 d_{kl} is the Kronecker delta function

E{} is the expected value or mean of the enclosed term(s)

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays when calculating the dynamic characteristics of a system, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Damping Ratio and Natural Frequency VI

Owning Palette: Dynamic Characteristics VIs

Installed With: Control Design and Simulation Module

Gives the damping ratios and natural frequencies of the poles of the input system. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Damping Ratio and Natural Frequency (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines damping ratio and natural frequency.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Damping Ratios** returns the damping ratios for each pole in the system.
- **Natural Frequencies** returns the natural frequencies for each pole in the system.
- **Poles** returns the eigenvalues of state matrix **A** in state-space

models.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Damping Ratio and Natural Frequency (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines damping ratio and natural frequency.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Damping Ratios** returns the damping ratios for each pole in the system.
- **Natural Frequencies** returns the natural frequencies for each pole in the system.

[CDB]

Poles returns the roots of the denominator in transfer function models.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Damping Ratio and Natural Frequency (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines damping ratio and natural frequency.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Damping Ratios** returns the damping ratios for each pole in the system.
- **Natural Frequencies** returns the natural frequencies for each pole in the system.
- **Poles** returns the poles in zero-pole-gain models.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Damping Ratio and Natural Frequency Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays when calculating the dynamic characteristics of a system, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD DC Gain VI

Owning Palette: <u>Dynamic Characteristics VIs</u>

Installed With: Control Design and Simulation Module

Calculates the DC gain, or the ratio of the outputs to the inputs of a system, after all the transients decay. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD DC Gain (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines DC gain.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DC Gain** returns the steady state gain, which is the gain of the system at low frequencies. **DC Gain** is a 2D-array where the *jj*th element gives the DC gain of the system due to the *j*th output and *j*th input.
- **Input-State DC Gain** is the ratio of the change in the steady state of the states divided by the change in the input under

consideration.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD DC Gain (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines DC gain.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DC Gain** returns the steady state gain, which is the gain of the system at low frequencies. **DC Gain** is a 2D-array where the *jj*th element gives the DC gain of the system due to the *j*th output and *j*th input.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD DC Gain (Zero-Pole Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines DC gain.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DC Gain** returns the steady state gain, which is the gain of the system at low frequencies. **DC Gain** is a 2D-array where the *jj*th element gives the DC gain of the system due to the *j*th output and *j*th input.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD DC Gain Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays when calculating the dynamic characteristics of a system, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Distribute Delay VI

Owning Palette: Dynamic Characteristics VIs

Installed With: Control Design and Simulation Module

Distributes the total delay of a system into input, output, and transport delays such that the transport delays are minimal. This VI allocates the majority of the delays as input or output delays. The remaining delays become transport delays. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Use the pull-down menu to select an instance of this VI.

Select an instance	•

■ Place on the block diagram ■ Find on the **Functions** palette

CD Distribute Delay (State-Space)



State-Space Model contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI distributes delay.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Input Delays** is the portion of the delay this VI transfers to the inputs.
- **Output Delays** is the portion of the delay this VI transfers to the outputs after distribution.
- **Transport Delays** is the portion of the delay this VI does not transfer to an input or output after distribution.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Distribute Delay (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI distributes delay.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Input Delays** is the portion of the delay this VI transfers to the inputs.
- **Output Delays** is the portion of the delay this VI transfers to the outputs after distribution.
- **Transport Delays** is the portion of the delay this VI does not

transfer to an input or output after distribution.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Distribute Delay (Zero-Pole-Gain)



Zero-Pole-Gain Model contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI distributes delay.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Input Delays** is the portion of the delay this VI transfers to the inputs.
- **Output Delays** is the portion of the delay this VI transfers to the outputs after distribution.
- **Transport Delays** is the portion of the delay this VI does not transfer to an input or output after distribution.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Norm VI

Owning Palette: <u>Dynamic Characteristics VIs</u>

Installed With: Control Design and Simulation Module

Calculates the infinity-norm and 2-norm of linear time-invariant (LTI) systems. The 2-norm is infinite for unstable systems and for state-space systems whose D matrix is not equal to zero. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

•

CD Norm (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines the norm.
- **Type** specifies the method for calculating the norm.

0 **2-norm** (default)—Calculates $||H||_2$ 1 **infinity-norm**—Calculates $||H||_{\infty}$

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Norm** returns the value of $||H||_2$ or $||H||_{\infty}$, depending on the

method specified in Type. $\|H\|_2 = \left(\frac{1}{2\pi} \int_{\infty}^{\infty} \operatorname{trace}[H(j\omega)^*H(j\omega)] d\omega\right)^{1/2}$

```
\|\mathbf{H}\|_{\omega} = \sup_{\boldsymbol{\omega}} \sigma_{\max}[\mathbf{H}(j\boldsymbol{\omega})]
```

The 2-norm is the RMS of the output when white noise of unit intensity excites the system. The infinity-norm is the maximum magnification of the frequency response of the system.

Frequency returns the value, in rad/s, at which this VI evaluates $||H||_{\infty}$. **Frequency** is undefined for $||H||_2$ and has a value of NaN.

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Norm (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines the norm.
- **Type** specifies the method for calculating the norm.

0 **2-norm** (default)—Calculates ||H||₂

1 infinity-norm—Calculates $||H||_{\infty}$

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Norm** returns the value of $||H||_2$ or $||H||_{\infty}$, depending on the

method specified in Type. $\|H\|_2 = \left(\frac{1}{2\pi} \int_{\infty}^{\infty} \operatorname{trace}[H(j\omega)^*H(j\omega)] d\omega\right)^{1/2}$

```
\|\mathbf{H}\|_{\omega} = \sup_{\boldsymbol{\omega}} \sigma_{\max}[\mathbf{H}(j\boldsymbol{\omega})]
```

The 2-norm is the RMS of the output when white noise of unit intensity excites the system. The infinity-norm is the maximum magnification of the frequency response of the system.

Frequency returns the value, in rad/s, at which this VI evaluates $||H||_{\infty}$. **Frequency** is undefined for $||H||_2$ and has a value of NaN.

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Norm (Zero-Pole-Gain)

Zero-Pole-Gain Model

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines the norm.
- **Type** specifies the method for calculating the norm.

0 **2-norm** (default)—Calculates ||H||₂ 1 **infinity-norm**—Calculates ||H||_∞

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Norm** returns the value of $||H||_2$ or $||H||_{\infty}$, depending on the

method specified in Type. $\|H\|_2 = \left(\frac{1}{2\pi} \int_{\infty}^{\infty} \operatorname{trace}[H(j\omega)^*H(j\omega)] d\omega\right)^{1/2}$

```
\|\mathbf{H}\|_{\omega} = \sup_{\boldsymbol{\omega}} \sigma_{\max}[\mathbf{H}(j\boldsymbol{\omega})]
```

The 2-norm is the RMS of the output when white noise of unit intensity excites the system. The infinity-norm is the maximum magnification of the frequency response of the system.

Frequency returns the value, in rad/s, at which this VI evaluates $||H||_{\infty}$. **Frequency** is undefined for $||H||_2$ and has a value of NaN.

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Norm Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays when calculating the dynamic characteristics of a system, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Pole-Zero Map VI

Owning Palette: Dynamic Characteristics VIs

Installed With: Control Design and Simulation Module

Plots the poles and zeros of a system model on an XY graph that represents a complex plane. You can display this data in the CD Pole-Zero S Grid or CD Pole-Zero Z Grid <u>indicator</u>. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

 \blacksquare Place on the block diagram \blacksquare Find on the Functions palette

CD Pole-Zero Map (State-Space)

State-Space Model

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI plots the poles and zeros on a complex plane.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Pole-Zero Map Plot** is a plot of the poles and zeros of a system model on the complex plane, where the real values are on the x-axis, and imaginary values are on the y-axis. Right-click this terminal on the block diagram and select **CreateIndicator** from the shortcut menu to display this data in a CD Pole-Zero indicator.
- **Poles** returns an array of all the system poles.

- **Zeros** returns all the system zeros. If the system is a MIMO system, this VI calculates the **Zeros** as transmission zeros of the model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Pole-Zero Map (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI plots the poles and zeros on a complex plane.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Pole-Zero Map Plot** is a plot of the poles and zeros of a system model on the complex plane, where the real values are on the x-axis, and imaginary values are on the y-axis. Right-click this terminal on the block diagram and select **CreateIndicator** from the shortcut menu to display this data in a CD Pole-Zero indicator.
- **Poles** returns an array of all the system poles.

- **Zeros** returns all the system zeros. If the system is a MIMO system, this VI calculates the **Zeros** as transmission zeros of the model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Pole-Zero Map (Zero-Pole-Gain)

Zero-Pole-Gain Model	Pole-Zero Map Plot
error in (no error)	
	error out

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI plots the poles and zeros on a complex plane.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Pole-Zero Map Plot** is a plot of the poles and zeros of a system model on the complex plane, where the real values are on the x-axis, and imaginary values are on the y-axis. Right-click this terminal on the block diagram and select **CreateIndicator** from the shortcut menu to display this data in a CD Pole-Zero indicator.
- **Poles** returns an array of all the system poles.

- **Zeros** returns all the system zeros. If the system is a MIMO system, this VI calculates the **Zeros** as transmission zeros of the model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Pole-Zero Map Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays when calculating the dynamic characteristics of a system, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Poles VI

Owning Palette: <u>Dynamic Characteristics VIs</u>

Installed With: Control Design and Simulation Module

Returns the model poles. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Poles (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI returns the poles.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Poles** returns the locations of the model poles. For multiple-input multiple-output (MIMO) system models, this VI calculates the least-common multiple for all the poles of transfer function or zero-pole-gain elements of the MIMO system matrix. These poles are not necessarily identical to the poles you can obtain by using the Smith-McMillan form.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Poles (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI returns the poles.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Poles** returns the locations of the model poles. For multiple-input multiple-output (MIMO) system models, this VI calculates the least-common multiple for all the poles of transfer function or zero-pole-gain elements of the MIMO system matrix. These poles are not necessarily identical to the poles you can obtain by using the Smith-McMillan form.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Poles (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI returns the poles.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Poles** returns the locations of the model poles. For multiple-input multiple-output (MIMO) system models, this VI calculates the least-common multiple for all the poles of transfer function or zero-pole-gain elements of the MIMO system matrix. These poles are not necessarily identical to the poles you can obtain by using the Smith-McMillan form.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Root Locus VI

Owning Palette: <u>Dynamic Characteristics VIs</u>

Installed With: Control Design and Simulation Module

Plots the Evans plot or trajectory of closed-loop poles of a single-input single-output (SISO) system as the feedback gain varies from zero to infinity. You can display this data in a root locus graph indicator. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

Place on the block diagram Find on the Functions palette

CD Root Locus (State-Space)



Root Locus Graph Reference specifies a reference to the Root Locus Graph.

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI generates the root locus, or Evans, plot.
- **Gain Info** specifies the initial and final gain values.
 - **K0** is the initial feedback gain value. The default is 0.
 - **Kf** is the final feedback gain value. The default is 0.
- **Gain** specifies the feedback gain to apply to each open-loop model pole. Use this parameter to determine the feedback gain necessary to place an open-loop pole in a particular location. Each element of the **Gain** parameter corresponds to a different open-loop pole.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Feedback** specifies the type of feedback, positive or negative.

0 **negative** (default)—The closed-loop system has negative feedback.

1 **positive**—The closed-loop system has positive feedback.

- Minimum Number of Points specifies the minimum number of points this VI uses for the root locus plot. The default is 300 points.
- **Root Locus Graph** displays the Evans plot of a SISO feedback system on an XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a root locus graph indicator.
- **Root Locus Data** returns information about the root locus calculation.
 - **K** is a 1D-array of *m* elements of all the gain values that this VI uses to create the root locus.
 - **Roots** is an $m \times n$ array, where n is the number of poles in the closed-loop transfer function. The *i*th row of **Roots** consists of the n closed-loop poles that result when the feedback gain has value equal to the *i*th element of **K**.
- **Root Locus Information** returns information about the root locus plot, such as the bifurcation points and open-loop poles and zeros.
 - **Open-Loop Zeros** returns the zeros of the open-loop system and the location of the closed-loop poles when the gain K equals infinity.
 - **Open-Loop Poles** returns the poles of the closed-loop system when the gain K equals 0.
 - **Bifurcation Points** returns the points in the root locus plot where a pair of real poles splits into a conjugate pair of

poles.

K at Bifurcation Point returns the gain values at which the bifurcation occurs in the root locus plot.

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Root Locus (Transfer Function)



Root Locus Graph Reference specifies a reference to the Root Locus Graph.

Transfer Function Model contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI generates the root locus, or Evans, plot.

Gain Info specifies the initial and final gain values.

K0 is the initial feedback gain value. The default is 0.

Kf is the final feedback gain value. The default is 0.

- **Gain** specifies the feedback gain to apply to each open-loop model pole. Use this parameter to determine the feedback gain necessary to place an open-loop pole in a particular location. Each element of the **Gain** parameter corresponds to a different open-loop pole.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Feedback** specifies the type of feedback, positive or negative.

0 **negative** (default)—The closed-loop system has negative feedback.

1 **positive**—The closed-loop system has positive feedback.

- Minimum Number of Points specifies the minimum number of points this VI uses for the root locus plot. The default is 300 points.
- **Root Locus Graph** displays the Evans plot of a SISO feedback system on an XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a root locus graph indicator.
- **Root Locus Data** returns information about the root locus calculation.
 - **K** is a 1D-array of *m* elements of all the gain values that this VI uses to create the root locus.
 - **Roots** is an $m \times n$ array, where n is the number of poles in the closed-loop transfer function. The *i*th row of **Roots** consists of the n closed-loop poles that result when the feedback gain has value equal to the *i*th element of **K**.
- **Root Locus Information** returns information about the root locus plot, such as the bifurcation points and open-loop poles and zeros.
 - **Open-Loop Zeros** returns the zeros of the open-loop system and the location of the closed-loop poles when the gain K equals infinity.
 - **Open-Loop Poles** returns the poles of the closed-loop system when the gain K equals 0.
 - **Bifurcation Points** returns the points in the root locus plot where a pair of real poles splits into a conjugate pair of

poles.

K at Bifurcation Point returns the gain values at which the bifurcation occurs in the root locus plot.

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Root Locus (Zero-Pole-Gain)



Root Locus Graph Reference specifies a reference to the Root Locus Graph.

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI generates the root locus, or Evans, plot.
- Gain Info specifies the initial and final gain values.
 - **K0** is the initial feedback gain value. The default is 0.
 - **Kf** is the final feedback gain value. The default is 0.
- **Gain** specifies the feedback gain to apply to each open-loop model pole. Use this parameter to determine the feedback gain necessary to place an open-loop pole in a particular location. Each element of the **Gain** parameter corresponds to a different open-loop pole.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Feedback** specifies the type of feedback, positive or negative.

0 **negative** (default)—The closed-loop system has negative feedback.

1 **positive**—The closed-loop system has positive feedback.

- Minimum Number of Points specifies the minimum number of points this VI uses for the root locus plot. The default is 300 points.
- **Root Locus Graph** displays the Evans plot of a SISO feedback system on an XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a root locus graph indicator.
- **Root Locus Data** returns information about the root locus calculation.
 - **K** is a 1D-array of *m* elements of all the gain values that this VI uses to create the root locus.
 - **Roots** is an $m \times n$ array, where n is the number of poles in the closed-loop transfer function. The *i*th row of **Roots** consists of the n closed-loop poles that result when the feedback gain has value equal to the *i*th element of **K**.
- **Root Locus Information** returns information about the root locus plot, such as the bifurcation points and open-loop poles and zeros.
 - **Open-Loop Zeros** returns the zeros of the open-loop system and the location of the closed-loop poles when the gain K equals infinity.
 - **Open-Loop Poles** returns the poles of the closed-loop system when the gain K equals 0.
 - **Bifurcation Points** returns the points in the root locus plot where a pair of real poles splits into a conjugate pair of

poles.

K at Bifurcation Point returns the gain values at which the bifurcation occurs in the root locus plot.

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Root Locus Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays when calculating the dynamics of a system, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Stability VI

Owning Palette: <u>Dynamic Characteristics VIs</u>

Installed With: Control Design and Simulation Module

Determines if the input system is stable, unstable, or marginally stable. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

•

CD Stability (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines stability.
- **Tolerance** specifies the tolerance this VI uses in determining if a pole is on the imaginary axis in continuous systems or in the unit circle in discrete systems. The default is 1E–8.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Stability** returns the stability of the system.

0 **Stable**—For continuous systems, the system is stable if all

	S	oles lie on the left half of the complex plane. For discrete ystems, the system is stable if all poles lie inside the unit rcle.	
	m th m	larginally stable —For continuous systems, the system is narginally stable (or asymptotically stable) if any pole lies on ne imaginary axis. For discrete systems, the system is narginally stable (or asymptotically stable) if any pole lies in the nit circle.	
	a m ol	nstable —For continuous systems, the system is unstable if ny pole lies on the right half of the complex plane or if the nultiplicity of any pole on the imaginary axis is greater than ne. For discrete systems, the system is unstable if any pole es outside the unit circle or if the multiplicity of any pole on the nit circle is greater than one.	
		ndetermined (default)—The VI cannot determine the stability f the system because there is an error or the system is empty.	
[CDB]	Poles returns the poles, which this VI uses to determine the type of Stability , of the input system.		
	erro sam that pan	or out contains error information. If error in indicates that an r occurred before this VI or function ran, error out contains the e error information. Otherwise, it describes the error status this VI or function produces. Right-click the error out front el indicator and select Explain Error from the shortcut menu nore information about the error. status is TRUE (X) if an error occurred or FALSE	
		(checkmark) to indicate a warning or that no error occurred.	
	132	code is the error or warning code. If status is TRUE, code is a nonzero <u>error code</u> . If status is FALSE, code is 0 or a warning code.	
	abc	source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced	

the error or warning.

CD Stability (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines stability.
- **Tolerance** specifies the tolerance this VI uses in determining if a pole is on the imaginary axis in continuous systems or in the unit circle in discrete systems. The default is 1E–8.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Stability** returns the stability of the system.

0 **Stable**—For continuous systems, the system is stable if all

	S	oles lie on the left half of the complex plane. For discrete ystems, the system is stable if all poles lie inside the unit rcle.	
	m th m	larginally stable —For continuous systems, the system is narginally stable (or asymptotically stable) if any pole lies on ne imaginary axis. For discrete systems, the system is narginally stable (or asymptotically stable) if any pole lies in the nit circle.	
	a m ol	nstable —For continuous systems, the system is unstable if ny pole lies on the right half of the complex plane or if the nultiplicity of any pole on the imaginary axis is greater than ne. For discrete systems, the system is unstable if any pole es outside the unit circle or if the multiplicity of any pole on the nit circle is greater than one.	
		ndetermined (default)—The VI cannot determine the stability f the system because there is an error or the system is empty.	
[CDB]	Poles returns the poles, which this VI uses to determine the type of Stability , of the input system.		
	erro sam that pan	or out contains error information. If error in indicates that an r occurred before this VI or function ran, error out contains the e error information. Otherwise, it describes the error status this VI or function produces. Right-click the error out front el indicator and select Explain Error from the shortcut menu nore information about the error. status is TRUE (X) if an error occurred or FALSE	
		(checkmark) to indicate a warning or that no error occurred.	
	132	code is the error or warning code. If status is TRUE, code is a nonzero <u>error code</u> . If status is FALSE, code is 0 or a warning code.	
	abc	source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced	

the error or warning.

CD Stability (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI determines stability.
- **Tolerance** specifies the tolerance this VI uses in determining if a pole is on the imaginary axis in continuous systems or in the unit circle in discrete systems. The default is 1E–8.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Stability** returns the stability of the system.

0 **Stable**—For continuous systems, the system is stable if all

	S	oles lie on the left half of the complex plane. For discrete ystems, the system is stable if all poles lie inside the unit rcle.	
	m th m	larginally stable —For continuous systems, the system is narginally stable (or asymptotically stable) if any pole lies on ne imaginary axis. For discrete systems, the system is narginally stable (or asymptotically stable) if any pole lies in the nit circle.	
	a m ol	nstable —For continuous systems, the system is unstable if ny pole lies on the right half of the complex plane or if the nultiplicity of any pole on the imaginary axis is greater than ne. For discrete systems, the system is unstable if any pole es outside the unit circle or if the multiplicity of any pole on the nit circle is greater than one.	
		ndetermined (default)—The VI cannot determine the stability f the system because there is an error or the system is empty.	
[CDB]	Poles returns the poles, which this VI uses to determine the type of Stability , of the input system.		
	erro sam that pan	or out contains error information. If error in indicates that an r occurred before this VI or function ran, error out contains the e error information. Otherwise, it describes the error status this VI or function produces. Right-click the error out front el indicator and select Explain Error from the shortcut menu nore information about the error. status is TRUE (X) if an error occurred or FALSE	
		(checkmark) to indicate a warning or that no error occurred.	
	132	code is the error or warning code. If status is TRUE, code is a nonzero <u>error code</u> . If status is FALSE, code is 0 or a warning code.	
	abc	source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced	

the error or warning.

CD Stability Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays when calculating the dynamic characteristics of a system, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Total Delay VI

Owning Palette: <u>Dynamic Characteristics VIs</u>

Installed With: Control Design and Simulation Module

Adds all the delays between each input and output pair and returns the total time delay. The total time delay includes the input, output, and transport delays. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

◄

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Total Delay (State-Space)

error in (no error)

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines total delay.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Total Delay** returns all the delays of the system. The *jjth* element of this array corresponds to the total time delay between the *jth* input and the *jth* output of the system the model represents. For continuous-time systems, the delay is in seconds. For discrete time systems, the delay is an integer multiple of the sampling time.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Total Delay (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines total delay.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Total Delay** returns all the delays of the system. The *jjth* element of this array corresponds to the total time delay between the *jth* input and the *jth* output of the system the model represents. For continuous-time systems, the delay is in seconds. For discrete time systems, the delay is an integer multiple of the sampling time.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Total Delay (Zero-Pole-Gain)

error in (no error)

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines total delay.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Total Delay** returns all the delays of the system. The *jjth* element of this array corresponds to the total time delay between the *jth* input and the *jth* output of the system the model represents. For continuous-time systems, the delay is in seconds. For discrete time systems, the delay is an integer multiple of the sampling time.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Total Delay Details

Refer to the <u>LabVIEW Control Design User Manual</u> for information about delay in system models.

CD Zeros VI

Owning Palette: Dynamic Characteristics VIs

Installed With: Control Design and Simulation Module

Returns the locations of the model zeros. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Zeros (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI returns the zeros.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Zeros** returns the locations of the model zeros. For multiple-input multiple-output (MIMO) system models, this VI converts transfer function and zero-pole-gain models to state-space form. This VI then calculates the transmission zeros of the state-space model. These poles are not necessarily identical to the poles you can obtain by using the Smith-McMillan form.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Zeros (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI returns the zeros.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Zeros** returns the locations of the model zeros. For multiple-input multiple-output (MIMO) system models, this VI converts transfer function and zero-pole-gain models to state-space form. This VI then calculates the transmission zeros of the state-space model. These poles are not necessarily identical to the poles you can obtain by using the Smith-McMillan form.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Zeros (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI returns the zeros.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Zeros** returns the locations of the model zeros. For multiple-input multiple-output (MIMO) system models, this VI converts transfer function and zero-pole-gain models to state-space form. This VI then calculates the transmission zeros of the state-space model. These poles are not necessarily identical to the poles you can obtain by using the Smith-McMillan form.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Frequency Response VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Frequency Response VIs to analyze a system model in the frequency domain.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD All</u> <u>Margins</u>	Calculates all the gain and phase margins, which result from the frequency response crossing at 0 decibels (magnitude) and –180 degrees (phase). You can use the CD All Margins VI only with single-input single-output (SISO) systems. You can use this polymorphic to calculate the gain and phase margins of state-space models, transfer function models, zero-pole-gain models, and stand-alone frequency response data. The data types you wire to the State-Space Model and Frequency Info inputs determine the polymorphic instance to use.
<u>CD</u> <u>Bandwidth</u>	Calculates the frequency, relative to the DC gain, at which the magnitude of the frequency response drops below Magnitude Drop (dB) . You can use the CD Bandwidth VI only with single-input single-output (SISO) systems. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Bode</u>	Produces the Bode magnitude and Bode phase plots of the system model on an XY graph. The data types you wire to the State-Space Model and Frequency Info inputs determine the polymorphic instance to use.
<u>CD</u> <u>Evaluate</u> <u>at</u> <u>Frequency</u>	Determines the magnitude and phase of the system at the given frequency. The data type you wire to the State-Space Model input determines the polymorphic instance to use.

<u>CD Gain</u> and <u>Phase</u> <u>Margin</u>	Calculates the gain and phase margins. If a system has multiple crossover frequencies, the CD Gain and Phase Margin VI returns the smallest gain and phase margins. You can use the CD Gain and Phase Margin VI only with single- input single-output (SISO) systems. The data types you wire to the State-Space Model and Frequency Info inputs determine the polymorphic instance to use.
<u>CD Get</u> Frequency Response Data	Gives access to the frequency response information the <u>Frequency Response</u> VIs return. The data types you wire to the Input and Output inputs determine the polymorphic instance to use. To get the frequency response data from all input-output pairs, you must <u>manually select the polymorphic</u> <u>instance</u> to use.
<u>CD</u> <u>Nichols</u>	Creates a Nichols plot of the input system for which this VI plots the magnitude, in decibels, of the frequency response against the phase. You can display this data in the <u>CD</u> <u>Nichols Plot</u> indicator. The data types you wire to the State-Space Model and Frequency Info inputs determine the polymorphic instance to use.
<u>CD</u> Nyquist	Produces the Nyquist plot of the input system for which this VI plots the imaginary part of the frequency response against its real part. You can display this data in the <u>CD Nyquist Plot</u> indicator. The data types you wire to the State-Space Model and Frequency Info inputs determine the polymorphic instance to use.
<u>CD</u> <u>Singular</u> <u>Values</u>	Calculates the singular values of the frequency response of the input model. The data types you wire to the State-Space Model and Frequency Info inputs determine the polymorphic instance to use.

CD All Margins VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Calculates all the gain and phase margins, which result from the frequency response crossing at 0 decibels (magnitude) and –180 degrees (phase). You can use the CD All Margins VI only with single-input single-output (SISO) systems. You can use this polymorphic to calculate the gain and phase margins of <u>state-space models</u>, <u>transfer function models</u>, <u>zero-pole-gain models</u>, and <u>stand-alone frequency</u> response data. The data types you wire to the **State-Space Model** and **Frequency Info** inputs determine the polymorphic instance to use.

This VI converts the state-space and zero-pole-gain models into transfer function models before calculating their margins.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD All Margins (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines gain and phase margins.
- **Frequency Range** contains the frequency information of the model.
 - **Initial frequency** is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear** (default)—Does not convert the magnitude of the frequency response to decibels.

1 **db**—Converts the magnitude of the frequency response to decibels.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Gain Margins** returns information about the gain margins.
 - **G.M. Frequency** lists all the frequencies at which the system phase crosses –180 degrees. The *i*th element of this array corresponds to the *i*th gain margin value in the **Gain Margins** array.
 - **Gain Margins** lists all the gain margins of the system.
- **Phase Margins** returns information about the phase margins.
 - **P.M. Frequency** lists all the frequencies at which the system magnitude crosses 0 decibels. The *j*th element of this array corresponds to the *j*th phase margin value in the **Phase Margins** array.
 - **Phase Margins** lists all the phase margins of the system.
- **Frequency Response Data** returns the data before this VI parameterizes it. To access the frequency response data, use the

<u>CD Get Frequency Response Data</u> VI.

- [DBL] Frequency is a 1D-array of frequency values (in radians/seconds) at which this VI calculates the magnitude and phase.
- [DBL] **Magnitude** is a 1D-array. For continuous-time systems *H*(s), the *i*th element of the array is defined by the following equation:

```
mag = |H(j\omega_j)|
```

For discrete-time systems, with sampling time *T* seconds, the *i*th element of the array is defined by the following equation:

```
mag = |H(e^{jT\omega})|
```

[DBL]

Phase is a 1D-array. For continuous-time systems, the *i*th element is defined as:

 $phase = \angle H(j\omega_i)$

For discrete-time systems, with sampling time T seconds, the *i*th element is defined as:

```
phase = \angle H(e^{jT \omega})
```

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- TF status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- 132 code is the error or warning code. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
- source describes the origin of the error or warning and is, in abc most cases, the name of the VI or function that produced

the error or warning.

CD All Margins (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines gain and phase margins.
- **Frequency Range** contains the frequency information of the model.
 - **Initial frequency** is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear** (default)—Does not convert the magnitude of the frequency response to decibels.

1 **db**—Converts the magnitude of the frequency response to decibels.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Gain Margins** returns information about the gain margins.
 - **G.M. Frequency** lists all the frequencies at which the system phase crosses –180 degrees. The *i*th element of this array corresponds to the *i*th gain margin value in the **Gain Margins** array.
 - **Gain Margins** lists all the gain margins of the system.
- **Phase Margins** returns information about the phase margins.
 - **P.M. Frequency** lists all the frequencies at which the system magnitude crosses 0 decibels. The *j*th element of this array corresponds to the *j*th phase margin value in the **Phase Margins** array.
 - **Phase Margins** lists all the phase margins of the system.
- **Frequency Response Data** returns the data before this VI parameterizes it. To access the frequency response data, use the

<u>CD Get Frequency Response Data</u> VI.

- [DBL] Frequency is a 1D-array of frequency values (in radians/seconds) at which this VI calculates the magnitude and phase.
- [DBL] **Magnitude** is a 1D-array. For continuous-time systems *H*(s), the *i*th element of the array is defined by the following equation:

```
mag = |H(j\omega_j)|
```

For discrete-time systems, with sampling time *T* seconds, the *i*th element of the array is defined by the following equation:

```
mag = |H(e^{jT\omega})|
```

[DBL]

Phase is a 1D-array. For continuous-time systems, the *i*th element is defined as:

 $phase = \angle H(j\omega_i)$

For discrete-time systems, with sampling time T seconds, the *i*th element is defined as:

```
phase = \angle H(e^{jT \omega})
```

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- TF status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- 132 code is the error or warning code. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
- source describes the origin of the error or warning and is, in abc most cases, the name of the VI or function that produced

the error or warning.

CD All Margins (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines gain and phase margins.
- **Frequency Range** contains the frequency information of the model.
 - **Initial frequency** is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear** (default)—Does not convert the magnitude of the frequency response to decibels.

1 **db**—Converts the magnitude of the frequency response to decibels.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Gain Margins** returns information about the gain margins.
 - **G.M. Frequency** lists all the frequencies at which the system phase crosses –180 degrees. The *i*th element of this array corresponds to the *i*th gain margin value in the **Gain Margins** array.
 - **Gain Margins** lists all the gain margins of the system.
- **Phase Margins** returns information about the phase margins.
 - **P.M. Frequency** lists all the frequencies at which the system magnitude crosses 0 decibels. The *j*th element of this array corresponds to the *j*th phase margin value in the **Phase Margins** array.
 - **Phase Margins** lists all the phase margins of the system.
- **Frequency Response Data** returns the data before this VI parameterizes it. To access the frequency response data, use the

<u>CD Get Frequency Response Data</u> VI.

- [DBL] Frequency is a 1D-array of frequency values (in radians/seconds) at which this VI calculates the magnitude and phase.
- [DBL] **Magnitude** is a 1D-array. For continuous-time systems *H*(s), the *i*th element of the array is defined by the following equation:

```
mag = |H(j\omega_j)|
```

For discrete-time systems, with sampling time *T* seconds, the *i*th element of the array is defined by the following equation:

```
mag = |H(e^{jT\omega})|
```

[DBL]

Phase is a 1D-array. For continuous-time systems, the *i*th element is defined as:

 $phase = \angle H(j\omega_i)$

For discrete-time systems, with sampling time T seconds, the *i*th element is defined as:

```
phase = \angle H(e^{jT \omega})
```

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- TF status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- 132 code is the error or warning code. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
- source describes the origin of the error or warning and is, in abc most cases, the name of the VI or function that produced

the error or warning.

CD All Margins (Frequency Response Data)



Frequency Response specifies the frequency response for which this VI calculates the margins.

- Magnitude specifies the magnitude component of the frequency response. The Magnitude Scale parameter defines the units of the Magnitude array.
- **Phase** specifies the phase component, in degrees, of the frequency response.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.
 - 0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear** (default)—Does not convert the magnitude of the frequency response to decibels.

1 **db**—Converts the magnitude of the frequency response to decibels.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Gain Margins** returns information about the gain margins.
 - **G.M. Frequency** lists all the frequencies at which the system phase crosses –180 degrees. The *i*th element of this array corresponds to the *i*th gain margin value in the **Gain Margins** array.
 - **Gain Margins** lists all the gain margins of the system.
- **Phase Margins** returns information about the phase margins.
 - **P.M. Frequency** lists all the frequencies at which the system magnitude crosses 0 decibels. The *i*th element of this array corresponds to the *i*th phase margin value in the **Phase Margins** array.
 - **Phase Margins** lists all the phase margins of the system.
- Frequency Response Data returns the data before this VI parameterizes it. To access the frequency response data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values (in radians/seconds) at which this VI calculates the magnitude

and phase.

Magnitude is a 1D-array. For continuous-time systems H(s), the *i*th element of the array is defined by the following equation:

 $mag = |H(j\omega_j)|$

For discrete-time systems, with sampling time T seconds, the *i*th element of the array is defined by the following equation:

```
mag = \left| H(e^{jT\omega}) \right|
```

[DBL]

Phase is a 1D-array. For continuous-time systems, the *i*th element is defined as:

 $phase = \angle H(j\omega_j)$

For discrete-time systems, with sampling time T seconds, the *i*th element is defined as:

 $phase = \angle H(e^{jT\omega})$

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD All Margins Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Bandwidth VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Calculates the frequency, relative to the DC gain, at which the magnitude of the frequency response drops below **Magnitude Drop (dB)**. You can use the CD Bandwidth VI only with single-input single-output (SISO) systems. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

▼

This VI converts state-space and zero-pole-gain models into transfer function models before calculating the bandwidth.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Bandwidth (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines bandwidth.
- Magnitude Drop (dB) is the magnitude in decibels that defines the level up to which this VI calculates the bandwidth frequency. The default value is –3 dB.
- **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the bandwidth.

1	0	Hz
	1	rad/s (default)

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Bandwidth** is the frequency, relative to the DC gain, at which the magnitude of the frequency response of the system falls below **Magnitude Drop**.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Bandwidth (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines bandwidth.
- Magnitude Drop (dB) is the magnitude in decibels that defines the level up to which this VI calculates the bandwidth frequency. The default value is –3 dB.
- **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the bandwidth.

1	0	Hz
	1	rad/s (default)

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Bandwidth** is the frequency, relative to the DC gain, at which the magnitude of the frequency response of the system falls below **Magnitude Drop**.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Bandwidth (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines bandwidth.
- Magnitude Drop (dB) is the magnitude in decibels that defines the level up to which this VI calculates the bandwidth frequency. The default value is –3 dB.
- **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the bandwidth.

0	Hz
1	rad/s (default)

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Bandwidth** is the frequency, relative to the DC gain, at which the magnitude of the frequency response of the system falls below **Magnitude Drop**.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Bandwidth Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Bode VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Produces the Bode magnitude and Bode phase plots of the system model on an XY graph. The data types you wire to the **State-Space Model** and **Frequency Info** inputs determine the polymorphic instance to use.

▼

This VI converts state-space and zero-pole-gain models to transfer function models before calculating the frequency response.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

CD Bode (Frequency Range, State-Space)



- Phase Graph Reference is a reference to the Bode Phase plot on an XY graph. Phase Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Graph Reference is a reference to the Bode Magnitude plot on an XY graph. Magnitude Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI produces Bode plots.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is –1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.



1 rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

1 **db** (default)—Converts the magnitude of the frequency response to decibels.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Plots Index specifies which input and which output to plot. If you do not specify the input #, this VI plots all the inputs.
 - input # is the index number of the specific input to the system. This VI displays the response to this input on the Bode Magnitude and Bode Phase plots on XY graphs. The

index is zero-based.

- **output** # is the index number of the specific output of the system that this VI displays in the **Bode Magnitude** and **Bode Phase** plots on XY graphs. The index is zero-based.
- **Bode Magnitude** is a plot on an XY graph that plots the linear or decibel magnitude of the given model against a set of frequencies. The **Plots Index** array determines the plots on this XY graph.
- **Bode Phase** is a plot on an XY graph that plots the phase (in degrees) of the model against a set of frequency values. The **Plots Index** array determines the plots on this XY graph.
- Bode Data returns information about the Bode plot. To access the Bode Data, use the <u>CD Get Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude** is a 3D-array. The *j*th column in the *n*th page of the array is the magnitude of the transfer function between the *j*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
 - **Phase** is a 3D-array. The *j*th column in the n^{th} page of the array is the phase of the transfer function between the *j*th output and the n^{th} input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a

warning code.

- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Frequency Info Out** returns the frequency information of the model.

CD Bode (Frequency Range, Transfer Function)



- Phase Graph Reference is a reference to the Bode Phase plot on an XY graph. Phase Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Graph Reference is a reference to the Bode Magnitude plot on an XY graph. Magnitude Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI produces Bode plots.
- **Frequency Range** contains the frequency information of the model.
 - **Initial frequency** is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is –1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.



1 rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

1 **db** (default)—Converts the magnitude of the frequency response to decibels.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Plots Index specifies which input and which output to plot. If you do not specify the input #, this VI plots all the inputs.
 - input # is the index number of the specific input to the system. This VI displays the response to this input on the Bode Magnitude and Bode Phase plots on XY graphs. The

index is zero-based.

- **output** # is the index number of the specific output of the system that this VI displays in the **Bode Magnitude** and **Bode Phase** plots on XY graphs. The index is zero-based.
- **Bode Magnitude** is a plot on an XY graph that plots the linear or decibel magnitude of the given model against a set of frequencies. The **Plots Index** array determines the plots on this XY graph.
- **Bode Phase** is a plot on an XY graph that plots the phase (in degrees) of the model against a set of frequency values. The **Plots Index** array determines the plots on this XY graph.
- Bode Data returns information about the Bode plot. To access the Bode Data, use the <u>CD Get Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude** is a 3D-array. The *j*th column in the *n*th page of the array is the magnitude of the transfer function between the *j*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
 - **Phase** is a 3D-array. The *j*th column in the n^{th} page of the array is the phase of the transfer function between the *j*th output and the n^{th} input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a

warning code.

- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Frequency Info Out** returns the frequency information of the model.

CD Bode (Frequency Range, Zero-Pole-Gain)



- Phase Graph Reference is a reference to the Bode Phase plot on an XY graph. Phase Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Graph Reference is a reference to the Bode Magnitude plot on an XY graph. Magnitude Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI produces Bode plots.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is –1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.



1 rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

1 **db** (default)—Converts the magnitude of the frequency response to decibels.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Plots Index specifies which input and which output to plot. If you do not specify the input #, this VI plots all the inputs.
 - input # is the index number of the specific input to the system. This VI displays the response to this input on the Bode Magnitude and Bode Phase plots on XY graphs. The

index is zero-based.

- **output** # is the index number of the specific output of the system that this VI displays in the **Bode Magnitude** and **Bode Phase** plots on XY graphs. The index is zero-based.
- **Bode Magnitude** is a plot on an XY graph that plots the linear or decibel magnitude of the given model against a set of frequencies. The **Plots Index** array determines the plots on this XY graph.
- **Bode Phase** is a plot on an XY graph that plots the phase (in degrees) of the model against a set of frequency values. The **Plots Index** array determines the plots on this XY graph.
- Bode Data returns information about the Bode plot. To access the Bode Data, use the <u>CD Get Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude** is a 3D-array. The *j*th column in the *n*th page of the array is the magnitude of the transfer function between the *j*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
 - **Phase** is a 3D-array. The *j*th column in the n^{th} page of the array is the phase of the transfer function between the *j*th output and the n^{th} input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a

warning code.

- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Frequency Info Out** returns the frequency information of the model.

CD Bode (Frequency Vector, State-Space)



- Phase Graph Reference is a reference to the Bode Phase plot on an XY graph. Phase Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Graph Reference is a reference to the Bode Magnitude plot on an XY graph. Magnitude Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI produces Bode plots.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

1 **db** (default)—Converts the magnitude of the frequency response to decibels.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** specifies which input and which output to plot. If you do not specify the **input** #, this VI plots all the inputs.
 - input # is the index number of the specific input to the system. This VI displays the response to this input on the Bode Magnitude and Bode Phase plots on XY graphs. The index is zero-based.
 - output # is the index number of the specific output of the system that this VI displays in the Bode Magnitude and Bode Phase plots on XY graphs. The index is zero-based.
- Bode Magnitude is a plot on an XY graph that plots the linear or decibel magnitude of the given model against a set of frequencies. The Plots Index array determines the plots on this XY graph.

- Bode Phase is a plot on an XY graph that plots the phase (in degrees) of the model against a set of frequency values. The **Plots Index** array determines the plots on this XY graph.
- Bode Data returns information about the Bode plot. To access the Bode Data, use the <u>CD Get Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude** is a 3D-array. The *i*th column in the *n*th page of the array is the magnitude of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
 - **Phase** is a 3D-array. The *j*th column in the n^{th} page of the array is the phase of the transfer function between the *j*th output and the n^{th} input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- Frequency Info Out returns the frequency information of the model.

CD Bode (Frequency Vector, Transfer Function)



- Phase Graph Reference is a reference to the Bode Phase plot on an XY graph. Phase Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Graph Reference is a reference to the Bode Magnitude plot on an XY graph. Magnitude Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI produces Bode plots.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

1 **db** (default)—Converts the magnitude of the frequency response to decibels.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** specifies which input and which output to plot. If you do not specify the **input** #, this VI plots all the inputs.
 - input # is the index number of the specific input to the system. This VI displays the response to this input on the Bode Magnitude and Bode Phase plots on XY graphs. The index is zero-based.
 - output # is the index number of the specific output of the system that this VI displays in the Bode Magnitude and Bode Phase plots on XY graphs. The index is zero-based.
- Bode Magnitude is a plot on an XY graph that plots the linear or decibel magnitude of the given model against a set of frequencies. The Plots Index array determines the plots on this XY graph.

- Bode Phase is a plot on an XY graph that plots the phase (in degrees) of the model against a set of frequency values. The **Plots Index** array determines the plots on this XY graph.
- Bode Data returns information about the Bode plot. To access the Bode Data, use the <u>CD Get Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude** is a 3D-array. The *i*th column in the *n*th page of the array is the magnitude of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
 - **Phase** is a 3D-array. The *j*th column in the n^{th} page of the array is the phase of the transfer function between the *j*th output and the n^{th} input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- Frequency Info Out returns the frequency information of the model.

CD Bode (Frequency Vector, Zero-Pole-Gain)



- Phase Graph Reference is a reference to the Bode Phase plot on an XY graph. Phase Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Graph Reference is a reference to the Bode Magnitude plot on an XY graph. Magnitude Graph Reference automatically configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI produces Bode plots.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

1 **db** (default)—Converts the magnitude of the frequency response to decibels.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** specifies which input and which output to plot. If you do not specify the **input** #, this VI plots all the inputs.
 - input # is the index number of the specific input to the system. This VI displays the response to this input on the Bode Magnitude and Bode Phase plots on XY graphs. The index is zero-based.
 - output # is the index number of the specific output of the system that this VI displays in the Bode Magnitude and Bode Phase plots on XY graphs. The index is zero-based.
- Bode Magnitude is a plot on an XY graph that plots the linear or decibel magnitude of the given model against a set of frequencies. The Plots Index array determines the plots on this XY graph.

- Bode Phase is a plot on an XY graph that plots the phase (in degrees) of the model against a set of frequency values. The **Plots Index** array determines the plots on this XY graph.
- Bode Data returns information about the Bode plot. To access the Bode Data, use the <u>CD Get Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude** is a 3D-array. The *i*th column in the *n*th page of the array is the magnitude of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
 - **Phase** is a 3D-array. The *j*th column in the n^{th} page of the array is the phase of the transfer function between the *j*th output and the n^{th} input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- Frequency Info Out returns the frequency information of the model.

CD Bode Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Evaluate at Frequency VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Determines the magnitude and phase of the system at the given frequency. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

Place	on the	block	diagram	Find	on the	Functions	nalette
- Flace		DIOCK	ulayiani			Functions	paielle

•

CD Evaluate at Frequency (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines frequency response.
- **Frequency (rad/s)** is the frequency at which this VI evaluates the system to obtain the magnitude and phase. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Response** returns the complex number that this VI obtains by evaluating the system at **Frequency (rad/s)**.
- **Magnitude** returns the magnitude of **Response** in decibels.

Phase (deg) returns the phase in degrees of **Response**.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Evaluate at Frequency (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines frequency response.
- **Frequency (rad/s)** is the frequency at which this VI evaluates the system to obtain the magnitude and phase. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Response** returns the complex number that this VI obtains by evaluating the system at **Frequency (rad/s)**.
- **Magnitude** returns the magnitude of **Response** in decibels.

Phase (deg) returns the phase in degrees of **Response**.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Evaluate at Frequency (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines frequency response.
- **Frequency (rad/s)** is the frequency at which this VI evaluates the system to obtain the magnitude and phase. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Response** returns the complex number that this VI obtains by evaluating the system at **Frequency (rad/s)**.
- **Magnitude** returns the magnitude of **Response** in decibels.

Phase (deg) returns the phase in degrees of **Response**.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Evaluate at Frequency Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Gain and Phase Margin VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Calculates the gain and phase margins. If a system has multiple crossover frequencies, the CD Gain and Phase Margin VI returns the smallest gain and phase margins. You can use the CD Gain and Phase Margin VI only with single-input single-output (SISO) systems. The data types you wire to the **State-Space Model** and **Frequency Info** inputs determine the polymorphic instance to use.

This VI converts state-space and zero-pole-gain models into transfer function models before calculating the margins.

▼

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Gain and Phase Margin (State-Space)



- Phase Plot Reference is a reference to the Phase Plot. Phase Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Plot Reference is a reference to the Magnitude Plot. Magnitude Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the SISO system for which this VI determines gain and phase margins.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is –1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

- 1 **db** (default)—Converts the magnitude of the frequency response to decibels.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Magnitude Plot** shows the magnitude of the frequency response of the system this VI plots against the frequency.
- **Phase Plot** shows the phase of the frequency response of the system in degrees this VI plots against the frequency.
- Gain and Phase Margins returns information about the gain and phase margins. To access the Gain and Phase Margins, use the

CD Get Frequency Response Data VI.

- **P.M. Frequency** is the 0 decibels crossover frequency that corresponds to the smallest phase margin.
- **Gain Margin** is the smallest gain margin of the system.
- **G.M. Frequency** is the –180 degrees crossover frequency that corresponds to the smallest gain margin.
- **Phase Margin** is the smallest phase margin of the system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Gain and Phase Margin (Transfer Function)



- Phase Plot Reference is a reference to the Phase Plot. Phase Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Plot Reference is a reference to the Magnitude Plot. Magnitude Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the SISO system for which this VI determines gain and phase margins.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is –1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

- 1 **db** (default)—Converts the magnitude of the frequency response to decibels.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Magnitude Plot** shows the magnitude of the frequency response of the system this VI plots against the frequency.
- **Phase Plot** shows the phase of the frequency response of the system in degrees this VI plots against the frequency.
- Gain and Phase Margins returns information about the gain and phase margins. To access the Gain and Phase Margins, use the

CD Get Frequency Response Data VI.

- **P.M. Frequency** is the 0 decibels crossover frequency that corresponds to the smallest phase margin.
- **Gain Margin** is the smallest gain margin of the system.
- **G.M. Frequency** is the –180 degrees crossover frequency that corresponds to the smallest gain margin.
- **Phase Margin** is the smallest phase margin of the system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Gain and Phase Margin (Zero-Pole-Gain)



- Phase Plot Reference is a reference to the Phase Plot. Phase Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Magnitude Plot Reference is a reference to the Magnitude Plot. Magnitude Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the SISO system for which this VI determines gain and phase margins.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is –1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

- 1 **db** (default)—Converts the magnitude of the frequency response to decibels.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Magnitude Plot** shows the magnitude of the frequency response of the system this VI plots against the frequency.
- **Phase Plot** shows the phase of the frequency response of the system in degrees this VI plots against the frequency.
- Gain and Phase Margins returns information about the gain and phase margins. To access the Gain and Phase Margins, use the

CD Get Frequency Response Data VI.

- **P.M. Frequency** is the 0 decibels crossover frequency that corresponds to the smallest phase margin.
- **Gain Margin** is the smallest gain margin of the system.
- **G.M. Frequency** is the –180 degrees crossover frequency that corresponds to the smallest gain margin.
- **Phase Margin** is the smallest phase margin of the system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Gain and Phase Margin (Frequency Response Data)



Magnitude Scale specifies how to scale the magnitude of the frequency response.

0 **linear**—Does not convert the magnitude of the frequency response to decibels.

1 **db** (default)—Converts the magnitude of the frequency response to decibels.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Magnitude Plot** shows the magnitude of the frequency response of the system this VI plots against the frequency.
- **Phase Plot** shows the phase of the frequency response of the system in degrees this VI plots against the frequency.
- Gain and Phase Margins returns information about the gain and phase margins. To access the Gain and Phase Margins, use the <u>CD Get Frequency Response Data</u> VI.

P.M. Frequency is the 0 decibels crossover frequency that

corresponds to the smallest phase margin.

Gain Margin is the smallest gain margin of the system.

G.M. Frequency is the –180 degrees crossover frequency that corresponds to the smallest gain margin.

Phase Margin is the smallest phase margin of the system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Gain and Phase Margin Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Get Frequency Response Data VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Gives access to the frequency response information the Frequency Response VIs return. The data types you wire to the **Input** and **Output** inputs determine the polymorphic instance to use. To get the frequency response data from all input-output pairs, you must <u>manually select the</u> polymorphic instance to use.

•

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Get Frequency Response Data (Input-Output Pair)



- **Frequency Response Data** contains information about the frequency response of a model. Refer to the <u>Details</u> section for more information about the frequency response data.
 - **Frequency** is the frequency response vector at which this VI plots the real versus imaginary parts of the transfer function H(w).
 - Magnitude/Real/Open-Loop Gain contains data about the frequency response of the real part of the elements in the matrix *H*(*w*).
 - **Phase/Imaginary/Open-Loop Phase** contains data about the frequency response of the imaginary part of the elements in the matrix *H*(*w*).
- **Input** determines the index number of each input for which you want to obtain data.
- **Output** determines the index number of each output for which you want to obtain data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Frequency** returns the frequency response vector at which this VI plots the real versus imaginary parts of the transfer function H(w).
- Magnitude/Real/Open-Loop Gain returns the magnitude or real part of the transfer function *H(w)*. For the <u>CD All Margins</u>, <u>CD</u> <u>Bode</u>, and <u>CD Nichols</u> VIs, Magnitude/Real/Open-Loop Gain returns the magnitude of the response. For the <u>CD Nyquist</u> VI, Magnitude/Real/Open-Loop Gain returns the real part of the response.
- Phase/Imaginary/Open-Loop Phase returns the phase or imaginary part of the transfer function *H(w)*. For the <u>CD All</u> <u>Margins</u>, <u>CD Bode</u>, and <u>CD Nichols</u> VIs, <u>Phase/Imaginary/Open-Loop Phase</u> returns the phase of the response. For the <u>CD</u> <u>Nyquist</u> VI, <u>Phase/Imaginary/Open-Loop Phase</u> returns the imaginary part of the response.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

Type of Response Data returns the type of response data.

- 0 bode-magnitude
- 1 bode-phase
- 2 nichols
- 3 nyquist
- 4 singular values (default)
- 5 step
- 6 impulse
- 7 initial
- 8 simulation

CD Get Frequency Response Data (Input-Output List)



- **Frequency Response Data** contains information about the frequency response of a model. Refer to the <u>Details</u> section for more information about the frequency response data.
 - **Frequency** is the frequency response vector at which this VI plots the real versus imaginary parts of the transfer function H(w).
 - Magnitude/Real/Open-Loop Gain contains data about the frequency response of the real part of the elements in the matrix *H*(*w*).
 - **Phase/Imaginary/Open-Loop Phase** contains data about the frequency response of the imaginary part of the elements in the matrix *H*(*w*).
- **Inputs** specifies a list of index numbers of the inputs for which you want to obtain data. The index is zero-based.
- **Outputs** specifies a list of index numbers of the outputs for which you want to obtain data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Frequency** returns the frequency response vector at which this VI plots the real versus imaginary parts of the transfer function H(w).
- Magnitude/Real/Open-Loop Gain returns the magnitude or real part of the transfer function *H(w)*. For the <u>CD All Margins</u>, <u>CD</u> <u>Bode</u>, and <u>CD Nichols</u> VIs, <u>Magnitude/Real/Open-Loop Gain</u> returns the magnitude of the response. For the <u>CD Nyquist</u> VI, <u>Magnitude/Real/Open-Loop Gain</u> returns the real part of the response.
- Phase/Imaginary/Open-Loop Phase returns the phase or imaginary part of the transfer function *H(w)*. For the <u>CD All</u> <u>Margins</u>, <u>CD Bode</u>, and <u>CD Nichols</u> VIs, Phase/Imaginary/Open-Loop Phase returns the phase of the response. For the <u>CD</u> <u>Nyquist</u> VI, Phase/Imaginary/Open-Loop Phase returns the imaginary part of the response.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

Type of Response Data returns the type of response data.

- 0 bode-magnitude
- 1 bode-phase
- 2 nichols
- 3 nyquist
- 4 singular values (default)
- 5 step
- 6 impulse
- 7 initial
- 8 simulation

CD Get Frequency Response Data (All Input-Output)



- **Frequency Response Data** contains information about the frequency response of a model. Refer to the <u>Details</u> section for more information about the frequency response data.
 - **Frequency** is the frequency response vector at which this VI plots the real versus imaginary parts of the transfer function H(w).
 - Magnitude/Real/Open-Loop Gain contains data about the frequency response of the real part of the elements in the matrix *H*(*w*).
 - **Phase/Imaginary/Open-Loop Phase** contains data about the frequency response of the imaginary part of the elements in the matrix *H*(*w*).
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Frequency** returns the frequency response vector at which this VI plots the real versus imaginary parts of the transfer function H(w).
- Magnitude/Real/Open-Loop Gain returns the magnitude or real part of the transfer function *H(w)*. For the <u>CD All Margins</u>, <u>CD</u> Bode, and <u>CD Nichols</u> VIs, Magnitude/Real/Open-Loop Gain returns the magnitude of the response. For the <u>CD Nyquist</u> VI, Magnitude/Real/Open-Loop Gain returns the real part of the response. Refer to the <u>Details</u> section for more information about the frequency response data.
- Phase/Imaginary/Open-Loop Phase returns the phase or imaginary part of the transfer function *H(w)*. For the <u>CD All</u> Margins, <u>CD Bode</u>, and <u>CD Nichols</u> VIs, Phase/Imaginary/Open-Loop Phase returns the phase of the response. For the <u>CD</u> Nyquist VI, Phase/Imaginary/Open-Loop Phase returns the imaginary part of the response. Refer to the <u>Details</u> section for more information about the frequency response data.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **Status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Type of Response Data** returns the type of response data.

- 0 bode-magnitude
- 1 bode-phase

- 2 nichols
 3 nyquist
 4 singular values (default)
 5 step
- 6 impulse
- 7 initial
- 8 simulation

CD Get Frequency Response Data Details

Frequency Response Data

The **Frequency Response Data** data type contains information about the frequency response of all the input-output pairs in the model. You can use the CD Get Frequency Response VI to customize how you want to view this information. The following explanation describes how to view and understand the frequency response data.

The **Frequency** is an array of frequencies this VI records at a certain interval. The **Magnitude/Real/Open-Loop Gain** and **Phase/Imaginary/Open-Loop Phase** arrays are three-dimensional arrays with three index displays. The first index display is the index number of the inputs. The third index display is the index number of the outputs. The data this VI displays in **Magnitude/Real/Open-Loop Gain** and **Phase/Imaginary/Open-Loop Phase** depends on the input-output pair these two index displays specify. The second index display is the index number of the response data. This index display is relative to the index display of the **Frequency**. The indexes are zero-based.

For example, consider the following Frequency Response Data.

<u></u>	Frequency	r
50	1.10	
	Magnitude/Real/Ope	n-Loop Gain
<u>/</u> 1	A) -1.43 < 0.0	0
() 50	-2.86 - 0.0	0
r) o	-4.35 0.0	0
Phase/Imaginary/Open-Loop Phase		
1	-241.22 0.0	00
/ 7)50	-245.92 0.0	00
$\left(\stackrel{A}{\tau} ight) 0$	-249.90 () 0.0	00

If you want to know the frequency response at w = 1.10, set the index displays for Magnitude/Real/Open-Loop Gain and Phase/Imaginary/Open-Loop Phase equal to the index display for Frequency. In this example, the index display is equal to 50. The

Magnitude/Real/Open-Loop Gain indicates that for the input-output pair (1, 0), the magnitude at that frequency is -1.43. The **Phase/Imaginary/Open-Loop Phase** indicates that for the input-output pair (1, 0), the phase is -241.22.

The **Frequency Response Data** displays the frequency response in terms of magnitude and phase when you use the <u>CD Bode</u> VI and <u>CD Nichols</u> VI to analyze the frequency response. The **Frequency Response Data** displays the frequency response in terms of real and imaginary parts when you use the <u>CD Nyquist</u> VI to analyze the frequency response.

Delay Support

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Nichols VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Creates a Nichols plot of the input system for which this VI plots the magnitude, in decibels, of the frequency response against the phase. You can display this data in the <u>CD Nichols Plot</u> indicator. The data types you wire to the **State-Space Model** and **Frequency Info** inputs determine the polymorphic instance to use.

•

This VI converts state-space and zero-pole-gain models into transfer function models before determining the frequency response.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Nichols (Frequency Range, State-Space)

Nichols Graph Reference	000000
State-Space Model	Wichols Plot
Frequency Range 🚽 🛛 🕅	Nichols Data
error in (no error)	error out
Plots Index	

- Nichols Graph Reference is a reference to the Nichols Plot on an XY graph. Nichols Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nichols plot.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nichols Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nichols Plot** on an XY graph. The index is zero-based.
- **Nichols Plot** returns the magnitude (in decibels) of the frequency response of the input system this VI plots against its phase (in degrees). The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nichols Plot indicator.
- Nichols Data returns information about the Nichols plot data. To access the Nichols Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this

VI calculates the magnitude and phase.

- **Magnitude (dB)** is a 3D-array. The *i*th column in the *n*th page of the array is the magnitude of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- **Phase (deg)** is a 3D-array. The *i*th column in the *n*th page of the array is the phase of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nichols (Frequency Range, Transfer Function)

Nichols Graph Reference Transfer Function Model Frequency Range error in (no error) Plots Index	Nichols Data
---	--------------

- Nichols Graph Reference is a reference to the Nichols Plot on an XY graph. Nichols Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nichols Plot.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nichols Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nichols Plot** on an XY graph. The index is zero-based.
- **Nichols Plot** returns the magnitude (in decibels) of the frequency response of the input system this VI plots against its phase (in degrees). The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nichols Plot indicator.
- Nichols Data returns information about the Nichols plot data. To access the Nichols Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this

VI calculates the magnitude and phase.

- **Magnitude (dB)** is a 3D-array. The *i*th column in the *n*th page of the array is the magnitude of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- **Phase (deg)** is a 3D-array. The *i*th column in the *n*th page of the array is the phase of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nichols (Frequency Range, Zero-Pole-Gain)

Zero-Pole-Gain Model

- Nichols Graph Reference is a reference to the Nichols Plot on an XY graph. Nichols Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model**contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nichols plot.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nichols Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nichols Plot** on an XY graph. The index is zero-based.
- **Nichols Plot** returns the magnitude (in decibels) of the frequency response of the input system this VI plots against its phase (in degrees). The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nichols Plot indicator.
- Nichols Data returns information about the Nichols plot data. To access the Nichols Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this

VI calculates the magnitude and phase.

- **Magnitude (dB)** is a 3D-array. The *i*th column in the *n*th page of the array is the magnitude of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- **Phase (deg)** is a 3D-array. The *i*th column in the *n*th page of the array is the phase of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nichols (Frequency Vector, State-Space)

Nichols Graph Reference State-Space Model Frequency Vector error in (no error)	Nichols Plot
Plots Index	

- Nichols Graph Reference is a reference to the Nichols Plot on an XY graph. Nichols Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nichols plot.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nichols Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nichols Plot** on an XY graph. The index is zero-based.
- **Nichols Plot** returns the magnitude (in decibels) of the frequency response of the input system this VI plots against its phase (in degrees). The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nichols Plot indicator.
- Nichols Data returns information about the Nichols plot data. To access the Nichols Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude (dB)** is a 3D-array. The *j*th column in the *n*th page of the array is the magnitude of the transfer function between the *j*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .

[DBL]

Phase (deg) is a 3D-array. The *i*th column in the *n*th page of the array is the phase of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nichols (Frequency Vector, Transfer Function)

Nichols Graph Reference Transfer Function Model Frequency Vector error in (no error) Plots Index	Nichols Data
---	--------------

- Nichols Graph Reference is a reference to the Nichols Plot on an XY graph. Nichols Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nichols Plot.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nichols Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nichols Plot** on an XY graph. The index is zero-based.
- **Nichols Plot** returns the magnitude (in decibels) of the frequency response of the input system this VI plots against its phase (in degrees). The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nichols Plot indicator.
- Nichols Data returns information about the Nichols plot data. To access the Nichols Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude (dB)** is a 3D-array. The *j*th column in the *n*th page of the array is the magnitude of the transfer function between the *j*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .

[DBL]

Phase (deg) is a 3D-array. The *i*th column in the *n*th page of the array is the phase of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nichols (Frequency Vector, Zero-Pole-Gain)

Nichols Graph Reference Zero-Pole-Gain Model Frequency Vector error in (no error) Plots Index	
---	--

- Nichols Graph Reference is a reference to the Nichols Plot on an XY graph. Nichols Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model**contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nichols plot.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nichols Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nichols Plot** on an XY graph. The index is zero-based.
- **Nichols Plot** returns the magnitude (in decibels) of the frequency response of the input system this VI plots against its phase (in degrees). The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nichols Plot indicator.
- Nichols Data returns information about the Nichols plot data. To access the Nichols Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Magnitude (dB)** is a 3D-array. The *j*th column in the *n*th page of the array is the magnitude of the transfer function between the *j*th output and the *n*th input. For example, the first two columns of the 0th page will be magnitudes of transfer functions between outputs y_0 and y_1 and the first input u_0 .

[DBL]

Phase (deg) is a 3D-array. The *i*th column in the *n*th page of the array is the phase of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page will be the phase of transfer functions between outputs y_0 and y_1 and the first input u_0 .

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nichols Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Nyquist VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Produces the Nyquist plot of the input system for which this VI plots the imaginary part of the frequency response against its real part. You can display this data in the <u>CD Nyquist Plot</u> indicator. The data types you wire to the **State-Space Model** and **Frequency Info** inputs determine the polymorphic instance to use.

•

This VI converts state-space and zero-pole-gain models into transfer function models before determining the frequency response.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Nyquist (Frequency Range, State-Space)



- Nyquist Graph Reference is a reference to the Nyquist Plot on an XY graph. Nyquist Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nyquist plot.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nyquist Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nyquist Plot** on an XY graph. The index is zero-based.
- **Nyquist Plot** returns the imaginary part of the complex frequency response of the input system this VI plots against its real part. The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **CreateIndicator** from the shortcut menu to display this data in a CD Nyquist Plot indicator.
- Nyquist Data returns information about the Nyquist plot data. To access the Nyquist Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this

VI calculates the magnitude and phase.

- **Real** returns data about the frequency response of the real part of the elements in the matrix H(W). The *i*th column in the n^{th} page of the array is the real part of the frequency response of the transfer function between the *i*th output and the n^{th} input. For example, the first two columns of the 0th page are real parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- **Imaginary** returns data about the frequency response of the imaginary part of the elements in the matrix H(w). The *i*th column in the *n*th page of the array is the imaginary part of the frequency response of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page are imaginary parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nyquist (Frequency Range, Transfer Function)



- Nyquist Graph Reference is a reference to the Nyquist Plot on an XY graph. Nyquist Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nyquist plot.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nyquist Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nyquist Plot** on an XY graph. The index is zero-based.
- **Nyquist Plot** returns the imaginary part of the complex frequency response of the input system this VI plots against its real part. The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **CreateIndicator** from the shortcut menu to display this data in a CD Nyquist Plot indicator.
- Nyquist Data returns information about the Nyquist plot data. To access the Nyquist Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this

VI calculates the magnitude and phase.

- **Real** returns data about the frequency response of the real part of the elements in the matrix H(W). The *i*th column in the n^{th} page of the array is the real part of the frequency response of the transfer function between the *i*th output and the n^{th} input. For example, the first two columns of the 0th page are real parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- **Imaginary** returns data about the frequency response of the imaginary part of the elements in the matrix H(w). The *i*th column in the *n*th page of the array is the imaginary part of the frequency response of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page are imaginary parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nyquist (Frequency Range, Zero-Pole-Gain)

Nyquist Graph Reference Zero-Pole-Gain Model Frequency Range error in (no error) Plots Index	iist Data
--	-----------

- Nyquist Graph Reference is a reference to the Nyquist Plot on an XY graph. Nyquist Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nyquist plot.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - **Frequency Unit** specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nyquist Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nyquist Plot** on an XY graph. The index is zero-based.
- **Nyquist Plot** returns the imaginary part of the complex frequency response of the input system this VI plots against its real part. The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **CreateIndicator** from the shortcut menu to display this data in a CD Nyquist Plot indicator.
- Nyquist Data returns information about the Nyquist plot data. To access the Nyquist Data, use the CD Get Frequency Response Data VI.
 - **Frequency** is a 1D-array of frequency values at which this

VI calculates the magnitude and phase.

- **Real** returns data about the frequency response of the real part of the elements in the matrix H(W). The *i*th column in the n^{th} page of the array is the real part of the frequency response of the transfer function between the *i*th output and the n^{th} input. For example, the first two columns of the 0th page are real parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- **Imaginary** returns data about the frequency response of the imaginary part of the elements in the matrix H(w). The *i*th column in the *n*th page of the array is the imaginary part of the frequency response of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page are imaginary parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nyquist (Frequency Vector, State-Space)



- Nyquist Graph Reference is a reference to the Nyquist Plot on an XY graph. Nyquist Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nyquist plot.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nyquist Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nyquist Plot** on an XY graph. The index is zero-based.
- **Nyquist Plot** returns the imaginary part of the complex frequency response of the input system this VI plots against its real part. The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nyquist Plot indicator.
- Nyquist Data returns information about the Nyquist plot data. To access the Nyquist Data, use the <u>CD Get Frequency Response</u> <u>Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Real** returns data about the frequency response of the real part of the elements in the matrix H(w). The *i*th column in the n^{th} page of the array is the real part of the frequency response of the transfer function between the *i*th output and the n^{th} input. For example, the first two columns of the 0th page are real parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .

- **Imaginary** returns data about the frequency response of the imaginary part of the elements in the matrix H(w). The *i*th column in the *n*th page of the array is the imaginary part of the frequency response of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page are imaginary parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nyquist (Frequency Vector, Transfer Function)



- Nyquist Graph Reference is a reference to the Nyquist Plot on an XY graph. Nyquist Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nyquist plot.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nyquist Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nyquist Plot** on an XY graph. The index is zero-based.
- **Nyquist Plot** returns the imaginary part of the complex frequency response of the input system this VI plots against its real part. The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nyquist Plot indicator.
- Nyquist Data returns information about the Nyquist plot data. To access the Nyquist Data, use the <u>CD Get Frequency Response</u> <u>Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Real** returns data about the frequency response of the real part of the elements in the matrix H(w). The *i*th column in the n^{th} page of the array is the real part of the frequency response of the transfer function between the *i*th output and the n^{th} input. For example, the first two columns of the 0th page are real parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .

- **Imaginary** returns data about the frequency response of the imaginary part of the elements in the matrix H(w). The *i*th column in the *n*th page of the array is the imaginary part of the frequency response of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page are imaginary parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nyquist (Frequency Vector, Zero-Pole-Gain)

Nyquist Graph Reference Zero-Pole-Gain Model Frequency Vector error in (no error) Plots Index	ì
---	---

- Nyquist Graph Reference is a reference to the Nyquist Plot on an XY graph. Nyquist Graph Reference configures the x-scale, yscale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI produces a Nyquist plot.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Plots Index** is an array of clusters of the following two elements.
 - **input** *#* is the index number of the specific input to the system. This VI displays the response to this input in the **Nyquist Plot** on an XY graph. The index is zero-based.
 - **output** # is the index number of the specific output of the system that this VI displays in the **Nyquist Plot** on an XY graph. The index is zero-based.
- **Nyquist Plot** returns the imaginary part of the complex frequency response of the input system this VI plots against its real part. The **Plots Index** input determines which plots to display on the XY graph. Right-click this terminal on the block diagram and select **Create»Indicator** from the shortcut menu to display this data in a CD Nyquist Plot indicator.
- Nyquist Data returns information about the Nyquist plot data. To access the Nyquist Data, use the <u>CD Get Frequency Response</u> <u>Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI calculates the magnitude and phase.
 - **Real** returns data about the frequency response of the real part of the elements in the matrix H(w). The *i*th column in the n^{th} page of the array is the real part of the frequency response of the transfer function between the *i*th output and the n^{th} input. For example, the first two columns of the 0th page are real parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .

- **Imaginary** returns data about the frequency response of the imaginary part of the elements in the matrix H(w). The *i*th column in the *n*th page of the array is the imaginary part of the frequency response of the transfer function between the *i*th output and the *n*th input. For example, the first two columns of the 0th page are imaginary parts of frequency response of transfer functions between outputs y_0 and y_1 and the first input u_0 .
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Nyquist Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Singular Values VI

Owning Palette: Frequency Response VIs

Installed With: Control Design and Simulation Module

Calculates the singular values of the frequency response of the input model. The data types you wire to the **State-Space Model** and **Frequency Info** inputs determine the polymorphic instance to use.

This VI converts transfer function and zero-pole-gain models into statespace models before determining the singular values.

•

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Singular Values (Frequency Range, State-Space)



- Singular Values Plot Reference is a reference to the Singular Values Plot. Singular Values Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines the frequency response singular values.
- **Frequency Range** contains the frequency information of the model.
 - Initial frequency is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - Frequency Unit specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Type specifies what type of singular value decomposition to perform.

0 H (default)—Calculates the singular value decomposition of H

1 **I+H**—Calculates the singular value decomposition of I+H

2 inverse(H)—Calculates the singular value decomposition of H⁻¹

3 I + inverse(H)—Calculates the singular value decomposition of $I+H^{-1}$

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Singular Values Plot** plots the singular values (in decibels) of the frequency response of the system against a set of frequencies.
- Singular Values Data returns information about the singular values data. To access the Singular Values Data, use the <u>CD Get</u> <u>Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI determines the singular values.

Singular Values is a 2D-array. The *i*th column of this array

contains the data for the *i*th singular value as a function of frequency.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Singular Values (Frequency Range, Transfer Function)



- Singular Values Plot Reference is a reference to the Singular Values Plot. Singular Values Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines the frequency response singular values.
- **Frequency Range** contains the frequency information of the model.
 - **Initial frequency** is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - Frequency Unit specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Type specifies what type of singular value decomposition to perform.

0 H (default)—Calculates the singular value decomposition of H

1 **I+H**—Calculates the singular value decomposition of I+H

2 inverse(H)—Calculates the singular value decomposition of H⁻¹

3 I + inverse(H)—Calculates the singular value decomposition of $I+H^{-1}$

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Singular Values Plot** plots the singular values (in decibels) of the frequency response of the system against a set of frequencies.
- Singular Values Data returns information about the singular values data. To access the Singular Values Data, use the <u>CD Get</u> <u>Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI determines the singular values.

Singular Values is a 2D-array. The *i*th column of this array

contains the data for the *i*th singular value as a function of frequency.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Singular Values (Frequency Range, Zero-Pole-Gain)



- Singular Values Plot Reference is a reference to the Singular Values Plot. Singular Values Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines the frequency response singular values.
- **Frequency Range** contains the frequency information of the model.
 - **Initial frequency** is the minimum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - **Final frequency** is the maximum frequency this VI uses in calculating the frequency response and producing the plots. The default is -1.
 - Minimum number of points is the minimum number of points this VI uses in calculating the frequency response and producing the plots. The default is 100 points.
 - Frequency Unit specifies the units of frequency, either in Hertz or radians/seconds, to use in calculating the frequency response and producing the plots.

0	Hz
1	rad/s (default)

Type specifies what type of singular value decomposition to perform.

 $0 | \mathbf{H}$ (default)—Calculates the singular value decomposition of H

1 **I+H**—Calculates the singular value decomposition of I+H

2 inverse(H)—Calculates the singular value decomposition of H⁻¹

3 I + inverse(H)—Calculates the singular value decomposition of $I+H^{-1}$

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Singular Values Plot** plots the singular values (in decibels) of the frequency response of the system against a set of frequencies.
- Singular Values Data returns information about the singular values data. To access the Singular Values Data, use the <u>CD Get</u> <u>Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI determines the singular values.

Singular Values is a 2D-array. The *i*th column of this array

contains the data for the *i*th singular value as a function of frequency.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Singular Values (Frequency Vector, State-Space)



- Singular Values Plot Reference is a reference to the Singular Values Plot. Singular Values Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines the frequency response singular values.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

Type specifies what type of singular value decomposition to perform.

0 **H** (default)—Calculates the singular value decomposition of H

1 I+H—Calculates the singular value decomposition of I+H

2 inverse(H)—Calculates the singular value decomposition of H-1

3 **I + inverse(H)**—Calculates the singular value decomposition of I+H⁻¹

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Singular Values Plot** plots the singular values (in decibels) of the frequency response of the system against a set of frequencies.
- Singular Values Data returns information about the singular values data. To access the Singular Values Data, use the <u>CD Get</u> <u>Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI determines the singular values.
 - **Singular Values** is a 2D-array. The *i*th column of this array contains the data for the *i*th singular value as a function of frequency.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Singular Values (Frequency Vector, Transfer Function)



- Singular Values Plot Reference is a reference to the Singular Values Plot. Singular Values Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines the frequency response singular values.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

Type specifies what type of singular value decomposition to perform.

0 **H** (default)—Calculates the singular value decomposition of H

1 I+H—Calculates the singular value decomposition of I+H

2 inverse(H)—Calculates the singular value decomposition of H-1

3 I + inverse(H)—Calculates the singular value decomposition of $I+H^{-1}$

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Singular Values Plot** plots the singular values (in decibels) of the frequency response of the system against a set of frequencies.
- Singular Values Data returns information about the singular values data. To access the Singular Values Data, use the <u>CD Get</u> <u>Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI determines the singular values.
 - **Singular Values** is a 2D-array. The *i*th column of this array contains the data for the *i*th singular value as a function of frequency.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Singular Values (Frequency Vector, Zero-Pole-Gain)



- Singular Values Plot Reference is a reference to the Singular Values Plot. Singular Values Plot Reference configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines the frequency response singular values.
- **Frequency Vector** specifies information about the frequencies this VI uses to excite the model.
 - **Frequency** specifies the frequencies this VI uses to excite the model.
 - **Frequency Unit** specifies the unit of measurement of the **Frequency** array.

0 **Hz**—Specifies that the frequency is measured in hertz.

1 **rad/s** (default)—Specifies that the frequency is measured in radians per second.

Type specifies what type of singular value decomposition to perform.

0 **H** (default)—Calculates the singular value decomposition of H

1 I+H—Calculates the singular value decomposition of I+H

2 inverse(H)—Calculates the singular value decomposition of H-1

3 I + inverse(H)—Calculates the singular value decomposition of $I+H^{-1}$

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Singular Values Plot** plots the singular values (in decibels) of the frequency response of the system against a set of frequencies.
- Singular Values Data returns information about the singular values data. To access the Singular Values Data, use the <u>CD Get</u> <u>Frequency Response Data</u> VI.
 - **Frequency** is a 1D-array of frequency values at which this VI determines the singular values.
 - **Singular Values** is a 2D-array. The *i*th column of this array contains the data for the *i*th singular value as a function of frequency.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Singular Values Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

Implementation VIs and Functions

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Implementation VIs and functions to simulate the dynamic response of a discrete system model, deploy a discrete model to a realtime target, implement a discrete Kalman filter, and implement current and predictive observers.

Palette Object	Description
<u>CD</u> <u>Discrete</u> <u>Kalman</u> <u>Filter</u>	Implements a discrete-time, linear time-variant, recursive Kalman filter. You define the system by specifying the stochastic state-space model and noise model as well as the inputs and outputs to the system. The CD Discrete Kalman Filter function calculates the predicted state estimates xhat(k+1 k), the corrected state estimates $xhat(k k)$, the corresponding gains used to calculate these estimates, and the associated estimation error covariances corresponding to these estimates. This function also calculates the estimated output $yhat(k)$.
<u>CD</u> <u>Discrete</u> <u>Observer</u>	Implements a discrete-time observer for a linear state-space system model.
<u>CD</u> <u>Discrete</u> <u>State-</u> <u>Space</u>	Implements a system model in discrete state-space form. You define the system model by specifying the input, output, state, and direct transmission matrices.
<u>CD</u> Discrete <u>Stochastic</u> <u>State-</u> <u>Space</u>	Implements a discrete-time, linear, stochastic state-space system. You define the system model by specifying the input, output, state, and direct transmission matrices. You also specify the matrices relating the process noise to the system states and outputs.
<u>CD</u> <u>Discrete</u>	Implements a system model in discrete transfer function form. You define the system model by specifying the

<u>Transfer</u> Function	Numerator and Denominator of the transfer function equation.
<u>CD</u> <u>Discrete</u> <u>Zero-</u> <u>Pole-Gain</u>	Implements a system model in discrete zero-pole-gain form. You define the system model by specifying the Zeros , Poles , and Gain of the zero-pole-gain equation.
<u>CD State</u> Feedback Controller	Implements a state-space controller where the Controller Action equals –Controller Gain * States.

CD Discrete Kalman Filter Function

Owning Palette: Implementation VIs and Functions

Installed With: Control Design and Simulation Module

Implements a discrete-time, linear time-variant, recursive Kalman filter. You define the system by specifying the stochastic state-space model and noise model as well as the inputs and outputs to the system. The CD Discrete Kalman Filter function calculates the predicted state estimates xhat(k+1|k), the corrected state estimates xhat(k|k), the corresponding gains used to calculate these estimates, and the associated estimation error covariances corresponding to these estimates. This function also calculates the estimated output yhat(k).

Refer to Chapter 16, *Using Stochastic System Models*, of the <u>LabVIEW</u> <u>Control Design User Manual</u> for information about using this function.

<u>Details</u>

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

Parameter	Description
	 Specifies the behavior of this function: Pred gain, w/ check—(Default) Performs prediction and correction with prediction gain; checks the model parameters. Pred gain, w/o check—Performs prediction and correction with prediction gain; does not check the model parameters. No pred gain, w/ check—Performs prediction and correction without prediction gain; checks the model parameters. No pred gain, w/ check—Performs prediction and correction without prediction gain; checks the model parameters. No pred gain, w/o check—Performs prediction and correction without prediction gain; does not check the model parameters. No pred gain, w/o check—Performs prediction and correction without prediction gain; does not check the model parameters. No corr, w/ check—Performs only prediction with prediction gain; checks the model parameters. No corr, w/o check—Performs only prediction with prediction gain; does not check the model parameters. Refer to the Details section for a description of the equations this function uses to calculate the outputs for
Feedthrough	 each of these polymorphic instances. Configures the function to be either a Direct or Indirect feedthrough function. Enable this control by selecting Stochastic State-Space Model from the Parameters list and then selecting Terminal from the Parameter source pull-down menu. If you select Configuration Dialog Box from the Parameter source pull-down menu, LabVIEW disables this control and calculates the feedthrough behavior automatically. If you specify Direct feedthrough, the transmission matrix D in the Stochastic State-Space Model is non-zero. If you specify Indirect feedthrough, the transmission matrix D in the Stochastic State-Space Model is zero. You can

	use the Corrected State Estimate xhat(k k) to design the input u(k) only if you specify Indirect feedthrough.
	Refer to the <u>Details</u> section for a description of the parameters that determine the feedthrough behavior of this function.
Parameters	Lists all the parameters associated with this function. Select a parameter from this list to configure the parameter. When you select a parameter, the parameter and its associated Parameter source control appear in the Parameter Information section of the configuration dialog box.
Preview	Displays a graphical preview, if available, of the function output or configuration.
Parameter Information	Contains the parameters you can configure for this function. You must select a parameter from the Parameters list to make that parameter and its associated Parameter source control visible in the Parameter Information section of the configuration dialog box.
Parameter source	Specifies whether you configure this parameter using the Configuration Dialog Box or a Terminal on the simulation diagram. The default value is Configuration Dialog Box . If you select Terminal , LabVIEW displays an input for that parameter on the simulation diagram, and you can wire values to that input to configure this function programmatically. If you select Configuration Dialog Box , LabVIEW removes that input from the simulation diagram. You then must set the value for this parameter inside the configuration dialog box.
Initial State Estimate xhat(0 –1)	Specifies the initial states from which this function begins estimating the model states. If you do not specify a value for this parameter and you select one of the w/ check instances of this function, Initial State Estimate xhat(0 -1) is a vector of zeros.
Initial Estimation	Specifies the initial covariance matrix of the estimation error. If you do not specify a value for this parameter and

Error Covariance P(0 -1)	you select one of the w / check instances of this function, Initial Estimation Error Covariance P(0 -1) is the identity matrix. Initial Estimation Error Covariance P(0 -1) must be symmetric and positive semi-definite such that $P(0 -1) = P^{T}(0 -1) \ge 0$.
Initialize?	Specifies whether to restart the calculation from any initial values you provide. The default is FALSE.
Second- Order Statistics Noise Model	 Specifies a mathematical representation of the noise model of a stochastic state-space model. You can create a noise model using the <u>CD Construct Noise Model</u> VI. E{w}—Specifies the expected value or mean of the process noise vector. The default is 0. Q—Specifies the covariance matrix of the process noise vector. The default is 0.01. E{v}—Specifies the expected value or mean of the measurement noise vector. The default is 0. R—Specifies the covariance matrix of the measurement noise vector. The default is 0.1. N—Specifies the cross-covariance matrix between the process noise vector. If these noise vectors are uncorrelated, N is a matrix of zeros. The default is 0.
Stochastic State-Space Model	 Specifies a mathematical representation of a stochastic system. You can construct a stochastic state-space model using the CD Construct Stochastic Model VI. A—Specifies the system matrix that describes the dynamics of the states of the system. B—Specifies the input matrix that relates the inputs to the states. G—Specifies the matrix that relates the process noise vector to the model states. C—Specifies the output matrix that relates the outputs to the states. D—Specifies the transmission matrix that relates

the inputs to the outputs.

- **H**—Specifies the matrix that relates the process noise vector to the model outputs.
- Sampling Time (s)—Specifies the sampling time of the system model and determines whether the model represents a continuous-time or discretetime system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discretetime system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the Sampling Time (s) to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the CD Convert Continuous Stochastic to Discrete VI to convert the continuous-time system to the discrete-time equivalent of the system.

Block Diagram Inputs

Parameter	Description
Initial State Estimate xhat(0 –1)	Specifies the initial states from which this function begins estimating the model states. If you do not specify a value for this parameter and you select one of the w/ check instances of this function, Initial State Estimate xhat(0 -1) is a vector of zeros.
Initial Estimation Error Covariance P(0 -1)	Specifies the initial covariance matrix of the estimation error. If you do not specify a value for this parameter and you select one of the w/ check instances of this function, Initial Estimation Error Covariance P(0 -1) is the identity matrix. Initial Estimation Error Covariance P(0 -1) must be symmetric and positive semi-definite such that $P(0 -1) = P^{T}(0 -1) \ge 0$.
Initialize?	Specifies whether to restart the calculation from any initial values you provide. The default is FALSE.
Second- Order Statistics Noise Model	 Specifies a mathematical representation of the noise model of a stochastic state-space model. You can create a noise model using the <u>CD Construct Noise Model</u> VI. E{w}—Specifies the expected value or mean of the process noise vector. The default is 0. Q—Specifies the covariance matrix of the process noise vector. The default is 0.01. E{v}—Specifies the expected value or mean of the measurement noise vector. The default is 0. R—Specifies the covariance matrix of the measurement noise vector. The default is 0.1. R—Specifies the covariance matrix of the measurement noise vector. The default is 0.1. R—Specifies the cross-covariance matrix between the process noise vector and the measurement noise vector and the measurement noise vector. The default is 0.1.
Stochastic	Specifies a mathematical representation of a stochastic
State-	system. You can construct a stochastic state-space model
Space	using the <u>CD Construct Stochastic Model</u> VI.

Model	 A—Specifies the system matrix that describes the dynamics of the states of the system. B—Specifies the input matrix that relates the inputs to the states. G—Specifies the matrix that relates the process noise vector to the model states. C—Specifies the output matrix that relates the outputs to the states. D—Specifies the transmission matrix that relates the inputs to the outputs. H—Specifies the matrix that relates the process noise vector to the model outputs. H—Specifies the matrix that relates the process noise vector to the model outputs. Sampling Time (s)—Specifies the sampling time of the system model and determines whether the model represents a continuous-time or discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0. Note If you use the inputs to create a continuous-time system, setting the Sampling Time (s) to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the CD Convert Continuous Stochastic to Discrete VI to convert the continuous-time system.
error in	Describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out . If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out . Use the <u>Simple Error Handler</u> or <u>General Error Handler</u> VIs to display the description of the

	error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
Output y(k)	Specifies measurements made on the Stochastic State- Space Model . You can use the <u>Discrete Stochastic State-</u> <u>Space</u> function to simulate the model and obtain this parameter. You also can wire in measurements made from a hardware sensor.
input u(k)	Specifies the control action this function applies to the model.

Block Diagram Outputs

Parameter	Description
Estimated Output yhat(k)	Returns the estimated model output at time <i>k</i> .
Corrected State Estimate xhat(k k)	Returns the corrected Kalman state estimate at time k , given all measurements up to and including time k . The length of this vector is equal to the number of model states. This parameter is not available if you select one of the No corr instances of this function.
Predicted State Estimate xhat(k+1 k)	Returns the predicted state estimate for the next time step $k + 1$, given all measurements up to and including time k . The length of this vector is equal to the number of model states.
Kalman Filter Gain M(k)	Returns the Kalman filtered gain matrix this function uses to calculate the Corrected State Estimate xhat(k k) . This parameter is not available if you select one of the No corr instances of this function.
Kalman Predictor Gain L(k)	Returns the gain matrix this function uses to calculate the Predicted State Estimate xhat(k+1 k) . This parameter is not available if you select one of the No pred gain instances of this function.
error out	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
	Returns the covariance matrix of the estimation error associated with the Corrected State Estimate xhat(k k) . This parameter is not available if you select one of the No corr instances of this function.
Prediction Error	Returns the covariance matrix of the estimation error associated with the Predicted State Estimate xhat(k+1 k) ,

Covariance	given all measurements up to and including time k .
P(k+1 k)	

CD Discrete Kalman Filter Details

This function adapts to changes in the **Stochastic State-Space Model** and the **Second-Order Statistics Noise Model** as long as the model dimensions do not change. Therefore, you can use this function with linear time-variant (LTV) models.

This function uses the **Second-Order Statistics Noise Model** to obtain the values of $E\{w(k)\}$, $E\{v(k)\}$, Q(k), R(k), and N(k). The following equations define these terms:

$$\begin{split} & \boldsymbol{Q}(k)\delta_{k/} = \mathsf{E}\{\boldsymbol{w}(k)\cdot\boldsymbol{w}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{w}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{w}(l)\} \\ & \boldsymbol{R}(k)\delta_{k/} = \mathsf{E}\{\boldsymbol{v}(k)\cdot\boldsymbol{v}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{v}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{v}(l)\} \\ & \boldsymbol{N}(k)\delta_{k/} = \mathsf{E}\{\boldsymbol{w}(k)\cdot\boldsymbol{v}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{w}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{v}(l)\} \end{split}$$

where δ_{kl} is the Kronecker delta function. The following equations defines this function: $\delta_{kl} = 1$ when k = l; $\delta_{kl} = 0$ when $k \neq l$.

w(*k*) is the process noise vector.

v(k) is the measurement noise vector.

 $\mathbf{Q}(k)$ is the covariance matrix of w(k).

 $\mathbf{R}(k)$ is the covariance matrix of $\mathbf{v}(k)$.

N(k) is the cross-covariance matrix between w(k) and v(k). If these noise vectors are uncorrelated, N(k) is a matrix of zeros.

E{} denotes the expected value or the mean of the enclosed term(s).

The noise covariance matrices must satisfy the following conditions:

 $\mathbf{Q}(k) = \mathbf{Q}^{\mathsf{T}}(k) \ge 0$ $\mathbf{R}(k) = \mathbf{R}^{\mathsf{T}}(k) > 0$ $\begin{bmatrix} \mathbf{Q} & \mathbf{N} \\ \mathbf{N}^{\mathsf{T}} & \mathbf{R} \end{bmatrix}^{\ge 0}$ $\begin{bmatrix} \overline{\mathbf{Q}} & \overline{\mathbf{N}} \\ \overline{\mathbf{N}}^{\mathsf{T}} & \overline{\mathbf{R}} \end{bmatrix}^{\ge 0}$ $\overline{\mathbf{Q}} = \mathbf{G}(k)\mathbf{Q}(k)\mathbf{G}^{\mathsf{T}}(k)$ $\overline{\mathbf{N}} = \mathbf{G}(k)\mathbf{N}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k)$

$\bar{\boldsymbol{R}}(k) = \boldsymbol{C}(k)\boldsymbol{P}(k|k-1)\boldsymbol{C}^{\mathsf{T}}(k) + \boldsymbol{R}(k) + \boldsymbol{H}(k)\boldsymbol{Q}(k)\boldsymbol{H}(k)^{\mathsf{T}} + \boldsymbol{H}(k)\boldsymbol{N}(k) + \boldsymbol{N}^{\mathsf{T}}(k)\boldsymbol{H}^{\mathsf{T}}(k) > 0$

This function assumes the process noise and measurement noise vectors to be temporally uncorrelated between time steps. This function also assumes the **Initial State Estimate xhat(0|–1)** to be uncorrelated with the noise vectors. If the noise vectors are Gaussian-distributed, this function is an optimal minimum mean square error (MMSE) estimator. However, if these vectors are not Gaussian-distributed, this function is an optimal affine MMSE estimator.

Feedthrough Behavior

The value you specify for the **D** matrix of the **Stochastic State-Space Model** parameter determines the feedthrough behavior of this function.

- If the **D** matrix is nonzero, all input/output pairs have direct feedthrough behavior.
- If the **D** matrix is zero, the following input/output pairs have indirect feedthrough behavior.
 - Input u(k) Corrected State Estimate xhat(k|k)
 - Input u(k) Predicted State Estimate xhat(k|k-1)

The remaining input/output pairs have direct feedthrough behavior.

When you use the configuration dialog box to configure the **D** matrix, LabVIEW verifies that the feedthrough behavior is correct. For example, if you set the **Feedthrough** parameter to **Indirect**, and you set the **D** matrix to nonzero, LabVIEW changes the **Feedthrough** parameter to **Direct**.

If you specify the value of the **D** matrix programmatically by wiring a value to the parameter terminal, LabVIEW does not adjust the feedthrough behavior for you. You must ensure that you specify the proper feedthrough behavior for the value of the **D** matrix that you specify.

Kalman Filter Structures

This function uses the following equations to calculate the outputs for each of the polymorphic instances.

Pred Gain Instances—Correction Stage

 $\begin{aligned} & \text{xhat}(k|k) = \text{xhat}(k|k-1) + M(k) \cdot [y(k) - \text{yhat}(k)] \\ & \text{yhat}(k) = \mathbf{C}(k) \cdot \text{xhat}(k|k-1) + \mathbf{D}(k) \cdot u(k) \\ & \text{M}(k) = P(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) \cdot [\mathbf{C}(k) \cdot P(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) + \mathbf{H}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k) + \\ & \mathbf{H}(k)\mathbf{N}(k) + \mathbf{N}^{\mathsf{T}}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{R}(k)]^{-1} \\ & P(k|k) = P(k|k-1) - M(k) \cdot \mathbf{C}(k) \cdot P(k|k-1) \end{aligned}$

Pred Gain Instances—Prediction Stage

 $xhat(k+1|k) = \mathbf{A}(k) \cdot xhat(k|k-1) + \mathbf{B}(k) \cdot u(k) + L(k) \cdot [y(k) - yhat(k)]$ $yhat(k) = \mathbf{C}(k) \cdot xhat(k|k-1) + \mathbf{D}(k) \cdot u(k)$

$$L(k) = [\mathbf{A}(k) \cdot \mathsf{P}(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{N}(k)] \cdot [\mathbf{C}(k) \cdot \mathsf{P}(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) + \mathbf{H}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{H}(k)\mathbf{N}(k) + \mathbf{N}^{\mathsf{T}}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{R}(k)]^{-1}$$

 $P(k+1|k) = [\mathbf{A}(k) \cdot P(k|k-1) \cdot \mathbf{A}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{G}^{\mathsf{T}}(k)] - L(k) \cdot [\mathbf{A}(k) \cdot P(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{N}(k)]^{\mathsf{T}}$

No Pred Gain Instances—Correction Stage

 $xhat(k|k) = xhat(k|k-1) + M(k) \cdot [y(k) - yhat(k)]$

 $yhat(k) = \mathbf{C}(k) \cdot xhat(k|k-1) + \mathbf{D}(k) \cdot u(k)$

$$\mathsf{M}(k) = \mathsf{P}(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) \cdot [\mathbf{C}(k) \cdot \mathsf{P}(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) + \mathbf{H}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k) + \mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k) + \mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k) + \mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k)\mathbf{H}(k) + \mathbf{H}(k)\mathbf$$

 $\mathbf{H}(k)\mathbf{N}(k) + \mathbf{N}^{\mathsf{T}}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{R}(k)]^{-1}$

 $\mathsf{P}(k|k) = \mathsf{P}(k|k-1) - \mathsf{M}(k) \cdot \mathbf{C}(k) \cdot \mathsf{P}(k|k-1)$

No Pred Gain Instances—Prediction Stage

 $\operatorname{xhat}(k+1|k) = \mathbf{A}(k)\operatorname{xhat}(k|k-1) + \mathbf{B}(k)u(k)$

No Corr Instances—Prediction Stage

 $\begin{aligned} & \text{xhat}(k+1|k) = \mathbf{A}(k) \cdot \text{xhat}(k|k-1) + \mathbf{B}(k) \cdot u(k) + L(k) \cdot [y(k) - \text{yhat}(k)] \\ & \text{yhat}(k) = \mathbf{C}(k) \cdot \text{xhat}(k|k-1) + \mathbf{D}(k) \cdot u(k) \\ & \text{L}(k) = [\mathbf{A}(k) \cdot P(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{N}(k)] \cdot [\mathbf{C}(k) \cdot P(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) + \mathbf{H}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{H}(k)\mathbf{N}(k) + \mathbf{N}^{\mathsf{T}}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{R}(k)]^{-1} \end{aligned}$

 $P(k+1|k) = [\mathbf{A}(k) \cdot P(k|k-1) \cdot \mathbf{A}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{G}^{\mathsf{T}}(k)] - L(k) \cdot [\mathbf{A}(k) \cdot P(k|k-1) \cdot \mathbf{C}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{H}^{\mathsf{T}}(k) + \mathbf{G}(k)\mathbf{N}(k)]^{\mathsf{T}}$

CD Discrete Observer Function

Owning Palette: Implementation VIs and Functions

Installed With: Control Design and Simulation Module

Implements a discrete-time observer for a linear state-space system model.

Details

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

Parameter	Description
Polymorphic instance	Specifies whether this function implements a current observer and corrects state estimates made at a previous time step, or whether this function implements a predictive observer and calculates the estimated states for the next time step $k + 1$. This option also specifies whether the observer checks the model parameters. The default is Current , w/ check. Refer to Chapter 15, <i>Estimating Model States</i> , of the LabVIEW Control Design User Manual for information about using current and predictive observers.
Feedthrough	Configures the function to be either a Direct or Indirect feedthrough function. Enable this control by selecting State-Space Model from the Parameters list and then selecting Terminal from the Parameter source pull-down menu. If you select Configuration Dialog Box from the Parameter source pull-down menu, LabVIEW disables this control and calculates the feedthrough behavior automatically.
	You can use the Corrected State Estimate xhat(k k) to design the Input u(k) only if you specify Indirect feedthrough.
	Refer to the <u>Details</u> section for a description of the parameters that determine the feedthrough behavior of this function.
Parameters	Lists all the parameters associated with this function. Select a parameter from this list to configure the parameter. When you select a parameter, the parameter and its associated Parameter source control appear in the Parameter Information section of the configuration dialog box.
Preview	Displays a graphical preview, if available, of the function output or configuration.

Parameter Information	Contains the parameters you can configure for this function. You must select a parameter from the Parameters list to make that parameter and its associated Parameter source control visible in the Parameter Information section of the configuration dialog box.
Parameter source	Specifies whether you configure this parameter using the Configuration Dialog Box or a Terminal on the simulation diagram. The default value is Configuration Dialog Box . If you select Terminal , LabVIEW displays an input for that parameter on the simulation diagram, and you can wire values to that input to configure this function programmatically. If you select Configuration Dialog Box , LabVIEW removes that input from the simulation diagram. You then must set the value for this parameter inside the configuration dialog box.
Initial State Estimate xhat(0 -1)	Specifies the initial states from which this function begins estimating the model states. If you do not specify a value for this parameter and you select one of the w/ check instances of this function, Initial State Estimate xhat(0 – 1) is a vector of zeros.
Observer Gain	Specifies the estimator gain matrix this VI applies to the difference between the observed output and the estimated output, which is Output y(k) – Estimated Output yhat(k) . You can use the <u>CD Pole Placement</u> VI or the <u>CD Ackermann</u> VI to calculate the Observer Gain . The Observer Gain for a current observer relates to the Observer Gain for a predictive observer through the following relationship: Observer Gain (Current) = $A^{-1} \cdot$ Observer Gain (Predictive), where <i>A</i> is the state matrix of the State-Space Model .
State-Space Model	Specifies the <u>mathematical representation</u> of and <u>information</u> about the system for which this function implements the observer.

•	Model name-	-Specifies the	e name	of the	state-
	space model.				

- **Sampling Time**—Specifies the sampling time of the system model and determines whether the model represents a continuous-time or discrete-time system. If the model represents a continuous-time system, **Sampling Time** must equal zero. If the model represents a discrete-time system, **Sampling Time** must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous Stochastic to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **A**—Specifies the *n* × *n* state matrix of the given system.
- **B**—Specifies the *n* × *m* input matrix of the given system.
- **C**—Specifies the *r* × *n* output matrix of the given system.
- **D**—Specifies the *r* × *m* direct transmission matrix of the given system.

where *m* is the number of inputs

n is the number of states

r is the number of outputs

Block Diagram Inputs

Parameter	Description			
Initial State Estimate xhat(0 -1)	Specifies the initial states from which this function begins estimating the model states. If you do not specify a value for this parameter and you select one of the w/ check instances of this function, Initial State Estimate xhat(0 -1) is a vector of zeros.			
Observer Gain				
Output y(k)	Specifies the measurements a sensor makes on the State- Space Model . You also can use the <u>Discrete State-Space</u> function to simulate the behavior of a state-space model.			
Initialize	Specifies whether to restart the calculation from any initial values you provide. The default is FALSE.			
Input u(k)	Specifies the control action this VI applies to the model.			
State- Space Model	Specifies the <u>mathematical representation</u> of and <u>information</u> about the system for which this function calculates the estimated states.			
error in	Describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out . If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out . Use the <u>Simple Error Handler</u> or <u>General Error</u>			

Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

Block Diagram Outputs

Parameter	Description
Estimated Output yhat(k)	Returns the estimated output of the State-Space Model .
Predicted State Estimate xhat(k+1 k)	Returns the predicted state estimates of the State-Space Model for the next time step $k + 1$. The length of this vector is equal to the number of model states.
Corrected State Estimate xhat(k k)	Returns the corrected state estimates at time <i>k</i> . The length of this vector is equal to the number of model states. This output appears only if you select Current , w/ check or Current , w/o check from the Polymorphic instance pull-down menu.
Predicted State Estimate xhat(k k-1)	Returns the estimated states of the State-Space Model estimated at the previous time step. This output appears only if you select Predictive , w/ check or Predictive , w/o check from the Polymorphic instance pull-down menu.
error out	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Discrete Observer Details

Current Observer

At each time step k, this function calculates the **Corrected State Estimate xhat(k|k)** using the **Estimated Output yhat(k)**, any control action **Input u(k)**, and the **Predicted State Estimate xhat(k|k-1)** calculated at the previous time step. This function also calculates the estimated states for the next time step k + 1, **Predicted State Estimate xhat(k+1|k)**.

This VI uses the following equations to calculate the outputs.

 $yhat(k) = C(k) \cdot xhat(k|k-1) + D(k) \cdot u(k)$

where **C** and **D** are the output and direct feedthrough matrices, respectively, of the **State-Space Model**.

 $\operatorname{xhat}(k|k) = \operatorname{xhat}(k|k-1) + \operatorname{Lc}[y(k) - \operatorname{yhat}(k)]$

where Lc is the Observer Gain.

 $\operatorname{xhat}(k+1|k) = \mathbf{A}(k) \cdot \operatorname{xhat}(k|k) + \mathbf{B}(k) \cdot u(k)$

where **A** and **B** are the state and input matrices, respectively, of the **State-Space Model**.

Predictive Observer

At each time step k, this function calculates the **Predicted State** Estimate xhat(k+1|k) using the Estimated Output yhat(k) and the Predicted State Estimate xhat(k|k-1).

The following equations define the outputs of this function:

 $yhat(k) = C(k) \cdot xhat(k|k-1) + D(k) \cdot u(k)$

 $xhat(k+1|k) = \mathbf{A}(k) \cdot xhat(k|k-1) + \mathbf{B}(k) \cdot u(k) + \mathbf{Lp}[y(k) - yhat(k)]$

where Lp is the Observer Gain.
Feedthrough Behavior

The value you specify for the **D** matrix of the **Stochastic State-Space Model** parameter determines the feedthrough behavior of this function.

- If the **D** matrix is nonzero, all input/output pairs have direct feedthrough behavior.
- If the **D** matrix is zero, the following input/output pairs have indirect feedthrough behavior.
 - Input u(k) Corrected State Estimate xhat(k|k)
 - Input u(k) Predicted State Estimate xhat(k|k-1)

The remaining input/output pairs have direct feedthrough behavior.

When you use the configuration dialog box to configure the **D** matrix, LabVIEW verifies that the feedthrough behavior is correct. For example, if you set the **Feedthrough** parameter to **Indirect**, and you set the **D** matrix to nonzero, LabVIEW changes the **Feedthrough** parameter to **Direct**.

If you specify the value of the **D** matrix programmatically by wiring a value to the parameter terminal, LabVIEW does not adjust the feedthrough behavior for you. You must ensure that you specify the proper feedthrough behavior for the value of the **D** matrix that you specify.

CD Discrete State-Space Function

Owning Palette: Implementation VIs and Functions

Installed With: Control Design and Simulation Module

Implements a system model in discrete state-space form. You define the system model by specifying the input, output, state, and direct transmission matrices.

Details

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

Dialog Box Options

_	
Parameter	Description
Polymorphic instance	Specifies whether this function is single-input single- output (SISO) or multiple-input multiple-output (MIMO). The default value is SISO .
Feedthrough	Configures the function to be either a Direct or Indirect feedthrough function. Enable this control by selecting a parameter from the Parameters list and then selecting Terminal from the Parameter source pull-down menu. If you select Configuration Dialog Box from the Parameter source pull-down menu, LabVIEW disables this control and calculates the feedthrough behavior automatically.
	Refer to the <u>Details</u> section for a description of the parameters that determine the feedthrough behavior of this function.
Parameters	Lists all the parameters associated with this function. Select a parameter from this list to configure the parameter. When you select a parameter, the parameter and its associated Parameter source control appear in the Parameter Information section of the configuration dialog box.
Preview	Displays a graphical preview, if available, of the function output or configuration.
Parameter Information	Contains the parameters you can configure for this function. You must select a parameter from the Parameters list to make that parameter and its associated Parameter source control visible in the Parameter Information section of the configuration dialog box.
Parameter source	Specifies whether you configure this parameter using the Configuration Dialog Box or a Terminal on the simulation diagram. The default value is Configuration Dialog Box. If you select Terminal, LabVIEW displays an input for that parameter on the simulation diagram, and

	you can wire values to that input to configure this function programmatically. If you select Configuration Dialog Box , LabVIEW removes that input from the simulation diagram. You then must set the value for this parameter inside the configuration dialog box.
State-Space	 Specifies the state-space model. Load Model—Loads model information from a data file. Save Model—Saves model information to a data file. This file is compatible with the <u>Control Design</u> VIs and functions. Copy to Clipboard—Copies the current model definition to the clipboard. From the clipboard, you can paste the model on the block diagram or into another configuration dialog box of the same model form. Paste from Clipboard—Pastes model information from the clipboard to the configuration dialog box. Model Dimensions—Use this section to specify the number of inputs, states, and outputs of the system model. Inputs—Specifies the number of model inputs. States—Specifies the number of model outputs. Φ—Specifies the <i>n</i> × <i>n</i> state matrix of the given system. C—Specifies the <i>n</i> × <i>n</i> output matrix of the given system. D—Specifies the <i>r</i> × <i>m</i> direct transmission matrix of the given system.

	where <i>m</i> is the number of inputs
	<i>n</i> is the number of states
	<i>r</i> is the number of outputs
initial state (x0)	Specifies a vector of initial states for the system. This vector must be of length <i>n</i> , where <i>n</i> is the number of states.

Block Diagram Inputs

Parameter	Description
initial state (x0)	Specifies a vector of initial states for the system. This vector must be of length n , where n is the number of states.
initialize	Sets the model state(s) to 0, when TRUE.
input u(k)	Specifies the input to the system. input u(k) must be a vector of length <i>m</i> , where <i>m</i> is the number of inputs.
State- Space	Specifies a state-space model. This input accepts either a block diagram constant or a model you created using the Control Design VIs and functions.
error in	Describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out . If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out . Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

Block Diagram Outputs

Parameter	Description
output y(k)	Returns the current output of the system. This vector must be of length <i>r</i> , where <i>r</i> is the number of outputs.
state x(k+1)	Returns the values of the model state(s) at time $k + 1$. The length of this vector is equal to the number of model states.
state x(k)	Returns a one-dimensional array that specifies the internal states. This array must be of length <i>n</i> , where <i>n</i> is the number of states. Each element of the array represents <i>x</i> [<i>n</i>].
error out	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Discrete State-Space Details

The following equations define the state-space system.

 $\begin{aligned} \boldsymbol{x}[t_{k+1}] &= \Phi \boldsymbol{x}[t_k] + \Gamma \boldsymbol{u}[t_k] \\ \boldsymbol{y}[t_k] &= \boldsymbol{C} \boldsymbol{x}[t_k] + \boldsymbol{D} \boldsymbol{u}[t_k] \end{aligned}$

where *u* is the input vector

y is the output vector

x is the state vector

 Φ , Γ , C, D are the state-space matrices. Refer to the **State-Space** parameter help for more information about these matrices.

If you place this function outside a Simulation Loop, $t_k = k *$ **Sampling Time (s)**, for k = 0, 1, 2, ...

If you place this function inside a Simulation Loop, $t_k = initial time +$ sample skew (s) + k * sample period (s), for k = 0, 1, 2, ...

Feedthrough Behavior

The value you specify for the **D** matrix of the **State-Space** parameter determines the feedthrough behavior of this function.

- If the *D* matrix is nonzero, all input/output pairs have direct feedthrough behavior.
- If the *D* matrix is zero, the following input/output pairs have indirect feedthrough behavior.
 - input u(k) output y(k)
 - input u(k) state x(k)

The remaining input/output pairs have direct feedthrough behavior.

When you use the configuration dialog box to configure the *D* matrix, LabVIEW verifies that the feedthrough behavior is correct. For example, if you set the **Feedthrough** parameter to **Indirect**, and you set the *D* matrix to nonzero, LabVIEW changes the **Feedthrough** parameter to **Direct**.

If you specify the value of the **D** matrix programmatically by wiring a value to the parameter terminal, LabVIEW does not adjust the feedthrough behavior for you. You must ensure that you specify the proper feedthrough behavior for the value of the **D** matrix that you specify.

CD Discrete Stochastic State-Space Function

Owning Palette: Implementation VIs and Functions

Installed With: Control Design and Simulation Module

Implements a discrete-time, linear, stochastic state-space system. You define the system model by specifying the input, output, state, and direct transmission matrices. You also specify the matrices relating the process noise to the system states and outputs.

If you use the **Internal Noise** instance of this function, this function generates samples of the noise vectors using the model you wire to the **Second-Order Statistics Noise Model** input. If you use the **External Noise** instance of this function, you can use the <u>CD Correlated Gaussian</u> Random Noise VI to generate samples of the noise vectors. You also can generate samples of the noise vectors using other VIs or specify the noise vectors to be deterministic disturbances.

<u>Details</u>

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

Parameter	Description
Polymorphic instance	Specifies whether this function uses internally-generated random samples of Gaussian-distributed noise vectors or externally-generated noise or disturbances. The default is Internal Noise .
Feedthrough	Configures the function to be either a Direct or Indirect feedthrough function. Enable this control by selecting Stochastic State-Space Model from the Parameters list and then selecting Terminal from the Parameter source pull-down menu. If you select Configuration Dialog Box from the Parameter source pull-down menu, LabVIEW disables this control and calculates the feedthrough behavior automatically. Refer to the <u>Details</u> section for a description of the
	parameters that determine the feedthrough behavior of this function.
Parameters	Lists all the parameters associated with this function. Select a parameter from this list to configure the parameter. When you select a parameter, the parameter and its associated Parameter source control appear in the Parameter Information section of the configuration dialog box.
Preview	Displays a graphical preview, if available, of the function output or configuration.
Parameter Information	Contains the parameters you can configure for this function. You must select a parameter from the Parameters list to make that parameter and its associated Parameter source control visible in the Parameter Information section of the configuration dialog box.
Parameter source	Specifies whether you configure this parameter using the Configuration Dialog Box or a Terminal on the simulation diagram. The default value is Configuration Dialog Box . If you select Terminal , LabVIEW displays an

	input for that parameter on the simulation diagram, and you can wire values to that input to configure this function programmatically. If you select Configuration Dialog Box , LabVIEW removes that input from the simulation diagram. You then must set the value for this parameter inside the configuration dialog box.
Stochastic State-Space Model	 Specifies a mathematical representation of a stochastic system. You can construct a stochastic state-space model using the CD Construct Stochastic Model VI. A—Specifies the system matrix that describes the dynamics of the states of the system. B—Specifies the input matrix that relates the inputs to the states. G—Specifies the matrix that relates the process noise vector to the model states. C—Specifies the transmission matrix that relates the outputs to the states. D—Specifies the transmission matrix that relates the inputs to the outputs. H—Specifies the matrix that relates the process noise vector to the model outputs. Sampling Time (s)—Specifies the sampling time of the system model and determines whether the model represents a continuous-time system. If the model represents a discrete-time system. Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0. Note If you use the inputs to create a continuous-time system, setting the Sampling Time (s) to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the CD Convert Continuous Stochastic to Discrete VI to convert the continuous-time

	system to the discrete-time equivalent of the system.
Second- Order Statistics Noise Model	 Specifies a mathematical representation of the noise model of a stochastic state-space model. You can create a noise model using the <u>CD Construct Noise Model</u> VI. This option is available only if you select Internal Noise from the Polymorphic instance pull-down menu. E{w}—Specifies the expected value or mean of the process noise vector. The default is 0. Q—Specifies the covariance matrix of the process noise vector. The default is 0.01. E{v}—Specifies the expected value or mean of the measurement noise vector. The default is 0. R—Specifies the covariance matrix of the measurement noise vector. The default is 0.1. N—Specifies the cross-covariance matrix between the process noise vector. If these noise vectors are uncorrelated, N is a matrix of zeros. The default is 0.
Initial State x(0)	Specifies the initial state of the model. If you do not specify a value for this parameter, this function assumes an initial state of 0.

Block Diagram Inputs

Parameter	Description
Stochastic State-Space Model	 Specifies a mathematical representation of a stochastic system. You can construct a stochastic state-space model using the <u>CD Construct Stochastic Model</u> VI. A—Specifies the system matrix that describes the dynamics of the states of the system. B—Specifies the input matrix that relates the inputs to the states. G—Specifies the model states. C—Specifies the output matrix that relates the outputs to the states. D—Specifies the output matrix that relates the outputs to the states. D—Specifies the output matrix that relates the outputs to the states. D—Specifies the transmission matrix that relates the outputs to the states. D—Specifies the matrix that relates the process noise vector to the model outputs. H—Specifies the matrix that relates the process noise vector to the model outputs. Sampling Time (s)—Specifies the sampling time of the system model and determines whether the model represents a continuous-time or discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0. Note If you use the inputs to create a continuous-time system. You must use the CD Convert Continuous Stochastic to Discrete VI to convert the continuous-time system to the discrete-time equivalent of the system.

Second- Order Statistics Noise Model	 Specifies a mathematical representation of the noise model of a stochastic state-space model. You can create a noise model using the <u>CD Construct Noise Model</u> VI. This option is available only if you select Internal Noise from the Polymorphic instance pull-down menu. E{w}—Specifies the expected value or mean of the process noise vector. The default is 0. Q—Specifies the covariance matrix of the process noise vector. The default is 0.01. E{v}—Specifies the expected value or mean of the measurement noise vector. The default is 0. R—Specifies the covariance matrix of the measurement noise vector. The default is 0.1. N—Specifies the cross-covariance matrix between the process noise vector. If these noise vectors are uncorrelated, N is a matrix of zeros. The default is 0.
Initial State x(0)	Specifies the initial state of the model. If you do not specify a value for this parameter, this function assumes an initial state of 0.
Initialize	Specifies whether to restart the calculation from any initial values you provide. The default is FALSE.
Input u(k)	Specifies the control action this function applies to the model. If you specify a vector of zeros for Input u(k) , this function does not apply a control action.
error in	Describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out . If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out . Use the <u>Simple Error Handler</u> or <u>General Error Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

Noise w(k)	Specifies the process noise vector. This input is available only if you select External Noise from the Polymorphic instance pull-down menu.
Noise v(k)	Specifies the measurement noise vector. This input is available only if you select External Noise from the Polymorphic instance pull-down menu.

Block Diagram Outputs

Parameter	Description
Output y(k)	Returns the values of the model output(s) at time <i>k</i> . The length of this vector is equal to the number of model outputs.
State x(k+1)	Returns the values of the model state(s) at time $k + 1$. The length of this vector is equal to the number of model states.
State x(k)	Returns the values of the model state(s) at time <i>k</i> . The length of this vector is equal to the number of model states.
error out	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Discrete Stochastic State-Space Details

Internal Noise

This function adapts to changes in the **Stochastic State-Space Model** and the **Second-Order Statistics Noise Model**, as long as the model dimensions do not change. Therefore, you can use this function to simulate linear time-variant (LTV) stochastic state-space models.

If you set the **Parameter source** to **Terminal** for the **Second-Order Statistics Noise Model** input, this function uses the **Second-Order Statistics Noise Model** to obtain the values of $E\{w(k)\}, E\{v(k)\}, Q(k), R(k), and N(k)$. The following equations define these terms:

$$\begin{split} & \boldsymbol{Q}(k)\delta_{kl} = \mathsf{E}\{\boldsymbol{w}(k)\cdot\boldsymbol{w}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{w}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{w}(l)\} \\ & \boldsymbol{R}(k)\delta_{kl} = \mathsf{E}\{\boldsymbol{v}(k)\cdot\boldsymbol{v}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{v}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{v}(l)\} \\ & \boldsymbol{N}(k)\delta_{kl} = \mathsf{E}\{\boldsymbol{w}(k)\cdot\boldsymbol{v}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{w}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{v}(l)\} \end{split}$$

where δ_{kl} is the Kronecker delta function. The following equations defines this function: $\delta_{kl} = 1$ when k = l; $\delta_{kl} = 0$ when $k \neq l$.

w(k) is the process noise vector.

v(k) is the measurement noise vector.

 $\mathbf{Q}(k)$ is the covariance matrix of $\mathbf{w}(k)$.

 $\mathbf{R}(k)$ is the covariance matrix of $\mathbf{v}(k)$.

N(k) is the cross-covariance matrix between w(k) and v(k). If these noise vectors are uncorrelated, N(k) is a matrix of zeros.

E{} denotes the expected mean of the enclosed term(s).

This function also uses the **Second-Order Statistics Noise Model** to generate Gaussian-distributed temporally-uncorrelated random samples of $\mathbf{w}(k)$ and $\mathbf{v}(k)$ at each time step.

External Noise

This function adapts to changes in the **Stochastic State-Space Model**, as long as the model dimensions do not change. Therefore, you can use this function to simulate linear time-variant (LTV) stochastic state-space models.

You can use the CD Correlated Gaussian Random Noise VI to generate external random samples of Gaussian-distributed noise vectors. You can use this VI and the **Process Noise w(k)** and **Measurement Noise v(k)** inputs of the Discrete Stochastic State-Space function to simulate the following conditions:

- Deterministic systems excited by deterministic disturbance vectors *w*(*k*) and *v*(*k*).
- Stochastic systems excited by stochastic noise vectors w(k) and v(k). These stochastic noise vectors can be zero-mean or nonzero-mean, stationary or nonstationary, and can have any density distribution.
- Deterministic or stochastic systems excited by *w*(*k*) and *v*(*k*), where one of these vectors is a deterministic disturbance and the other is a stochastic noise.

Feedthrough Behavior

The value you specify for the **D** matrix of the **Stochastic State-Space Model** parameter determines the feedthrough behavior of this function.

- If the **D** matrix is nonzero, all input/output pairs have direct feedthrough behavior.
- If the **D** matrix is zero, the following input/output pairs have indirect feedthrough behavior.
 - Input u(k) Output y(k)
 - Input u(k) State x(k)

The remaining input/output pairs have direct feedthrough behavior.

When you use the configuration dialog box to configure the **D** matrix, LabVIEW verifies that the feedthrough behavior is correct. For example, if you set the **Feedthrough** parameter to **Indirect**, and you set the **D** matrix to nonzero, LabVIEW changes the **Feedthrough** parameter to **Direct**.

If you specify the value of the **D** matrix programmatically by wiring a value to the parameter terminal, LabVIEW does not adjust the feedthrough behavior for you. You must ensure that you specify the proper feedthrough behavior for the value of the **D** matrix that you specify.

CD Discrete Transfer Function Function

Owning Palette: Implementation VIs and Functions

Installed With: Control Design and Simulation Module

Implements a system model in discrete transfer function form. You define the system model by specifying the **Numerator** and **Denominator** of the transfer function equation.

Details

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

-	-]
Parameter	Description
Polymorphic instance	Specifies whether this function is single-input single- output (SISO) or multiple-input multiple-output (MIMO). The default value is SISO .
Feedthrough	Configures the function to be either a Direct or Indirect feedthrough function. Enable this control by selecting a parameter from the Parameters list and then selecting Terminal from the Parameter source pull-down menu. If you select Configuration Dialog Box from the Parameter source pull-down menu, LabVIEW disables this control and calculates the feedthrough behavior automatically.
	Refer to the <u>Details</u> section for a description of the parameters that determine the feedthrough behavior of this function.
Parameters	Lists all the parameters associated with this function. Select a parameter from this list to configure the parameter. When you select a parameter, the parameter and its associated Parameter source control appear in the Parameter Information section of the configuration dialog box.
Preview	Displays a graphical preview, if available, of the function output or configuration.
Parameter Information	Contains the parameters you can configure for this function. You must select a parameter from the Parameters list to make that parameter and its associated Parameter source control visible in the Parameter Information section of the configuration dialog box.
Parameter source	Specifies whether you configure this parameter using the Configuration Dialog Box or a Terminal on the simulation diagram. The default value is Configuration Dialog Box. If you select Terminal, LabVIEW displays an input for that parameter on the simulation diagram, and

 Transfer Function Specifies the transfer function in terms of numerator and denominator polynomial functions. Load Model—Loads model information from a data file. Save Model—Saves model information to a data file. This file is compatible with the <u>Control Design</u> VIs and functions. Copy to Clipboard—Copies the current model definition to the clipboard. From the clipboard, you can paste the model on the block diagram or into another configuration dialog box of the same model form. Paste from Clipboard—Pastes model information from the clipboard to the configuration dialog box. Model Dimensions—Use this section to specify the number of inputs and outputs of the system model. You also use this section to specify the input-output location of the equation you want to edit. This section is available only if you select MIMO from the Polymorphic instance pull-down menu. Inputs—Specifies the number of model inputs. Outputs—Specifies the current column of the Input-Output Model. Current Input—Specifies the current row of the Input-Output Model. Input-Output Model. Input-Output Model. 	you can wire values to that input to configure this function programmatically. If you select Configuration Dialog Box , LabVIEW removes that input from the simulation diagram. You then must set the value for this parameter inside the configuration dialog box.
	 Specifies the transfer function in terms of numerator and denominator polynomial functions. Load Model—Loads model information from a data file. Save Model—Saves model information to a data file. This file is compatible with the Control Design VIs and functions. Copy to Clipboard—Copies the current model definition to the clipboard. From the clipboard, you can paste the model on the block diagram or into another configuration dialog box of the same model form. Paste from Clipboard—Pastes model information from the clipboard to the configuration dialog box. Model Dimensions—Use this section to specify the number of inputs and outputs of the system model. You also use this section to specify the input-output location of the equation you want to edit. This section is available only if you select MIMO from the Polymorphic instance pull-down menu. Inputs—Specifies the number of model outputs. Current Input—Specifies the current column of the Input-Output Model. Current Output—Specifies the current row of the Input-Output Model. Input-Output Model. Input-Output Model. Input-Output Model. Input-Output Model. Input-Output Model.

	edit the equation at that input-output location. You also can use the Current Input and Current Output controls to specify the location of the equation you want to edit. Numerator —Specifies the coefficients of the numerator polynomial function in ascending powers of <i>z</i> , where <i>z</i> is the <i>z</i> -transform variable. For MIMO models, Numerator applies to the equation that the Current Input and Current Output parameters specify.
•	Denominator —Specifies the coefficients of the denominator polynomial function in ascending powers of <i>z</i> , where <i>z</i> is the <i>z</i> -transform variable. For MIMO models, Denominator applies to the equation that the Current Input and Current Output parameters specify. For each equation, the order of the Denominator must be greater than or equal to the order of the Numerator .

Block Diagram Inputs

Parameter	Description
initialize	Sets the model state(s) to 0, when TRUE.
input u(k)	Specifies the input to the system. input u(k) must be a vector of length <i>m</i> , where <i>m</i> is the number of inputs.
Transfer Function	Specifies a transfer function model. This input accepts either a block diagram constant or a model you created using Control Design VIs and functions.
error in	Describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out . If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out . Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

Block Diagram Outputs

Parameter	Description
-	Returns the current output of the system. This vector must be of length <i>r</i> , where <i>r</i> is the number of outputs.
	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Discrete Transfer Function Details

For SISO models, this function uses the following equation to calculate the output:

 $H(z) = \frac{b_0 + b_1 z + \dots + b_{m-1} z^{m-1} + b_m z^m}{a_0 + a_1 z + \dots + a_{n-1} z^{n-1} + a_n z^n}$

For MIMO models, this function calculates the output as $H = [H_{ij}]$.

where $b_0...b_m$ are the coefficients of the numerator polynomial

 $a_0 \dots a_n$ are the coefficients of the denominator polynomial

m is the order of the numerator

n is the order of the denominator

z is the z-transform variable

i is the index number of the input

j is the index number of the output

Feedthrough Behavior

The values you specify for the **Numerator** and **Denominator** subparameters determine the feedthrough behavior of this function. Given *m* as the order of the **Numerator** and *n* as the order of the **Denominator**:

- If n = m, all input/output pairs have direct feedthrough behavior.
- If n > m, the input u(k) input has indirect feedthrough to the output y(k) output.
- If *n* < *m*, LabVIEW returns an error.

When you use the configuration dialog box to configure *n* and *m*, LabVIEW verifies that the feedthrough behavior is correct. For example, if you set the **Execution Mode** parameter to **Indirect**, and you set *n* equal to *m*, LabVIEW changes the **Execution Mode** parameter to **Direct**.

If you define the transfer function programmatically, LabVIEW does not adjust the feedthrough behavior for you. You must ensure that you specify the proper feedthrough behavior for the orders of *n* and *m* you specify.

CD Discrete Zero-Pole-Gain Function

Owning Palette: Implementation VIs and Functions

Installed With: Control Design and Simulation Module

Implements a system model in discrete zero-pole-gain form. You define the system model by specifying the **Zeros**, **Poles**, and **Gain** of the zero-pole-gain equation.

Details

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

-	-]
Parameter	Description
Polymorphic instance	Specifies whether this function is single-input single- output (SISO) or multiple-input multiple-output (MIMO). The default value is SISO .
Feedthrough	Configures the function to be either a Direct or Indirect feedthrough function. Enable this control by selecting a parameter from the Parameters list and then selecting Terminal from the Parameter source pull-down menu. If you select Configuration Dialog Box from the Parameter source pull-down menu, LabVIEW disables this control and calculates the feedthrough behavior automatically.
	Refer to the <u>Details</u> section for a description of the parameters that determine the feedthrough behavior of this function.
Parameters	Lists all the parameters associated with this function. Select a parameter from this list to configure the parameter. When you select a parameter, the parameter and its associated Parameter source control appear in the Parameter Information section of the configuration dialog box.
Preview	Displays a graphical preview, if available, of the function output or configuration.
Parameter Information	Contains the parameters you can configure for this function. You must select a parameter from the Parameters list to make that parameter and its associated Parameter source control visible in the Parameter Information section of the configuration dialog box.
Parameter source	Specifies whether you configure this parameter using the Configuration Dialog Box or a Terminal on the simulation diagram. The default value is Configuration Dialog Box. If you select Terminal, LabVIEW displays an input for that parameter on the simulation diagram, and

	you can wire values to that input to configure this function programmatically. If you select Configuration Dialog Box , LabVIEW removes that input from the simulation diagram. You then must set the value for this parameter inside the configuration dialog box.
Zeros-Poles-	Specifies the zero-pole-gain model.
Gain	 Load Model—Loads model information from a data file.
	 Save Model—Saves model information to a data file. This file is compatible with the <u>Control Design</u> VIs and functions.
	• Copy to Clipboard —Copies the current model definition to the clipboard. From the clipboard, you can paste the model on the block diagram or into another configuration dialog box of the same model form.
	 Paste from Clipboard—Pastes model information from the clipboard to the configuration dialog box.
	• Model Dimensions—Use this section to specify the number of inputs and outputs of the system model. You also use this section to specify the input-output location of the equation you want to edit. This section is available only if you select MIMO from the Polymorphic instance pull-down menu.
	 Inputs—Specifies the number of model inputs. Outputs—Specifies the number of model
	outputs.
	 Current Input—Specifies the current column of the Input-Output Model.
	 Current Output—Specifies the current row of the Input-Output Model.
	• Input-Output Model —Displays a graphical representation of the MIMO model. Click a cell to edit the equation at that input-output location. You

	 also can use the Current Input and Current Output controls to specify the location of the equation you want to edit. Gain—Specifies a real value representing the common gain of the zero-pole-gain model. For MIMO models, Gain applies to the equation that the Current Input and Current Output parameters specify. Zeros—Specifies zeros of the zero-pole-gain model. For all zeros with an imaginary part, the conjugate complex number also must belong to this array. For MIMO models, Zeros applies to the equation that the Current Input and Current Output parameters specify. Poles—Specifies the poles of the zero-pole-gain model. For all poles with an imaginary part, the conjugate complex number also must belong to this array. For MIMO models, Zeros applies to the equation that the Current Input and Current Output parameters specify. Poles—Specifies the poles of the zero-pole-gain model. For all poles with an imaginary part, the conjugate complex number also must belong to this array. For MIMO models, Poles applies to the equation that the Current Input and Current Output parameters specify.
--	---

Block Diagram Inputs

Parameter	Description
initialize	Sets the model state(s) to 0, when TRUE.
input u(k)	Specifies the input to the system. input u(k) must be a vector of length <i>m</i> , where <i>m</i> is the number of inputs.
Zeros- Poles- Gain	Specifies a zero-pole-gain model. This input accepts either a block diagram constant or a model you created using the Control Design VIs and functions.
error in	Describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out . If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out . Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

Block Diagram Outputs

Parameter	Description
-	Returns the current output of the system. This vector must be of length <i>r</i> , where <i>r</i> is the number of outputs.
	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Discrete Zero-Pole-Gain Details

For SISO models, this function uses the following equation to calculate the output:

 $\mathsf{H}(z) \; = \; \frac{k(z\!-\!Z_1)(z\!-\!Z_2)...(z\!-\!Z_m)}{(z\!-\!P_1)(z\!-\!P_2)...(z\!-\!P_n)}$

For MIMO models, this function calculates the output as $H = [H_{ij}]$.

where k is the gain

Z[*m*] is the array of zeros

P[n] is the array of poles

m is the order of the numerator

n is the order of the denominator

z is the z-transform variable

i is the index number of the input

j is the index number of the output

The function is based on the <u>CD Discrete Transfer Function</u> function, which is represented in zero-pole-gain notation.

Feedthrough Behavior

The values you specify for the **Zeros** and **Poles** subparameters determine the feedthrough behavior of this function. Given *Z* as the **Zeros** subparameter and *P* as the **Poles** subparameter:

- If the order of *Z* = the order of *P*, the function has direct feedthrough behavior.
- If the order of *Z* < the order of *P*, the **input u(k)** input has indirect feedthrough to the **output y(k)** output. All other input/output pairs have direct feedthrough behavior.
- If the order of Z > the order of P, LabVIEW returns an error.

When you use the configuration dialog box to configure *Z* and *P*, LabVIEW verifies that the feedthrough behavior is correct. For example, if you set the **Execution Mode** parameter to **Indirect**, and you set the order of *Z* equal to the order of *P*, LabVIEW changes the **Execution Mode** parameter to **Direct**.

If you define the zero-pole-gain equation programmatically, LabVIEW does not adjust the feedthrough behavior for you. You must ensure that you specify the proper feedthrough behavior for the orders of *Z* and *P* you specify.
CD State Feedback Controller VI

Owning Palette: Implementation VIs and Functions

Installed With: Control Design and Simulation Module

Implements a state-space controller where the **Controller Action** equals –**Controller Gain** * **States**.

▼

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD State Feedback Controller (Single Output)



States specifies the initial states of the controller.

- **Controller Gain** specifies the gains that premultiply the **States** to calculate the manipulated input to drive the system towards or back to a reference.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Controller Action** returns the current value of the control action.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD State Feedback Controller (Multiple Outputs)



Initialize specifies whether to restart the calculation from any initial values you provide. The default is FALSE.

- **States** specifies the initial states of the controller.
- **Controller Gain** specifies the gains that premultiply the **States** to calculate the manipulated inputs to drive the system towards or back to a reference.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Controller Action** returns the current values of the control action.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Model Construction VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Model Construction VIs to create linear system models and modify the properties of a system model. You also can use the Model Construction VIs to save a system model to a file, read a system model from a file, or obtain a visual representation of a model.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD</u> <u>Construct</u> <u>Filter</u> <u>Model</u>	Constructs a digital filter model. You must <u>manually select the</u> polymorphic instance to use.
	Constructs a phase-lead or a phase-lag controller model in transfer function form. You must <u>manually select the</u> polymorphic instance to use.
CD Construct PID Model	Constructs a PID model in transfer function form. You must manually select the polymorphic instance to use.
<u>CD</u> <u>Construct</u> <u>Random</u> <u>Model</u>	Creates a random model in state-space, transfer function, or zero-pole-gain representation. You can specify the sampling time, location of uncontrollable or unobservable states (for state-space), and location of poles and zeros in the different regions of the complex plane. You must <u>manually select the</u> <u>polymorphic instance</u> to use.
<u>CD</u> <u>Construct</u> <u>Special</u> TF Model	Creates commonly used transfer function models. You must manually select the polymorphic instance to use.

<u>CD</u> <u>Construct</u> State-	Creates a <u>deterministic state-space representation</u> of a system using the matrices A , B , C , and D , and the Sampling Time (s) You must <u>manually select the polymorphic instance</u>
Space Model	to use.
<u>CD</u> Construct Transfer Function Model	Creates a <u>transfer function representation</u> of a system using the Sampling Time (s) , Numerator , Denominator , and Delay . This VI also produces a transfer function model which specifies the data in symbolic form. You must <u>manually select</u> <u>the polymorphic instance</u> to use.
<u>CD</u> <u>Construct</u> <u>Zero-</u> <u>Pole-</u> <u>Gain</u> <u>Model</u>	Creates a <u>zero-pole-gain representation</u> of a system using the Zeros , Poles , Gain , Delay , and Sampling Time (s) . This VI also produces a zero-pole-gain model which specifies the data in symbolic form. You must <u>manually select the</u> <u>polymorphic instance</u> to use.
<u>CD Draw</u> <u>State-</u> <u>Space</u> Equation	Displays the state-space equation of the State-Space Model .
<u>CD Draw</u> <u>Transfer</u> <u>Function</u> Equation	Displays the transfer function equation of the model. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Draw</u> <u>Zero-</u> <u>Pole-</u> <u>Gain</u> Equation	Displays the zero-pole-gain equation of the model. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Read</u> <u>Model</u> from File	Opens a file the <u>Write Model to File</u> VI created and reads all the records in the file. Each record contains a separate model. To retrieve all records in the file, all the models must be in the same model representation. You must <u>manually</u> <u>select the polymorphic instance</u> to use.
<u>CD Write</u> <u>Model to</u>	Creates a new file or appends to an existing file, writes the specified number of records to the file, then closes the file

<u>File</u>	and checks for errors. Each record is a model. The data type
	you wire to the State-Space Model input determines the
	polymorphic instance to use.

CD Construct Filter Model VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Constructs a digital filter model. You must <u>manually select the</u> <u>polymorphic instance</u> to use.

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

•

CD Construct Filter Model (Coefficients)



Frequency (Hz) specifies the frequency, in hertz, of the digital filter this VI creates. The default value is 1 Hz.

- Reverse Coefficients (Denominator) specifies the reverse coefficients of the digital filter this VI creates. The reverse coefficients also are the coefficients of the denominator polynomial function of the Transfer Function Model.
- **Forward Coefficients (Numerator)** specifies the forward coefficients of the digital filter this VI creates. The forward coefficients also are the coefficients of the numerator polynomial function of the **Transfer Function Model**.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **Transfer Function Model** returns the transfer function this VI constructs based on the inputs. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Filter Model (Specifications)



- **Specification** specifies the passband ripple and stopband attenuation of the filter.
 - **passband ripple** specifies the passband ripple, in decibels. The default value is 0.1 dB.
 - stopband attenuation specifies the stopband attenuation, in decibels. This parameter is valid only if you specify
 Elliptic for the Coefficients Type parameter. The default value of stopband attenuation is 60 dB.
- **Sampling Frequency** specifies the sampling frequency, in hertz, of the filter. The value of this parameter must be greater 0, or LabVIEW returns an error. The default value is 10 Hz.
- **Coefficients Type** specifies the type of coefficients this VI generates to create the filter. You also can specify the **Order** of these coefficients.

0	Butterworth Chebyshev	
1		
2	Elliptic (default)	

Filter Type specifies the type of filter this VI creates.

	0	Lowpass	
	1 Highpass		
2 Bandpass			
		Bandstop (default)	
	3	Danusiop (delauit)	

Order specifies the order of the coefficients this VI generates to create the filter. The default value is 2.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- High Cutoff Frequency specifies the upper cutoff frequency, in hertz, of the filter. The default value is 2 Hz. This parameter is valid only if you specify Bandpass or Bandstop for the Filter Type parameter. The value of the High Cutoff Frequency must be greater than the value of the Low Cutoff Frequency and observe the Nyquist criterion.
- Low Cutoff Frequency specifies the lower cutoff frequency, in hertz, of the filter. The default value is 0.01 Hz. This parameter is valid only if you specify Bandpass or Bandstop for the Filter Type parameter. The value of the Low Cutoff Frequency must be less than the value of the High Cutoff Frequency and observe the Nyquist criterion.
 - $\overline{\mathbb{N}}$
 - Note If the Low Cutoff Frequency is less than or equal to 0 or is greater than half the Sampling Frequency, this VI returns an error.

- **Zero-Pole-Gain Model** returns the <u>zero-pole-gain form</u> of the digital filter this VI creates.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Lead-Lag Controller VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Constructs a phase-lead or a phase-lag controller model in transfer function form. You must <u>manually select the polymorphic instance</u> to use.

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

•

■ Place on the block diagram ■ Find on the **Functions** palette

CD Construct Phase-Lag Controller



Unit Gain Normalization specifies the type of lead or lag controller model you want to construct.



- **Gain (K)** specifies the gain of the controller model equation.
- **Parameter (tau)** specifies the tau parameter of the controller model equation.
- Beta specifies the beta parameter of the phase-lag controller model equation. The value of this parameter must be greater than 1. If you enter a value less than 1, LabVIEW coerces the value to be 1.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Transfer Function Model** returns the <u>mathematical representation</u> of the phase-lag or phase-lead controller.
- wn returns the frequency, in rad/s, at which the Max Phase of the Transfer Function Model occurs.
- Max Phase returns the maximum phase, in degrees, of the Transfer Function Model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Phase-Lead Controller



Unit Gain Normalization specifies the type of lead or lag controller model you want to construct.

	Below 0 dB
1	Above 0 dB

- **Gain (K)** specifies the gain of the controller model equation.
- **Parameter (tau)** specifies the tau parameter of the controller model equation.
- Alpha specifies the alpha parameter of the phase-lead controller model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Transfer Function Model** returns the <u>mathematical representation</u> of the phase-lag or phase-lead controller.
- wn returns the frequency, in rad/s, at which the Max Phase of the Transfer Function Model occurs.
- Max Phase returns the maximum phase, in degrees, of the Transfer Function Model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Lead-Lag Controller Details

This VI uses the following equations to calculate the outputs:

	Phase-Lag Controller	Phase-Lead Controller
Transfer Function Model, if Unit Gain Normalization is Below 0dB	$\boldsymbol{K} \cdot \boldsymbol{\beta} \cdot \left(\frac{\tau s + 1}{\beta \tau s + 1}\right) \boldsymbol{\beta} \ge 1$	$\boldsymbol{K} \cdot \left(\frac{\tau s + 1}{\alpha \tau s + 1}\right) 0 \le \alpha \le 1$
Transfer Function Model, if Unit Gain Normalization is Above 0dB	$\boldsymbol{K} \cdot \left(\frac{\tau s + 1}{\beta \tau s + 1}\right) \beta \ge 1$	$\boldsymbol{K} \cdot \boldsymbol{\alpha} \cdot \left(\frac{\tau s + 1}{\alpha \tau s + 1}\right) 0 \le \alpha \le 1$
wn	$\frac{1}{\tau\sqrt{\beta}}$	$\frac{1}{\tau \sqrt{\alpha}}$
Max Phase	$\frac{180}{\pi} \cdot \operatorname{asin}\left(\frac{1-\beta}{1+\beta}\right)$	$\frac{180}{\pi} \cdot \operatorname{asin}\left(\frac{1-\alpha}{1+\alpha}\right)$

CD Construct PID Model VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Constructs a PID model in transfer function form. You must <u>manually</u> <u>select the polymorphic instance</u> to use.

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

•

■ Place on the block diagram ■ Find on the **Functions** palette

CD Construct PID Model (Academic)



- **Proportional Gain (Kc)** represents the proportional gain of the controller. In the equation that defines the PID Academic form, K_c represents the proportional gain. The default is 1.
- **Integral Time [s] (Ti)** is the controller parameter that adjusts the effect of the error integral term E(s)/s on the controller output U(s). In the equation that defines the PID Academic form, T_i represents the integral time. The default is 0.
- **Derivative Time [s] (Td)** is the controller parameter that adjusts the effect of the error derivative term sE(s) on the controller output U(s). In the equation that defines the PID Academic form, T_d represents the derivative time. The default is 0. The default PID Academic controller does not use derivative time.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **High Frequency Time Constant [s] (Tf)** is the low pass filter time constant T_f this VI uses to make the PID model a proper system. The default is 0.
- **Transfer Function Model** returns the transfer function this VI constructs based on the inputs. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct PID Model (Parallel, Continuous)



- **Proportional Gain (Kp)** represents the proportional gain of the controller. In the equation that defines the PID Parallel form, K_P represents the proportional gain. The default is 1.
- **Integral Gain (Ki)** is the controller parameter that adjusts the effect of the error integral term E(s)/s on the controller output U(s). In the equation that defines the PID Parallel form, K_i represents the integral gain. The default is 0.
- **Derivative Gain (Kd)** is the controller parameter that adjusts the effect of the error derivative term sE(s) on the controller output U(s). In the equation that defines the PID Parallel form, K_d represents the derivative gain. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- High Frequency Time Constant [s] (Tf) is the low pass filter time constant T_f this VI uses to make the PID model a proper system. The default is 0.
- Transfer Function Model returns the transfer function this VI constructs based on the inputs. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct PID Model (Parallel, Discrete)

 $\overline{\mathbb{N}}$

Note This instance uses <u>bilinear-transformation integration</u> to construct the PID model.



- **Sampling Time** specifies the sampling time of the discrete PID model. The default is 1.
- **Proportional Gain (Kp)** represents the proportional gain of the controller. In the equation that defines the PID Parallel form, K_P represents the proportional gain. The default is 1.
- **Integral Gain (Ki)** is the controller parameter that adjusts the effect of the error integral term E(s)/s on the controller output U(s). In the equation that defines the PID Parallel form, K_i represents the integral gain. The default is 0.
- **Derivative Gain (Kd)** is the controller parameter that adjusts the effect of the error derivative term sE(s) on the controller output U(s). In the equation that defines the PID Parallel form, K_d represents the derivative gain. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple</u> <u>Error Handler</u> or <u>General Error Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is an <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Transfer Function Model returns the transfer function this VI constructs based on the inputs. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct PID Model (Series)



- **Proportional Gain (KC)** represents the proportional gain of the controller. In the equation that defines the PID Series form, K_C represents the proportional gain. The default is 1.
- **Integral Time [s] (TI)** is the controller parameter that adjusts the effect of the error integral term E(s)/s on the controller output U(s). In the equation that defines the PID Series form, T_1 represents the integral time. The default is 0.
- **Derivative Time [s] (TD)** is the controller parameter that adjusts the effect of the error derivative term sE(s) on the controller output U(s). In the equation that defines the PID Series form, T_D represents the derivative time. The default is 0. The default PID Series controller does not use derivative time.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **High Frequency Time Constant [s] (Tf)** is the low pass filter time constant T_f this VI uses to make the PID model a proper system. The default is 0.
- **Transfer Function Model** returns the transfer function this VI constructs based on the inputs. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct PID Model Details

This VI constructs transfer function models in the following forms:

PID Academic

$$\frac{U(s)}{E(s)} = \mathcal{K}_{c} (1 + \frac{1}{T_{i}s} + \frac{T_{d}s}{T_{f}s + 1})$$

PID Parallel

$$\frac{U(s)}{E(s)} = \mathcal{K}_p + \frac{\mathcal{K}_i}{s} + \frac{\mathcal{K}_d s}{T_f s + 1}$$

PID Parallel Discrete

$$D(z) = \frac{\left(2K_{p}T + K_{j}T^{2} + 2K_{d}\right)z^{2} + (K_{j}T^{2} - 2K_{p}T - 4K_{d})z + 2K_{d}}{2Tz(z-1)}$$

Refer to the book <u>Digital Control Systems</u> for more information about bilinear-transformation integration.

PID Series

$$\frac{U(s)}{E(s)} = \mathcal{K}_C(1 + \frac{1}{T_I s}) + (\frac{T_D s + 1}{T_f s + 1})$$

CD Construct Random Model VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Creates a random model in state-space, transfer function, or zero-polegain representation. You can specify the sampling time, location of uncontrollable or unobservable states (for state-space), and location of poles and zeros in the different regions of the complex plane. You must manually select the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Construct Random Model (State-Space)



- If **Strictly Proper?** is TRUE, the model is strictly proper. For a state-space model to be strictly proper, the **D** matrix must be a null matrix. For a transfer function or a zero-pole-gain model to be strictly proper, the order of the polynomial of the numerator must be less than the order of the polynomial of the denominator. The default is TRUE.
- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Number of Inputs** specifies the number of inputs for the random system this VI creates. The default is 1.
- Number of Outputs specifies the number of outputs for the random system this VI creates. The default is 1.
- System Order specifies the order (dimension of matrix A) of the multiple-input multiple-output (MIMO) system this VI uses for the random model generation. The default is 2. LabVIEW does not use this input if you wire values to the **Poles** input. By default, the poles that the **System Order** specifies are stable.
- error in describes error conditions that occur before this VI or

function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Element Range** defines the absolute value of the range of the integer coefficients this VI uses to define the model. The default is 10.
- **Poles** determines the number and location in the complex plane of randomly generated poles.
 - **Non Random Poles** specifies information about the controllability and observability of non-random poles.
 - **Controllable and Observable Modes** specifies the location of the controllable and observable poles of the system model.
 - **Uncontrollable Modes** specifies the location of the uncontrollable poles of the system model.
 - **Unobservable Modes** specifies the location of the unobservable poles of the system model.

[CDB]

Uncontrollable and Unobservable Modes specifies the location of uncontrollable and unobservable poles of the system model.

- **Random Real** specifies the location of the stable, unstable, and marginally stable random real poles.
 - **Real Stable** specifies the number of stable random real poles.
 - **Real Unstable** specifies the number of unstable random real poles.
 - **Real Marginally Stable** specifies the number of marginally stable random real poles.
- Random Complex Pairs specifies the location of the stable, unstable, and marginally stable random complex pair poles.
 - **Complex Stable** specifies the number of stable random complex pair poles.
 - **Complex Unstable** specifies the number of unstable random complex pair poles.
 - **Complex Marginally Stable** specifies the number of marginally stable random complex pair poles.
- **Random Model** is the random model that this VI creates. To access and modify the data in the model, use the <u>Model</u> <u>Information</u> VIs.
- Minimum Realization? indicates if the random model is a minimum realization. The model must be controllable and observable to be a minimum realization.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Random Model (Transfer Function)



- **Zeros** determines the number and location in the complex plane of randomly generated zeros.
 - **Random Number of Zeros?** determines if this VI generates the number and location of the zeros randomly. If **Random Number of Zeros?** is TRUE, the random system is strictly proper.
 - **Non Random Zeros** determines the location of non-random zeros.
 - Random Real specifies the location of the minimum phase, non-minimum phase, and marginally minimum phase random real zeros.
 - **Real Minimum Phase** specifies the number of minimum phase random real zeros.
 - **Real Non Minimum Phase** specifies the number of non-minimum phase random real zeros.
 - Real Marginally Minimum Phase specifies the number of marginally minimum phase random real zeros.
 - Random Complex Pairs specifies the location of the minimum phase, non-minimum phase, and marginally minimum phase random complex pair zeros.
 - **Complex Minimum Phase** specifies the number of minimum phase random complex pair zeros.
 - Complex Non Minimum Phase specifies the number of non-minimum phase random complex pair zeros.
 - **Complex Marginally Minimum Phase** specifies the
number of marginally minimum phase random complex pair zeros.

- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Number of Inputs** specifies the number of inputs for the random system this VI creates. The default is 1.
- **Number of Outputs** specifies the number of outputs for the random system this VI creates. The default is 1.
- System Order specifies the number of poles for each element of the transfer function or zero-pole-gain system this VI uses for the random model generation. The default is 2. LabVIEW does not use this input if you wire values to the **Poles** input. By default, the poles **System Order** specifies are stable.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI or

function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Element Range** defines the absolute value of the range of the integer coefficients this VI uses to define the model. The default is 10.
- **Poles** determines the number and location in the complex plane of randomly generated poles.
 - **Non Random Poles** specifies the location of the non-random poles.
 - **Random Real** specifies the location of the stable, unstable, and marginally stable random real poles.
 - **Real Stable** specifies the number of stable random real poles.
 - **Real Unstable** specifies the number of unstable random real poles.
 - **Real Marginally Stable** specifies the number of marginally stable random real poles.
 - Random Complex Pairs specifies the location of the stable, unstable, and marginally stable random complex pair poles.
 - **Complex Stable** specifies the number of stable random complex pair poles.
 - **Complex Unstable** specifies the number of unstable random complex pair poles.
 - **Complex Marginally Stable** specifies the number of marginally stable random complex pair poles.

Random Model is the random model that this VI creates. To access and modify the data in the model, use the <u>Model</u> <u>Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Random Model (Zero-Pole-Gain)



- **Zeros** determines the number and location in the complex plane of randomly generated zeros.
 - **Random Number of Zeros?** determines if this VI generates the number and location of the zeros randomly. If **Random Number of Zeros?** is TRUE, the random system is strictly proper.
 - **Non Random Zeros** determines the location of non-random zeros.
 - Random Real specifies the location of the minimum phase, non-minimum phase, and marginally minimum phase random real zeros.
 - **Real Minimum Phase** specifies the number of minimum phase random real zeros.
 - **Real Non Minimum Phase** specifies the number of non-minimum phase random real zeros.
 - Real Marginally Minimum Phase specifies the number of marginally minimum phase random real zeros.
 - Random Complex Pairs specifies the location of the minimum phase, non-minimum phase, and marginally minimum phase random complex pair zeros.
 - **Complex Minimum Phase** specifies the number of minimum phase random complex pair zeros.
 - Complex Non Minimum Phase specifies the number of non-minimum phase random complex pair zeros.
 - **Complex Marginally Minimum Phase** specifies the

number of marginally minimum phase random complex pair zeros.

- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Number of Inputs** specifies the number of inputs for the random system this VI creates. The default is 1.
- **Number of Outputs** specifies the number of outputs for the random system this VI creates. The default is 1.
- System Order specifies the number of poles for each element of the transfer function or zero-pole-gain system this VI uses for the random model generation. The default is 2. LabVIEW does not use this input if you wire values to the **Poles** input. By default, the poles **System Order** specifies are stable.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI or

function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Element Range** defines the absolute value of the range of the integer coefficients this VI uses to define the model. The default is 10.
- **Poles** determines the number and location in the complex plane of randomly generated poles.
 - **Non Random Poles** specifies the location of the non-random poles.
 - **Random Real** specifies the location of the stable, unstable, and marginally stable random real poles.
 - **Real Stable** specifies the number of stable random real poles.
 - **Real Unstable** specifies the number of unstable random real poles.
 - **Real Marginally Stable** specifies the number of marginally stable random real poles.
 - Random Complex Pairs specifies the location of the stable, unstable, and marginally stable random complex pair poles.
 - **Complex Stable** specifies the number of stable random complex pair poles.
 - **Complex Unstable** specifies the number of unstable random complex pair poles.
 - **Complex Marginally Stable** specifies the number of marginally stable random complex pair poles.

Random Model is the random model that this VI creates. To access and modify the data in the model, use the <u>Model</u> <u>Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Random Model Details

This VI supports delays. To represent a delay in the model, you must specify the delay in this VI. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Construct Special TF Model VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Creates commonly used transfer function models. You must <u>manually</u> <u>select the polymorphic instance</u> to use.

The first order instance generates a transfer function with the following equation:

▼

 $H(s) = \frac{K}{\varpi + 1} e^{-sT}$

where K is the gain and t is the time constant.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

CD Construct Special TF Model (1st Order)



Static Gain specifies the DC gain of the system, which is the gain when the frequency is 0. The default value is 1.

- **Time Constant (s)** specifies the system time constant, or the time required for the output to reach 63% of its final response. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay** is the delay this VI uses to construct the special model. The default is NaN. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

Transfer Function Model returns the transfer function this VI constructs based on the inputs. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Special TF Model (2nd Order)

The second order instance generates a transfer function with the following equation:

$$H(s) = \frac{\omega_D^2}{s^2 + 2\zeta\omega_D s + \omega_D^2} e^{-sT}$$

where w_n is the natural frequency and z is the damping ratio.



- **Static Gain** specifies the DC gain of the system, which is the gain when the frequency is 0. The default value is 1.
- **Damping Ratio** specifies the damping ratio, which reflects the damping level as a fraction of the critical damping value where the poles become real, of the second-order transfer function model. The default is zero.
- **Natural Frequency (rad/s)** specifies the natural frequency, in radians/seconds, of the second-order transfer function model. The default is zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay** is the delay this VI uses to construct the special model. The default is NaN. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.
- Transfer Function Model returns the transfer function this VI constructs based on the inputs. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Special TF Model (Delay Pade Approximation)

Delay refers to a continuous time value in which the Pade polynomial approximates behavior.



- **Transport Delay** specifies the transport time delay that you want to use to create a transfer function using Pade approximation. The default is 0.
- **Polynomial Order** specifies the order of this polynomial approximation. The default is 3.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Pade Approximation Model** returns the transfer function model

with delays this VI incorporates. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Special TF Model Details

This VI supports delays. To represent a delay in the model, you must specify the delay in this VI. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Construct State-Space Model VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Creates a <u>deterministic state-space representation</u> of a system using the matrices **A**, **B**, **C**, and **D**, and the **Sampling Time (s)** You must <u>manually</u> <u>select the polymorphic instance</u> to use.

This VI constructs a deterministic state-space model. You can use the <u>CD Construct Stochastic Model</u> VI to construct a stochastic state-space model.



Note This VI supports specific combinations of the system matrices A, B, C, and D. Refer to Required Matrices input of the <u>CD Verify MIMO Properties</u> VI for a complete list of the combinations.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an ii	nstance
--------------	---------

■ Place on the block diagram ■ Find on the **Functions** palette

CD Construct State-Space Model (Numeric)



- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- A specifies the system matrix that describes the dynamics of the states of the system.
- **B** specifies the input matrix that relates the inputs to the states.
- **C** specifies the output matrix that relates the outputs to the states.
- **D** specifies the transmission matrix that relates the inputs to the outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- State-Space Model returns the created numeric model. The data consists of the arrays A, B, C, and D. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct State-Space Model (Symbolic)



- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Symbolic A** is the <u>symbolic representation</u> of the system matrix that describes the dynamics of the states of the system.
- **Symbolic B** is the symbolic representation of the input matrix of the system that relates the inputs to the states.
- **Symbolic C** is the symbolic representation of the output matrix that relates the outputs to the states.
- **Symbolic D** is the symbolic representation of the transmission matrix that relates the inputs to the outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to

treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Variables** contains the name and value of each variable.
 - **Name** is a variable name this VI uses to define the data of the system model. Variable names can be a combination of letters and numbers. A variable name that begins with a capital letter E can produce unpredictable errors if parts of the original string represent numbers like 1E–2. Avoid terms beginning with E in such cases.
 - **Value** is the numeric value this VI associates with the variable. The VI uses this value to evaluate the model.
- State-Space Model returns the created model. The data consists of the arrays Symbolic A, Symbolic B, Symbolic C, and Symbolic D in which this VI evaluates string elements using data of the Variables array. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct State-Space Model Details

This VI does not support delays directly. To add a delay to the model, use the <u>CD Set Delays to Model</u> VI. Refer to the <u>LabVIEW Control Design</u> <u>User Manual</u> for more information about delays.

CD Construct Transfer Function Model VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Creates a <u>transfer function representation</u> of a system using the **Sampling Time (s)**, **Numerator**, **Denominator**, and **Delay**. This VI also produces a transfer function model which specifies the data in symbolic form. You must <u>manually select the polymorphic instance</u> to use.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

•

■ Place on the block diagram ■ Find on the **Functions** palette

CD Construct Transfer Function Model (SISO)



- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Numerator** contains the constant coefficients, in ascending order, of a polynomial that represents the numerator of a SISO transfer function. The coefficients take the following form:

 $b_0 + b_1 s + \dots + b_m s^m$.

- **Denominator** contains the constant coefficients, in ascending order, of a polynomial that represents the denominator of a SISO transfer function. The coefficients take the following form: $a_0 + a_1 s + \dots + a_n s^n$.
- **Delay** is the transport time delay that might exist in the system. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Transfer Function Model** is the system model this VI creates. When the sampling time is zero (for continuous-time), the **Numerator** and **Denominator** collectively represent the mathematical model (in Laplace transformation) of a dynamic system H(s) that provides the relationship between the input U(s)and output Y(s) of the system. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Transfer Function Model (SISO Symbolic)



- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- Symbolic Numerator is the symbolic representation of the constant coefficients of a polynomial that represent the numerator of a SISO transfer function. The *i*th element of Symbolic Numerator is the coefficient of the *i*th order term of the polynomial. The index is zero-based.

Symbolic Denominator is the <u>symbolic representation</u> of the constant coefficients of a polynomial that represent the denominator of a SISO transfer function. The *i*th element of **Symbolic Denominator** is the coefficient of the *i*th order term of the polynomial. The index is zero-based.

- Symbolic Delay is the <u>symbolic representation</u> of the transport time delay that may exist in the system. Refer to the <u>LabVIEW</u> <u>Control Design User Manual</u> for more information about delays.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error

occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Variables** contains the name and value of each variable.
 - **Name** is a variable name this VI uses to define the data of the system model. Variable names can be a combination of letters and numbers. A variable name that begins with a capital letter E can produce unpredictable errors if parts of the original string represent numbers like 1E–2. Avoid terms beginning with E in such cases.
 - **Value** is the numeric value this VI associates with the variable. The VI uses this value to evaluate the model.
- **Transfer Function Model** is the numeric model that this VI evaluates using data from **Variables**. This VI converts empty array elements into numeric zeros. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Transfer Function Model (MIMO)

Sampling Time (s) Transfer Function(s) error in (no error)

- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Transfer Function(s)** is an array of transfer function models that you can use to define the dynamics of a multiple-input single-output (MISO), a single-input multiple-output (SIMO), or a multiple-input multiple-output (MIMO) system.
 - **Numerator** contains the constant coefficients of a polynomial that represent the numerator of a transfer function.
 - **Denominator** contains the constant coefficients of a polynomial that represent the denominator of a transfer function.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to

treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Transfer Function Model is the system model this VI creates. The size of the Transfer Function(s) array determines if the system is SISO, MIMO, MISO, or SIMO. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Transfer Function Model (MIMO Symbolic)



- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- Symbolic Transfer Function(s) is the symbolic representation of the transfer functions that you can use to define the dynamics of a single-input single-output (SISO), multiple-input single-output (MISO), single-input multiple-output (SIMO), or multiple-input multiple-output (MIMO) system.
 - **Numerator** contains the constant coefficients of a polynomial that represent the numerator of a transfer function.
 - **Denominator** contains the constant coefficients of a polynomial that represent the denominator of a transfer function.
 - **delay** is the symbolic representation of the transport time delay that may exist in the system. Refer to the <u>LabVIEW</u> <u>Control Design User Manual</u> for more information about delays.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value

to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Variables** contains the name and value of each variable.
 - **Name** is a variable name this VI uses to define the data of the system model. Variable names can be a combination of letters and numbers. A variable name that begins with a capital letter E can produce unpredictable errors if parts of the original string represent numbers like 1E–2. Avoid terms beginning with E in such cases.
 - **Value** is the numeric value this VI associates with the variable. The VI uses this value to evaluate the model.
- **Transfer Function Model** is the system model this VI creates. The size of the **Transfer Function(s)** array determines if the system is SISO, MIMO, MISO, or SIMO. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Transfer Function Model Details

Only the <u>SISO</u> and <u>SISO Symbolic</u> instances of this VI support delays. To represent a delay in the model, you must specify the delay in this VI. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Construct Zero-Pole-Gain Model VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Creates a <u>zero-pole-gain representation</u> of a system using the **Zeros**, **Poles**, **Gain**, **Delay**, and **Sampling Time (s)**. This VI also produces a zero-pole-gain model which specifies the data in symbolic form. You must <u>manually select the polymorphic instance</u> to use.

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

•

■ Place on the block diagram ■ Find on the **Functions** palette
CD Construct Zero-Pole-Gain Model (SISO)



- **Complete Complex Conjugate?** specifies, when TRUE, to calculate the complex conjugate for any zero or pole that has only a real or pure complex root specified. The default is FALSE.
- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Zeros** is the array of zeros of the SISO system. The zeros can be real or complex. If they are complex, they must be in complex conjugate pairs.
- **Poles** is the array of poles of the SISO system. The poles can be real or complex. If they are complex, they must be in complex conjugate pairs.
- **Gain** is the scalar gain of the SISO system. The default is NaN.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay** is the transport time delay that may exist in the system. The default is NaN. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.
- **Zero-Pole-Gain Model** is the system model this VI creates. To access and modify the data in the model, use the <u>Model</u> <u>Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Zero-Pole-Gain Model (SISO Symbolic)



- **Complete Complex Conjugate?** specifies, when TRUE, to calculate the complex conjugate for any zero or pole that has only a real or pure complex root specified. The default is FALSE.
- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Symbolic Zeros** is the array of zeros of the system. The zeros can be real or complex. If they are complex, they must be in complex conjugate pairs.
- Symbolic Poles is the array of poles of the system. The poles can be real or complex. If they are complex, they must be in complex conjugate pairs.
- Symbolic Gain is the scalar gain of the system. The default is NaN.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Symbolic Delay is the <u>symbolic representation</u> of the transport time delay that may exist in the system. Refer to the <u>LabVIEW</u> <u>Control Design User Manual</u> for more information about delays.
- **Variables** contains the name and value of each variable.
 - **Name** is a variable name this VI uses to define the data of the system model. Variable names can be a combination of letters and numbers. A variable name that begins with a capital letter E can produce unpredictable errors if parts of the original string represent numbers like 1E–2. Avoid terms beginning with E in such cases.
 - **Value** is the numeric value this VI associates with the variable. The VI uses this value to evaluate the model.
- **Zero-Pole-Gain Model** is the system model this VI creates. To access and modify the data in the model, use the <u>Model</u> <u>Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Zero-Pole-Gain Model (MIMO)



Complete Complex Conjugate? specifies, when TRUE, to calculate the complex conjugate for any zero or pole that has only a real or pure complex root specified. The default is FALSE.

Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.

- Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- **Zeros-Poles-Gains** is an array of zero-pole-gain models that you can use to define the dynamics of a multiple-input single-output (MISO), single-input multiple-output (SIMO), or multiple-input multiple-output (MIMO) system.
 - **Gain** is the gain of the SISO system. The default is NaN.
 - **Zeros** is the array of zeros of the SISO system. The zeros can be real or complex. If they are complex, they must be in complex conjugate pairs.
 - **Poles** is the array of poles of the SISO system. The poles can be real or complex. If they are complex, they must be in complex conjugate pairs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error

occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Zero-Pole-Gain Model** is the system model this VI creates. To access and modify the data in the model, use the <u>Model</u> <u>Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Zero-Pole-Gain Model (MIMO Symbolic)



- **Complete Complex Conjugate?** specifies, when TRUE, to calculate the complex conjugate for any zero or pole that has only a real or pure complex root specified. The default is FALSE.
- Sampling Time (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time (s) must equal zero. If the model represents a discrete-time system, Sampling Time (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time (s)** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous to</u> <u>Discrete</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- Symbolic Zero-Pole-Gain Model(s) is the <u>symbolic</u> representation of the zero-pole-gain model.
 - **Symbolic Gain** is the scalar gain of the system. The default is NaN.
 - Symbolic Delay is the <u>symbolic representation</u> of the transport time delay that may exist in the system. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.
 - Symbolic Zeros is the array of zeros of the system. The zeros can be real or complex. If they are complex, they must be in complex conjugate pairs.
 - Symbolic Poles is the array of poles of the system. The poles can be real or complex. If they are complex, they must be in complex conjugate pairs.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Variables** contains the name and value of each variable.
 - Name is a variable name this VI uses to define the data of the system model. Variable names can be a combination of letters and numbers. A variable name that begins with a capital letter E can produce unpredictable errors if parts of the original string represent numbers like 1E–2. Avoid terms beginning with E in such cases.
 - **Value** is the numeric value this VI associates with the variable. The VI uses this value to evaluate the model.
- **Zero-Pole-Gain Model** is the system model this VI creates. To access and modify the data in the model, use the <u>Model</u> <u>Information</u> VIs.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Zero-Pole-Gain Model Details

Only the <u>SISO</u> and <u>SISO Symbolic</u> instances of this VI support delays. To represent a delay in the model, you must specify the delay in this VI. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Draw State-Space Equation VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Displays the state-space equation of the State-Space Model.

Note Real-time targets do not support this VI.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

- **Format Coefficients** specifies how to format the resulting equation. You can enter any of the <u>LabVIEW format specifier</u> <u>syntax elements</u>. The default value is %#_g, which removes trailing zeros and specifies that LabVIEW uses fractional or scientific notation depending on the exponent of the number.
- **Display Format** specifies the format in which this VI draws the equation.

0 **Equation** (default)—Draws the equation in the following form: dx/dt = Ax + Bu, y = Cx + Du

1 **Compound Matrix**—Draws the equation with the states and inputs in one vector.

2 **Quadruple**—Draws only the **A**, **B**, **C**, and **D** matrices within one matrix.

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI draws an equation.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error

occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Origin** specifies the upper-left position of the equation this VI draws.
 - **x** specifies the *x*-coordinate of the origin. The default value is 10.
 - **y** specifies the *y*-coordinate of the origin. The default value is 10.
- **Equation** draws a picture of the model equation this VI defines.
- **Draw Area Size** specifies the size of the area in the picture control this VI uses to draw the model equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Draw State-Space Equation Details

This VI does not display the delay information in the state-space equation of the model. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Draw Transfer Function Equation VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Displays the transfer function equation of the model. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

▾

Note Real-time targets do not support this VI.

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Draw Transfer Function Equation (State-Space)



- **Format Coefficients** specifies how to format the resulting equation. You can enter any of the <u>LabVIEW format specifier</u> <u>syntax elements</u>. The default value is %#_g, which removes trailing zeros and specifies that LabVIEW uses fractional or scientific notation depending on the exponent of the number.
- **display format** specifies the format in which this VI displays the equation.

0	Standard
1	Engineering
2	Time Constant
3	Power 1/z

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI draws an equation.
- **Output (row)** specifies the index number of the output row from which to draw the transfer function matrix. The index is zero-based. The default is -1, which draws all outputs.
- **Input (column)** specifies the index number of the input column from which to draw the transfer function matrix. The index is zero-based. The default is -1, which draws all inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Origin** specifies the upper-left position of the equation this VI draws.
 - **x** specifies the *x*-coordinate of the origin. The default value is 10.
 - **y** specifies the *y*-coordinate of the origin. The default value is 10.
- **Function Name** specifies the function this VI draws. For example, if you enter H, this VI displays **H**(s)= for continuous equations and **H**(z)= for discrete equations.
- **Equation** draws a picture of the model equation this VI defines.
- **Draw Area Size** specifies the size of the area in the picture control this VI uses to draw the model equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Draw Transfer Function Equation (Transfer Function)



- **Format Coefficients** specifies how to format the resulting equation. You can enter any of the <u>LabVIEW format specifier</u> <u>syntax elements</u>. The default value is %#_g, which removes trailing zeros and specifies that LabVIEW uses fractional or scientific notation depending on the exponent of the number.
- **display format** specifies the format in which this VI displays the equation.

0	Standard
1	Engineering
2	Time Constant
3	Power 1/z

- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI draws an equation.
- **Output (row)** specifies the index number of the output row from which to draw the transfer function matrix. The index is zero-based. The default is -1, which draws all outputs.
- **Input (column)** specifies the index number of the input column from which to draw the transfer function matrix. The index is zero-based. The default is -1, which draws all inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Origin** specifies the upper-left position of the equation this VI draws.
 - **x** specifies the *x*-coordinate of the origin. The default value is 10.
 - **y** specifies the *y*-coordinate of the origin. The default value is 10.
- **Function Name** specifies the function this VI draws. For example, if you enter H, this VI displays **H**(s)= for continuous equations and **H**(z)= for discrete equations.
- **Equation** draws a picture of the model equation this VI defines.
- **Draw Area Size** specifies the size of the area in the picture control this VI uses to draw the model equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Draw Transfer Function Equation (Zero-Pole-Gain)



- **Format Coefficients** specifies how to format the resulting equation. You can enter any of the <u>LabVIEW format specifier</u> <u>syntax elements</u>. The default value is %#_g, which removes trailing zeros and specifies that LabVIEW uses fractional or scientific notation depending on the exponent of the number.
- **display format** specifies the format in which this VI displays the equation.

0	Standard
1	Engineering
2	Time Constant
3	Power 1/z

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI draws an equation.
- **Output (row)** specifies the index number of the output row from which to draw the transfer function matrix. The index is zero-based. The default is -1, which draws all outputs.
- **Input (column)** specifies the index number of the input column from which to draw the transfer function matrix. The index is zero-based. The default is -1, which draws all inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Origin** specifies the upper-left position of the equation this VI draws.
 - **x** specifies the *x*-coordinate of the origin. The default value is 10.
 - **y** specifies the *y*-coordinate of the origin. The default value is 10.
- **Function Name** specifies the function this VI draws. For example, if you enter H, this VI displays **H**(s)= for continuous equations and **H**(z)= for discrete equations.
- **Equation** draws a picture of the model equation this VI defines.
- **Draw Area Size** specifies the size of the area in the picture control this VI uses to draw the model equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Draw Transfer Function Equation Details

This VI does not display the delay information in the transfer function equation of the model. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Draw Zero-Pole-Gain Equation VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Displays the zero-pole-gain equation of the model. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Ŧ

Note Real-time targets do not support this VI.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Draw Zero-Pole-Gain Equation (State-Space)



- **Format Coefficients** specifies how to format the resulting equation. You can enter any of the <u>LabVIEW format specifier</u> <u>syntax elements</u>. The default value is %#_g, which removes trailing zeros and specifies that LabVIEW uses fractional or scientific notation depending on the exponent of the number.
- **display format** specifies the format in which this VI displays the equation.

0	Standard
1	Engineering
2	Time Constant
3	Power 1/z

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI draws an equation.
- **Output (row)** specifies the index number of the output row from which to draw the zero-pole-gain matrix. The index is zero-based. The default is -1, which draws all outputs.
- **Input (column)** specifies the index number of the input column from which to draw the zero-pole-gain matrix. The index is zero-based. The default is -1, which draws all inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Origin** specifies the upper-left position of the equation this VI draws.
 - **x** specifies the *x*-coordinate of the origin. The default value is 10.
 - **y** specifies the *y*-coordinate of the origin. The default value is 10.
- **Function Name** specifies the function this VI draws. For example, if you enter H, this VI displays **H**(s)= for continuous equations and **H**(z)= for discrete equations.
- **Equation** draws a picture of the model equation this VI defines.
- **Draw Area Size** specifies the size of the area in the picture control this VI uses to draw the model equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Draw Zero-Pole-Gain Equation (Transfer Function)



- **Format Coefficients** specifies how to format the resulting equation. You can enter any of the <u>LabVIEW format specifier</u> <u>syntax elements</u>. The default value is %#_g, which removes trailing zeros and specifies that LabVIEW uses fractional or scientific notation depending on the exponent of the number.
- **display format** specifies the format in which this VI displays the equation.

0	Standard
1	Engineering
2	Time Constant
3	Power 1/z

- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI draws an equation.
- **Output (row)** specifies the index number of the output row from which to draw the zero-pole-gain matrix. The index is zero-based. The default is -1, which draws all outputs.
- **Input (column)** specifies the index number of the input column from which to draw the zero-pole-gain matrix. The index is zero-based. The default is -1, which draws all inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Origin** specifies the upper-left position of the equation this VI draws.
 - **x** specifies the *x*-coordinate of the origin. The default value is 10.
 - **y** specifies the *y*-coordinate of the origin. The default value is 10.
- **Function Name** specifies the function this VI draws. For example, if you enter H, this VI displays **H**(s)= for continuous equations and **H**(z)= for discrete equations.
- **Equation** draws a picture of the model equation this VI defines.
- **Draw Area Size** specifies the size of the area in the picture control this VI uses to draw the model equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Draw Zero-Pole-Gain Equation (Zero-Pole-Gain)



- **Format Coefficients** specifies how to format the resulting equation. You can enter any of the <u>LabVIEW format specifier</u> <u>syntax elements</u>. The default value is %#_g, which removes trailing zeros and specifies that LabVIEW uses fractional or scientific notation depending on the exponent of the number.
- **display format** specifies the format in which this VI displays the equation.

0	Standard
1	Engineering
2	Time Constant
3	Power 1/z

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI draws an equation.
- **Output (row)** specifies the index number of the output row from which to draw the zero-pole-gain matrix. The index is zero-based. The default is -1, which draws all outputs.
- **Input (column)** specifies the index number of the input column from which to draw the zero-pole-gain matrix. The index is zero-based. The default is -1, which draws all inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Origin** specifies the upper-left position of the equation this VI draws.
 - **x** specifies the *x*-coordinate of the origin. The default value is 10.
 - **y** specifies the *y*-coordinate of the origin. The default value is 10.
- **Function Name** specifies the function this VI draws. For example, if you enter H, this VI displays **H**(s)= for continuous equations and **H**(z)= for discrete equations.
- **Equation** draws a picture of the model equation this VI defines.
- **Draw Area Size** specifies the size of the area in the picture control this VI uses to draw the model equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Draw Zero-Pole-Gain Equation Details

This VI does not display the delay information in the zero-pole-gain equation of the model. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Read Model from File VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Opens a file the <u>Write Model to File</u> VI created and reads all the records in the file. Each record contains a separate model. To retrieve all records in the file, all the models must be in the same model representation. You must <u>manually select the polymorphic instance</u> to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Read Model from File (State-Space)



- **File Path (dialog if empty)** specifies the file path to the model location.
- **Offset (records:0)** specifies in bytes how far below the start of the file to start reading from the file. The default is 0.
- Records to Read (-1, all) determines from which record to read the file. If you set Records to Read (-1, all) to -1, the CD Read Model from File VI returns all the records in the file. The default is -1.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- new file path returns the path to the file.
- **First State-Space Model in Record** returns the first state-space model in the record.
- State-Space Model returns all the models in the file if Records to Read is –1. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- Mark after Read (records) is the location of the file mark after the VI reads the records. It points to the character (record) in the file following the last character read.

CD Read Model from File (Transfer Function)



- **File Path (dialog if empty)** specifies the file path to the model location.
- **Offset (records:0)** specifies in bytes how far below the start of the file to start reading from the file. The default is 0.
- Records to Read (-1, all) determines from which record to read the file. If you set Records to Read (-1, all) to -1, the CD Read Model from File VI returns all the records in the file. The default is -1.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **new file path** returns the path to the file.
- **First Transfer Function Model in Record** returns the first transfer function model in the record.
- Transfer Function Model returns all the models in the file if Records to Read is –1. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- Mark after Read (records) is the location of the file mark after the VI reads the records. It points to the character (record) in the file following the last character read.

CD Read Model from File (Zero-Pole-Gain)



- **File Path (dialog if empty)** specifies the file path to the model location.
- **Offset (records:0)** specifies in bytes how far below the start of the file to start reading from the file. The default is 0.
- Records to Read (-1, all) determines from which record to read the file. If you set Records to Read (-1, all) to -1, the CD Read Model from File VI returns all the records in the file. The default is -1.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- new file path returns the path to the file.
- **First Zero-Pole-Gain Model in Record** returns the first zero-pole-gain model in the record.
- **Zero-Pole-Gain Model** returns all the models in the file if **Records to Read** is –1. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- Mark after Read (records) is the location of the file mark after the VI reads the records. It points to the character (record) in the file following the last character read.

CD Write Model to File VI

Owning Palette: Model Construction VIs

Installed With: Control Design and Simulation Module

Creates a new file or appends to an existing file, writes the specified number of records to the file, then closes the file and checks for errors. Each record is a model. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Write Model to File (State-Space)



- **File Path (dialog if empty)** specifies the file path to the model location.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system that you want to write to a file.
- If **append to file?** is TRUE, the VI appends data to an existing file. If **append to file?** is FALSE (default), the VI replaces data in an existing file. If there is no existing file, the VI creates a new file.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **new file path** returns the path to the file.

- **Offset after Write (Records)** is the location in the datalog file where the VI writes the next record.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Write Model to File (Transfer Function)



- **File Path (dialog if empty)** specifies the file path to the model location.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system that you want to write to a file.
- If **append to file?** is TRUE, the VI appends data to an existing file. If **append to file?** is FALSE (default), the VI replaces data in an existing file. If there is no existing file, the VI creates a new file.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **new file path** returns the path to the file.

- **Offset after Write (Records)** is the location in the datalog file where the VI writes the next record.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Write Model to File (Zero-Pole-Gain)



- **File Path (dialog if empty)** specifies the file path to the model location.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system that you want to write to a file.
- If **append to file?** is TRUE, the VI appends data to an existing file. If **append to file?** is FALSE (default), the VI replaces data in an existing file. If there is no existing file, the VI creates a new file.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **new file path** returns the path to the file.

- **Offset after Write (Records)** is the location in the datalog file where the VI writes the next record.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Model Conversion VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Model Conversion VIs to convert a system model from one representation to another, from a continuous-time to a discrete-time model, or from a discrete-time to a continuous-time model. You also can use the Model Conversion VIs to convert a control design model into a simulation model or a simulation model into a control design model.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD Convert</u> <u>Continuous to</u> <u>Discrete</u>	Converts a continuous-time model to a discrete-time model using the Sampling Time (s) and the Method you specify. The Zero-Order-Hold conversion method supports input and output delays that are not an integer multiple of the Sampling Time (s) . The data type you wire to the Continuous State-Space Model input determines the polymorphic instance to use.
<u>CD Convert</u>	Converts a <u>control design</u> model into a model you can
<u>Control</u>	use for <u>simulation</u> . The data type you wire to the State-
<u>Design to</u>	Space Model input determines the polymorphic instance
<u>Simulation</u>	to use.
<u>CD Convert</u>	Incorporates delays into discrete system models by
<u>Delay to</u>	adding poles at the origin to account for the specified
<u>Poles at</u>	delay. The data type you wire to the State-Space Model
<u>Origin</u>	input determines the polymorphic instance to use.
<u>CD Convert</u>	Incorporates time delays in a continuous-time system
<u>Delay with</u>	model using Pade approximation, which converts all
<u>Pade</u>	residuals. The data type you wire to the State-Space
Approximation	Model input determines the polymorphic instance to use.
<u>CD Convert</u> <u>Discrete to</u>	Converts a discrete-time model to a continuous-time model by using the specified Method . The data type you

<u>Continuous</u>	wire to the Discrete State-Space Model input determines the polymorphic instance to use.
<u>CD Convert</u> <u>Discrete to</u> <u>Discrete</u>	Changes the sampling time of a discrete-time system model. The data type you wire to the Discrete State- Space Model input determines the polymorphic instance to use.
<u>CD Convert</u> <u>Simulation to</u> <u>Control</u> <u>Design</u>	Converts a <u>simulation</u> model into a model you can use for <u>control design</u> . You must <u>manually select the</u> <u>polymorphic instance</u> to use.
<u>CD Convert to</u> <u>State-Space</u> <u>Model</u>	Converts a system model to state-space form. This VI produces a full or minimum realization by specifying the Realization Type . The data type you wire to the Transfer Function Model input determines the polymorphic instance to use.
CD Convert to Transfer Function Model	Converts a system model to transfer function form. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Convert to</u> <u>Zero-Pole-</u> <u>Gain Model</u>	Converts a system model to zero-pole-gain form. The data type you wire to the State-Space Model input determines the polymorphic instance to use.

CD Convert Continuous to Discrete VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Converts a continuous-time model to a discrete-time model using the **Sampling Time (s)** and the **Method** you specify. The **Zero-Order-Hold** conversion method supports input and output delays that are not an integer multiple of the **Sampling Time (s)**. The data type you wire to the **Continuous State-Space Model** input determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance	
--------------------	--

■ Place on the block diagram ■ Find on the **Functions** palette

CD Convert Continuous to Discrete (State-Space)



Matching Frequency (rad/s) specifies the frequency at which the gains of the continuous and discrete systems match. The value of this parameter must be between zero and the Nyquist frequency.

This parameter is valid only if you specify **Prewarp** or **Matched Pole-Zero** for the **Method** parameter. The default value is 0, which matches the discrete system to the DC gain of the continuous system.

- **Continuous State-Space Model** is the continuous-time system model that this VI converts into a discrete-time equivalent.
- **Sampling Time (s)** is the fixed time period between successive digital samples that a computer produces. The default is 1.
- Method is the algorithm this VI uses to calculate the discrete equivalent of the continuous-time system model.



error in describes error conditions that occur before this VI or

function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Discrete State-Space Model** is the discrete-time equivalent of the input continuous-time system model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- **Discrete IC Multiplier** is the matrix **P** which this VI uses to convert continuous time initial conditions vector *xc0* to discrete-time initial conditions vector *xd0* for a state-space system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code**

is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Continuous to Discrete (Transfer Function)



DBL Matching Frequency (rad/s) specifies the frequency at which the gains of the continuous and discrete systems match. The value of this parameter must be between zero and the Nyquist frequency.

This parameter is valid only if you specify **Prewarp** or **Matched** Pole-Zero for the Method parameter. The default value is 0, which matches the discrete system to the DC gain of the continuous system.

- Continuous Transfer Function Model is the continuous-time system model that this VI converts into a discrete-time equivalent.
- DBL Sampling Time (s) is the fixed time period between successive digital samples that a computer produces. The default is 1.
- **Method** is the algorithm this VI uses to calculate the discrete equivalent of the continuous-time system model.

0	Zero-Order-Hold (default)
1	Tustin (Bilinear) $s \rightarrow \frac{2(z-1)}{7(z+1)}$
2	Prewarp , where $r = \frac{2 \tan\left(\frac{w \times T}{2}\right)}{w}$
3	Forward $s \rightarrow \frac{z-1}{\tau}$
4	Backward $s \rightarrow \frac{z-1}{zT}$
5	Z-Transform
6	First-Order-Hold
7	Matched Pole-Zero

error in describes error conditions that occur before this VI or

function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Discrete Transfer Function Model** is the discrete-time equivalent of the input continuous-time system model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Continuous to Discrete (Zero-Pole-Gain)



DBL Matching Frequency (rad/s) specifies the frequency at which the gains of the continuous and discrete systems match. The value of this parameter must be between zero and the Nyquist frequency.

This parameter is valid only if you specify **Prewarp** or **Matched** Pole-Zero for the Method parameter. The default value is 0, which matches the discrete system to the DC gain of the continuous system.

- **Continuous Zero-Pole-Gain Model** is the continuous-time system model that this VI converts into a discrete-time equivalent.
- DBL Sampling Time (s) is the fixed time period between successive digital samples that a computer produces. The default is 1.
- **Method** is the algorithm this VI uses to calculate the discrete equivalent of the continuous-time system model.

0	Zero-Order-Hold (default)
1	Tustin (Bilinear) $s \rightarrow \frac{2(z-1)}{7(z+1)}$
2	Prewarp , where $r = \frac{2 \tan\left(\frac{w \times T}{2}\right)}{w}$
3	Forward $s \rightarrow \frac{z-1}{\tau}$
4	Backward $s \rightarrow \frac{z-1}{zT}$
5	Z-Transform
6	First-Order-Hold
7	Matched Pole-Zero

error in describes error conditions that occur before this VI or

function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Discrete Zero-Pole-Gain Model** is the discrete-time equivalent of the input continuous-time system model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Continuous to Discrete Details

This VI supports delays. This VI divides the delays by the **Sampling Time (s)** and if the results contain residues, this VI incorporates the delay information into the resulting discrete model as long as the residue is in the input and/or outputs and you specified a value of **Zero-Order-Hold** for the **Method** parameter. Otherwise, this VI ignores the residues and gives a warning.

If you specify a value of **Zero-Order-Hold** for the **Method** parameter and some of the input and/or output delays are non-integer multiples of the **Sampling Time (s)**, then the resulting discrete model contains n + pd + qd states.

where n is the number of states in the system

pd is the number of inputs with delays that are non-integer multiples of the sampling time

qd is the number of outputs with delays that are non-integer multiples of the sampling time

Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Convert Control Design to Simulation VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Converts a <u>control design</u> model into a model you can use for <u>simulation</u>. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

◄

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Convert Control Design to Simulation (State-Space)

State-Space Model

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the control design model you want to convert into a simulation model. You can use the resulting simulation model with the Simulation VIs and functions.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Simulation State-Space Model returns the simulation model equivalent of a control design model. You can use the resulting simulation model with the Simulation VIs and functions.
- Sampling Time [dt] (s) returns the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a

continuous-time system or a discrete-time system. If the model represents a continuous-time system, **Sampling Time [dt] (s)** equals zero. If the model represents a discrete-time system, **Sampling Time [dt] (s)** is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Control Design to Simulation (Transfer Function)

Transfer Function Model Simulation Transfer Functio... error in (no error)

- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the control design model you want to convert into a simulation model. You can use the resulting simulation model with the Simulation VIs and functions.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Simulation Transfer Function Model returns the simulation model equivalent of a control design model. You can use the resulting simulation model with the Simulation VIs and functions.
- **Sampling Time [dt] (s)** returns the sampling time of the model.

Sampling Time [dt] (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, **Sampling Time [dt] (s)** equals zero. If the model represents a discrete-time system, **Sampling Time [dt] (s)** is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Control Design to Simulation (Zero-Pole-Gain)

Zero-Pole-Gain Model	
error in (no error)	

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the control design model you want to convert into a simulation model. You can use the resulting simulation model with the Simulation VIs and functions.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Simulation Zero-Pole-Gain Model returns the simulation model equivalent of a control design model. You can use the resulting simulation model with the Simulation VIs and functions.
- Sampling Time [dt] (s) returns the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a

continuous-time system or a discrete-time system. If the model represents a continuous-time system, **Sampling Time [dt] (s)** equals zero. If the model represents a discrete-time system, **Sampling Time [dt] (s)** is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Control Design to Simulation (SISO Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the control design model you want to convert into a simulation model. You can use the resulting simulation model with the Simulation VIs and functions.
- **Input** specifies the index number of the input from which this VI converts the transfer function matrix to the simulation model. The index is zero-based. The default is zero.
- **Output** specifies the index number of the output from which this VI converts the transfer function matrix to the simulation model. The index is zero-based. The default is zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in
most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Simulation Transfer Function Model returns the simulation model equivalent of a control design model. You can use the resulting simulation model with the Simulation VIs and functions.
- Sampling Time [dt] (s) returns the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time [dt] (s) equals zero. If the model represents a discrete-time system, Sampling Time [dt] (s) is greater than zero and equal to the sampling rate, in seconds, of the discrete system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Control Design to Simulation (SISO Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the control design model you want to convert into a simulation model. You can use the resulting simulation model with the Simulation VIs and functions.
- **Input** specifies the index number of the input from which this VI converts the zero-pole-gain matrix to the simulation model. The index is zero-based. The default is zero.
- **Output** specifies the index number of the output from which this VI converts the zero-pole-gain matrix to the simulation model. The index is zero-based. The default is zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Simulation Zero-Pole-Gain Model returns the simulation model equivalent of a control design model. You can use the resulting simulation model with the Simulation VIs and functions.
- Sampling Time [dt] (s) returns the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time [dt] (s) equals zero. If the model represents a discrete-time system, Sampling Time [dt] (s) is greater than zero and equal to the sampling rate, in seconds, of the discrete system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Control Design to Simulation Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. When converting to simulation models from control design models, this VI does not transfer the delay from the control design model. To account for the delay when converting a control design model to a simulation model, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation VI</u> (continuous models) or the <u>CD Convert Delay to Poles at Origin VI</u> (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade approximation.

CD Convert Delay to Poles at Origin VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Incorporates delays into discrete system models by adding poles at the origin to account for the specified delay. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

-

■ Place on the block diagram ■ Find on the **Functions** palette

CD Convert Delay to Poles at Origin (State-Space)

State-Space Model

- **State-Space Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the discrete-time system for which this VI converts time delays to poles at the origin.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- State-Space Model Out returns the discrete-time system for which this VI incorporates the total time delay between each inputoutput pair into the model by adding poles at the origin. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Delay to Poles at Origin (Transfer Function)

Transfer Function Model

- **Transfer Function Model In** contains a <u>mathematical</u> <u>representation</u> of and <u>information</u> about the discrete-time system for which this VI converts time delays to poles at the origin.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Transfer Function Model Out is the discrete-time system for which this VI incorporates the total time delay between each inputoutput pair into the model by adding poles at the origin. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Delay to Poles at Origin (Zero-Pole-Gain)

Zero-Pole-Gain Model

- **Zero-Pole-Gain Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the discrete-time system for which this VI converts time delays to poles at the origin.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Zero-Pole-Gain Model Out** is the discrete-time system for which this VI incorporates the total time delay between each input-output pair into the model by adding poles at the origin. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Delay to Poles at Origin Details

Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Convert Delay with Pade Approximation VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Incorporates time delays in a continuous-time system model using Pade approximation, which converts all residuals. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

▼

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Convert Delay with Pade Approximation (State-Space)

State-Space Model

- **State-Space Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the continuous-time system for which this VI incorporates time delays using Pade approximation.
- **Polynomial Order** is the order of the Pade approximation polynomials. The default value is 3.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- State-Space Model Out returns the system for which this VI incorporates the total time delay between each input-output pair into the transfer function between that pair using Pade approximation of **Polynomial Order**. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Delay with Pade Approximation (Transfer Function)

Transfer Function Model

- Transfer Function Model In contains a <u>mathematical</u> representation of and <u>information</u> about the continuous-time system for which this VI incorporates time delays using Pade approximation.
- **Polynomial Order** is the order of the Pade approximation polynomials. The default value is 3.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Transfer Function Model Out** is the system for which this VI incorporates the total time delay between each input-output pair

into the transfer function between that pair using Pade approximation of **Polynomial Order**. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Delay with Pade Approximation (Zero-Pole-Gain)

Transfer Function Model

- **Zero-Pole-Gain Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the continuous-time system for which this VI incorporates time delays using Pade approximation.
- **Polynomial Order** is the order of the Pade approximation polynomials. The default value is 3.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Zero-Pole-Gain Model Out is the system for which this VI incorporates the total time delay between each input-output pair into the transfer function between that pair using Pade

approximation of **Polynomial Order**. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Delay with Pade Approximation Details

Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Convert Discrete to Continuous VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Converts a discrete-time model to a continuous-time model by using the specified **Method**. The data type you wire to the **Discrete State-Space Model** input determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

- P		e
-	I Place on the block diagram 🔳 Find on the Functions p	alatta
100	i Place on the block diadram 🔲 Find on the Functions r	ыепе
_		Juictic

CD Convert Discrete to Continuous (State-Space)



- **Prewarp Frequency (rad/s)** calculates the continuous-time model with the inverse algorithm of the Tustin (bilinear) transformation with prewarping.
- **Discrete State-Space Model** is the discrete-time system model that this VI converts into a continuous-time equivalent.
- Method is the algorithm this VI uses to calculate the continuoustime equivalent of the discrete-time system model.



error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Continuous State-Space Model** is the continuous-time equivalent of the input discrete-time system model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Discrete to Continuous (Transfer Function)



- **Prewarp Frequency (rad/s)** calculates the continuous-time model with the inverse algorithm of the Tustin (bilinear) transformation with prewarping.
- **Discrete Transfer Function Model** is the discrete-time system model that this VI converts into a continuous-time equivalent.
- Method is the algorithm this VI uses to calculate the continuoustime equivalent of the discrete-time system model.



error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Continuous Transfer Function Model is the continuous-time equivalent of the input discrete-time system model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Discrete to Continuous (Zero-Pole-Gain)



- **Prewarp Frequency (rad/s)** calculates the continuous-time model with the inverse algorithm of the Tustin (bilinear) transformation with prewarping.
- **Discrete Zero-Pole-Gain Model** is the discrete-time system model that this VI converts into a continuous-time equivalent.
- Method is the algorithm this VI uses to calculate the continuoustime equivalent of the discrete-time system model.



error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Continuous Zero-Pole-Gain Model is the continuous-time equivalent of the input discrete-time system model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Discrete to Continuous Details

This VI supports delays. This VI multiplies delays by the sampling time in the conversion process. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Convert Discrete to Discrete VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Changes the sampling time of a discrete-time system model. The data type you wire to the **Discrete State-Space Model** input determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Convert Discrete to Discrete (State-Space)

Prewarp Frequency (rad/s) Discrete State-Space Model Resampling Time (s) Method error in (no error)	Ti ∰T₂ error out
---	------------------

- Prewarp Frequency (rad/s) is the frequency that this VI uses to produce a Tustin (bilinear) transformation with pre-warping. This VI only uses Prewarp Frequency (rad/s) when the Method is Prewarp.
- **Discrete State-Space Model** is the discrete-time system model that this VI resamples with the new sampling time.
- **Resampling Time (s)** is the time this VI uses to resample the input discrete-time model. The default is 1.
- Method determines the conversion type this VI uses from discrete to continuous and continuous to discrete when resampling.

0	Zero-Order-Hold (default)
1	Tustin (Bilinear)
2	Prewarp
3	Forward
4	Backward
5	Z-Transform

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Resampled State-Space Model is the modified discrete-time system model with the new sampling time. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Discrete to Discrete (Transfer Function)



- Prewarp Frequency (rad/s) is the frequency that this VI uses to produce a Tustin (bilinear) transformation with pre-warping. This VI only uses Prewarp Frequency (rad/s) when the Method is Prewarp.
- **Discrete Transfer Function Model** is the discrete-time system model that this VI resamples with the new sampling time.
- **Resampling Time (s)** is the time this VI uses to resample the input discrete-time model. The default is 1.
- Method determines the conversion type this VI uses from discrete to continuous and continuous to discrete when resampling.

0	Zero-Order-Hold (default)
1	Tustin (Bilinear)
2	Prewarp
3	Forward
4	Backward
5	Z-Transform

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Resampled Transfer Function Model** is the modified discretetime system model with the new sampling time. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Discrete to Discrete (Zero-Pole-Gain)



- Prewarp Frequency (rad/s) is the frequency that this VI uses to produce a Tustin (bilinear) transformation with pre-warping. This VI only uses Prewarp Frequency (rad/s) when the Method is Prewarp.
- **Discrete Zero-Pole-Gain Model** is the discrete-time system model that this VI resamples with the new sampling time.
- **Resampling Time (s)** is the time this VI uses to resample the input discrete-time model. The default is 1.
- Method determines the conversion type this VI uses from discrete to continuous and continuous to discrete when resampling.

0	Zero-Order-Hold (default)
1	Tustin (Bilinear)
2	Prewarp
3	Forward
4	Backward
5	Z-Transform

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Resampled Zero-Pole-Gain Model** is the modified discrete-time system model with the new sampling time. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Discrete to Discrete Details

This VI supports time delays. In the conversion process, if the results of the delay conversion contain residues, this VI ignores the residues and gives a warning. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.
CD Convert Simulation to Control Design VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Converts a <u>simulation</u> model into a model you can use for <u>control design</u>. You must <u>manually select the polymorphic instance</u> to use.

▾

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Convert Simulation to Control Design (State-Space)



- Simulation State-Space Model represents the simulation model this VI converts into a control design model. You can create a simulation model by using the Simulation VIs and functions.
- Sampling Time [dt] (s) specifies the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time [dt] (s) must equal zero. If the model represents a discrete-time system, Sampling Time [dt] (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **State-Space Model** returns the control design model equivalent of a simulation model. You can use this model with the Control Design VIs and functions.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Simulation to Control Design (Transfer Function)



- Simulation Transfer Function Model represents the simulation model this VI converts into a control design model. You can create a simulation model by using the Simulation VIs and functions.
- Sampling Time [dt] (s) specifies the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time [dt] (s) must equal zero. If the model represents a discrete-time system, Sampling Time [dt] (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Transfer Function Model returns the control design model equivalent of a simulation model. You can use this model with the Control Design VIs and functions.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Simulation to Control Design (Zero-Pole-Gain)



- Simulation Zero-Pole-Gain Model represents the simulation model this VI converts into a control design model. You can create a simulation model by using the Simulation VIs and functions.
- Sampling Time [dt] (s) specifies the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time [dt] (s) must equal zero. If the model represents a discrete-time system, Sampling Time [dt] (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **Zero-Pole-Gain Model** returns the control design model equivalent of a simulation model. You can use this model with the Control Design VIs and functions.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Simulation to Control Design (SISO Transfer Function)



- Simulation Transfer Function Model represents the simulation model this VI converts into a control design model. You can create a simulation model by using the Simulation VIs and functions.
- Sampling Time [dt] (s) specifies the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time [dt] (s) must equal zero. If the model represents a discrete-time system, Sampling Time [dt] (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Transfer Function Model returns the control design model equivalent of a simulation model. You can use this model with the Control Design VIs and functions.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Simulation to Control Design (SISO Zero-Pole-Gain)



- Simulation Zero-Pole-Gain Model represents the simulation model this VI converts into a control design model. You can create a simulation model by using the Simulation VIs and functions.
- Sampling Time [dt] (s) specifies the sampling time of the model. Sampling Time [dt] (s) defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time [dt] (s) must equal zero. If the model represents a discrete-time system, Sampling Time [dt] (s) must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Zero-Pole-Gain Model** returns the control design model equivalent of a simulation model. You can use this model with the Control Design VIs and functions.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Simulation to Control Design Details

This VI does not support delays because simulation models do not directly include delay in their representation of a system. When converting a simulation model to a control design model, this VI does not transfer delay to the control design model. To account for the delays, you must linearize the simulation model before converting it. Refer to the LabVIEW Control Design User Manual for more information about delays.

CD Convert to State-Space Model VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Converts a system model to state-space form. This VI produces a full or minimum realization by specifying the **Realization Type**. The data type you wire to the **Transfer Function Model** input determines the polymorphic instance to use.

This VI converts zero-pole-gain models to transfer function models before converting a zero-pole-gain model into a state-space model.

▼

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Convert Transfer Function to State-Space Model

Transfer Function Model Realization Type Tolerance error out error in (no error)

- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system this VI converts to an equivalent state-space form. In a SISO system, the degree of the numerator must be less than or equal to the degree of the denominator. In a MIMO system, the degree of all numerators must be less than or equal to the degree of their respective denominators.
- **Realization Type** specifies the type of state-space realization to use.

0 **minimum**—Gives a state-space realization after removing zero-pole cancellations and all states that do not affect the output of the system

1 **full** (default)—Provides a state-space realization without reducing any states

- **Tolerance** determines zero-pole cancellations. If the difference between the location of a pole and a zero is within the **Tolerance**, this VI removes the zero-pole pair if the **Realization Type** is **minimum**. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- State-Space Model returns the state-space representation of the given input system model. The properties are the same as that of the input model. This VI returns a state-space model in controllability form. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Zero-Pole-Gain Model to State-Space Model

Zero-Pole-Gain Model Realization Type Tolerance error in (no error)

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system this VI converts to an equivalent state-space form. In a SISO system, the degree of the numerator must be less than or equal to the degree of the denominator. In a MIMO system, the degree of all numerators must be less than or equal to the degree of their respective denominators.
- **Realization Type** specifies the type of state-space realization to use.

minimum—Gives a state-space realization after removing zero-pole cancellations and all states that do not affect the output of the system

1 **full** (default)—Provides a state-space realization without reducing any states

- **Tolerance** determines zero-pole cancellations. If the difference between the location of a pole and a zero is within the **Tolerance**, this VI removes the zero-pole pair if the **Realization Type** is **minimum**. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI or

function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **State-Space Model** returns the state-space representation of the given input system model. The properties are the same as that of the input model. This VI returns a state-space model in controllability form. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert to State-Space Model Details

This VI supports delays. This VI transfers the delay information from the input model to the resulting model. Refer to the <u>LabVIEW Control Design</u> <u>User Manual</u> for more information about delays.

CD Convert to Transfer Function Model VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Converts a system model to transfer function form. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the bloc	ck diagram 🔳	Find on the	Functions	palette

CD Convert State-Space to Transfer Function Model

State-Space Model Realization Type Tolerance error in (no error)

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system this VI converts into an equivalent transfer function form.
- **Realization Type** specifies the type of state-space realization to use.

0 **minimum** (default)—Gives a state-space realization after removing zero-pole cancellations and all states that do not affect the output of the system

- 1 **full**—Provides a state-space realization without reducing any states
- **Tolerance** is the maximum distance for zero-pole pair cancellation when this VI requests minimum realization. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Transfer Function Model returns the transfer function representation of the input system model. The properties are the same as that of the input model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Zero-Pole-Gain to Transfer Function Model



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system this VI converts into an equivalent transfer function form.
- **Realization Type** specifies the type of state-space realization to use.

0 **minimum** (default)—Gives a state-space realization after removing zero-pole cancellations and all states that do not affect the output of the system

- 1 **full**—Provides a state-space realization without reducing any states
- **Tolerance** is the maximum distance for zero-pole pair cancellation when this VI requests minimum realization. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Transfer Function Model returns the transfer function representation of the input system model. The properties are the same as that of the input model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert to Transfer Function Model Details

This VI supports delays. This VI transfers the delay information from the input model to the resulting model. Refer to the <u>LabVIEW Control Design</u> <u>User Manual</u> for more information about delays.

CD Convert to Zero-Pole-Gain Model VI

Owning Palette: Model Conversion VIs

Installed With: Control Design and Simulation Module

Converts a system model to zero-pole-gain form. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

This VI converts state-space models to transfer function models before converting a state-space model to a zero-pole-gain model.

▼

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Convert State-Space to Zero-Pole-Gain Model



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system this VI converts into an equivalent zero-pole-gain form.
- **Realization Type** specifies the type of state-space realization to use.

0 **minimum** (default)—Gives a state-space realization after removing zero-pole cancellations and all states that do not affect the output of the system

- 1 **full**—Provides a state-space realization without reducing any states
- **Tolerance** is the maximum distance for zero-pole pair cancellation when this VI requests minimum realization. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Zero-Pole-Gain Model** returns the zero-pole-gain representation of the input system model. The properties are the same as that of the input model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Transfer Function to Zero-Pole-Gain Model



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system this VI converts into an equivalent zero-pole-gain form.
- **Realization Type** specifies the type of state-space realization to use.

0 **minimum** (default)—Gives a state-space realization after removing zero-pole cancellations and all states that do not affect the output of the system

1 **full**—Provides a state-space realization without reducing any states

- **Tolerance** is the maximum distance for zero-pole pair cancellation when this VI requests minimum realization. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Zero-Pole-Gain Model** returns the zero-pole-gain representation of the input system model. The properties are the same as that of the input model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert to Zero-Pole-Gain Model Details

This VI supports delays. This VI transfers the delay information from the input model to the resulting model. Refer to the <u>LabVIEW Control Design</u> <u>User Manual</u> for more information about delays.

Model Information VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Model Information VIs to obtain or set parameters, data, and names of a system model. Model information includes properties such as the system delay, system dimensions, sampling time, and names of inputs, outputs, and states.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD Get</u> Data from <u>Mode</u> l	Obtains data that describes the dynamics of the given system model. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Get</u> <u>Delays</u> from Model	Obtains all the delays present in the system model. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Get</u> <u>Names</u> from Model	Obtains the names this VI associates with the system model. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Get</u> Sampling Time from Model	Obtains the sampling time of the system. The sampling time is zero for continuous systems and greater than zero for discrete systems. The data type you wire to the State- Space Model input determines the polymorphic instance to use.
<u>CD Get</u> System Dimensions	Obtains the number of inputs, outputs and, if applicable, the number of states of the system. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Set</u> <u>Data to</u> <u>Model</u>	Sets the given values to the system model. The data type you wire to the State-Space Model In input determines the polymorphic instance to use.

<u>CD Set</u> <u>Delays to</u> <u>Model</u>	Sets the given delays in the system model. The data type you wire to the State-Space Model In input determines the polymorphic instance to use.
<u>CD Set</u> <u>Names to</u> <u>Model</u>	Sets the various names this VI associates with the inputs, outputs, and, if applicable, states of the model. The data type you wire to the State-Space Model In input determines the polymorphic instance to use.
<u>CD Set</u> Sampling Time to Model	Sets the sampling time of the system. The data type you wire to the State-Space Model In input determines the polymorphic instance to use.
<u>CD Verify if</u> <u>Delayed</u>	Checks if the input model has any nonzero input delays, output delays, or transport delays. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Verify if</u> <u>Discrete</u>	Checks if the input model represents a discrete-time system. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Verify</u> <u>MIMO</u> <u>Properties</u>	Determines if the dimensions of the system matrices or transfer function matrix are consistent with the properties of the model variables, for example, delays and names. This VI calculates the number of inputs, outputs, and states (for state-space models only), and generates error messages for matrix dimensions that are inconsistent with the model. The data type you wire to the State-Space Model In input determines the polymorphic instance to use.
<u>CD Verify</u> <u>Model Type</u>	Determines the type of system models based on the number of inputs or outputs. A system model can be single- input single-output (SISO), multiple-input multiple-output (MIMO), single-input multiple-output (SIMO), or multiple- input single-output (MISO). The data type you wire to the State-Space Model input determines the polymorphic instance to use.

CD Get Data from Model VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Obtains data that describes the dynamics of the given system model. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Get Data from Model (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a linear time-invariant (LTI) system for which this VI returns data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Matrix A returns the system matrix that describes the dynamics of the states of the system.
- **Matrix B** returns the input matrix of the system that relates the inputs to the states.
- Matrix C returns the output matrix that relates the outputs to the

states.

- Matrix D returns the transmission matrix that relates the inputs to the outputs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Data from Model (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about a linear time-invariant system for which this VI returns data.
- **Output (row)** specifies the output, or row, number of the transfer function in which this VI retrieves the data. This number is necessary when the model represents a MIMO system. The default value is zero.
- Input (column) specifies the index number of the input column of the transfer function in which this VI retrieves the data. This number is necessary when the model represents a MIMO system. The index is zero-based. The default is zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in
most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Numerator** returns the numerator of the transfer function as **Output (row)** and **Input (column)** specify.
- **Denominator** returns the denominator of the transfer function as **Output (row)** and **Input (column)** specify.
- **Delay** returns the transport time delay between the **Output (row)** and **Input (column)** pair.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Data from Model (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about a linear time-invariant system for which this VI returns data.
- **Output (row)** specifies the output, or row, number of the transfer function (in zero-pole-gain form) in which this VI returns data. This number is necessary when the model represents a MIMO system. The index is zero-based. The default value is zero.
- Input (column) specifies the index number of the input column of the transfer function (in zero-pole-gain form) in which this VI returns data. This number is necessary when the model represents a MIMO system. The index is zero-based. The default is zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Zeros** returns the zeros of the SISO system as **Output (row)** and **Input (column)** specify.
- **Poles** returns the poles of the SISO system as **Output (row)** and **Input (column)** specify.
- Gain returns the scalar gain of the SISO system as Output (row) and Input (column) specify.
- **Delay** returns the transport time delay between the **Output (row)** and **Input (column)** pair.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Delays from Model VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Obtains all the delays present in the system model. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Get Delays from Model (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains time delays.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Input Delays** returns the input delays present in the system model. The *i*th element of the array is the time delay in the *i*th input of the system.
- **Output Delays** returns the output delays present in the system model. The *i*th element of the array is the time delay in the *i*th

output of the system.

- **Transport Delays** is a 2D-array. The *ij*th element of this array is the time delay between the *i*th output and *j*th input pair of the system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Delays from Model (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and additional <u>model information</u> about the system for which this VI obtains time delays.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Input Delays** returns the input delays present in the system model. The *i*th element of the array is the time delay in the *i*th input of the system.
- **Output Delays** returns the output delays present in the system model. The *i*th element of the array is the time delay in the *i*th

output of the system.

- **Transport Delays** is a 2D-array. The *ij*th element of this array is the time delay between the *i*th output and *j*th input pair of the system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Delays from Model (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains time delays.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Input Delays** returns the input delays present in the system model. The *i*th element of the array is the time delay in the *i*th input of the system.
- **Output Delays** returns the output delays present in the system model. The *i*th element of the array is the time delay in the *i*th

output of the system.

- **Transport Delays** is a 2D-array. The *ij*th element of this array is the time delay between the *i*th output and *j*th input pair of the system.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Names from Model VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Obtains the names this VI associates with the system model. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Get Names from Model (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains inputs, outputs, and other names.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Model Name** is a string that specifies the name of the model.
- **Input Names** returns the names of the system inputs. The *i*th element of the array gives the name of the *i*th input to the model.
- **Output Names** returns the names of the system outputs. The *i*th

element of the array gives the name of the *i*th output of the model.

State Names returns the names of the states of the model. The *j*th element of the array gives the name of the *j*th state of the model.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Notes** returns information you entered from the original model.

CD Get Names from Model (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains inputs, outputs, and other names.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Model Name** is a string that specifies the name of the model.
- **Input Names** returns the names of the system inputs. The *i*th element of the array gives the name of the *i*th input to the model.
- **Output Names** returns the names of the system outputs. The *i*th

element of the array gives the name of the i^{th} output of the model.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Notes** returns information you entered from the original model.

CD Get Names from Model (Zero-Pole-Gain)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains inputs, outputs, and other names.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Model Name** is a string that specifies the name of the model.
- **Input Names** returns the names of the system inputs. The *i*th element of the array gives the name of the *i*th input to the model.
- **Output Names** returns the names of the system outputs. The *i*th

element of the array gives the name of the i^{th} output of the model.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Notes** returns information you entered from the original model.

CD Get Sampling Time from Model VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Obtains the sampling time of the system. The sampling time is zero for continuous systems and greater than zero for discrete systems. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Use the pull-down menu to select an instance of this VI.

Select an instance

~

■ Place on the block diagram ■ Find on the **Functions** palette

CD Get Sampling Time from Model (State-Space Model)

State-Space Model Sampling Time

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains sampling time.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Sampling Time is the sampling time this VI associates with the system model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time equals zero. If the model represents a discrete-time system, Sampling Time is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Sampling Time from Model (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains sampling time.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Sampling Time is the sampling time this VI associates with the system model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time equals zero. If the model represents a discrete-time system, Sampling Time is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Sampling Time from Model (Zero-Pole-Gain)

Zero-Pole-Gain Model ---- Sampling Time ජය error in (no error) ------ error out

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains sampling time.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Sampling Time is the sampling time this VI associates with the system model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time equals zero. If the model represents a discrete-time system, Sampling Time is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get System Dimensions VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Obtains the number of inputs, outputs and, if applicable, the number of states of the system. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Get System Dimensions (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains system dimensions.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Number of Inputs returns the number of inputs in the system model.
- **Number of Outputs** returns the number of outputs in the system model.
- **Number of States** returns the number of states in the system

model.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get System Dimensions (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains system dimensions.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Number of Inputs returns the number of inputs in the system model.
- **Number of Outputs** returns the number of outputs in the system model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get System Dimensions (Zero-Pole-Gain)

Zero-Pole-Gain Model	Number of Inputs
error in (no error)	toor toor

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI obtains system dimensions.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Number of Inputs** returns the number of inputs in the system model.
- **Number of Outputs** returns the number of outputs in the system model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Data to Model VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Sets the given values to the system model. The data type you wire to the **State-Space Model In** input determines the polymorphic instance to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Set Data to Model (State-Space)



- **State-Space Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI modifies data.
- Matrix A is the system matrix that describes the dynamics of the states of the system. If you do not wire an input to Matrix A, this VI uses the matrix A from the State-Space Model In input.
- Matrix B is the input matrix of the system that relates the inputs to the states. If you do not wire an input to Matrix B, this VI uses the matrix B from the State-Space Model In input.
- Matrix C is the output matrix that relates the outputs to the states. If you do not wire an input to Matrix C, this VI uses the matrix C from the State-Space Model In input.
- Matrix D is the transmission matrix that relates the inputs of the systems to its outputs. If you do not wire an input to Matrix D, this VI uses the matrix D from the State-Space Model In input.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Allow Properties Change? specifies if the new data values in Matrix A, Matrix B, Matrix C, or Matrix D change the number of inputs, outputs, or states in the system model. The default value is FALSE.
- Model Out is the modified form of State-Space Model In with the values that the Matrix A, Matrix B, Matrix C, and Matrix D inputs specify. The properties are the same as that of State-Space Model In. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Data to Model (Transfer Function)



- Transfer Function Model In contains a <u>mathematical</u> representation of and <u>information</u> about the system for which this VI modifies data.
- **Numerator** represents the numerator, in ascending order, of the transfer function as **Output (row)** and **Input (column)** specify.
- **Denominator** represents the denominator, in ascending order, of the transfer function as **Output (row)** and **Input (column)** specify.
- **Output (row)** specifies the output, or row, number of the transfer function in which this VI sets data. This number is necessary when the model represents a MIMO system. The index is zero-based. The default is zero.
- Input (column) specifies the index number of the input column of the transfer function in which this VI sets data. This number is necessary when the model represents a MIMO system. The index is zero-based. The default is zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Delay is the transport time delay between the Output (row) and Input (column) pair. The default is Nan. Refer to the LabVIEW Control Design User Manual for more information about delays.
- Model Out is the modified form of Transfer Function In with the values as the Numerator, Denominator, and Delay inputs specify. The properties are the same as that of Transfer Function Model In. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Set Data to Model (Zero-Pole-Gain)



- **Zero-Pole-Gain Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI modifies data.
- **Zeros** represents the array of zeros, in ascending order, of the system that **Output (row)** and **Input (column)** specify.
- **Poles** represents the array of poles, in ascending order, of the system that **Output (row)** and **Input (column)** specify.
- **Output (row)** specifies the output, or row, number of the transfer function (in zero-pole-gain form) in which this VI returns data. This number is necessary when the model represents a MIMO system. The index is zero-based. The default value is zero.
- Input (column) specifies the index number of the input column of the transfer function (in zero-pole-gain form) in which this VI returns data. This number is necessary when the model represents a MIMO system. The index is zero-based. The default is zero.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Delay is the transport time delay between the Output (row) and Input (column) pair. The default is Nan. Refer to the LabVIEW Control Design User Manual for more information about delays.
- Gain is the scalar gain of the system that Output (row) and Input (column) specify.
- Model Out is the modified form of Zero-Pole-Gain Model In with the values that the Zeros, Poles, Delay, and Gain inputs specify. The properties are the same as that of Zero-Pole-Gain Model In. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Delays to Model VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Sets the given delays in the system model. The data type you wire to the **State-Space Model In** input determines the polymorphic instance to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Set Delays to Model (State-Space)



- **State-Space Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI sets time delays.
- **Input Delays** is an array of input delays where the *i*th element of this array is the time delay in the *i*th input of the system. If you do not wire an input to **Input Delays**, the VI uses the input delays from the **State-Space Model In** input.
- **Output Delays** is an array of output delays where the *i*th element of this array is the time delay in the *i*th output of the system. If you do not wire an input to **Output Delays**, the VI uses the output delays from the **State-Space Model In** input.
- **Transport Delays** is a 2D-array. The *ij*th element of this array is the time delay between the *i*th output and *j*th input pair of the system.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out is the modified form of the model with its delays set according to the values given in Input Delays, Output Delays, and Transport Delays. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Delays to Model (Transfer Function)



- Transfer Function Model In contains a <u>mathematical</u> <u>representation</u> of and <u>information</u> about the system for which this VI sets time delays.
- **Input Delays** is an array of input delays where the *i*th element of this array is the time delay in the *i*th input of the system. If you do not wire an input to **Input Delays**, the VI uses the input delays from the **State-Space Model In** input.
- **Output Delays** is an array of output delays where the *i*th element of this array is the time delay in the *i*th output of the system. If you do not wire an input to **Output Delays**, the VI uses the output delays from the **State-Space Model In** input.
- **Transport Delays** is a 2D-array. The *ij*th element of this array is the time delay between the *i*th output and *j*th input pair of the system.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out is the modified form of the model with its delays set according to the values given in Input Delays, Output Delays, and Transport Delays. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Delays to Model (Zero-Pole-Gain)



- **Zero-Pole-Gain Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI sets time delays.
- **Input Delays** is an array of input delays where the *i*th element of this array is the time delay in the *i*th input of the system. If you do not wire an input to **Input Delays**, the VI uses the input delays from the **State-Space Model In** input.
- **Output Delays** is an array of output delays where the *i*th element of this array is the time delay in the *i*th output of the system. If you do not wire an input to **Output Delays**, the VI uses the output delays from the **State-Space Model In** input.
- **Transport Delays** is a 2D-array. The *ij*th element of this array is the time delay between the *i*th output and *j*th input pair of the system.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out is the modified form of the model with its delays set according to the values given in Input Delays, Output Delays, and Transport Delays. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Names to Model VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Sets the various names this VI associates with the inputs, outputs, and, if applicable, states of the model. The data type you wire to the **State-Space Model In** input determines the polymorphic instance to use.

Ŧ

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Set Names to Model (State-Space)



- **State-Space Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI sets inputs, outputs, and other names.
- **Model Name** is a string that specifies the name of the system model.
- **Input Names** specifies the names of the inputs of the model. The *i*th element of the array sets the name of the *i*th input to the model.
- **Output Names** specifies the names of the outputs of the model. The *j*th element of the array sets the name of the *j*th output of the model.
- **State Names** specifies the names of the states of the model. The *i*th element of the array sets the name of the *i*th state of the model. This output is associated with state-space models only.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Notes** specifies comments or information you want to store with the system model.
- Model Out is the modified system model with the new names properties. To access and modify the data in the model, use the Model Information VIs.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Names to Model (Transfer Function)



- Transfer Function Model In contains a <u>mathematical</u> <u>representation</u> of and <u>information</u> about the system for which this VI sets inputs, outputs, and other names.
- **Model Name** is a string that specifies the name of the system model.
- **Input Names** specifies the names of the inputs of the model. The *j*th element of the array sets the name of the *j*th input to the model.
- **Output Names** specifies the names of the outputs of the model. The *j*th element of the array sets the name of the *j*th output of the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Notes** specifies comments or information you want to store with the system model.
- Model Out is the modified system model with the new names properties. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Names to Model (Zero-Pole-Gain)



- **Zero-Pole-Gain Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI sets inputs, outputs, and other names.
- **Model Name** is a string that specifies the name of the system model.
- **Input Names** specifies the names of the inputs of the model. The *j*th element of the array sets the name of the *j*th input to the model.
- **Output Names** specifies the names of the outputs of the model. The *j*th element of the array sets the name of the *j*th output of the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Notes** specifies comments or information you want to store with the system model.
- Model Out is the modified system model with the new names properties. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Sampling Time to Model VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Sets the sampling time of the system. The data type you wire to the **State-Space Model In** input determines the polymorphic instance to use.

▾

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Set Sampling Time to Model (State-Space)



- **State-Space Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI sets sampling time.
- Sampling Time specifies the sampling time of the model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time must equal zero. If the model represents a discrete-time system, Sampling Time must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- Model Out is the modified system model with a new sampling time. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Sampling Time to Model (Transfer Function)



Transfer Function Model In contains a <u>mathematical</u> <u>representation</u> of and <u>information</u> about the system for which this VI sets sampling time.

- Sampling Time specifies the sampling time of the model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time must equal zero. If the model represents a discrete-time system, Sampling Time must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- Model Out is the modified system model with a new sampling time. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Set Sampling Time to Model (Zero-Pole-Gain)



- **Zero-Pole-Gain Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI sets sampling time.
- Sampling Time specifies the sampling time of the model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time must equal zero. If the model represents a discrete-time system, Sampling Time must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- Model Out is the modified system model with a new sampling time. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify if Delayed VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Checks if the input model has any nonzero input delays, output delays, or transport delays. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

▼

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Verify if Delayed (State-Space)

State-Space Model contains a <u>mathematical representation</u> of and <u>information</u> about a system that this VI checks for the presence of any time delays.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- If **Transport Delay?** is FALSE, the system transport delay is zero.
- If **Delay?** is TRUE, a time delay exists in the model.
- If Input Delay? is FALSE, the system input delay is zero.
- If **Output Delay?** is FALSE, the system output delay is zero.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify if Delayed (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system that this VI checks for the presence of any time delays.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- If **Transport Delay?** is FALSE, the system transport delay is zero.
- If **Delay?** is TRUE, a time delay exists in the model.
- If Input Delay? is FALSE, the system input delay is zero.
- If **Output Delay?** is FALSE, the system output delay is zero.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify if Delayed (Zero-Pole-Gain)



- Zero-Pole-Gain Model contains a <u>mathematical representation</u> of and <u>information</u> about a system that this VI checks for the presence of any time delays.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- If **Transport Delay?** is FALSE, the system transport delay is zero.
- If **Delay?** is TRUE, a time delay exists in the model.
- If Input Delay? is FALSE, the system input delay is zero.
- If **Output Delay?** is FALSE, the system output delay is zero.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify if Discrete VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Checks if the input model represents a discrete-time system. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Use the pull-down menu to select an instance of this VI.

Select an instance

-

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

CD Verify if Discrete (State-Space)



- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system that this VI checks to determine if it represents a discrete-time system.
- If **Discrete?** is TRUE, the model represents a discrete-time system and the **Sampling Time** of the model is greater than zero.
- Sampling Time returns the sampling time of the model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time equals zero. If the model represents a discrete-time system, Sampling Time is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

CD Verify if Discrete (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system that this VI checks to determine if it represents a discrete-time system.
- If **Discrete?** is TRUE, the model represents a discrete-time system and the **Sampling Time** of the model is greater than zero.
- Sampling Time returns the sampling time of the model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time equals zero. If the model represents a discrete-time system, Sampling Time is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

CD Verify if Discrete (Zero-Pole-Gain)

Zero-Pole-Gain Model ------ Discrete?

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system that this VI checks to determine if it represents a discrete-time system.
- If **Discrete?** is TRUE, the model represents a discrete-time system and the **Sampling Time** of the model is greater than zero.
- Sampling Time returns the sampling time of the model. Sampling Time defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, Sampling Time equals zero. If the model represents a discrete-time system, Sampling Time is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

CD Verify MIMO Properties VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Determines if the dimensions of the system matrices or transfer function matrix are consistent with the properties of the model variables, for example, delays and names. This VI calculates the number of inputs, outputs, and states (for state-space models only), and generates error messages for matrix dimensions that are inconsistent with the model. The data type you wire to the **State-Space Model In** input determines the polymorphic instance to use.

Use the pull-down menu to select an instance of this VI.

f		-	ŝ
I			1
I	Select an instance	-	1
I			1
I			48

■ Place on the block diagram ■ Find on the **Functions** palette

CD Verify MIMO Properties (State-Space)



- **State-Space Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI verifies the properties.
- **Required Matrices** specifies the system matrices that this VI requires to continue execution.

0	None (default)
1	Α
2	В
2 3 4	А, В
4	С
5	A, C
6	B, C
7	A, B, C
8	D
9	A, D
10	B, D
11	A, B, D
12	C, D
13	A, C, D
14	B, C, D
15	A, B, C, D

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error
occurred before this VI or function runs. If an error occurs before this VI executes, the VI only closes the reference passed to it. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- If Static Gain Case? is TRUE, the system has a gain when the frequency is zero or when there is direct feedthrough between the input and output.
- State-Space Model Out returns the model after this VI verifies dimensions and specification requirements. To access and modify the data in the model, use the Model Information VIs.
- Number of States is the number of states based on system matrix dimensions.
- **Number of Inputs** is the number of inputs based on model dimensions.
- **Number of Outputs** is the number of outputs based on model dimensions.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify MIMO Properties (Transfer Function)



Transfer Function Model In contains a <u>mathematical</u> <u>representation</u> of and <u>information</u> about the system for which this VI verifies the properties.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs before this VI executes, the VI only closes the reference passed to it. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- If **Static Gain Case?** is TRUE, the system has a gain when the frequency is zero or when there is direct feedthrough between the input and output.
- **Transfer Function Model Out** is the resulting model after this VI

verifies dimensions and specification requirements. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- Number of Inputs is the number of inputs based on model dimensions.
- **Number of Outputs** is the number of outputs based on model dimensions.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify MIMO Properties (Zero-Pole-Gain)



- **Zero-Pole-Gain Model In** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI verifies the properties.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs before this VI executes, the VI only closes the reference passed to it. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- If **Static Gain Case?** is TRUE, the system has a gain when the frequency is zero or when there is direct feedthrough between the input and output.
- Zero-Pole-Gain Model Out is the resulting model after this VI

verifies dimensions and specification requirements. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- Number of Inputs is the number of inputs based on model dimensions.
- **Number of Outputs** is the number of outputs based on model dimensions.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify Model Type VI

Owning Palette: Model Information VIs

Installed With: Control Design and Simulation Module

Determines the type of system models based on the number of inputs or outputs. A system model can be single-input single-output (SISO), multiple-input multiple-output (MIMO), single-input multiple-output (SIMO), or multiple-input single-output (MISO). The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

◄

Use the pull-down menu to select an instance of this VI.

Select an instance	
--------------------	--

■ Place on the block diagram ■ Find on the **Functions** palette

CD Verify Model Type (State-Space)

State-Space Model

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI verifies the model type.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Type of Model** uses the number of inputs and outputs of the system model to determine if the model is SISO, MIMO, SIMO, or MISO.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify Model Type (Transfer Function)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI verifies the model type.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Type of Model** uses the number of inputs and outputs of the system model to determine if the model is SISO, MIMO, SIMO, or MISO.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify Model Type (Zero-Pole-Gain)

Zero-Pole-Gain Model

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI verifies the model type.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Type of Model** uses the number of inputs and outputs of the system model to determine if the model is SISO, MIMO, SIMO, or MISO.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Model Interconnection VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Model Interconnection VIs to perform different types of linear system interconnections. You can build a large system model by connecting smaller system models together.

The Model Interconnection VIs do not support the <u>Stochastic Systems</u> VIs.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD Add</u> <u>Models</u>	Adds Model 1 to Model 2 such that the Model Out represents the addition of the two models. Model 1, Model 2, and Model Out are of the same type and have the same number of inputs and outputs. Therefore, this operation is equivalent to a parallel connection of models that share the same set of inputs, and whose outputs are added accordingly. The models either must be continuous-time models or must have the same sampling time if they are discrete-time models.
<u>CD Append</u>	Connects Model 1 and Model 2 to produce an augmented system which is the Appended Model . The inputs and outputs of the Appended Model are the collective inputs and outputs of Model 1 and Model 2 . The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models. The data types you wire to the Model 1 and Model 2 inputs determine the polymorphic instance to use.
<u>CD Divide</u> <u>Models</u>	Divides Model 1 by a constant Gain such that the Model Out represents the quotient of these inputs. This operation is equivalent to multiplying the Model 1 by the

	inverse of the Gain . Therefore, if this VI cannot calculate the inverse of the Gain , LabVIEW returns an error.
<u>CD Feedback</u>	Connects Model 1 with Model 2 in feedback configuration and produces the Closed-Loop Model . You can specify which outputs this VI feeds back to which inputs of the system. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models. The data types you wire to the Model 1 and/or Model 2 inputs determine the polymorphic instance to use.
	Concatenates Model 1 and Model 2 such that the Model Out is an array that contains these models as columns. The number of inputs of Model Out equals the number of inputs to Model 1 plus the number of inputs to Model 2 . The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models.
<u>CD Multiply</u> <u>Models</u>	Multiplies two models by using matrix multiplication. If you are multiplying two models, the number of outputs of the Model 1 must be equal to the number of inputs of Model 2 . The models either must be continuous-time models or must have the same sampling time if they are discrete-time models.
<u>CD Parallel</u>	Connects Model 1 and Model 2 such that the Parallel Model represents the two input models connected in parallel as the Input Connections and Output Connections specify. The system models must either be continuous-time models or have the same sampling time if they are discrete-time models. The data types you wire to the Model 1 and Model 2 inputs determine the polymorphic instance to use.
<u>CD Series</u>	Connects Model 1 and Model 2 such that the Series Model represents the two input models connected in series as Connections specifies. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models. The data types you wire to the Model 1 and Model 2

	inputs determine the polymorphic instance to use.
<u>CD Subtract</u> <u>Models</u>	Subtracts Model 2 from Model 1 such that the Model Out represents the subtraction of the two models. Model 1 , Model 2 , and Model Out are of the same type and have the same number of inputs and outputs. Therefore, this operation is equivalent to a parallel connection of models that share the same set of inputs, and whose outputs are subtracted accordingly.
<u>CD Transpose</u> <u>Model</u>	Transposes a model such that the <i>ij</i> -th element of Model In becomes the <i>ji</i> -th element of Model Out .
<u>CD Unit</u> <u>Feedback</u>	Connects Model 1 with Model 2 such that the Closed- Loop Model represents these models in a unit feedback configuration. You can specify which outputs this VI feeds back to which inputs of the system. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models. The data types you wire to the Model 1 and/or Model 2 inputs determine the polymorphic instance to use.
CD Vertical Concatenation	Concatenates Model 1 and Model 2 such that the Model Out is an array that contains these models as rows. The number of outputs of Model Out equals the number of outputs from Model 1 plus the number of outputs from Model 2 . The system models either must be continuous- time models or must have the same sampling time if they are discrete-time models.

Subpalette	Description
	Use the Rational Polynomial VIs to perform calculations and evaluations with <u>rational polynomials</u> .

CD Add Models VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Adds **Model 1** to **Model 2** such that the **Model Out** represents the addition of the two models. **Model 1**, **Model 2**, and **Model Out** are of the same type and have the same number of inputs and outputs. Therefore, this operation is equivalent to a parallel connection of models that share the same set of inputs, and whose outputs are added accordingly. The models either must be continuous-time models or must have the same sampling time if they are discrete-time models.

You also can add a single-input single-output (SISO) model or a constant **Gain** to each element of a multiple-input multiple-output (MIMO) model.

The data types you wire to the **Model 1** and **Model 2** inputs determine the polymorphic instance to use.

Details

Use the pull-down menu to select an instance of this VI.

lect an instance

■ Place on the block diagram ■ Find on the Functions palette

CD Add Models (State-Space and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the

resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Add Models (State-Space and Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- **Gain** specifies the gain to add to the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Add Models (Transfer Function and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the

resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Add Models (Transfer Function and Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- **Gain** specifies the gain to add to the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Add Models (Zero-Pole-Gain and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the

resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Add Models (Zero-Pole-Gain and Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- **Gain** specifies the gain to add to the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Add Models Details

This VI supports delays. This VI transfers the delay information from the input models to the augmented model. Refer to the <u>LabVIEW Control</u> <u>Design User Manual</u> for more information about delays.

CD Append VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Connects **Model 1** and **Model 2** to produce an augmented system which is the **Appended Model**. The inputs and outputs of the **Appended Model** are the collective inputs and outputs of **Model 1** and **Model 2**. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models. The data types you wire to the **Model 1** and **Model 2** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about appending models.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Append State-Space and State-Space



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space

model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append (State-Space and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space

model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append (State-Space and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append (Transfer Function and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append Transfer Function and Transfer Function



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append Transfer Function and Zero-Pole-Gain



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append Zero-Pole-Gain and State-Space



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append Zero-Pole-Gain and Transfer Function



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append Zero-Pole-Gain and Zero-Pole-Gain



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Appended Model is the augmented model this VI produces from the two input models. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Append Details

This VI supports delays. This VI transfers the delay information from the input models to the augmented model. Refer to the <u>LabVIEW Control</u> <u>Design User Manual</u> for more information about delays.

CD Divide Models VI

Owning Palette: Model Interconnection VIs

Installed With: Control Design and Simulation Module

Divides **Model 1** by a constant **Gain** such that the **Model Out** represents the quotient of these inputs. This operation is equivalent to multiplying the **Model 1** by the inverse of the **Gain**. Therefore, if this VI cannot calculate the inverse of the **Gain**, LabVIEW returns an error.

You also can divide two models by one another. In this situation, both models either must be continuous-time models or must have the same sampling time if they are discrete-time models.

The data types you wire to the **Model 1** and **Gain** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about appending models.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the Functions palette

CD Divide Models (State-Space by Gain)



- **Model 1** specifies the model that is the dividend.
- **Gain** specifies the gain that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models (Gain by State-Space)



- **Gain** specifies the gain that is the dividend.
- **Model 2** specifies the model that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models (Transfer Function by Gain)



- **Model 1** specifies the model that is the dividend.
- **Gain** specifies the gain that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models (Gain by Transfer Function)



Gain specifies the gain that is the dividend.

- **Model 2** specifies the model that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models (Zero-Pole-Gain by Gain)



- **Model 1** specifies the model that is the dividend.
- **Gain** specifies the gain that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models (Gain by Zero-Pole-Gain)



Gain specifies the gain that is the dividend.

- **Model 2** specifies the model that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models (State-Space by State-Space)



- **Model 1** specifies the model that is the dividend.
- **Model 2** specifies the model that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models (Transfer Function by Transfer Function)



- **Model 1** specifies the model that is the dividend.
- **Model 2** specifies the model that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models (Zero-Pole-Gain by Zero-Pole-Gain)



- **Model 1** specifies the model that is the dividend.
- **Model 2** specifies the model that is the divisor.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Divide Models Details

This VI supports delays. This VI transfers the delay information from the input models to the augmented model. Refer to the <u>LabVIEW Control</u> <u>Design User Manual</u> for more information about delays.

CD Feedback VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Connects **Model 1** with **Model 2** in feedback configuration and produces the **Closed-Loop Model**. You can specify which outputs this VI feeds back to which inputs of the system. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models. The data types you wire to the **Model 1** and/or **Model 2** inputs determine the polymorphic instance to use.

 $\overline{\mathbb{N}}$

Note You can use the <u>CD Unit Feedback</u> VI to connect two models in a unit feedback configuration.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about connecting models in a feedback configuration.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

Place on the block diagram Find on the Functions palette

CD Feedback State-Space and State-Space



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

Model Output specifies the index number of the output of
- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback State-Space and Transfer Function



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback State-Space and Zero-Pole-Gain



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback Transfer Function and Transfer Function



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback Transfer Function and State-Space



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback Transfer Function and Zero-Pole-Gain



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback Zero-Pole-Gain and Zero-Pole-Gain



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback Zero-Pole-Gain and State-Space



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback Zero-Pole-Gain and Transfer Function



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 is the first model this VI uses to create the Closed-Loop Model. This model represents the system in the forward loop path.
- Model 2 is the second model this VI uses to create the Closed-Loop Model. This model represents the system in the feedback path. If you do not wire a model to Model 2, then this VI assumes a unit feedback by defining Model 2 as the unit gain matrix. The number of connections in Feedback Connections defines the size of the unit gain matrix.
- Feedback Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. This VI adds or subtracts the output value of Model 1 or Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased. If you do not specify any Feedback Connections, this VI connects as many input-output pairs as possible from Model 1 to Model 2, and Feedback Sign specifies the sign of all feedback connections. If you specify only one model, this VI feeds back the outputs from Model 1 to the inputs of Model 1. If you specify only one model and you do not specify any Feedback Connections, this VI applies unit feedback to Model 1 with as many connections as possible in ascending order.

- Model 1 Input specifies the index number of the input of
 Model 1 to which you want to connect an output of Model 1
 or Model 2. The indexes are zero-based.
- Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define input-output pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default is negative.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Output Connections uses the index number of the input and output to define each input-output pair, from Model 1 and Model 2, in a feedback loop. If you do not specify any Output Connections, this VI connects as many input-output pairs as possible from Model 2 to Model 1. If you specify only one model in this VI, the VI does not use the Output Connections in feedback calculations.
 - Model 1 Output specifies the index number of the output of Model 1 to which you want to connect an input of Model 2. Signal determines if this VI adds or subtracts the output of Model 1 to the reference input. The indexes are zerobased.
 - Model 2 Input specifies the index number of the input of
 Model 2 to which you want to connect the output of Model
 1. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model 1 Output and Model 2 Input specify. Signal specifies if this VI adds or subtracts the output of Model 1 to a reference to define the input of Model 2. The default is negative.
- **Closed-Loop Model** represents the closed-loop system that results from this VI connecting **Model 1** and **Model 2** according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

CD Feedback Details

This VI does not support delays for models in a closed-loop configuration. If a model has a delay, this VI ignores the delays and gives a warning. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.
CD Horizontal Concatenation VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Concatenates **Model 1** and **Model 2** such that the **Model Out** is an array that contains these models as columns. The number of inputs of **Model Out** equals the number of inputs to **Model 1** plus the number of inputs to **Model 2**. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models.

The data types you wire to the **Model 1** and **Model 2** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about appending models.

•

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Horizontal Concatenation (State-Space and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Model Out** returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation (State-Space and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation (State-Space and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation (Transfer Function and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation (Transfer Function and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation (Transfer Function and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation (Zero-Pole-Gain and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation (Zero-Pole-Gain and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation (Zero-Pole-Gain and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Horizontal Concatenation Details

This VI supports delays. This VI transfers the delay information from the input models to the augmented model. Refer to the <u>LabVIEW Control</u> <u>Design User Manual</u> for more information about delays.

CD Multiply Models VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Multiplies two models by using matrix multiplication. If you are multiplying two models, the number of outputs of the **Model 1** must be equal to the number of inputs of **Model 2**. The models either must be continuous-time models or must have the same sampling time if they are discrete-time models.

You also can multiply a multiple-input multiple-output (MIMO) model by a single-input single-output (SISO) model or by a constant **Gain**. This operation involves element multiplication instead of matrix multiplication.

The data types you wire to the **Model 1** and **Model 2** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about appending models.

▾

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Multiply Models (Transfer Function and Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- **Gain** specifies the gain by which this VI multiplies the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Multiply Models (Zero-Pole-Gain and Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- **Gain** specifies the gain by which this VI multiplies the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Multiply Models (State-Space and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the

resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Multiply Models (Transfer Function and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Multiply Models (Zero-Pole-Gain and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the

resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Multiply Models (State-Space and Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- **Gain** specifies the gain by which this VI multiplies the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Multiply Models Details

This VI supports delays. This VI transfers the delay information from the input models to the augmented model. Refer to the <u>LabVIEW Control</u> <u>Design User Manual</u> for more information about delays.

CD Parallel VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Connects **Model 1** and **Model 2** such that the **Parallel Model** represents the two input models connected in parallel as the **Input Connections** and **Output Connections** specify. The system models must either be continuous-time models or have the same sampling time if they are discrete-time models. The data types you wire to the **Model 1** and **Model 2** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about connecting models in parallel.

▾

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

CD Parallel State-Space and State-Space



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or
that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - Output Model 2 is the output of the second model in which this VI adds value to or subtracts value from the output of the first model Output Model 1 specifies. Signal 2 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do

not want to adjust the model for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel State-Space and Transfer Function

This VI converts transfer function models into state-space models before connecting the models.



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Output Model 2** is the output of the second model in which this VI adds value to or subtracts value from the output of the first model **Output Model 1** specifies. **Signal 2** specifies if this VI adds or subtracts this input. **Output Connections** uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the Adjust Model for Delay parameter,

this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel State-Space and Zero-Pole-Gain

This VI converts zero-pole-gain models into state-space models before connecting the models.



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Output Model 2** is the output of the second model in which this VI adds value to or subtracts value from the output of the first model **Output Model 1** specifies. **Signal 2** specifies if this VI adds or subtracts this input. **Output Connections** uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the Adjust Model for Delay parameter,

this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel Transfer Function and State-Space

This VI converts transfer function models into state-space models before connecting the models.



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Output Model 2** is the output of the second model in which this VI adds value to or subtracts value from the output of the first model **Output Model 1** specifies. **Signal 2** specifies if this VI adds or subtracts this input. **Output Connections** uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the Adjust Model for Delay parameter,

this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel Transfer Function and Transfer Function



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - Output Model 2 is the output of the second model in which this VI adds value to or subtracts value from the output of the first model Output Model 1 specifies. Signal 2 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do

not want to adjust the model for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel Transfer Function and Zero-Pole-Gain

This VI converts zero-pole-gain models into transfer function models before connecting the models.



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Output Model 2** is the output of the second model in which this VI adds value to or subtracts value from the output of the first model **Output Model 1** specifies. **Signal 2** specifies if this VI adds or subtracts this input. **Output Connections** uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the Adjust Model for Delay parameter,

this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel Zero-Pole-Gain and State-Space

This VI converts zero-pole-gain models into state-space models before connecting the models.



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Output Model 2** is the output of the second model in which this VI adds value to or subtracts value from the output of the first model **Output Model 1** specifies. **Signal 2** specifies if this VI adds or subtracts this input. **Output Connections** uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the Adjust Model for Delay parameter,

this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel Zero-Pole-Gain and Transfer Function

This VI converts zero-pole-gain models into transfer function models before connecting the models.



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Output Model 2** is the output of the second model in which this VI adds value to or subtracts value from the output of the first model **Output Model 1** specifies. **Signal 2** specifies if this VI adds or subtracts this input. **Output Connections** uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the Adjust Model for Delay parameter,

this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel Zero-Pole-Gain and Zero-Pole-Gain



- Input Connections specifies the pair of inputs, from Model 1 and Model 2, connected in parallel.
 - Input Model 1 is the input of the first model that shares the same input value as the input of the second model, as Input Model 2 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
 - Input Model 2 is the input of the second model that shares the same input value as the input of the first model, as Input Model 1 specifies. Input Connections uses the index number of the inputs to identify the pair of inputs. The indexes are zero-based.
- Model 1 is the first model this VI uses in creating the **Parallel** Model.
- Model 2 is the second model this VI uses in creating the **Parallel** Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Output Connections** specifies the pair of outputs, from **Model 1** and **Model 2**, connected in parallel.
 - Output Model 1 is the output of the first model in which this VI adds value to or subtracts value from the output of the second model Output Model 2 specifies. Signal 1 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - Output Model 2 is the output of the second model in which this VI adds value to or subtracts value from the output of the first model Output Model 1 specifies. Signal 2 specifies if this VI adds or subtracts this input. Output Connections uses the index number of the outputs to identify the pair of outputs. The indexes are zero-based.
 - **Signal 1** specifies if this VI adds or subtracts the output of **Model 1** to the output of **Model 2**.
 - **Signal 2** specifies if this VI adds or subtracts the output of **Model 2** to the output of **Model 1**.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do

not want to adjust the model for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Parallel Model returns the new system that results from this VI connecting Model 1 and Model 2 in parallel according to the Input Connections and Output Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parallel Details

This VI supports delays for single-input single-output models. For multiple-input multiple-output models, this VI transfers the delays of the outputs of the first model, which connect to the inputs of the second model, to the inputs of the first model. Likewise, this VI transfers the delays of the inputs of the second model, which connect to the outputs of the first model, to the outputs of the second model.

This VI transfers the delay because the connected input-output pair disappears from the resulting series model. You can lose some transport delay information when this VI eliminates such connected input-output pairs in the resulting series model. You can configure this VI to incorporate delays into the resulting series model by using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Series VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Connects **Model 1** and **Model 2** such that the **Series Model** represents the two input models connected in series as **Connections** specifies. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models. The data types you wire to the **Model 1** and **Model 2** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about connecting models in series.

▼

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

CD Series State-Space and State-Space



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series State-Space and Transfer Function

This VI converts transfer function models into state-space models before connecting the models.



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the **Connections** specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series State-Space and Zero-Pole-Gain

This VI converts zero-pole-gain models into state-space models before connecting the models.



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the **Connections** specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu
for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series Transfer Function and State-Space

This VI converts transfer function models into state-space models before connecting the models.



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the **Connections** specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series Transfer Function and Transfer Function



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series Transfer Function and Zero-Pole-Gain

This VI converts zero-pole-gain models into transfer function models before connecting the models.



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the **Connections** specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series Zero-Pole-Gain and State-Space

This VI converts zero-pole-gain models into state-space models before connecting the models.



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the **Connections** specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series Zero-Pole-Gain and Transfer Function

This VI converts zero-pole-gain models into transfer function models before connecting the models.



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the **Connections** specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series Zero-Pole-Gain and Zero-Pole-Gain



- Model 1 is the first model this VI uses to create the **Connected** Model.
- Model 2 is the second model this VI uses to create the Connected Model.
- **Connections** specifies which outputs of the first model this VI connects to which inputs of the second model.
 - Output Model 1 specifies the output number of the first model that is connected to an input of the second model as specified by Input Model 2. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
 - Input Model 2 is the input number of the second model that is connected to the output of the first model as specified by Output Model 1. Connections uses the index number of the input and output to identify the input-output pair. The indexes are zero-based.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.
- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Series Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the Connections specification. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Series Details

This VI supports delays for single-input single-output models. For multiple-input multiple-output models, this VI transfers the delays of the outputs of the first model, which this VI connects to the inputs of the second model, to the inputs of the first model. Likewise, this VI transfers the delays of the inputs of the second model, which this VI connects to the outputs of the first model, to the outputs of the second model.

This VI transfers the delay because the connected input-output pair disappears from the resulting series model. You can lose some transport delay information when this VI eliminates such connected input-output pairs in the resulting series model. You can configure this VI to incorporate delays into the resulting series model by using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Subtract Models VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Subtracts **Model 2** from **Model 1** such that the **Model Out** represents the subtraction of the two models. **Model 1**, **Model 2**, and **Model Out** are of the same type and have the same number of inputs and outputs. Therefore, this operation is equivalent to a parallel connection of models that share the same set of inputs, and whose outputs are subtracted accordingly.

You also can subtract a single-input single-output (SISO) model or a constant **Gain** from each element of a multiple-input multiple-output (MIMO) model and vice-versa. The models either must be continuous-time models or must have the same sampling time if they are discrete-time models.

The data types you wire to the **Model 1** and **Model 2** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about appending models.

▼

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Subtract Models (State-Space from State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the

resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models (Gain from State-Space)



- **Model In** specifies the model from which you want to subtract information.
- **Gain** specifies the gain to subtract from the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models (State-Space from Gain)



- **Gain** specifies the gain from which this VI subtracts the model.
- **Model In** specifies the model this VI subtracts from the gain.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models (Transfer Function from Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models (Gain from Transfer Function)



- **Model In** specifies the model from which you want to subtract information.
- **Gain** specifies the gain to subtract from the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the

data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models (Transfer Function from Gain)



- **Gain** specifies the gain from which this VI subtracts the model.
- **Model In** specifies the model this VI subtracts from the gain.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models (Zero-Pole-Gain from Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the

resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models (Gain from Zero-Pole-Gain)



- **Model In** specifies the model from which you want to subtract information.
- **Gain** specifies the gain to subtract from the model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the
data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models (Zero-Pole-Gain from Gain)



- **Gain** specifies the gain from which this VI subtracts the model.
- **Model In** specifies the model this VI subtracts from the gain.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Subtract Models Details

This VI supports delays. This VI transfers the delay information from the input models to the augmented model. Refer to the <u>LabVIEW Control</u> <u>Design User Manual</u> for more information about delays.

CD Transpose Model VI

Owning Palette: Model Interconnection VIs

Installed With: Control Design and Simulation Module

Transposes a model such that the *ij*-th element of **Model In** becomes the *ji*-th element of **Model Out**.

The data type you wire to the **Model In** input determines the polymorphic instance to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Transpose Model (State-Space)



Model In specifies the model this VI transposes.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Model Out** returns the transpose of **Model In**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Transpose Model (Transfer Function)



Model In specifies the model this VI transposes.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Model Out** returns the transpose of **Model In**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Transpose Model (Zero-Pole-Gain)



Model In specifies the model this VI transposes.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Model Out** returns the transpose of **Model In**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback VI

Owning Palette: <u>Model Interconnection VIs</u>

Installed With: Control Design and Simulation Module

Connects **Model 1** with **Model 2** such that the **Closed-Loop Model** represents these models in a unit feedback configuration. You can specify which outputs this VI feeds back to which inputs of the system. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models. The data types you wire to the **Model 1** and/or **Model 2** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about connecting models in a feedback configuration.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Unit Feedback (State-Space and State-Space)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback (State-Space and Transfer Function)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback (State-Space and Zero-Pole-Gain)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback (Transfer Function and State-Space)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback (Transfer Function and Transfer Function)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback (Transfer Function and Zero-Pole-Gain)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback (Zero-Pole-Gain and State-Space)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback (Zero-Pole-Gain and Transfer Function)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Unit Feedback (Zero-Pole-Gain and Zero-Pole-Gain)



Feedback Sign specifies the sign of all feedback connections. If **Feedback Sign** is positive (TRUE), all feedback connections are positive. If **Feedback Sign** is negative (FALSE), all feedback connections are negative. The default is negative (FALSE).

When you specify connections using **Feedback Connections**, this VI ignores **Feedback Sign**.

- Model 1 specifies the first model this VI uses to create the Closed-Loop Model.
- Model 2 specifies the second model this VI uses to create the Closed-Loop Model.
- Series Connections specifies the outputs of Model 1 to connect to the inputs of Model 2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Feedback Connections uses the index number of the output and input to define each output-input pair, from Model 2 and Model 1, in a feedback loop. This VI adds or subtracts the output value of Model 2 from a reference input and assigns a value to the input of Model 1. Signal determines if this VI adds or subtracts the output of Model 2 to the reference input. The indexes are zero-based. If you do not specify any Feedback Connections, this VI connects as many output-input pairs as possible from Model 2 to Model 1, and Feedback Sign specifies the sign of all feedback connections.
 - Model Output specifies the index number of the output of Model 2 to which you want to connect an input of Model 1.
 - Model 1 Input specifies the index number of the input of Model 1 to which you want to connect an output of Model 2. The indexes are zero-based.
 - Signal defines the feedback sign for the connection that Model Output and Model 1 Input specify. If you define output-input pairs using Feedback Connections, this VI uses Signal to define the feedback sign instead of Feedback Sign. The default value is negative.
- **Delay Adjustment Settings** approximates the delay that this VI eliminated by connecting the models. With continuous systems, the approximation is based on Pade approximation, while discrete systems increase the system order to account for delay.

If you specify FALSE for the **Adjust Model for Delay** parameter, this VI does not make any approximation for any delay.

Adjust Model for Delay specifies if you want to adjust the model for delay. The default is FALSE, which means you do not want to adjust the model for delay.

- **Pade Approximation Order** specifies the Pade approximation order. The default is 2.
- Closed-Loop Model represents the closed-loop system that results from this VI connecting Model 1 and Model 2 according to the connections you specify. When the two input models are different model types, this VI determines the model type of the resulting model by the following model hierarchy: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the resulting model is a state-space model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Unit Feedback Details

This VI does not support delays for models in a closed-loop configuration. If a model has a delay, this VI ignores the delays and gives a warning. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Vertical Concatenation VI

Owning Palette: Model Interconnection VIs

Installed With: Control Design and Simulation Module

Concatenates **Model 1** and **Model 2** such that the **Model Out** is an array that contains these models as rows. The number of outputs of **Model Out** equals the number of outputs from **Model 1** plus the number of outputs from **Model 2**. The system models either must be continuous-time models or must have the same sampling time if they are discrete-time models.

The data types you wire to the **Model 1** and **Model 2** inputs determine the polymorphic instance to use.

This VI might convert one or both input models to different forms before connecting the models. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about appending models.

•

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Vertical Concatenation (State-Space and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: statespace>transfer function>zero-pole-gain. For example, if one input is a state-space model and the other is a zero-pole-gain model, the

resulting model is a state-space model. To access and modify the data in the model, use the <u>Model Information</u> VIs.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation (State-Space and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation (State-Space and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation (Transfer Function and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation (Transfer Function and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation (Transfer Function and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation (Zero-Pole-Gain and Zero-Pole-Gain)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation (Zero-Pole-Gain and State-Space)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation (Zero-Pole-Gain and Transfer Function)



- Model 1 specifies the first model this VI uses to create the Model Out.
- Model 2 specifies the second model this VI uses to create the Model Out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Model Out returns the model this VI produces. When the two input models are not of the same model type, the following model hierarchy determines the model type of the resulting model: state-space>transfer function>zero-pole-gain. For example, if one input

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Vertical Concatenation Details

This VI supports delays. This VI transfers the delay information from the input models to the augmented model. Refer to the <u>LabVIEW Control</u> <u>Design User Manual</u> for more information about delays.

Model Reduction VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Model Reduction VIs to perform a zero-pole cancellation or to reduce the number of states in state-space models. You also can use the Model Reduction VIs to eliminate inputs and outputs that are uncontrollable or unobservable.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD</u> <u>Minimal</u> <u>Realization</u>	Simplifies the model by using the minimum order possible to fully describe the dynamic system. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD</u> <u>Minimal</u> <u>State</u> <u>Realization</u>	Calculates a reduced State-Space Model based on null rows and columns in the controllability and observability matrices, respectively. Even though the controllability and observability matrices might be rank deficient, state elimination only occurs when columns or rows on such matrices are below a tolerance.
<u>CD Model</u> <u>Order</u> <u>Reduction</u>	Eliminates state dynamics that you want to ignore in the State-Space Model . This VI can eliminate the state dynamics by deleting the states or by assuming steady state conditions for states with fast dynamics, also known as pseudo-steady-state assumption. The pseudo steady state assumption does not affect the discrete and continuous system gain of the original system, that is, the remaining states will have the same gain as when considering the full order system.
<u>CD</u> <u>Remove</u> <u>IO from</u>	Selects the inputs, outputs and/or states that you want to remove from the original system model. If you want to keep inputs, outputs, or states from the original system model,

<u>Model</u>	use the <u>CD Select IO from Model</u> VI. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Select</u> <u>IO from</u> <u>Model</u>	Selects the inputs, outputs, and/or states that you want to keep from the original system model. If you want to remove inputs, outputs, or states from the original system model, use the <u>CD Remove IO from Model</u> VI. The data type you wire to the State-Space Model input determines the polymorphic instance to use.

CD Minimal Realization VI

Owning Palette: Model Reduction VIs

Installed With: Control Design and Simulation Module

Simplifies the model by using the minimum order possible to fully describe the dynamic system. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

A minimal realization for a state-space model is a state-space representation in which you remove all states that are not observable or controllable. A minimal realization for transfer function and zero-pole-gain models is a representation where you use zero-pole cancellations to reduce the number of modes required to describe the system dynamics.

For state-space models, this VI uses the staircase transformation to first obtain a group of controllable states. After identifying the controllable states, this VI uses the staircase transformation again to specify which of the controllable states also are observable. The minimal realization consists of the controllable and observable states of the system.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

-

■ Place on the block diagram ■ Find on the **Functions** palette

CD Minimal Realization (State-Space)

State-Space Model Tolerance Vumber of States Removed error in (no error)

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the original state-space model this VI reduces.
- **Tolerance** specifies the threshold this VI uses to determine modes to eliminate. For state-space models, **Tolerance** specifies the controllability and observability thresholds. For transfer function and zero-pole-gain models, **Tolerance** also specifies zero-pole cancellation closeness. The default is 1E–12.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally even if an error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Reduced Model** represents the minimal realization of the system

model. To access and modify the data in the model, use the <u>Model</u> <u>Information</u> VIs.

- Number of States Removed returns the number of states this VI removes from the model to generate a minimal realization of the original state-space model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Minimal Realization (Transfer Function)

For transfer function models, this VI searches for the closest zero to each pole of the dynamic system. If the closest zero is within the threshold that the **Tolerance** specifies, this VI removes the zero-pole pair and reduces the system order.



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the original transfer function model this VI reduces.
- **Tolerance** specifies the threshold this VI uses to determine modes to eliminate. For state-space models, **Tolerance** specifies the controllability and observability thresholds. For transfer function and zero-pole-gain models, **Tolerance** also specifies zero-pole cancellation closeness. The default is 1E–12.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally even if an error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Reduced Model represents the minimal realization of the system model. To access and modify the data in the model, use the Model Information VIs.
- **Number of Poles-Zeros Removed** returns the number of polezero pairs this VI removes from each input-output pair of a model. The *ij*th element of this parameter corresponds to the number of pole-zero pairs this VI removes from the equation at the *i*th output and *j*th input location.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Minimal Realization (Zero-Pole-Gain)

For zero-pole-gain models, this VI searches for the closest zero to each pole of the dynamic system. If the closest zero is within the threshold the **Tolerance** specifies, this VI removes the zero-pole pair and reduces the system order.



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the original zero-pole-gain model this VI reduces.
- **Tolerance** specifies the threshold this VI uses to determine modes to eliminate. For state-space models, **Tolerance** specifies the controllability and observability thresholds. For transfer function and zero-pole-gain models, **Tolerance** also specifies zero-pole cancellation closeness. The default is 1E–12.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally even if an error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- Reduced Model represents the minimal realization of the system model. To access and modify the data in the model, use the Model Information VIs.
- **Number of Poles-Zeros Removed** returns the number of polezero pairs this VI removes from each input-output pair of a model. The *ij*th element of this parameter corresponds to the number of pole-zero pairs this VI removes from the equation at the *i*th output and *j*th input location.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Minimal Realization Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Minimal State Realization VI

Owning Palette: Model Reduction VIs

Installed With: Control Design and Simulation Module

Calculates a reduced **State-Space Model** based on null rows and columns in the controllability and observability matrices, respectively. Even though the controllability and observability matrices might be rank deficient, state elimination only occurs when columns or rows on such matrices are below a tolerance.

<u>Details</u>



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI reduces the states.
- **Tolerance** is the threshold this VI uses to determine if a row or column of the controllability or observability matrix is null. The default is 2E–16.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Reduced Model returns the minimal state-space realization. To access and modify the data in the model, use the Model Information VIs.
- **Removed States** returns the index number of the states that this VI eliminated from the original system. The index is zero-based.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Minimal State Realization Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Model Order Reduction VI

Owning Palette: Model Reduction VIs

Installed With: Control Design and Simulation Module

Eliminates state dynamics that you want to ignore in the **State-Space Model**. This VI can eliminate the state dynamics by deleting the states or by assuming steady state conditions for states with fast dynamics, also known as pseudo-steady-state assumption. The pseudo steady state assumption does not affect the discrete and continuous system gain of the original system, that is, the remaining states will have the same gain as when considering the full order system.

<u>Details</u>



Place on the block diagram Find on the Functions palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI reduces the states.
- **States** lists the index number of the states you want to eliminate. The index is zero-based.
- **Procedure** determines how to eliminate the states.

0 **Match Gain** (default)—The VI considers the states, which you want to eliminate, to be in a steady state. This VI substitutes the steady state values of the removed states into the dynamics of the reduced order system.

1 **Simple Detection**—The VI removes the states from the system model.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Reduced Model is the resulting reduced State-Space Model. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Model Order Reduction Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Remove IO from Model VI

Owning Palette: Model Reduction VIs

Installed With: Control Design and Simulation Module

Selects the inputs, outputs and/or states that you want to remove from the original system model. If you want to keep inputs, outputs, or states from the original system model, use the <u>CD Select IO from Model</u> VI. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

CD Remove IO from State-Space Model



- States specifies the index number of the states you want to remove from the State-Space Model. The index is zero-based. If you wire a value of -1 to this parameter, this VI removes all model states.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI removes the states, inputs and/or outputs from the resulting model.
- **Inputs** specifies the index numbers of inputs you want to remove from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI removes all model inputs.
- **Outputs** specifies the index numbers of outputs you want to remove from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI removes all model outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Reduced Model is the resulting system model without the states, inputs, or outputs you specified to remove. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Remove IO from Transfer Function Model



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system in which this VI removes the inputs and/or outputs from the resulting model.
- **Inputs** specifies the index numbers of inputs you want to remove from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI removes all model inputs.
- **Outputs** specifies the index numbers of outputs you want to remove from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI removes all model outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **Reduced Model** is the resulting system model without the states, inputs, or outputs you specified to remove. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Remove IO from Zero-Pole-Gain Model



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system in which this VI removes the inputs and/or outputs from the resulting model.
- **Inputs** specifies the index numbers of inputs you want to remove from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI removes all model inputs.
- **Outputs** specifies the index numbers of outputs you want to remove from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI removes all model outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **Reduced Model** is the resulting system model without the states, inputs, or outputs you specified to remove. To access and modify the data in the model, use the <u>Model Information</u> VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Remove IO from Model Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

CD Select IO from Model VI

Owning Palette: Model Reduction VIs

Installed With: Control Design and Simulation Module

Selects the inputs, outputs, and/or states that you want to keep from the original system model. If you want to remove inputs, outputs, or states from the original system model, use the <u>CD Remove IO from Model</u> VI. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.



Note If you specify a value of [-1] for the **States**, **Inputs**, or **Outputs** parameter, this VI selects all model states/inputs/outputs, respectively.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Select IO from State-Space Model



Allow Reorder is TRUE if you want this VI to reorder the model states/inputs/outputs in the order you select them. Allow Reorder is FALSE if you do not want to reorder the states/inputs/outputs. The default value is FALSE.

For example, consider a model with three outputs in the following order:

Output Number	Index Number
1	0
2	1
3	2

You specify a value of [1 0] for the **Outputs** parameter. If **Allow Reorder?** is FALSE, the **Selected Model** contains the first two outputs in the original order. However, if **Allow Reorder?** is TRUE, the **Selected Model** contains the first two outputs in the following order:

Output Number	Index Number
1	1
2	0

- **States** is the index number of the states you want to select from the **State-Space Model**. The index is zero-based. If you wire a value of -1 to this parameter, this VI selects all model states.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI keeps the states, inputs and/or outputs in the resulting model.

- **Inputs** specifies the index numbers of the inputs you want to select from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI selects all model inputs.
- **Outputs** specifies the index numbers of the outputs you want to select from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI selects all model outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Selected Model is the resulting system model with only the states, inputs, or outputs you selected. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Select IO from Transfer Function Model



TF

Allow Reorder is TRUE if you want this VI to reorder the model states/inputs/outputs in the order you select them. Allow Reorder is FALSE if you do not want to reorder the states/inputs/outputs. The default value is FALSE.

For example, consider a model with three outputs in the following order:

Output Number	Index Number
1	0
2	1
3	2

You specify a value of [1 0] for the **Outputs** parameter. If **Allow Reorder?** is FALSE, the **Selected Model** contains the first two outputs in the original order. However, if **Allow Reorder?** is TRUE, the **Selected Model** contains the first two outputs in the following order:

Output Number	Index Number
1	1
2	0

- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system in which this VI keeps the inputs and/or outputs in the resulting model.
- Inputs specifies the index numbers of the inputs you want to select from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI selects all model inputs.

- **Outputs** specifies the index numbers of the outputs you want to select from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI selects all model outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Selected Model is the resulting system model with only the states, inputs, or outputs you selected. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Select IO from Zero-Pole-Gain Model





Allow Reorder is TRUE if you want this VI to reorder the model states/inputs/outputs in the order you select them. Allow Reorder is FALSE if you do not want to reorder the states/inputs/outputs. The default value is FALSE.

For example, consider a model with three outputs in the following order:

Output Number	Index Number
1	0
2	1
3	2

You specify a value of [1 0] for the **Outputs** parameter. If **Allow Reorder?** is FALSE, the **Selected Model** contains the first two outputs in the original order. However, if **Allow Reorder?** is TRUE, the **Selected Model** contains the first two outputs in the following order:

Output Number	Index Number
1	1
2	0

- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system in which this VI keeps the inputs and/or outputs in the resulting model.
- Inputs specifies the index numbers of the inputs you want to select from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI selects all model inputs.

- **Outputs** specifies the index numbers of the outputs you want to select from the system model. The index is zero-based. If you wire a value of -1 to this parameter, this VI selects all model outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Selected Model is the resulting system model with only the states, inputs, or outputs you selected. To access and modify the data in the model, use the Model Information VIs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Select IO from Model Details

This VI supports delays. Refer to the <u>LabVIEW Control Design User</u> <u>Manual</u> for more information about delays.

Predictive Control VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Predictive Control VIs to construct and implement a predictive controller model for a state-space plant.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the <u>LabVIEW Control Design User Manual</u> for information about using these VIs.

Palette Object	Description
<u>CD</u> <u>Create</u> <u>MPC</u> <u>Controller</u>	Creates a model predictive control (MPC) controller for a state-space model. You must <u>manually select the</u> polymorphic instance to use.
<u>CD</u> <u>Create</u> <u>MPC</u> <u>FIFO</u>	Creates a queue or real-time (RT) FIFO for an MPC controller. You use this queue or RT FIFO to update setpoint and/or disturbance profiles dynamically.
<u>CD Delete</u> <u>MPC</u> <u>FIFO</u>	Deletes the MPC FIFO . After you delete this FIFO, the <u>CD</u> <u>Write MPC FIFO</u> VI stops writing data to the FIFO and the loop that contains this VI terminates.
<u>CD</u> Implement <u>MPC</u> Controller	Calculates the Control Action u(k) to apply to the plant. This VI uses the Output Reference Window , Disturbance Window , and Control Action Reference Window parameters to calculate the control action along the control horizon at time <i>k</i> .
<u>CD Read</u> <u>MPC</u> <u>FIFO</u>	Reads a portion, or window, of profile values from the MPC FIFO .
CD Set MPC Controller	Updates specified parameters of a model predictive control (MPC) controller for a state-space model. You must <u>manually</u> <u>select the polymorphic instance</u> to use.

<u>CD Step</u> <u>Forward</u> <u>MPC</u> <u>Window</u>	Calculates the appropriate portion, or window, of the setpoint and/or disturbance profiles. You wire these windows to the appropriate input(s) of the <u>CD Implement MPC Controller</u> VI.
<u>CD</u> <u>Update</u> <u>MPC</u> <u>Window</u>	Calculates the appropriate portion, or window, of the setpoint or disturbance profile of a signal from time k to time k + Prediction Horizon . You wire these windows to the appropriate input(s) of the <u>CD Implement MPC Controller</u> VI.
<u>CD Write</u> <u>MPC</u> <u>FIFO</u>	Writes a control action setpoint, output setpoint, or disturbance profile window to the MPC FIFO . You then use the <u>CD Read MPC FIFO</u> VI to read values from this MPC FIFO.

CD Create MPC Controller VI

Owning Palette: Predictive Control VIs

Installed With: Control Design and Simulation Module

Creates a model predictive control (MPC) controller for a state-space model. You must <u>manually select the polymorphic instance</u> to use.

If you want to create an MPC controller for a transfer function model or a zero-pole-gain model, you must first convert the model to a state-space model.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the <u>LabVIEW Control Design User Manual</u> for information about using this VI.

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

_

■ Place on the block diagram ■ Find on the **Functions** palette

CD Create MPC Controller (Dual)



- **Dd** specifies the direct feedthrough matrix for the disturbance in the system.
- **Bd** specifies the input matrix for the disturbance in the system.
- **MPC Controller Parameters** specifies execution parameters for the model predictive controller.
 - **Prediction Horizon** specifies the number of future discrete time samples that this VI considers for the output error term in the cost function.
 - **Control Horizon** specifies the number of future discrete time samples that this VI considers when calculating the control action. The value of this parameter must be less than the value of the **Prediction Horizon** parameter.
 - **Initial Window** specifies the initial range, or window, of plant output errors that this VI does not consider for the cost function. National Instruments recommends you specify the value of this parameter to be, at most, one sample less than the minimum delay of the model.

National Instruments recommends that you adjust the value of this parameter when the model has a delay of 10 or more samples. This adjustment reduces the internal calculations of the MPC controller. Because the control action does not affect the plant outputs for the duration of the delays, the initial window adjustment reduces the controller size significantly.

Integral Action? specifies whether the **MPC Controller** includes integral action. Integral action is important when the MPC controller model does not match the plant model.

- State-Space Model specifies a discrete state-space representation of the plant you want to control. Providing an accurate model improves the performance of the MPC controller this VI creates.
- **MPC Cost Weights** specifies the weight matrices this VI uses in the cost function.
 - Note These weight matrices are different from the **Q**, **R**, and **N** parameters that define the state estimator.
 - **Output Error Weightings** specifies the weight matrix **Q** this VI uses for each plant output error in the cost function. The dimensions of this matrix must match the number of plant outputs.

To prevent this VI from returning sub-optimal values, this matrix must be positive semi-definite.

Control Action Change Weightings specifies the weight matrix *R* this VI uses for each rate of control action change in the cost function. The dimensions of this matrix must match the number of plant inputs.

To prevent this VI from returning sub-optimal values, this matrix must be positive definite.

Control Action Error Weightings specifies the weight matrix *N* this VI uses for each control action error change in the cost function.

National Instruments recommends that you specify **Control Action Error Weightings** for systems with more inputs than outputs.

Output Error Factors specifies the factors this VI uses to increase or decrease the weights for each plant output error along the **Prediction Horizon** in the cost function. This VI applies these factors at subsequent sample times within the **Prediction Horizon**.

Output Error Factors must be greater than 0. For each

element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the plant output error at each sample time within the **Prediction Horizon**.

Control Action Change Factors specifies the factors this VI uses to increase or decrease the weights for each rate of control action change along the Control Horizon in the cost function. This VI applies these factors at subsequent sample times within the Prediction Horizon.

Control Action Change Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the rate of control action change at each sample time within the **Prediction Horizon**.

Control Action Error Factors specifies the factors this VI uses to increase or decrease the weights for each control action error along the Control Horizon in the cost function. This VI applies these factors at subsequent sample times within the Prediction Horizon.

Control Action Error Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the control action error at each sample time within the **Prediction Horizon**.

- **MPC Constraints (Dual)** specifies any constraints on the quadratic programming (QP) optimization algorithm using the dual optimization method. Each parameter specifies a minimum or maximum limit on the initial or final control action *u*, plant output *y*, or rate of change in control action *du*. If you specify only a minimum or a maximum limit for a parameter, LabVIEW assumes a constant profile for that parameter.
 - **u min init** specifies the initial minimum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

u min final specifies the final minimum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

u max init specifies the initial maximum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

u max final specifies the final maximum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

y min init specifies the initial minimum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

y min final specifies the final minimum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

y max init specifies the initial maximum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

y max final specifies the final maximum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

du min init specifies the initial minimum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

du min final specifies the final minimum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

du max init specifies the initial maximum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

du max final specifies the final maximum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

- **stopping criteria** is the collection of conditions that terminate the optimization. If (**function tolerance** AND **parameter tolerance** AND **gradient tolerance**) OR **max iterations** OR **max function calls** then optimization terminates.
 - **function tolerance** is the relative change in function value and is defined as abs(current f prev f)/(abs(curr f)+machine eps). If the relative change in the function value falls below **function tolerance**, the optimization terminates.
 - **parameter tolerance** is the relative change in parameter values and is defined as abs(current p – prev p)/(abs(curr p)+machine eps). If the relative change of all the parameter values falls below

parameters tolerance, the optimization terminates.

- **gradient tolerance** is the 2–norm of the gradient. If the 2–norm of the gradient falls below **gradient tolerance**, the optimization terminates.
- **max iterations** is the largest number of iterations of the major loop of the optimization. If the number of major loop iterations exceeds **max iterations**, the optimization terminates.
- **max function calls** is the largest number of objective function calls allowed before terminating the optimization process.
- **max time (sec)** is the maximum amount of time LabVIEW allows between the start and the end of the optimization process. The default is -1. -1 indicates never to time out.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

MPC State Estimator Parameters specifies the parameters of the state estimator that the MPC Controller uses to estimate the states of the plant.

Use the **Q**, **R**, and **N** parameters to define a state estimator by using the **Noise Covariance** method. This method results in a state estimator that uses the Kalman filter gain.



Note These parameters are different from the **Q**, **R**, and **N** weight matrices in the cost function.

Use the **Observer Poles** parameter to define a state estimator by using the **Pole Placement** method.



Defining a state estimator is optional. If you do not define an estimator and can measure the plant states, you can wire the plant states to the Estimated State xhat(k) input terminal of the <u>CD Implement MPC Controller</u> VI.

- **Internal Estimator Gain Calculation** specifies the method this VI uses to define the state estimator.
 - 0 **None** (default)—Specifies that this VI does not define an internal state estimator. In this situation, you can provide the states to the CD Implement MPC Controller VI.
 - 1 **Pole Placement**—Specifies that this VI defines a state estimator by using the value of the **Observer Poles** parameter.
 - 2 **Noise Covariance**—Specifies that this VI defines a state estimator by using the values of the **Q**, **R**, and **N** parameters. This method results in a Kalman filter.
- **Observer Poles** specifies the locations of the estimator poles. This parameter is valid only if you specify **Pole Placement** for the **Internal Estimator Gain Calculation** parameter.

[DBL]

 ${\bf Q}$ specifies the auto-covariance matrix of the process noise vector.

- **N** specifies the cross-covariance matrix between the process noise and the measurement noise vectors.
 - Note If the process noise and measurement noise vectors are uncorrelated, either specify a matrix of zeros for this parameter or do not wire a value to this parameter.
- **R** specifies the auto-covariance matrix of the measurement noise vector.
- MPC Initial Conditions specifies the initial conditions for the MPC Controller.
 - **Control Action (u)** specifies the initial value(s) of the control action. The default is a vector of zeros.
 - **Output (y)** specifies the initial value(s) of the plant output. The default is a vector of zeros.
 - **Disturbance (d)** specifies the initial value(s) of the disturbance that affects the plant. The default is a vector of zeros.
 - **State (x)** specifies the initial value(s) of the states of the plant. The default is a vector of zeros.
 - **Control Action Change (Du)** specifies the initial values of the rate of change in control action. The default is a vector of zeros.
- **MPC Controller** returns the predictive controller model that corresponds to the parameters this VI specifies.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Create MPC Controller (Barrier)



- **Dd** specifies the direct feedthrough matrix for the disturbance in the system.
- **Bd** specifies the input matrix for the disturbance in the system.
- **MPC Controller Parameters** specifies execution parameters for the model predictive controller.
 - **Prediction Horizon** specifies the number of future discrete time samples that this VI considers for the output error term in the cost function.
 - **Control Horizon** specifies the number of future discrete time samples that this VI considers when calculating the control action. The value of this parameter must be less than the value of the **Prediction Horizon** parameter.
 - **Initial Window** specifies the initial range, or window, of plant output errors that this VI does not consider for the cost function. National Instruments recommends you specify the value of this parameter to be, at most, one sample less than the minimum delay of the model.

National Instruments recommends that you adjust the value of this parameter when the model has a delay of 10 or more samples. This adjustment reduces the internal calculations of the MPC controller. Because the control action does not affect the plant outputs for the duration of the delays, the initial window adjustment reduces the controller size significantly.

Integral Action? specifies whether the **MPC Controller** includes integral action. Integral action is important when the MPC controller model does not match the plant model.
- State-Space Model specifies a discrete state-space representation of the plant you want to control. Providing an accurate model improves the performance of the MPC controller this VI creates.
- **MPC Cost Weights** specifies the weight matrices this VI uses in the cost function.
 - Note These weight matrices are different from the **Q**, **R**, and **N** parameters that define the state estimator.
 - **Output Error Weightings** specifies the weight matrix **Q** this VI uses for each plant output error in the cost function. The dimensions of this matrix must match the number of plant outputs.

To prevent this VI from returning sub-optimal values, this matrix must be positive semi-definite.

Control Action Change Weightings specifies the weight matrix *R* this VI uses for each rate of control action change in the cost function. The dimensions of this matrix must match the number of plant inputs.

To prevent this VI from returning sub-optimal values, this matrix must be positive definite.

Control Action Error Weightings specifies the weight matrix *N* this VI uses for each control action error change in the cost function.

National Instruments recommends that you specify **Control Action Error Weightings** for systems with more inputs than outputs.

Output Error Factors specifies the factors this VI uses to increase or decrease the weights for each plant output error along the **Prediction Horizon** in the cost function. This VI applies these factors at subsequent sample times within the **Prediction Horizon**.

Output Error Factors must be greater than 0. For each

element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the plant output error at each sample time within the **Prediction Horizon**.

Control Action Change Factors specifies the factors this VI uses to increase or decrease the weights for each rate of control action change along the Control Horizon in the cost function. This VI applies these factors at subsequent sample times within the Prediction Horizon.

Control Action Change Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the rate of control action change at each sample time within the **Prediction Horizon**.

Control Action Error Factors specifies the factors this VI uses to increase or decrease the weights for each control action error along the Control Horizon in the cost function. This VI applies these factors at subsequent sample times within the Prediction Horizon.

Control Action Error Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the control action error at each sample time within the **Prediction Horizon**.

- **MPC Constraints (Barrier)** specifies any constraints on the quadratic programming (QP) optimization algorithm using the barrier function method. Each parameter specifies a minimum limit, maximum limit, tolerance, or penalty on the initial or final control action *u*, plant output *y*, or rate of change in control action *du*. If you specify only a minimum or a maximum limit for a parameter, LabVIEW assumes a constant profile for that parameter.
 - **u min init** specifies the initial minimum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

u min final specifies the final minimum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

- **u min tolerance** specifies the tolerance range for the minimum limit of the control action. The penalty on the cost function becomes active when *u* drops below **u min + u min tolerance**.
- **u min penalty** specifies the penalty on the cost function for the minimum limit of the control action when *u* equals **u min**. The penalty on the cost function becomes active when *u* drops below **u min** + **u min tolerance**.
- **u max init** specifies the initial maximum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

u max final specifies the final maximum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

- **u max tolerance** specifies the tolerance range for the maximum limit of the control action. The penalty on the cost function becomes active when *u* exceeds **u max u max tolerance**.
- **u max penalty** specifies the penalty on the cost function for the maximum limit of the control action when *u* equals **u max**. The penalty on the cost function becomes active when *u* exceeds **u max u max tolerance**.
- **y min init** specifies the initial minimum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

y min final specifies the final minimum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

- **y** min tolerance specifies the tolerance range for the minimum limit of the plant output. The penalty on the cost function becomes active when *y* drops below **y** min + **y** min tolerance.
- **y** min penalty specifies the penalty on the cost function for the minimum limit of the plant output when *y* equals **y** min. The penalty on the cost function becomes active when *y* drops below **y** min + **y** min tolerance.
- **y max init** specifies the initial maximum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

y max final specifies the final maximum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

- **y max tolerance** specifies the tolerance range for the maximum limit of the plant output. The penalty on the cost function becomes active when *y* exceeds **y max y max tolerance**.
- **y max penalty** specifies the penalty on the cost function for the maximum limit of the plant output when *y* equals **y max**. The penalty on the cost function becomes active when *y* exceeds **y max y max tolerance**.
- **du min init** specifies the initial minimum limit of the rate of change in control action. This VI uses this limit for constraint

optimization along the Control Horizon.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

du min final specifies the final minimum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

- **du min tolerance** specifies the tolerance range for the minimum limit of the rate of change in control action. The penalty on the cost function becomes active when *du* drops below **du min + du min tolerance**.
- **du min penalty** specifies the penalty on the cost function for the minimum limit of the rate of change in control action when *du* equals **du min**. The penalty on the cost function becomes active when *du* drops below **du min + du min tolerance**.
- **du max init** specifies the initial maximum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

du max final specifies the final maximum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

- **du max tolerance** specifies the tolerance range for the maximum limit of the rate of change in control action. The penalty on the cost function becomes active when *du* exceeds **du max du max tolerance**.
- **bet** du max penalty specifies the penalty on the cost function for the maximum limit of the rate of change in control action when *du* equals **du max**. The penalty on the cost function becomes active when *du* exceeds **du max du max** tolerance.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **MPC State Estimator Parameters** specifies the parameters of the state estimator that the **MPC Controller** uses to estimate the states of the plant.

Use the **Q**, **R**, and **N** parameters to define a state estimator by using the **Noise Covariance** method. This method results in a state estimator that uses the Kalman filter gain.



Note These parameters are different from the **Q**, **R**, and **N** weight matrices in the cost function.

Use the **Observer Poles** parameter to define a state estimator by using the **Pole Placement** method.



Defining a state estimator is optional. If you do not define an estimator and can measure the plant states, you can wire

the plant states to the **Estimated State xhat(k)** input terminal of the <u>CD Implement MPC Controller</u> VI.

- Internal Estimator Gain Calculation specifies the method this VI uses to define the state estimator.
 - 0 **None** (default)—Specifies that this VI does not define an internal state estimator. In this situation, you can provide the states to the CD Implement MPC Controller VI.
 - 1 **Pole Placement**—Specifies that this VI defines a state estimator by using the value of the **Observer Poles** parameter.
 - 2 Noise Covariance—Specifies that this VI defines a state estimator by using the values of the Q, R, and N parameters. This method results in a Kalman filter.
- **Observer Poles** specifies the locations of the estimator poles. This parameter is valid only if you specify **Pole Placement** for the **Internal Estimator Gain Calculation** parameter.
- **Q** specifies the auto-covariance matrix of the process noise vector.
- **N** specifies the cross-covariance matrix between the process noise and the measurement noise vectors.
 - Note If the process noise and measurement noise vectors are uncorrelated, either specify a matrix of zeros for this parameter or do not wire a value to this parameter.
- **R** specifies the auto-covariance matrix of the measurement noise vector.
- MPC Initial Conditions specifies the initial conditions for the MPC Controller.
 - **Control Action (u)** specifies the initial value(s) of the control action. The default is a vector of zeros.
 - **Output (y)** specifies the initial value(s) of the plant output. The default is a vector of zeros.

- **Disturbance (d)** specifies the initial value(s) of the disturbance that affects the plant. The default is a vector of zeros.
- **State (x)** specifies the initial value(s) of the states of the plant. The default is a vector of zeros.
- **Control Action Change (Du)** specifies the initial values of the rate of change in control action. The default is a vector of zeros.
- **MPC Controller** returns the predictive controller model that corresponds to the parameters this VI specifies.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the CD Create MPC Controller VI:

• CDEx MPC with Barrier Constraints VI: labview\examples\Control and Simulation\Control Design\MPC

■ Open example ■ Browse related examples

• CDEx MPC with Dual Constraints VI: labview\examples\Control and Simulation\Control Design\MPC

Open example Browse related examples

CD Create MPC FIFO VI

Owning Palette: Predictive Control VIs

Installed With: Control Design and Simulation Module

Creates a queue or real-time (RT) FIFO for an MPC controller. You use this queue or RT FIFO to update setpoint and/or disturbance profiles dynamically.

You can create up to three types of queues/RT FIFOs: one each for the output setpoint, control action setpoint, and disturbance profiles.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the LabVIEW Control Design User Manual for information about using this VI.



■ Place on the block diagram ■ Find on the **Functions** palette

- MPC Controller In specifies the predictive controller model this VI uses. You create this model by using the <u>CD Create MPC</u> <u>Controller</u> VI.
- **Signal Type** specifies whether the MPC FIFO contains values of an output setpoint profile, control action setpoint profile, or disturbance profile. You can choose from the following options:

0 **Outputs** (default)—Specifies this MPC FIFO contains values of an output setpoint profile.

1 **Inputs**—Specifies this MPC FIFO contains values of a control action setpoint profile.

2 **Disturbances**—Specifies this MPC FIFO contains values of a disturbance profile.

MPC FIFO Size In specifies the maximum number of elements in the MPC FIFO. The type of elements depends on the value you specify for the Signal Type parameter. The default value of MPC FIFO Size In is –1, which specifies no limit.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **MPC FIFO Name** specifies an optional name for the MPC FIFO.
- **MPC Controller Out** returns the predictive controller model.
- **MPC FIFO** returns the queue or FIFO this VI creates.
- **MPC FIFO Size Out** returns the maximum number of elements in the MPC FIFO.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

132

code is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Delete MPC FIFO VI

Owning Palette: Predictive Control VIs

Installed With: Control Design and Simulation Module

Deletes the **MPC FIFO**. After you delete this FIFO, the <u>CD Write MPC</u> <u>FIFO</u> VI stops writing data to the FIFO and the loop that contains this VI terminates.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the <u>LabVIEW Control Design User Manual</u> for information about using this VI.



■ Place on the block diagram ■ Find on the **Functions** palette

- **MPC FIFO** specifies the MPC FIFO this VI deletes. You create this FIFO by using the <u>CD Create MPC FIFO</u> VI.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Implement MPC Controller VI

Owning Palette: <u>Predictive Control VIs</u>

Installed With: Control Design and Simulation Module

Calculates the **Control Action u(k)** to apply to the plant. This VI uses the **Output Reference Window**, **Disturbance Window**, and **Control Action Reference Window** parameters to calculate the control action along the control horizon at time k.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the <u>LabVIEW Control Design User Manual</u> for information about using this VI.

Use the pull-down menu to select an instance of this VI.

Select an instance

-

■ Place on the block diagram ■ Find on the **Functions** palette

CD Implement MPC Controller (MIMO)



FIFO VI.

National Instruments recommends that you specify **Control Action Reference Window** for systems with more inputs than outputs.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Measured Output y(k)** specifies the measured outputs of the plant at the current sample time *k*. You can obtain these outputs by using sensors to measure the plant output.



Note If you are simulating the MPC controller, you can use the <u>Discrete State-Space</u> function to represent the plant.

Estimated State xhat(k) specifies the estimated values of the plant states at the current sample time *k*. Providing state values is optional but improves the accuracy of the **MPC Controller**.



Note If you do not wire a value to this parameter, you can estimate these states by using the CD Create MPC

Controller VI to define a state estimator. You also can use the Discrete State-Space function to simulate the plant and obtain the actual state values.

- Measured Control Action u(k–1) specifies the measured control action from the previous sample time *k*–1. This information is useful because the actuator might not be able to implement the Control Action u(k) this VI returns. In this situation, the MPC Controller takes the limitations of the actuator into account.
 - Note If the Control Action u(k) differs from the Measured Control Action u(k–1), saturation might be present in the dynamic system.
- **Control Action u(k)** returns the control action to apply to the plant at the current sample time *k*. This control action results in the **Estimated Output yhat(k)** from the plant.
- MPC Final Conditions returns the final data for the MPC Controller.
 - **Control Action (u)** returns the current values of the control actions.
 - **Output (y)** returns the current values of the plant outputs.
 - **Disturbance (d)** returns the current values of the disturbance.
 - **State (x)** returns the current values of the plant states.
 - **Control Action Change (Du)** returns the current rates at which the control actions are changing.
- **Estimated Output yhat(k)** returns the predicted output of the plant after applying all control action changes along the control horizon.
- **Lagrangian Multipliers** returns the coefficients of the lagrangian function that corresponds to the equality and inequality constraints on the MPC controller.

For example, if you specified three equality constraints and two inequality constraints by using the **MPC Optimization Parameters** parameter of the CD Create MPC Controller VI, the first three values of the **Lagrangian Multipliers** parameter correspond to the equality constraints and the last two correspond to the inequality constraints.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Implement MPC Controller (SISO)



FIFO VI.

National Instruments recommends that you specify **Control** Action Reference Window for systems with more inputs than outputs.

- **error in** describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - TFI status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - 132 code is the error or warning code. The default is 0. If status is TRUE, **code** is a nonzero error code. If **status** is FALSE, code is 0 or a warning code.
 - abc **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- DBL **Measured Output y(k)** specifies the measured output of the plant at the current sample time k. You can obtain this output by using sensors to measure the plant output.



Note If you are simulating the MPC controller, you can use the **Discrete State-Space** function to represent the plant.

[DBL] Estimated State xhat(k) specifies the estimated values of the plant states at the current sample time k. Providing state values is optional but improves the accuracy of the MPC Controller.



Note If you do not wire a value to this parameter, you can estimate these states by using the CD Create MPC

Controller VI to define a state estimator. You also can use the Discrete State-Space function to simulate the plant and obtain the actual state values.

- Measured Control Action u(k-1) specifies the measured control action from the previous sample time k-1. This information is useful because the actuator might not be able to implement the Control Action u(k) this VI returns. In this situation, the MPC Controller takes the limitations of the actuator into account.
 - Note If the Control Action u(k) differs from the Measured Control Action u(k–1), saturation might be present in the dynamic system.
- Control Action u(k) returns the control action to apply to the plant at the current sample time k. This control action results in the Estimated Output yhat(k) from the plant.
- MPC Final Conditions returns the final data for the MPC Controller.
 - **Control Action (u)** returns the current values of the control actions.
 - **Output (y)** returns the current values of the plant outputs.
 - **Disturbance (d)** returns the current values of the disturbance.
 - **State (x)** returns the current values of the plant states.
 - **Control Action Change (Du)** returns the current rates at which the control actions are changing.
- **Estimated Output yhat(k)** returns the predicted output of the plant after applying the control action change along the control horizon.
- **Lagrangian Multipliers** returns the coefficients of the lagrangian function that corresponds to the equality and inequality constraints on the MPC controller.

For example, if you specified three equality constraints and two inequality constraints by using the **MPC Optimization Parameters** parameter of the CD Create MPC Controller VI, the first three values of the **Lagrangian Multipliers** parameter correspond to the equality constraints and the last two correspond to the inequality constraints.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Read MPC FIFO VI

Owning Palette: <u>Predictive Control VIs</u>

Installed With: Control Design and Simulation Module

Reads a portion, or window, of profile values from the MPC FIFO.

You wire the **Profile Window** output of this VI to either the **Output Reference Window**, **Control Action Reference Window**, or **Disturbance Window** input of the <u>CD Implement MPC Controller</u> VI. The correct input depends on the value of the **Signal Type** parameter.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the LabVIEW Control Design User Manual for more information about using this VI.



■ Place on the block diagram ■ Find on the **Functions** palette

- **Initial Profile Window** specifies the values this VI uses to initialize the **MPC FIFO**.
- **Initialize?** is TRUE if you want to initialize the **MPC FIFO** by using the values of the **Initial Profile Window** parameter. **Initialize** is FALSE if you do not want to initialize the MPC FIFO. The default value is FALSE.
- MPC Controller In specifies the predictive controller model this VI uses. You create this model by using the <u>CD Create MPC</u> <u>Controller</u> VI.
- MPC FIFO specifies the MPC FIFO this VI uses. You create this MPC FIFO by using the <u>CD Create MPC FIFO</u> VI.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **MPC Controller Out** returns the predictive controller model.
- **Profile Window** returns the requested profile window from the **MPC FIFO**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **MPC FIFO Empty?** returns TRUE if the **MPC FIFO** contains no

elements. This indicator returns FALSE if the **MPC FIFO** contains at least one element.

CD Set MPC Controller VI

Owning Palette: <u>Predictive Control VIs</u>

Installed With: Control Design and Simulation Module

Updates specified parameters of a model predictive control (MPC) controller for a state-space model. You must <u>manually select the</u> <u>polymorphic instance</u> to use.

If you want to update an MPC controller for a transfer function model or a zero-pole-gain model, you must first convert the model to a state-space model.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the <u>LabVIEW Control Design User Manual</u> for information about using this VI.

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

▼

CD Set MPC Controller (Dual)



- **Dd** specifies the direct feedthrough matrix for the disturbance in the system.
- **Bd** specifies the input matrix for the disturbance in the system.
- **Reset** updates the parameters of the controller that you specify.
 - **Reset Option** specifies which parameter(s) of the controller to update.
 - 0 **All**—Updates the entire controller, including the prediction and control horizons, controller model, cost weights, optimization constraints, state estimator, and initial conditions.
 - 1 **Parameters**—Updates the prediction and control horizons, cost weights, optimization constraints, and state estimator of the controller.
 - 2 **Model**—Updates the model used to generate the controller prediction and updates the state estimator of the controller. This model must be state-space and discrete. The model also must have the same number of inputs and outputs as the original model.
 - 3 Weights—Updates the cost weights of the controller.
 - 4 **Constraints**—Updates the optimization constraints of the controller.
 - 5 **Estimator**—Updates the state estimator of the controller.
 - 6 **Initial Data**—Updates the initial conditions at which the controller starts.

Reset? applies the parameter updates to the controller when TRUE.

- MPC Controller In specifies the predictive controller model this VI uses. You create this model by using the <u>CD Create MPC</u> <u>Controller</u> VI.
- **MPC Controller Parameters** specifies execution parameters for the model predictive controller.
 - **Prediction Horizon** specifies the number of future discrete time samples that this VI considers for the output error term in the cost function.
 - **Control Horizon** specifies the number of future discrete time samples that this VI considers when calculating the control action. The value of this parameter must be less than the value of the **Prediction Horizon** parameter.
 - **Initial Window** specifies the initial range, or window, of plant output errors that this VI does not consider for the cost function. National Instruments recommends you specify the value of this parameter to be, at most, one sample less than the minimum delay of the model.

National Instruments recommends that you adjust the value of this parameter when the model has a delay of 10 or more samples. This adjustment reduces the internal calculations of the MPC controller. Because the control action does not affect the plant outputs for the duration of the delays, the initial window adjustment reduces the controller size significantly.

- **Integral Action?** specifies whether the **MPC Controller** includes integral action. Integral action is important when the MPC controller model does not match the plant model.
- **State-Space Model** specifies a discrete state-space representation of the plant you want to control. Providing an accurate model improves the performance of the MPC controller this VI creates.
- **MPC Cost Weights** specifies the weight matrices this VI uses in the cost function.



Note These weight matrices are different from the **Q**, **R**, and **N** parameters that define the state estimator.

Output Error Weightings specifies the weight matrix *Q* this VI uses for each plant output error in the cost function. The dimensions of this matrix must match the number of plant outputs.

To prevent this VI from returning sub-optimal values, this matrix must be positive semi-definite.

Control Action Change Weightings specifies the weight matrix *R* this VI uses for each rate of control action change in the cost function. The dimensions of this matrix must match the number of plant inputs.

To prevent this VI from returning sub-optimal values, this matrix must be positive definite.

Control Action Error Weightings specifies the weight matrix *N* this VI uses for each control action error change in the cost function.

National Instruments recommends that you specify **Control Action Error Weightings** for systems with more inputs than outputs.

Output Error Factors specifies the factors this VI uses to increase or decrease the weights for each plant output error along the **Prediction Horizon** in the cost function. This VI applies these factors at subsequent sample times within the **Prediction Horizon**.

Output Error Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the plant output error at each sample time within the **Prediction Horizon**.

Control Action Change Factors specifies the factors this VI uses to increase or decrease the weights for each rate of control action change along the Control Horizon in the cost function. This VI applies these factors at subsequent sample times within the **Prediction Horizon**.

Control Action Change Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the rate of control action change at each sample time within the **Prediction Horizon**.

Control Action Error Factors specifies the factors this VI uses to increase or decrease the weights for each control action error along the Control Horizon in the cost function. This VI applies these factors at subsequent sample times within the Prediction Horizon.

Control Action Error Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the control action error at each sample time within the **Prediction Horizon**.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **MPC Constraints (Dual)** specifies any constraints on the quadratic programming (QP) optimization algorithm using the dual optimization method. Each parameter specifies a minimum or maximum limit on the initial or final control action *u*, plant output *y*, or rate of change in control action *du*. If you specify only a minimum or a maximum limit for a parameter, LabVIEW assumes a constant profile for that parameter.
 - **u min init** specifies the initial minimum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

u min final specifies the final minimum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

u max init specifies the initial maximum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

u max final specifies the final maximum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

y min init specifies the initial minimum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$. [DBL] **y min final** specifies the final minimum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**. If you do not specify a limit, LabVIEW uses a limit of $-\infty$. [DBL] y max init specifies the initial maximum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**. If you do not specify a limit, LabVIEW uses a limit of ∞ . [DBL] **y max final** specifies the final maximum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**. If you do not specify a limit, LabVIEW uses a limit of ∞ . [DBL] du min init specifies the initial minimum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**. If you do not specify a limit, LabVIEW uses a limit of $-\infty$. [DBL] du min final specifies the final minimum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**. If you do not specify a limit, LabVIEW uses a limit of $-\infty$. [DBL] du max init specifies the initial maximum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**. If you do not specify a limit, LabVIEW uses a limit of ∞ . [DBL] du max final specifies the final maximum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**. If you do not specify a limit, LabVIEW uses a limit of ∞ .

- **stopping criteria** is the collection of conditions that terminate the optimization. If (**function tolerance** AND **parameter tolerance** AND **gradient tolerance**) OR **max iterations** OR **max function calls** then optimization terminates.
 - **function tolerance** is the relative change in function value and is defined as abs(current f prev f)/(abs(curr f)+machine eps). If the relative change in the function value falls below **function tolerance**, the optimization terminates.
 - **parameter tolerance** is the relative change in parameter values and is defined as abs(current p – prev p)/(abs(curr p)+machine eps). If the relative change of all the parameter values falls below **parameters tolerance**, the optimization terminates.
 - **gradient tolerance** is the 2–norm of the gradient. If the 2–norm of the gradient falls below **gradient tolerance**, the optimization terminates.
 - **max iterations** is the largest number of iterations of the major loop of the optimization. If the number of major loop iterations exceeds **max iterations**, the optimization terminates.
 - **max function calls** is the largest number of objective function calls allowed before terminating the optimization process.
 - **max time (sec)** is the maximum amount of time LabVIEW allows between the start and the end of the optimization process. The default is -1. -1 indicates never to time out.
- **MPC State Estimator Parameters** specifies the parameters of the state estimator that the **MPC Controller** uses to estimate the states of the plant.

Use the **Q**, **R**, and **N** parameters to define a state estimator by using the **Noise Covariance** method. This method results in a state estimator that uses the Kalman filter gain.



Note These parameters are different from the **Q**, **R**, and **N** weight matrices in the cost function.

Use the **Observer Poles** parameter to define a state estimator by using the **Pole Placement** method.

 $\overline{\mathbb{N}}$

Defining a state estimator is optional. If you do not define an estimator and can measure the plant states, you can wire the plant states to the **Estimated State xhat(k)** input terminal of the <u>CD Implement MPC Controller</u> VI.

Internal Estimator Gain Calculation specifies the method this VI uses to define the state estimator.

- 0 **None** (default)—Specifies that this VI does not define an internal state estimator. In this situation, you can provide the states to the CD Implement MPC Controller VI.
- 1 **Pole Placement**—Specifies that this VI defines a state estimator by using the value of the **Observer Poles** parameter.

2 **Noise Covariance**—Specifies that this VI defines a state estimator by using the values of the **Q**, **R**, and **N** parameters. This method results in a Kalman filter.

- **Observer Poles** specifies the locations of the estimator poles. This parameter is valid only if you specify **Pole Placement** for the **Internal Estimator Gain Calculation** parameter.
- **Q** specifies the auto-covariance matrix of the process noise vector.
- **N** specifies the cross-covariance matrix between the process noise and the measurement noise vectors.
 - Note If the process noise and measurement noise vectors are uncorrelated, either specify a matrix of zeros for this parameter or do not wire a value to this parameter.
- **R** specifies the auto-covariance matrix of the measurement

noise vector.

- MPC Initial Conditions specifies the initial conditions for the MPC Controller.
 - **Control Action (u)** specifies the initial value(s) of the control action. The default is a vector of zeros.
 - **Output (y)** specifies the initial value(s) of the plant output. The default is a vector of zeros.
 - **Disturbance (d)** specifies the initial value(s) of the disturbance that affects the plant. The default is a vector of zeros.
 - **State (x)** specifies the initial value(s) of the states of the plant. The default is a vector of zeros.
 - **Control Action Change (Du)** specifies the initial values of the rate of change in control action. The default is a vector of zeros.
- **MPC Controller Out** returns the predictive controller model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
CD Set MPC Controller (Barrier)



- **Dd** specifies the direct feedthrough matrix for the disturbance in the system.
- **Bd** specifies the input matrix for the disturbance in the system.
- **Reset** updates the parameters of the controller that you specify.
 - **Reset Option** specifies which parameter(s) of the controller to update.
 - 0 **All**—Updates the entire controller, including the prediction and control horizons, controller model, cost weights, optimization constraints, state estimator, and initial conditions.
 - 1 **Parameters**—Updates the prediction and control horizons, cost weights, optimization constraints, and state estimator of the controller.
 - 2 **Model**—Updates the model used to generate the controller prediction and updates the state estimator of the controller. This model must be state-space and discrete. The model also must have the same number of inputs and outputs as the original model.
 - 3 Weights—Updates the cost weights of the controller.
 - 4 **Constraints**—Updates the optimization constraints of the controller.
 - 5 **Estimator**—Updates the state estimator of the controller.
 - 6 **Initial Data**—Updates the initial conditions at which the controller starts.

Reset? applies the parameter updates to the controller when TRUE.

- MPC Controller In specifies the predictive controller model this VI uses. You create this model by using the <u>CD Create MPC</u> <u>Controller</u> VI.
- **MPC Controller Parameters** specifies execution parameters for the model predictive controller.
 - **Prediction Horizon** specifies the number of future discrete time samples that this VI considers for the output error term in the cost function.
 - **Control Horizon** specifies the number of future discrete time samples that this VI considers when calculating the control action. The value of this parameter must be less than the value of the **Prediction Horizon** parameter.
 - **Initial Window** specifies the initial range, or window, of plant output errors that this VI does not consider for the cost function. National Instruments recommends you specify the value of this parameter to be, at most, one sample less than the minimum delay of the model.

National Instruments recommends that you adjust the value of this parameter when the model has a delay of 10 or more samples. This adjustment reduces the internal calculations of the MPC controller. Because the control action does not affect the plant outputs for the duration of the delays, the initial window adjustment reduces the controller size significantly.

- **Integral Action?** specifies whether the **MPC Controller** includes integral action. Integral action is important when the MPC controller model does not match the plant model.
- **State-Space Model** specifies a discrete state-space representation of the plant you want to control. Providing an accurate model improves the performance of the MPC controller this VI creates.
- **MPC Cost Weights** specifies the weight matrices this VI uses in the cost function.



Note These weight matrices are different from the **Q**, **R**, and **N** parameters that define the state estimator.

Output Error Weightings specifies the weight matrix *Q* this VI uses for each plant output error in the cost function. The dimensions of this matrix must match the number of plant outputs.

To prevent this VI from returning sub-optimal values, this matrix must be positive semi-definite.

Control Action Change Weightings specifies the weight matrix *R* this VI uses for each rate of control action change in the cost function. The dimensions of this matrix must match the number of plant inputs.

To prevent this VI from returning sub-optimal values, this matrix must be positive definite.

Control Action Error Weightings specifies the weight matrix *N* this VI uses for each control action error change in the cost function.

National Instruments recommends that you specify **Control Action Error Weightings** for systems with more inputs than outputs.

Output Error Factors specifies the factors this VI uses to increase or decrease the weights for each plant output error along the **Prediction Horizon** in the cost function. This VI applies these factors at subsequent sample times within the **Prediction Horizon**.

Output Error Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the plant output error at each sample time within the **Prediction Horizon**.

Control Action Change Factors specifies the factors this VI uses to increase or decrease the weights for each rate of control action change along the Control Horizon in the cost function. This VI applies these factors at subsequent sample times within the **Prediction Horizon**.

Control Action Change Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the rate of control action change at each sample time within the **Prediction Horizon**.

Control Action Error Factors specifies the factors this VI uses to increase or decrease the weights for each control action error along the Control Horizon in the cost function. This VI applies these factors at subsequent sample times within the Prediction Horizon.

Control Action Error Factors must be greater than 0. For each element, a value greater than 1 increases the weight. A value less than 1 decreases the weight. The default is 1, which specifies the same weight for the control action error at each sample time within the **Prediction Horizon**.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **MPC Constraints (Barrier)** specifies any constraints on the quadratic programming (QP) optimization algorithm using the barrier function method. Each parameter specifies a minimum limit, maximum limit, tolerance, or penalty on the initial or final control action *u*, plant output *y*, or rate of change in control action *du*. If you specify only a minimum or a maximum limit for a parameter, LabVIEW assumes a constant profile for that parameter.
 - **u min init** specifies the initial minimum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

u min final specifies the final minimum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

- **u min tolerance** specifies the tolerance range for the minimum limit of the control action. The penalty on the cost function becomes active when *u* drops below **u min + u min tolerance**.
- **u min penalty** specifies the penalty on the cost function for the minimum limit of the control action when *u* equals **u min**. The penalty on the cost function becomes active when *u* drops below **u min** + **u min tolerance**.
- **u max init** specifies the initial maximum limit of the control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

u max final specifies the final maximum limit of the control

action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

- **u max tolerance** specifies the tolerance range for the maximum limit of the control action. The penalty on the cost function becomes active when *u* exceeds **u max u max tolerance**.
- **u max penalty** specifies the penalty on the cost function for the maximum limit of the control action when *u* equals **u max**. The penalty on the cost function becomes active when *u* exceeds **u max u max tolerance**.
- **y min init** specifies the initial minimum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

y min final specifies the final minimum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

- **y** min tolerance specifies the tolerance range for the minimum limit of the plant output. The penalty on the cost function becomes active when *y* drops below **y** min + **y** min tolerance.
- **y** min penalty specifies the penalty on the cost function for the minimum limit of the plant output when *y* equals **y** min. The penalty on the cost function becomes active when *y* drops below **y** min + **y** min tolerance.
- **y max init** specifies the initial maximum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

[DBL]

y max final specifies the final maximum limit of the plant output. This VI uses this limit for constraint optimization along the **Prediction Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

- **y max tolerance** specifies the tolerance range for the maximum limit of the plant output. The penalty on the cost function becomes active when *y* exceeds **y max y max tolerance**.
- **y max penalty** specifies the penalty on the cost function for the maximum limit of the plant output when *y* equals **y max**. The penalty on the cost function becomes active when *y* exceeds **y max y max tolerance**.
- **du min init** specifies the initial minimum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

du min final specifies the final minimum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of $-\infty$.

- **du min tolerance** specifies the tolerance range for the minimum limit of the rate of change in control action. The penalty on the cost function becomes active when *du* drops below **du min + du min tolerance**.
- **du min penalty** specifies the penalty on the cost function for the minimum limit of the rate of change in control action when *du* equals **du min**. The penalty on the cost function becomes active when *du* drops below **du min + du min tolerance**.
- **du max init** specifies the initial maximum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

du max final specifies the final maximum limit of the rate of change in control action. This VI uses this limit for constraint optimization along the **Control Horizon**.

If you do not specify a limit, LabVIEW uses a limit of ∞ .

- **du max tolerance** specifies the tolerance range for the maximum limit of the rate of change in control action. The penalty on the cost function becomes active when *du* exceeds **du max du max tolerance**.
- **du max penalty** specifies the penalty on the cost function for the maximum limit of the rate of change in control action when *du* equals **du max**. The penalty on the cost function becomes active when *du* exceeds **du max** – **du max tolerance**.
- **MPC State Estimator Parameters** specifies the parameters of the state estimator that the **MPC Controller** uses to estimate the states of the plant.

Use the **Q**, **R**, and **N** parameters to define a state estimator by using the **Noise Covariance** method. This method results in a state estimator that uses the Kalman filter gain.



Note These parameters are different from the **Q**, **R**, and **N** weight matrices in the cost function.

Use the **Observer Poles** parameter to define a state estimator by using the **Pole Placement** method.

- Defining a state estimator is optional. If you do not define an estimator and can measure the plant states, you can wire the plant states to the **Estimated State xhat(k)** input terminal of the <u>CD Implement MPC Controller</u> VI.
- Internal Estimator Gain Calculation specifies the method this VI uses to define the state estimator.

- 0 **None** (default)—Specifies that this VI does not define an internal state estimator. In this situation, you can provide the states to the CD Implement MPC Controller VI.
- 1 **Pole Placement**—Specifies that this VI defines a state estimator by using the value of the **Observer Poles** parameter.
- 2 Noise Covariance—Specifies that this VI defines a state estimator by using the values of the Q, R, and N parameters. This method results in a Kalman filter.
- **Observer Poles** specifies the locations of the estimator poles. This parameter is valid only if you specify **Pole Placement** for the **Internal Estimator Gain Calculation** parameter.
- **Q** specifies the auto-covariance matrix of the process noise vector.
- **N** specifies the cross-covariance matrix between the process noise and the measurement noise vectors.
 - Note If the process noise and measurement noise vectors are uncorrelated, either specify a matrix of zeros for this parameter or do not wire a value to this parameter.
- **R** specifies the auto-covariance matrix of the measurement noise vector.
- MPC Initial Conditions specifies the initial conditions for the MPC Controller.
 - **Control Action (u)** specifies the initial value(s) of the control action. The default is a vector of zeros.
 - **Output (y)** specifies the initial value(s) of the plant output. The default is a vector of zeros.
 - **Disturbance (d)** specifies the initial value(s) of the disturbance that affects the plant. The default is a vector of zeros.
 - **State (x)** specifies the initial value(s) of the states of the plant. The default is a vector of zeros.

- **Control Action Change (Du)** specifies the initial values of the rate of change in control action. The default is a vector of zeros.
- **MPC Controller Out** returns the predictive controller model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the CDEx Update MPC Controller VI in the labview\examples\Control and Simulation\Control Design\MPC directory for an example of using the CD Set MPC Controller VI.

■ Open example ■ Browse related examples

CD Step Forward MPC Window VI

Owning Palette: Predictive Control VIs

Installed With: Control Design and Simulation Module

Calculates the appropriate portion, or window, of the setpoint and/or disturbance profiles. You wire these windows to the appropriate input(s) of the <u>CD Implement MPC Controller</u> VI.

The CD Step Forward MPC Window VI also moves the control and prediction horizons forward.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the <u>LabVIEW Control Design User Manual</u> for information about using this VI.



■ Place on the block diagram ■ Find on the Functions palette

- **Timestep Index** specifies the sample time *k* at which this VI provides the correct window for the setpoint and/or disturbance profiles.
- **Initialize?** is TRUE if you want to restart the calculation from any initial values you provide. **Initialize** is FALSE if you do not want to restart this calculation. The default value is FALSE.
- MPC Controller In specifies the predictive controller model this VI uses. You create this model by using the <u>CD Create MPC</u> <u>Controller</u> VI.
- **Interleaved Output Setpoint** specifies the interleaved values of the output setpoint profile.
- **Interleaved Disturbance Profile** specifies the interleaved values of the disturbance profile.
- **Interleaved Control Action Setpoint** specifies the interleaved values of the control action profile.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **MPC Controller Out** returns the predictive controller model.
- **Output Reference Window** returns an interleaved window of output setpoints for the current prediction horizon.
- **Disturbance Window** returns an interleaved window of disturbance values for the current prediction horizon.
- **Control Action Reference Window** returns an interleaved window of control action setpoints for the current prediction horizon.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

) TF	status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
132	code is the error or warning code. If status is TRUE, code is a nonzero <u>error code</u> . If status is FALSE, code is 0 or a warning code.
Jabc	source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
Profile Current Values returns the current values of the inputs, outputs, disturbances, states, and rates of change of control action profile.	
(DBL)	Control Action (u) returns the current values of the control actions.
[DBL]	Output (y) returns the current values of the plant outputs.
[DBL]	Disturbance (d) returns the current values of the disturbance.
[DBL]	State (x) returns the current values of the plant states.
[DBL]	Control Action Change (Du) returns the current rates at which the control actions are changing.

CD Update MPC Window VI

Owning Palette: Predictive Control VIs

Installed With: Control Design and Simulation Module

Calculates the appropriate portion, or window, of the setpoint or disturbance profile of a signal from time k to time k + **Prediction Horizon**. You wire these windows to the appropriate input(s) of the <u>CD</u> <u>Implement MPC Controller</u> VI.

•

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Update MPC Window (Single)



- Initialize specifies whether to reset the calculation of the setpoint or disturbance profile. If Initialize is TRUE and you do not specify an Initial Profile Window, this VI sets the setpoint or disturbance value to the Predicted Value(s) for a period of time equivalent to the Prediction Horizon. If Initialize is TRUE and you specify an Initial Profile Window, this VI specifies the Initial Profile Window as the profile of the signal from a time period equivalent to the Prediction Horizon prior to the current time.
- **Predicted Value** specifies the setpoint or disturbance value at time k + **Prediction Horizon**.
- **Initial Profile Window** specifies the profile of the signal for a time period equivalent to the **Prediction Horizon** prior to the current time. This VI uses the **Initial Profile Window** when **Initialize** is TRUE.
- **Prediction Horizon** specifies the prediction horizon to use to calculate the window size of the signal profile.
- **Current Value** returns the setpoint or disturbance value at time *k*.
- **Predicted Profile Window** returns the window of the signal profile from time k to time k + **Prediction Horizon**.

CD Update MPC Window (Multiple)



- Initialize specifies whether to reset the calculation of the setpoint or disturbance profile. If Initialize is TRUE and you do not specify an Initial Profile Window, this VI sets the setpoint or disturbance value to the Predicted Value(s) for a period of time equivalent to the Prediction Horizon. If Initialize is TRUE and you specify an Initial Profile Window, this VI specifies the Initial Profile Window as the profile of the signal from a time period equivalent to the Prediction Horizon prior to the current time.
- **Predicted Values** specifies the interleaved values of the setpoint or disturbance profile at time k + **Prediction Horizon**.
- Initial Profile Window specifies the profile of the signal for a time period equivalent to the Prediction Horizon prior to the current time. This VI uses the Initial Profile Window when Initialize is TRUE.
- **Prediction Horizon** specifies the prediction horizon to use to calculate the window size of the signal profile.
- **Current Values** returns the interleaved values of the setpoint or disturbance profile at time *k*.
- **Predicted Profile Window** returns the window of the signal profile from time k to time k + **Prediction Horizon**.

Example

Refer to the CDEx Update MPC Controller VI in the labview\examples\Control and Simulation\Control Design\MPC directory for an example of using the CD Update MPC Window VI.

■ Open example ■ Browse related examples

CD Write MPC FIFO VI

Owning Palette: Predictive Control VIs

Installed With: Control Design and Simulation Module

Writes a control action setpoint, output setpoint, or disturbance profile window to the **MPC FIFO**. You then use the <u>CD Read MPC FIFO</u> VI to read values from this MPC FIFO.



Note Refer to Chapter 18, *Creating and Implementing a Model Predictive Controller*, of the <u>LabVIEW Control Design User Manual</u> for information about using this VI.



■ Place on the block diagram ■ Find on the **Functions** palette

- MPC Controller specifies the predictive controller model this VI uses. You create this model by using the <u>CD Create MPC</u> <u>Controller</u> VI.
- MPC FIFO specifies the MPC FIFO this VI uses. You create this MPC FIFO by using the <u>CD Create MPC FIFO</u> VI.
- **Interleaved Profile** specifies the values of the control action setpoint, output setpoint, or disturbance profile window to write to the **MPC FIFO**.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- timeout (n*Sample Time) specifies the time, in number of samples, to wait for an empty slot in the MPC FIFO. This VI multiplies the timeout value *n* by the sample time. The default value is 1.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Solvers VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Solvers VIs to compute the solutions to the continuous and discrete algebraic Riccati equations, the continuous and discrete Lyapunov equations, and integrals involving matrix exponentials.

Palette Object	Description
<u>CD Calculate</u> Integrals with <u>Matrix</u> Exponential	Calculates integrals involving matrix exponentials. These integrals are called Van Loan integrals.
<u>CD Continuous</u> <u>Algebraic Riccati</u> <u>Equations</u>	Calculates the symmetric positive definite, or stabilizing, matrix X that solves the following continuous algebraic Riccati equation (CARE).
<u>CD Continuous</u> Lyapunov Equations	Calculates the matrix X that solves the following continuous Lyapunov equation:
<u>CD Discrete</u> <u>Algebraic Riccati</u> <u>Equations</u>	Calculates the symmetric positive definite, or stabilizing, matrix X that solves the following discrete algebraic Riccati equation (DARE).
<u>CD Discrete</u> Lyapunov Equations	Calculates the matrix X that solves the following discrete Lyapunov equation:

CD Calculate Integrals with Matrix Exponential VI

Owning Palette: Solvers VIs

Installed With: Control Design and Simulation Module

Calculates integrals involving matrix exponentials. These integrals are called Van Loan integrals.

<u>Details</u>



Place on the block diagram Find on the Functions palette

- **Ac** specifies an $n \ge n$ state matrix, where n is the number of states.
- **Bc** specifies an *n* x *m* input matrix, where *m* is the number of inputs. The default value is the transpose of **A**.
- **Qc** specifies the state weight matrix. **Q** must be symmetric and positive semi-definite.
- **Rc** specifies the input weight matrix. **R** must be symmetric and positive definite. The default is an identity matrix of appropriate dimensions.
- **Nc** specifies the state-input cross weight matrix. The value of **Nc** must be such that the matrix (**Qc**–**Nc***inv(**Rc**)***Nc**) is positive semidefinite. The default value of **Nc** is an empty matrix of appropriate dimensions.
- **Ts** specifies the upper limit of the integral of the matrix this VI calculates. Refer to the <u>Details</u> section for the matrix this VI calculates.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Qd** returns the 1 x 1 block of the matrix this VI calculates. Refer to the <u>Details</u> section for the matrix this VI calculates.
- **Rd** returns the 2 x 2 block of the matrix this VI calculates. Refer to the <u>Details</u> section for the matrix this VI calculates.
- **Nd** returns the 1 x 2 block of the matrix this VI calculates. Refer to the <u>Details</u> section for the matrix this VI calculates.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning.

CD Calculate Integrals with Matrix Exponential Details

This VI calculates the outputs according to the following equations:

$$\begin{bmatrix} \mathbf{Q}_{d} & \mathbf{N}_{d} \\ \mathbf{N}_{d} & \mathbf{R}_{d} \end{bmatrix} = \int_{0}^{\mathsf{Ts}} \begin{bmatrix} \mathbf{A}_{d}^{\mathsf{T}}(\tau) & \mathbf{0} \\ \mathbf{B}_{d}^{\mathsf{T}}(\tau) & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{Q}_{c} & \mathbf{N}_{c} \\ \mathbf{N}_{c} & \mathbf{R}_{c} \end{bmatrix} \begin{bmatrix} \mathbf{A}_{d}(\tau) & \mathbf{B}_{d}(\tau) \\ \mathbf{0} & \mathbf{I} \end{bmatrix} d^{d}\tau$$
$$= \int_{0}^{\mathsf{Ts}} \begin{bmatrix} \mathbf{A}_{d}^{\mathsf{T}}(\tau) \mathbf{Q}_{c} \mathbf{A}_{d}(\tau) & \mathbf{A}_{d}^{\mathsf{T}}(\tau) \mathbf{Q}_{c} \mathbf{B}_{d} + \mathbf{A}_{d}^{\mathsf{T}}(\tau) \mathbf{N}_{c} \\ \mathbf{B}_{d}^{\mathsf{T}}(\tau) \mathbf{Q}_{c} \mathbf{A}_{d} + \mathbf{N}_{c}^{\mathsf{T}} \mathbf{A}_{d}(\tau) & \mathbf{B}_{d}^{\mathsf{T}}(\tau) \mathbf{N}_{c} \mathbf{B}_{d} + \mathbf{R}_{c} \end{bmatrix} d^{d}\tau$$

where

$$A_{d}(\tau) = e^{A_{c}\tau}$$

$$B_{d}(\tau) = \int_{0}^{\tau} e^{A_{c}\eta} d\eta \cdot B_{c}$$

$$Q_{c} = Q_{c}^{\top} \ge 0$$

$$R_{c} = R_{c}^{\top} \ge 0$$

$$Q_{c} - N_{c}R_{c}^{-1}N_{c}^{\top} \ge 0$$

If you interpret the inputs to this VI as describing information about a continuous system model, you can interpret the outputs of this VI as discretized versions of these inputs, where **Ts** is the sampling time.

For example, if you make the following assumptions:

- Ac represents the continuous system matrix that describes the states of the system.
- Bc represents the continuous input matrix that relates the inputs to the states.
- **Qc** represents the continuous cost matrix penalizing the system states. This matrix also can represent the continuous process noise spectral intensity matrix.
- **Rc** represents the continuous cost matrix penalizing the system inputs. This matrix also can represent the continuous measurement noise spectral intensity matrix.
- **Nc** represents the continuous cost matrix penalizing the cross product between the system states and the system inputs. This matrix also can represent the continuous cross-spectral intensity matrix between the process noise and the measurement noise.
- **Ts** represents the sampling time this VI uses to discretize the continuous matrices **Ac**, **Bc**, **Qc**, **Rc**, and **Nc**.

then the following statements are true:

- **Qd** is the discrete equivalent of **Qc**.
- **Rd** is the discrete equivalent of **Rc**.
- Nd is the discrete equivalent of Nc.

CD Continuous Algebraic Riccati Equations VI

Owning Palette: <u>Solvers VIs</u>

Installed With: Control Design and Simulation Module

Calculates the symmetric positive definite, or stabilizing, matrix **X** that solves the following continuous algebraic Riccati equation (CARE).

$\mathbf{A}'.\mathbf{X} + \mathbf{X}.\mathbf{A} - [\mathbf{X}.\mathbf{B} + \mathbf{N}].inv(\mathbf{R}).[\mathbf{X}.\mathbf{B} + \mathbf{N}]' + \mathbf{Q} = 0$



■ Place on the block diagram ■ Find on the **Functions** palette

- **A** specifies an $n \times n$ state matrix, where *n* is the number of states.
- **B** specifies an $n \ge m$ input matrix, where m is the number of inputs.
- **Q** specifies the state weight matrix. **Q** must be symmetric and positive semi-definite.
- **R** specifies the input weight matrix. **R** must be symmetric and positive definite. The default is an identity matrix of appropriate dimensions.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **N** specifies the state-input cross weight matrix. The value of **N** must be such that the matrix (**Q**–**N***inv(**R**)***N**) is positive semidefinite. The default value of **N** is an empty matrix of appropriate dimensions.
- **Riccati Solution (X)** returns the solution to the algebraic Riccati equation.
- **Gain (K)** returns the gain matrix **K** such that $\mathbf{K} = inv(\mathbf{R}).(\mathbf{B}'\mathbf{X}+\mathbf{N}')$.
- **Closed-Loop Eigenvalues** returns the eigenvalues of the matrix (**A**–**BK**). These eigenvalues are the closed-loop pole locations.
- **Relative Residual** returns the relative residual of the **Riccati Solution (X)**. The **Relative Residual** is defined by the following equation:

Relative Residual = ||Residual|| / ||X||

where ||Residual|| is the is the Frobenius norm of the continuous algebraic Riccati equation achieved by substituting the solution X into (A'.X + X.A–[X.B+N].inv(R).[X.B+N]' + Q)

||X|| is the Frobenius norm of the Riccati Solution (X)

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

132

code is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Continuous Lyapunov Equations VI

Owning Palette: <u>Solvers VIs</u>

Installed With: Control Design and Simulation Module

Calculates the matrix ${\bf X}$ that solves the following continuous Lyapunov equation:

AX+XB = -Q



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

- **A** specifies an $n \times n$ state matrix, where n is the number of states.
- **B** specifies an $n \ge m$ input matrix, where m is the number of inputs. The default value is the transpose of **A**.
- **Q** specifies the state weight matrix. **Q** must be symmetric and positive semi-definite.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Lyapunov Solution (X)** returns the solution to the Lyapunov equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Discrete Algebraic Riccati Equations VI

Owning Palette: <u>Solvers VIs</u>

Installed With: Control Design and Simulation Module

Calculates the symmetric positive definite, or stabilizing, matrix **X** that solves the following discrete algebraic Riccati equation (DARE).

X = A'XA - [A'XB+N]*inv(B'XB+R)*[A'XB+N]'+Q



■ Place on the block diagram ■ Find on the **Functions** palette

- **A** specifies an $n \times n$ state matrix, where n is the number of states.
- **B** specifies an $n \ge m$ input matrix, where m is the number of inputs.
- **Q** specifies the state weight matrix. **Q** must be symmetric and positive semi-definite.
- **R** specifies the input weight matrix. **R** must be symmetric and positive definite. The default is an identity matrix of appropriate dimensions.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **N** specifies the state-input cross weight matrix. The value of **N** must be such that the matrix (**Q**–**N***inv(**R**)***N**) is positive semidefinite. The default value of **N** is an empty matrix of appropriate dimensions.
- **Riccati Solution (X)** returns the solution to the algebraic Riccati equation.
- **Gain (K)** returns the gain matrix **K** such that $\mathbf{K} = inv(\mathbf{R}).(\mathbf{B}'\mathbf{X}+\mathbf{N}')$.
- **Closed-Loop Eigenvalues** returns the eigenvalues of the matrix (**A**–**BK**). These eigenvalues are the closed-loop pole locations.
- **Relative Residual** returns the relative residual of the **Riccati Solution (X)**. The **Relative Residual** is defined by the following equation:

Relative Residual = ||Residual|| / ||X||

where ||Residual|| is the is the Frobenius norm of the discrete algebraic Riccati equation achieved by substituting the solution X into X = A'XA–[A'XB+N]*inv(B'XB+R)* [A'XB+N]'+Q

||X|| is the Frobenius norm of the Riccati Solution (X)

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Discrete Lyapunov Equations VI

Owning Palette: Solvers VIs

Installed With: Control Design and Simulation Module

Calculates the matrix ${\bf X}$ that solves the following discrete Lyapunov equation:

AXA'-X+Q = -Q



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

- **A** specifies an $n \times n$ state matrix, where *n* is the number of states.
- **Q** specifies the state weight matrix. **Q** must be symmetric and positive semi-definite.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **Lyapunov Solution (X)** returns the solution to the Lyapunov equation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
State Feedback Design VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the State Feedback Design VIs to calculate controller and observer gains for closed-loop state feedback control or to estimate a state-space model. You also can use State Feedback Design VIs to configure and test state-space controllers and state estimators in time domains.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD</u> <u>Ackermann</u>	Uses the Ackermann formula with the controllability matrix to determine the controller feedback gain matrix <i>K</i> that places the closed-loop poles in the locations you specify. You also can use this VI to determine the observer gain matrix <i>L</i> that places the observer poles in the locations you specify.
<u>CD</u> <u>Augment</u> <u>Output with</u> <u>States</u>	Augments a space-state model by appending states to the outputs.
<u>CD</u> <u>Kalman</u> <u>Gain</u>	Calculates the optimal steady-state Kalman gain <i>L</i> that minimizes the covariance of the estimation error for a continuous or discrete model affected by noise. You can use this VI to calculate the Kalman gain for a stochastic or deterministic model. You also can use this VI to discretize automatically a continuous stochastic or continuous deterministic model before calculating <i>L</i> . You must manually <u>select the polymorphic instance</u> you want to use.
<u>CD Linear</u> Quadratic Regulator	Calculates the optimal steady-state feedback gain matrix <i>K</i> that minimizes a linear quadratic cost function you specify. You can use this VI to calculate <i>K</i> for a continuous or discrete state-space model. You also can use this VI to

	calculate automatically a discrete <i>K</i> for a continuous state- space model. This calculation uses a specified sampling time and an associated continuous cost function. You must <u>manually select the polymorphic instance</u> to use.
<u>CD Pole</u> <u>Placement</u>	Determines the Gain that places the closed-loop poles at desired locations in a system with full state feedback. You can use this VI with multiple-input multiple-output (MIMO) systems. However, if you have a single-input single-output system, use the <u>CD Ackermann</u> VI.
<u>CD State-</u> <u>Space</u> <u>Controller</u>	Builds a state-space controller based on known inputs, manipulated inputs, measured outputs, linear state-space model, estimator gain, and controller gain. You must <u>manually select the polymorphic instance</u> you want to use.
<u>CD State</u> <u>Estimator</u>	Builds the state estimator based on a list of known inputs, measured outputs, the linear state-space model, and the estimator gain. The data type you wire to the Estimator Gain (L) input determines the polymorphic instance to use.

CD Ackermann VI

Owning Palette: <u>State Feedback Design VIs</u>

Installed With: Control Design and Simulation Module

Uses the Ackermann formula with the controllability matrix to determine the controller feedback gain matrix K that places the closed-loop poles in the locations you specify. You also can use this VI to determine the observer gain matrix L that places the observer poles in the locations you specify.

You can use the CD Ackermann VI with single-input multiple-output (SIMO) or multiple-input single-output (MISO) systems when placing closed-loop control or observer poles, respectively.

Details



Place on the block diagram Find on the Functions palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI places closedloop poles in desired locations using the **Gain** in the state feedback. You also can use <u>pole placement</u> to calculate the estimator gain and set the pole locations of the full state observer.
- **Poles** specifies the closed-loop pole locations.
- Gain Type specifies the type of gain this VI returns in the Gain parameter. Use the Controller Gain with the <u>CD State-Space</u> <u>Controller</u> VI. Use the Observer Gain (Predictive) and Observer Gain (Current) with the <u>CD Discrete Observer</u> function and the <u>CD Continuous Observer</u> function.

For discrete systems, the **Observer Gain (Predictive)** relates to the **Observer Gain (Current)** through the following relationship:

Observer Gain (Predictive) = A. **Observer Gain (Current)**.

For continuous systems, the notion of current and predictive time

does not apply, so specifying either observer gain returns the correct gain value.

	controllability matrix must be non-singular.
	must be a single-input system and controllable, meaning the
	locations you specify using the Poles parameter. The system
	controller, such that the controller pole locations are the
С	Controller Gain (default)—Calculates feedback gain <i>K</i> for a

- **Observer Gain (Predictive)**—Calculates feedback gain *Lp* for a predictive observer, such that the observer pole locations are the locations you specify using the **Poles** parameter. The system must be a single-output system and observable, meaning the observability matrix must be non-singular.
- 2 **Observer Gain (Current)**—Calculates feedback gain *Lc* for a current observer, such that the observer pole locations are the locations you specify using the **Poles** parameter. The system must be a single-output system and observable, meaning the observability matrix must be non-singular. For continuous systems, the notion of current and predictive time does not apply; either observer gain returns the correct gain value. Also, if the system is discrete, *A* must be invertible.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Gain** returns the feedback gain, according to the value you specify for the **Gain Type** parameter, that produces a closed-loop system such that the poles are equal to the values you specify in the **Poles** parameter.
- Actual Poles are the eigenvalues of the closed-loop system matrix $ilde{A}$. The definition of $ilde{A}$ depends on the value you specify for the Gain Type parameter.

If Gain Type is Controller Gain, $\tilde{A} = A - BK$.

If Gain Type is Observer Gain (Predictive), $\tilde{A} = A - LC$, where L is the Gain this VI returns.

If Gain Type is Observer Gain (Current), $\tilde{A} = A - LCA$.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Ackermann Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller or state estimator, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Augment Output with States VI

Owning Palette: <u>State Feedback Design VIs</u>

Installed With: Control Design and Simulation Module

Augments a space-state model by appending states to the outputs.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system that this VI augments.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Augmented Model** returns the augmented state-space model.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Augment Output with States Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD Convert Delay with Pade</u> Approximation VI (continuous models) or the <u>CD Convert Delay to Poles</u> at Origin VI (discrete models). Refer to the <u>LabVIEW Control Design User</u> Manual for more information about delays and the limitations of Pade Approximation.

CD Kalman Gain VI

Owning Palette: <u>State Feedback Design VIs</u>

Installed With: Control Design and Simulation Module

Calculates the optimal steady-state Kalman gain L that minimizes the covariance of the estimation error for a continuous or discrete model affected by noise. You can use this VI to calculate the Kalman gain for a stochastic or deterministic model. You also can use this VI to discretize automatically a continuous stochastic or continuous deterministic model before calculating L. You must manually select the polymorphic instance you want to use.

▾

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Kalman Gain (Stochastic)



- **Stochastic State-Space Model In** specifies a <u>mathematical</u> <u>representation</u> of a stochastic system.
- Second-Order Statistics Noise Model specifies a <u>mathematical</u> representation of the noise model of the Stochastic State-Space Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Closed-Loop Eigenvalues** returns the poles of the Kalman filter when using the **Steady-State Kalman Gain (L)**.

- Steady-State Kalman Gain (L) returns the optimal gain matrix L that minimizes the Steady-State Estimation Error Covariance (P) for the given model and noise.
- **Steady-State Estimation Error Covariance (P)** returns the covariance of the estimation error.
- Steady-State Innovation Gain (M) returns the optimal gain matrix M that minimizes the Steady-State Error Covariance of Updated Estimate (Z) in the discrete estimation process.
- **Steady-State Error Covariance of Updated Estimate (Z)** returns the covariance of the error between the actual states and the updated state estimates in the discrete estimation process.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Kalman Gain (Deterministic)



- **G** specifies the matrix that relates the process noise vector to the model states.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates optimal estimator gain.
- \bigcirc **Q** specifies the auto-covariance matrix of **w**.
- **R** specifies the auto-covariance matrix of v.
- **N** specifies the cross-covariance matrix between the process noise vector *w* and the measurement noise vector *v*. If *w* and *v* are uncorrelated, **N** is a matrix of zeros.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **H** specifies the matrix that relates the process noise vector to the model outputs.
- **Closed-Loop Eigenvalues** returns the poles of the Kalman filter when using the **Steady-State Kalman Gain (L)**.
- Steady-State Kalman Gain (L) returns the optimal gain matrix L that minimizes the Steady-State Estimation Error Covariance (P) for the given model and noise.
- **Steady-State Estimation Error Covariance (P)** returns the covariance of the estimation error.
- Steady-State Innovation Gain (M) returns the optimal gain matrix M that minimizes the Steady-State Error Covariance of Updated Estimate (Z) in the discrete estimation process.
- **Steady-State Error Covariance of Updated Estimate (Z)** returns the covariance of the error between the actual states and the updated state estimates in the discrete estimation process.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Discretized Kalman Gain (Stochastic)



- **Continuous Stochastic State-Space Model** specifies a <u>mathematical representation</u> of a continuous stochastic system.
- Continuous Second-Order Statistics Noise Model specifies a mathematical representation of the continuous noise model of the Continuous Stochastic State-Space Model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Sampling Time (s)** specifies the sampling time this VI uses to discretize the continuous model(s) before calculating the estimator gain.

- **Closed-Loop Eigenvalues** returns the poles of the Kalman filter when using the **Steady-State Kalman Gain (L)**.
- Steady-State Kalman Gain (L) returns the optimal gain matrix L that minimizes the Steady-State Estimation Error Covariance (P) for the given model and noise.
- **Steady-State Estimation Error Covariance (P)** returns the covariance of the estimation error.
- Steady-State Innovation Gain (M) returns the optimal gain matrix M that minimizes the Steady-State Error Covariance of Updated Estimate (Z) in the discrete estimation process.
- **Steady-State Error Covariance of Updated Estimate (Z)** returns the covariance of the error between the actual states and the updated state estimates in the discrete estimation process.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Discretized Kalman Gain (Deterministic)



- **G** specifies the matrix that relates the process noise vector to the model states.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates optimal estimator gain.
- **Q** specifies the auto-covariance matrix of **w**.
- **R** specifies the auto-covariance matrix of v.
- **N** specifies the cross-covariance matrix between the process noise vector *w* and the measurement noise vector *v*. If *w* and *v* are uncorrelated, **N** is a matrix of zeros.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **H** specifies the matrix that relates the process noise vector to the model outputs.
- **Sampling Time (s)** specifies the sampling time this VI uses to discretize the continuous model(s) before calculating the estimator gain.
- **Closed-Loop Eigenvalues** returns the poles of the Kalman filter when using the **Steady-State Kalman Gain (L)**.
- Steady-State Kalman Gain (L) returns the optimal gain matrix L that minimizes the Steady-State Estimation Error Covariance (P) for the given model and noise.
- **Steady-State Estimation Error Covariance (P)** returns the covariance of the estimation error.
- Steady-State Innovation Gain (M) returns the optimal gain matrix M that minimizes the Steady-State Error Covariance of Updated Estimate (Z) in the discrete estimation process.
- **Steady-State Error Covariance of Updated Estimate (Z)** returns the covariance of the error between the actual states and the updated state estimates in the discrete estimation process.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning.

CD Kalman Gain Details

Continuous Stochastic and Deterministic Models

For continuous models, the estimation error is defined as $e(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t)$. The **Steady-State Estimation Error Covariance (P)** is calculated as

 $\boldsymbol{P} = \lim_{t \to \infty} \mathbb{E}\{\boldsymbol{e}(t) \cdot \boldsymbol{e}^{\mathsf{T}}(t)\}$

The Kalman filter minimizes this quantity. This VI computes the **Steady-State Kalman Gain (L)** to apply to the Kalman filter by solving the continuous algebraic Riccati equation expressed in terms of P. This expression is defined as

 $AP + PA^{\mathsf{T}} + GQG^{\mathsf{T}} - PC^{\mathsf{T}}[HQH^{\mathsf{T}} + HN + N^{\mathsf{T}}H^{\mathsf{T}} + R]^{-1} \cdot CP$ $-[GQH^{\mathsf{T}} + GN][HQH^{\mathsf{T}} + HN + N^{\mathsf{T}}H^{\mathsf{T}} + R]^{-1}[GQH^{\mathsf{T}} + GN]^{\mathsf{T}} = 0$

The Steady-State Kalman Gain (L) is calculated as

 $L = [PC^{\mathsf{T}} + GQH^{\mathsf{T}} + GN][HQH^{\mathsf{T}} + HN + N^{\mathsf{T}}H^{\mathsf{T}} + R]^{-1}$

The following equations define the noise covariance matrices Q, R, and N.

```
\begin{aligned} \boldsymbol{Q}(t)\delta(t-\tau) &= \mathsf{E}\{\boldsymbol{w}(t)\cdot\boldsymbol{w}^{\mathsf{T}}(\tau)\} - \mathsf{E}\{\boldsymbol{w}(t)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{w}(\tau)\}\\ \boldsymbol{R}(t)\delta(t-\tau) &= \mathsf{E}\{\boldsymbol{v}(t)\cdot\boldsymbol{v}^{\mathsf{T}}(\tau)\} - \mathsf{E}\{\boldsymbol{v}(t)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{v}(\tau)\}\\ \boldsymbol{N}(t)\delta(t-\tau) &= \mathsf{E}\{\boldsymbol{w}(t)\cdot\boldsymbol{v}^{\mathsf{T}}(\tau)\} - \mathsf{E}\{\boldsymbol{w}(t)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{v}(\tau)\}\end{aligned}
```

Discrete Stochastic and Deterministic Models

For discrete models, the prediction estimation error is defined as $e(k+1|k) = \mathbf{x}(k) - \hat{\mathbf{x}}(k+1|k)$. The **Steady-State Estimation Error Covariance** (P) is calculated as

 $\boldsymbol{P} = \lim_{k \to \infty} \mathbb{E} \{ \boldsymbol{e}(k+1|k) \cdot \boldsymbol{e}^{\mathsf{T}}(k+1|k) \}$

The Kalman filter minimizes this quantity. This VI computes the **Steady-State Kalman Gain (L)** to apply to the Kalman filter by solving the discrete algebraic Riccati equation expressed in terms of P. This expression is defined as

 $P = APA^{\mathsf{T}} + GQG^{\mathsf{T}} - L[APC^{\mathsf{T}} + GQH^{\mathsf{T}} + GN]^{\mathsf{T}}$

This VI calculates the Steady-State Kalman Gain (L) as

```
L = [APC^{\mathsf{T}} + GQH^{\mathsf{T}} + GN][CPC^{\mathsf{T}} + HQH^{\mathsf{T}} + HN + N^{\mathsf{T}}H^{\mathsf{T}} + R]^{-1}
```

The following equations define the noise covariance matrices Q, R, and N.

 $\begin{aligned} \boldsymbol{Q}(k)\delta_{kl} &= \mathsf{E}\{\boldsymbol{w}(k)\cdot\boldsymbol{w}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{w}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{w}(l)\} \\ \boldsymbol{R}(k)\delta_{kl} &= \mathsf{E}\{\boldsymbol{v}(k)\cdot\boldsymbol{v}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{v}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{v}(l)\} \\ \boldsymbol{N}(k)\delta_{kl} &= \mathsf{E}\{\boldsymbol{w}(k)\cdot\boldsymbol{v}^{\mathsf{T}}(l)\} - \mathsf{E}\{\boldsymbol{w}(k)\}\cdot\mathsf{E}^{\mathsf{T}}\{\boldsymbol{v}(l)\} \end{aligned}$

This VI calculates the Steady-State Innovation Gain (M) as

 $\boldsymbol{M} = \boldsymbol{P}\boldsymbol{C}^{\mathsf{T}} [\boldsymbol{C}\boldsymbol{P}\boldsymbol{C}^{\mathsf{T}} + \boldsymbol{H}\boldsymbol{Q}\boldsymbol{H}^{\mathsf{T}} + \boldsymbol{H}\boldsymbol{N} + \boldsymbol{N}^{\mathsf{T}}\boldsymbol{H}^{\mathsf{T}} + \boldsymbol{R}]^{-1}$

The updated estimation error is defined as $e(k|k) = \mathbf{x}(k) - \hat{\mathbf{x}}(k|k)$. The **Steady-State Error Covariance of Updated Estimate (Z)** is defined as $\mathbf{Z} = \mathbf{P} - \mathbf{MCP}$.

Matrix Restrictions

For both continuous and discrete models, which either are stochastic or deterministic, Q, R, and N must satisfy the following conditions:

- **Q** is a symmetric, positive semi-definite matrix
- **R** is a symmetric, positive definite matrix
- **N** satisfies the following relationship:

$$\begin{bmatrix} \boldsymbol{Q} & \boldsymbol{N} \\ \boldsymbol{N}^{\mathsf{T}} & \boldsymbol{R} \end{bmatrix} \geq \boldsymbol{0}$$

The following conditions also must be satisfied.

 $\overline{\boldsymbol{Q}} = \overline{\boldsymbol{Q}}^{\mathsf{T}} \ge \boldsymbol{0}$ $\overline{\boldsymbol{R}} = \overline{\boldsymbol{R}}^{\mathsf{T}} \ge \boldsymbol{0}$

The matrix \overline{R} must satisfy the following relationship:

$$\begin{bmatrix} \bar{\boldsymbol{Q}} & \bar{\boldsymbol{N}} \\ \bar{\boldsymbol{N}}^{\mathsf{T}} & \bar{\boldsymbol{R}} \end{bmatrix} \ge 0$$

The pair (\bar{c}_1, \bar{A}) is detectable, and the pair (\bar{A}, \bar{B}_1) is stabilizable.

where the following definitions apply:

$$\bar{A} = A^{\mathsf{T}} - C^{\mathsf{T}} \bar{R}^{-1} \bar{N}^{\mathsf{T}}$$
$$\bar{Q} = G Q G^{\mathsf{T}}$$
$$\bar{R} = R + H Q H^{\mathsf{T}} + H N + N H^{\mathsf{T}}$$
$$\bar{N} = G N + G Q H^{\mathsf{T}}$$

The matrix \bar{c}_1 is the full-rank factorization of the matrix $\Gamma = \bar{c}^T R^{-1} c$, such that $\Gamma = \bar{c}_1^T \bar{c}_1$ and $\operatorname{rank}[\Gamma] = \operatorname{rank}[\bar{c}_1]$.

The matrix \bar{B}_1 is the full-rank factorization of the matrix $\Phi = Q - \bar{N}\bar{R}^{-1}\bar{N}^{T}$, such that $\Phi = \bar{B}_1\bar{B}_1^{T}$ and $\operatorname{rank}[\Phi] = \operatorname{rank}[\bar{B}_1]$.

Discretized Kalman Gain

These instances convert the *A*, *B*, *C*, and *D* matrices using the numerical integration method as proposed by Van Loan. Refer to the following sources for more information about this method.

- G.F. Franklin, J.D. Powell, and M. Workman, *Digital Control of Dynamic Systems*, 3rd ed. Menlo Park, CA: Addison Wesley, 1997.
- C.F.V. Loan, "Computing integrals involving the matrix exponential," *IEEE Transactions on Automatic Control*, vol. 23, no. 3, pp. 395–404, June 1978.

Delay Support

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Linear Quadratic Regulator VI

Owning Palette: <u>State Feedback Design VIs</u>

Installed With: Control Design and Simulation Module

Calculates the optimal steady-state feedback gain matrix *K* that minimizes a linear quadratic cost function you specify. You can use this VI to calculate *K* for a continuous or discrete state-space model. You also can use this VI to calculate automatically a discrete *K* for a continuous state-space model. This calculation uses a specified sampling time and an associated continuous cost function. You must <u>manually select the</u> <u>polymorphic instance</u> to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Linear Quadratic Regulator (LQR)



- Weighting Type determines the type of cost function this VI minimizes. Refer to the <u>Details</u> section for the cost function equations.
 - 0 **state weighting** (default)—Selects a cost function that weights the model states.
 - 1 **output weighting, Dimension Q =** Ny—Selects a cost function that weights the model outputs, where Q is in terms of the model outputs. If you select this option, Q must be an $r \times r$ matrix, where r is the number of model outputs.

2 **output weighting, Dimension Q=** Nx—Selects a cost function that weights the model outputs, where where Q is in terms of the model states. If you select this option, Q must be an $n \times n$ matrix, where n is the number of model states.

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI calculates state feedback optimal gain.
- **Q** specifies a symmetric, positive semi-definite matrix that penalizes the state vector **x** in the cost function. If you specify a cost function that weights the model outputs, **Q** penalizes the output vector **y** in the cost function.
- **R** specifies a symmetric, positive definite matrix that penalizes the input vector \boldsymbol{u} in the cost function.
- **N** specifies a matrix of appropriate dimensions that penalizes the cross-product between the input and state vectors. If you do not wire a value to this parameter, this VI sets **N** to an appropriately-sized matrix of zeros.
- **error in** describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Optimal Gain (K)** is the resulting state feedback gain matrix **K** that minimizes the cost function.
- **Eigenvalues** are the poles of the resulting closed-loop system. The poles are equal to the eigenvalues of **A**–**BK**.
- **Riccati Solution (X)** returns the symmetric, positive semi-definite solution *X* to the continuous or discrete algebraic Riccati equation, depending whether the model you wire to the **State-Space Model** input of this VI is continuous or discrete. However, the **Discretized LQR** instance of this VI always returns the solution to the discrete algebraic Riccati equation. Refer to the **Details** section for the definitions of the Riccati equations.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Linear Quadratic Regulator (Discretized LQR)



- Weighting Type determines the type of cost function this VI minimizes. Refer to the <u>Details</u> section for the cost function equations.
 - 0 **state weighting** (default)—Selects a cost function that weights the model states.
 - 1 **output weighting, Dimension Q =** Ny—Selects a cost function that weights the model outputs, where Q is in terms of the model outputs. If you select this option, Q must be an $r \times r$ matrix, where r is the number of model outputs.

2 **output weighting, Dimension Q=** Nx—Selects a cost function that weights the model outputs, where where Q is in terms of the model states. If you select this option, Q must be an $n \times n$ matrix, where n is the number of model states.

- **Continuous State-Space Model** specifies the continuous statespace model for which this VI designs a discrete-time LQR controller.
- **Q** specifies a symmetric, positive semi-definite matrix that penalizes the state vector **x** in the cost function. If you specify a cost function that weights the model outputs, **Q** penalizes the output vector **y** in the cost function.
- **R** specifies a symmetric, positive definite matrix that penalizes the input vector **u** in the cost function.
- **N** specifies a matrix of appropriate dimensions that penalizes the cross-product between the input and state vectors. If you do not wire a value to this parameter, this VI sets **N** to an appropriately-sized matrix of zeros.
- **error in** describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Sampling Time specifies the discrete sampling time this VI uses to discretize the continuous state-space model and cost function. The resulting discrete **Optimal Gain (K)** is associated with this sampling time. The default value is 1 second.
- **Optimal Gain (K)** is the resulting state feedback gain matrix **K** that minimizes the cost function.
- **Eigenvalues** are the poles of the resulting closed-loop system. The poles are equal to the eigenvalues of **A**–**BK**.
- **Riccati Solution (X)** returns the symmetric, positive semi-definite solution *X* to the continuous or discrete algebraic Riccati equation, depending whether the model you wire to the **State-Space Model** input of this VI is continuous or discrete. However, the **Discretized LQR** instance of this VI always returns the solution to the discrete algebraic Riccati equation. Refer to the **Details** section for the definitions of the Riccati equations.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Linear Quadratic Regulator Details

The value you specify for the **Weighting Type** parameter determines the cost function this VI minimizes. The cost function also depends on whether the model is continuous or discrete.

Continuous Models

State Weighting

$$J = \int_{0}^{\infty} [\mathbf{x}^{\mathsf{T}}(t)\mathbf{Q}\,\mathbf{x}(t) + \mathbf{u}^{\mathsf{T}}(t)\mathbf{R}\,\mathbf{u}(t) + 2\,\mathbf{x}^{\mathsf{T}}(t)\mathbf{N}\,\mathbf{u}(t)]dt$$

Output Weighting, Dim[Q] = Ny

 $J = \int_{0}^{\infty} [\boldsymbol{y}^{\mathsf{T}}(t)\boldsymbol{Q}\boldsymbol{y}(t) + \boldsymbol{u}^{\mathsf{T}}(t)\boldsymbol{R}\boldsymbol{u}(t) + 2\boldsymbol{y}^{\mathsf{T}}(t)\boldsymbol{N}\boldsymbol{u}(t)]dt$

Output Weighting, Dim[Q] = Nx

 $J = \int_{0}^{\infty} [\boldsymbol{y}^{\mathsf{T}}(t) \cdot \boldsymbol{u}^{\mathsf{T}}(t)] \begin{bmatrix} \boldsymbol{c} & \boldsymbol{p} \\ 0 & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{N} \\ \boldsymbol{N}^{\mathsf{T}} & \boldsymbol{R} \end{bmatrix} \begin{bmatrix} \boldsymbol{c}^{\mathsf{T}} & \boldsymbol{0}^{\mathsf{T}} \\ \boldsymbol{p}^{\mathsf{T}} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}(t) \\ \boldsymbol{u}(t) \end{bmatrix} dt$

For continuous models, **Optimal Gain (K)** = $R^{-1}[B^TX+N^T]$.

This VI solves the continuous algebraic Riccati equation, defined as $A^TX+XA-[XB+N]R^{-1}[XB+N]^T+Q = 0.$

Discrete Models

State Weighting

$$J = \sum_{k=0}^{\infty} [\mathbf{x}^{\mathsf{T}}(k)\mathbf{Q}\mathbf{x}(k) + \mathbf{u}^{\mathsf{T}}(k)\mathbf{R}\mathbf{u}(k) + 2\mathbf{x}^{\mathsf{T}}(k)\mathbf{N}\mathbf{u}(k)]$$

Output Weighting, Dim[Q] = Ny

$$J = \sum_{k=0}^{\infty} [\mathbf{y}^{\mathsf{T}}(k)\mathbf{Q}\mathbf{y}(k) + \mathbf{u}^{\mathsf{T}}(k)\mathbf{R}\mathbf{u}(k) + 2\mathbf{y}^{\mathsf{T}}(k)\mathbf{N}\mathbf{u}(k)]$$

Output Weighting, Dim[Q] = Nx

$$J = \sum_{k=0}^{\infty} [\mathbf{y}^{\mathsf{T}}(k) \cdot \mathbf{u}^{\mathsf{T}}(k)] \begin{bmatrix} \mathbf{c} & \mathbf{p} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{q} & \mathbf{N} \\ \mathbf{N}^{\mathsf{T}} & \mathbf{R} \end{bmatrix} \begin{bmatrix} \mathbf{c}^{\mathsf{T}} & \mathbf{0}^{\mathsf{T}} \\ \mathbf{p}^{\mathsf{T}} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{y}(k) \\ \mathbf{u}(k) \end{bmatrix}$$

For discrete models, **Optimal Gain (K)** = $[\mathbf{B}^{\mathsf{T}}\mathbf{X}\mathbf{B}+\mathbf{R}]^{-1}[\mathbf{B}^{\mathsf{T}}\mathbf{X}\mathbf{A}+\mathbf{N}^{\mathsf{T}}]$.

This VI solves the discrete algebraic Riccati equation, defined as $A^T X A - [A^T X B + N] [B^T X B + R]^{-1}] [A^T X B + N]^T + Q = X.$

Matrix Restrictions

For continuous models, you must be able to stabilize the pair (\hat{A}, B_1) .

```
where \hat{A} = A - BR^{-1}N^{T}
```

 B_1 is the full-rank factorization of $G = BR^{-1}B^T$, that is, $G = B_1B^T_1$ and rank $[G] = rank[B_1]$

For discrete models, you must be able to stabilize the pair (\hat{A}, B) .

For continuous and discrete models, you must be able to detect the pair $(c_1, \hat{A})_1$.

where C_1 is the full-rank factorization of $H = Q - NR^{-1}N^T$, that is, $H = C_1C_1^T$ and rank[H] = rank[C_1]

 ${\bf Q}$ and ${\bf R}$ must satisfy the following relationships, respectively:

 $\boldsymbol{Q} = \boldsymbol{Q}^{\mathsf{T}} \ge 0$ $\boldsymbol{R} = \boldsymbol{R}^{\mathsf{T}} > 0$

N must satisfy the following relationship:

$$\begin{bmatrix} \boldsymbol{Q} & \boldsymbol{N} \\ \boldsymbol{N}^{\mathsf{T}} & \boldsymbol{R} \end{bmatrix} \ge 0$$

where \boldsymbol{u} is the input

y is the output

x is the states

t is continuous time

k is discrete time

Q is the state-weighting matrix

R is the input-weighting matrix

N is the cross-weighting matrix between the states and inputs

B is the input matrix

C is the output matrix

D is the direct transmission matrix

I is the identity matrix

 \pmb{X} is the solution to the continuous or discrete algebraic Riccati equation

Discretized Linear Quadratic Regulator

This instance converts the *A*, *B*, *C*, and *D* matrices using the numerical integration method as proposed by Van Loan. Refer to the following sources for more information about this method.

- G.F. Franklin, J.D. Powell, and M. Workman, *Digital Control of Dynamic Systems*, 3rd ed. Menlo Park, CA: Addison Wesley, 1997.
- C.F.V. Loan, "Computing integrals involving the matrix exponential," *IEEE Transactions on Automatic Control*, vol. 23, no. 3, pp. 395–404, June 1978.

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.
CD Pole Placement VI

Owning Palette: <u>State Feedback Design VIs</u>

Installed With: Control Design and Simulation Module

Determines the **Gain** that places the closed-loop poles at desired locations in a system with full state feedback. You can use this VI with multiple-input multiple-output (MIMO) systems. However, if you have a single-input single-output system, use the <u>CD Ackermann</u> VI.

Details



■ Place on the block diagram ■ Find on the Functions palette

- Threshold specifies the tolerance, which is the maximum difference between the **Poles** you specify and the poles this VI calculates. This VI displays a warning if the calculated poles are not within the tolerance of the specified poles.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI places closedloop poles in desired locations using the **Gain** in the state feedback. You also can use <u>pole placement</u> to calculate the estimator gain and set the pole locations of the full state observer.
- **Poles** specifies the poles this VI defines for the closed-loop system.
- Gain Type specifies the type of gain this VI returns in the Gain parameter. Use the Controller Gain with the <u>CD State-Space</u> <u>Controller</u> VI. Use the Observer Gain (Predictive) and the Observer Gain (Current) with the <u>CD Discrete Observer</u> function and the <u>CD Continuous Observer</u> function.

For discrete systems, the **Observer Gain (Predictive)** relates to the **Observer Gain (Current)** through the following relationship:

Observer Gain (Predictive) = A. **Observer Gain (Current)**.

For continuous systems, the notion of current and predictive time does not apply, so specifying either observer gain returns the correct gain value.

- 0 **Controller Gain** (default)—Calculates feedback gain **K** for a controller, such that the controller pole locations are the locations you specify using the **Poles** parameter. The system must be controllable, meaning the controllability matrix must be full row rank.
- 1 **Observer Gain (Predictive)**—Calculates feedback gain **Lp** for a predictive observer, such that the observer pole locations are the locations you specify using the **Poles** parameter. The system must be observable, meaning the observability matrix must be full column rank.
- 2 **Observer Gain (Current)**—Calculates feedback gain **Lc** for a current observer, such that the observer pole locations are the locations you specify using the **Poles** parameter. The system must be observable, meaning the observability matrix must be full column rank. Also, if the system is discrete, **A** must be invertible.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Gain** returns the feedback gain, according to the value you specify for the **Gain Type** parameter, that produces a closed-loop system such that the poles are equal to the values you specify in the **Poles** parameter.
- Actual Poles are the eigenvalues of the closed-loop system matrix $ilde{A}$. The definition of $ilde{A}$ depends on the value you specify for the Gain Type parameter.

If Gain Type is Controller Gain, $\tilde{A} = A - BK$.

If Gain Type is Observer Gain (Predictive), $\tilde{A} = A - LC$, where L is the Gain this VI returns.

If Gain Type is Observer Gain (Current), $\tilde{A} = A - LCA$.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Pole Placement Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD State Estimator VI

Owning Palette: <u>State Feedback Design VIs</u>

Installed With: Control Design and Simulation Module

Builds the state estimator based on a list of known inputs, measured outputs, the linear state-space model, and the estimator gain. The data type you wire to the **Estimator Gain (L)** input determines the polymorphic instance to use.

A matrix estimator gain indicates that the system is a <u>multiple-output</u> model. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about building state estimators. A general system configuration for the state estimator is one where this VI appends the original model states (x) to the estimation model states (\hat{x}) to represent the estimator, as shown in the following equations.

$$\begin{bmatrix} \dot{x} \\ \dot{x} \\ \dot{x} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{x} \end{bmatrix} + \begin{bmatrix} \mathbf{B} & \mathcal{L} \\ \mathbf{B} & 0 \end{bmatrix} \begin{bmatrix} u \\ v & -\hat{v} \end{bmatrix}$$
$$\begin{bmatrix} \dot{v} \\ y \end{bmatrix} = \begin{bmatrix} \mathbf{C} & 0 \\ 0 & \mathbf{C} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{x} \end{bmatrix} + \begin{bmatrix} \mathbf{D} & 0 \\ \mathbf{D} & 0 \end{bmatrix} \begin{bmatrix} u \\ v & -\hat{v} \end{bmatrix} + \begin{bmatrix} 0 \\ r_{y} \end{bmatrix}$$

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

•

State Estimator (Single Output)



Configuration specifies if this VI includes the system in the estimator or considers noise as an input of the estimator. The estimator also can be a standalone estimator.

0 Standalone (default)

1 System Included

2 System Included with Noise

- **Estimator Gain (L)** is the gain that premultiplies the output error to correct the model state dynamics towards the physical states of the system.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the model that you want to use to build the estimator.
- Measured Outputs lists the index numbers of all the outputs that you can measure for state estimation. The index is zero-based. By default, this VI measures all outputs.
- **Known Inputs** lists the index numbers of all the inputs that are known for state estimation. The index is zero-based. By default, all inputs are known.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check

errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Estimator Model** provides the state-space model for the state and output estimation given known inputs and measured outputs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

State Estimator (Multiple Output)



Configuration specifies if this VI includes the system in the estimator or considers noise as an input of the estimator. The estimator also can be a standalone estimator.

0 Standalone (default)

1 System Included

2 System Included with Noise

- **Estimator Gain** is the gain that premultiplies the output error to correct the model state dynamics towards the physical states of the system.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the model that you want to use to build the estimator.
- Measured Outputs lists the index numbers of all the outputs that you can measure for state estimation. The index is zero-based. By default, this VI measures all outputs.
- **Known Inputs** lists the index numbers of all the inputs that are known for state estimation. The index is zero-based. By default, all inputs are known.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check

errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Estimator Model** provides the state-space model for the state and output estimation given known inputs and measured outputs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD State Estimator Details

Configuration Types

There are different configurations to synthesize the estimator. These configurations are system included, system included with noise, and standalone.

The system included configuration indicates that this VI calculates the states of the system and the outputs of the system internally. You do not need to provide the system outputs as inputs of the estimator to synthesize the estimator.

The system included with noise configuration incorporates sensor noise into the system included configuration. Therefore, sensor noise is an input of the controller model. Sensor noise affects the estimated states that you use to calculate the control action.

The standalone configuration configures the estimator to accept the actual outputs of the systems as inputs for state estimation. This configuration is useful for implementing the estimator on a real-time target.

Refer to the <u>LabVIEW Control Design User Manual</u> for more information about building a state-space controller.

Delay Support

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD State-Space Controller VI

Owning Palette: <u>State Feedback Design VIs</u>

Installed With: Control Design and Simulation Module

Builds a state-space controller based on known inputs, manipulated inputs, measured outputs, linear state-space model, estimator gain, and controller gain. You must <u>manually select the polymorphic instance</u> you want to use.

A general system configuration for the state compensator is one where this VI appends the original model states (x) to the estimation model states (\hat{x}) to represent the compensator with an estimator, as shown in the following equations.

$$\begin{bmatrix} \dot{x} \\ \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \mathbf{A} - \mathbf{B} \times \mathbf{0} \\ -\mathbf{B} \times \mathbf{A} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{x} \end{bmatrix} + \begin{bmatrix} \mathbf{B} \times \mathbf{L} \\ \mathbf{B} \times \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \mathbf{y} - \dot{y} \end{bmatrix}$$
$$\begin{bmatrix} \mathbf{u} \\ \mathbf{c} \\ -\mathbf{D} \times \mathbf{0} \\ \mathbf{c} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{x} \end{bmatrix} + \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ \mathbf{D} \times \mathbf{0} \\ \mathbf{D} \times \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \mathbf{y} - \dot{y} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{r} \\ \mathbf{y} - \dot{y} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{r} \\ \mathbf{r} \end{bmatrix}$$

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

```
Select an instance
```

■ Place on the block diagram ■ Find on the **Functions** palette

•

State-Space Controller (Compensator)



Configuration specifies if this VI includes the system in the estimator or considers noise as an input of the estimator. The estimator also can be a standalone estimator.

0 Standalone (default)

1 System Included

2 System Included with Noise

- **Estimator Gain (L)** is the gain that premultiplies the output error to correct the model state dynamics towards the physical states of the system.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the process that you want to use to build the controller.
- Measured Outputs lists the index numbers of all the outputs that you can measure for state estimation. The index is zero-based. By default, this VI measures all outputs.
- **Known Inputs** lists the index numbers of all the inputs that are known for state estimation. The index is zero-based. By default, all inputs are known.
- Manipulated Inputs are the inputs that this VI adjusts in the system and uses for control.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error

<u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Controller Gain (K)** is the gain that premultiplies the estimated state to calculate the manipulated inputs to drive the system towards or back to a reference.
- **Controller Model** returns the state-space model for the controller. The inputs, number of states, and outputs are a function of the configuration you select.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

State-Space Controller (Regulator)

A general system configuration for the state regulator is one where this VI appends the original model states (x) to the estimation model states (\hat{x}) to represent the state regulator with an estimator, as shown in the following equations.



- **Configuration** specifies if this VI includes the system in the estimator or considers noise as an input of the estimator. The estimator also can be a standalone estimator.
 - 0 Standalone (default)

1 System Included

2 System Included with Noise

- **Estimator Gain (L)** is the gain that premultiplies the output error to correct the model state dynamics towards the physical states of the system.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the process that you want to use to build the controller.
- Measured Outputs lists the index numbers of all the outputs that you can measure for state estimation. The index is zero-based. By default, this VI measures all outputs.
- **Known Inputs** lists the index numbers of all the inputs that are known for state estimation. The index is zero-based. By default, all inputs are known.

- Manipulated Inputs are the inputs that this VI adjusts in the system and uses for control.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Controller Gain (K)** is the gain that premultiplies the estimated state to calculate the manipulated inputs to drive the system towards or back to a reference.
- **Controller Model** returns the state-space model for the controller. The inputs, number of states, and outputs are a function of the configuration you select.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

State-Space Controller (Regulator with Integral)

A general system configuration for the state regulator with integral action is one where this VI appends the output error integrator (z) to the estimation model states (\hat{x}). In addition, this VI augments the resulting vector (\hat{x} , z) with the original model states (x) to represent the state regulator with integral action and an estimator.



Configuration specifies if this VI includes the system in the estimator or considers noise as an input of the estimator. The estimator also can be a standalone estimator.

0	Standalone	(default)
---	------------	-----------

1 System Included

2 System Included with Noise

- **Estimator Gain (L)** is the gain that premultiplies the output error to correct the model state dynamics towards the physical states of the system.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the process that you want to use to build the controller.
- Measured Outputs lists the index numbers of all the outputs that you can measure for state estimation. The index is zero-based. By default, this VI measures all outputs.

[032]

Known Inputs lists the index numbers of all the inputs that are known for state estimation. The index is zero-based. By default, all inputs are known.

- Manipulated Inputs are the inputs that this VI adjusts in the system and uses for control.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Controller Gain (K)** is the gain that premultiplies the estimated state to calculate the manipulated inputs to drive the system towards or back to a reference.
- **Integral Gain (Ki)** is active when this VI independently tunes the integral action from the controller gain **K**. **Integral Gain (Ki)** is empty when this VI includes the integral action gain in the controller gain **K**.
- **Controller Model** returns the state-space model for the controller. The inputs, number of states, and outputs are a function of the

configuration you select.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD State-Space Controller Details

The LabVIEW Control Design and Simulation Module provides three types of state controllers—a compensator, a regulator, and a regulator with integral action. The difference in these controllers is in how you calculate the control action. There are four different configurations to synthesize a state controller. These configurations are system included, system included with noise, standalone with estimator, and standalone without estimator. Both the system included and system included with noise configurations automatically include an estimator in the configuration. The configuration of a controller depends on the inputs.

The system included configuration bases the control action on the estimated states. This VI calculates the states and outputs of the system internally. You do not need to provide the system outputs as the inputs of the estimator to synthesize the controller.

The system included with noise configuration incorporates sensor noise into the system included configuration. Therefore, sensor noise is an input of the controller model. Sensor noise affects the estimated states that you use to calculate the control action.

The standalone with estimator configuration bases the control action on the estimated states using the estimator gain this VI provides. The estimated states are based on the actual outputs of the system. The system outputs are inputs of the standalone controller. This configuration is useful for implementing the estimator on a real-time target.

The standalone without estimator configuration bases the control action on the actual states because it does not require an estimator. This configuration considers a full state feedback structure and the controller internally calculates the system outputs.

Refer to the <u>LabVIEW Control Design User Manual</u> for more information about building a state-space controller.

State-Space Model Analysis VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use State-Space Model Analysis VIs to calculate properties of a given state-space model, such as observability, detectability, controllability, stabilizability, similarity transformations, model balance, and system Grammians.

The State-Space Model Analysis VIs do not support the <u>Stochastic</u> <u>Systems VIs</u>.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD Balance</u> <u>State-Space</u> <u>Model</u> (Diagonal)	Balances the State-Space Model using a diagonal similarity transformation. This transformation reduces the ratios of rows and columns norms of the system matrix A , or of the matrix S defined by the following equation.
<u>CD Balance</u> <u>State-Space</u> <u>Model</u> (Grammians)	Calculates a Balanced State-Space Model based on Grammians. The resulting balanced transformation has identical controllability and observability diagonal Grammians.
<u>CD</u> <u>Canonical</u> <u>State-Space</u> <u>Realization</u>	Transforms the State-Space Model to a canonical form that Form Type specifies. This VI also returns the similarity Transformation Matrix that this VI uses to transform the given system.
<u>CD</u> Controllability Matrix	Calculates the Controllability Matrix of the State-Space Model . You can use the controllability matrix Q to determine if the given system is controllable. A system of order <i>n</i> is controllable if Q is full rank, meaning the rank of Q is equal to <i>n</i> . This VI also determines if the given system is stabilizable. A system is stabilizable if all the unstable eigenvalues are controllable.

<u>CD</u> <u>Controllability</u> <u>Staircase</u>	Calculates the controllable staircase similarity transformation of the State-Space Model . You can use the staircase representation to identify controllable and uncontrollable states by simple inspection of the A and B matrices of the transformed model.
<u>CD</u> <u>Grammians</u>	Calculates the controllability or observability Grammian of the State-Space Model for a stable system. The system can be continuous or discrete. You can use this VI to balance state-space models and to study controllability and observability properties of the system.
<u>CD</u> Observability Matrix	Calculates the Observability Matrix of the State-Space Model . You can use the observability matrix N to determine if the given system is observable. A system of order <i>n</i> is observable if N is full rank, meaning the rank of N is equal to <i>n</i> . This VI also determines if the given system is detectable. A system is detectable if all the unstable eigenvalues are observable.
<u>CD</u> <u>Observability</u> <u>Staircase</u>	Calculates the observable staircase similarity transformation of a State-Space Model . You can use the staircase representation to identify observable and unobservable states by simple inspection of the A and C matrices of the transformed model.
<u>CD State</u> <u>Similarity</u> <u>Transform</u>	Applies a similarity transformation on State-Space Model using the given Transformation Matrix (T) .

CD Balance State-Space Model (Diagonal) VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Balances the **State-Space Model** using a diagonal similarity transformation. This transformation reduces the ratios of rows and columns norms of the system matrix **A**, or of the matrix **S** defined by the following equation.

$S = [\mathbf{A} \mathbf{B} | \mathbf{C} 0]$

This VI returns the similarity **Transformation Matrix** that this VI uses to transform the given system. This VI renames the names of the states using the default names after the transformation.

<u>Details</u>



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system that this VI transforms.
- **Tolerance** limits the ill-conditioning of matrix **T**. This VI estimates this condition by the ratio of maximum and minimum norms of the eigenvalues of **T**. The default is Inf.
- Balance Type defines which matrix this VI balances and if this VI allows column/row permutations in the similarity transformation.
 - 0 A with permutations (default)—This VI considers only matrix A in the calculation of T. This VI can have permutations among rows and columns of matrix A.
 - 1 **A without permutations**—This VI considers only matrix **A** in the calculation of **T**. This VI can not have permutations among rows and columns of matrix **A**.
 - 2 **[A B| C 0]**—This VI considers matrix **S** in the calculation of **T**. This VI can not have permutations.



error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Balanced Model returns the model that results from applying the similarity transformation **T** to the **State-Space Model**.
- **Transformation Matrix (T)** returns the matrix **T** that this VI uses to transform the system. The calculation of **T** is based on LAPACK balance routine.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code**

is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Balance State-Space Model (Diagonal) Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Balance State-Space Model (Grammians) VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Calculates a **Balanced State-Space Model** based on Grammians. The resulting balanced transformation has identical controllability and observability diagonal Grammians.

This VI returns the similarity **Transformation Matrix** that this VI uses in transforming the given system. This VI renames the names of the states using the default names after the transformation.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system that this VI transforms.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Balanced Model returns the model that results from applying the Transformation Matrix (T) to the State-Space Model.
- **Transformation Matrix (T)** returns the matrix **T** that this VI uses to transform the system. The calculation of **T** is based on the Cholesky decomposition of the Grammians of the original system.
- **Diagonal of Grammian** returns the resulting observability and controllability Grammians of the **Balanced Model**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Balance State-Space Model (Grammians) Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Canonical State-Space Realization VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Transforms the **State-Space Model** to a canonical form that **Form Type** specifies. This VI also returns the similarity **Transformation Matrix** that this VI uses to transform the given system.

This VI renames the names of the states using the default names after the transformation.

Details

State-Space Model Transformed Model [⁷ 7] Form Type -Transformation Matrix (T) error in (no error) === error out

■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system that this VI transforms into the specified canonical representation.
- **Form Type** specifies the canonical form this VI uses to transform the state-space model.

Modal (default)—A is a bi-diagonal matrix. The elements in the main diagonal are the real part of the eigenvalues of A, while elements in the upper secondary or lower secondary diagonal represent the imaginary part of the eigenvalues of A. Therefore, A is strictly diagonal if and only if all eigenvalues are real.

- 1 **Controllability**—**A** has the coefficients of the characteristic polynomial in its bottom row.
- 2 **Controller Companion**—**A** has the coefficients of the characteristic polynomial of **A** in its top row.
- 3 **Observability**—**A** has the coefficients of the characteristic polynomial in its last column.
- 4 **Observer Companion**—**A** has the coefficients of the characteristic polynomial in its first column.



error in describes error conditions that occur before this VI or

function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Transformed Model** returns the canonical representation of the **State-Space Model** according to the **Form Type**.
- **Transformation Matrix (T)** returns the matrix **T** that this VI uses to apply a similarity transform on the **State-Space Model** to produce the **Canonical State-Space Model**. To obtain the controllability or controllable companion form, the system must be controllable, while to obtain the observability or observable companion form, the system must be observable.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Canonical State-Space Realization Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Controllability Matrix VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Calculates the **Controllability Matrix** of the **State-Space Model**. You can use the controllability matrix \mathbf{Q} to determine if the given system is controllable. A system of order *n* is controllable if \mathbf{Q} is full rank, meaning the rank of \mathbf{Q} is equal to *n*. This VI also determines if the given system is stabilizable. A system is stabilizable if all the unstable eigenvalues are controllable.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines controllability matrix.
- **Tolerance** is the threshold this VI uses to determine if the controllability matrix is row rank deficient. The default is 1E–6.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Controllability Matrix** is the matrix **Q** for a system with system matrix **A** of order *n*, and input matrix **B**. The following equation defines the controllability matrix.

 $\boldsymbol{Q} = \left[\boldsymbol{B} |\boldsymbol{A}\boldsymbol{B}| .. \boldsymbol{A}^{n-1} \boldsymbol{B}\right]$

- If **Is Controllable?** is TRUE, the system is controllable.
- If **Is Stabilizable?** is TRUE, the system is stabilizable. A system is stabilizable if all unstable eigenvalues are controllable.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Controllability Matrix Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.
CD Controllability Staircase VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Calculates the controllable staircase similarity transformation of the **State-Space Model**. You can use the staircase representation to identify controllable and uncontrollable states by simple inspection of the **A** and **B** matrices of the transformed model.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines staircase transformation.
- **Tolerance** determines the threshold value below which this VI considers a diagonal entry in the system matrix **A** of the **State-Space Model** zero. The default is 0.0001.
- **Controllable Block Location** determines if the controllable block is in the upper left or bottom right corner of the transformed matrix.

0 **up** (default)—The controllable block is in the upper left corner of the transformed matrix.

1 **down**—The controllable block is in the bottom right corner of the transformed matrix.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Transformed Model** returns the transformed system model in staircase form.
- **Transformation Matrix (T)** returns the matrix use for the similarity transformation.
- Controllable States returns the number of controllable states this VI finds per iteration of the calculation that the VI performs to get T. The sum of elements in this array is the total number of controllable states.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

CD Controllability Staircase Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Grammians VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Calculates the controllability or observability **Grammian** of the **State-Space Model** for a stable system. The system can be continuous or discrete. You can use this VI to balance state-space models and to study controllability and observability properties of the system.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI calculates Grammian.
- **Tolerance** is the threshold this VI uses to determine if the system is controllable and stable. The default is 1E–6.
- **Type** specifies if this VI calculates the controllability or observability Grammian.

0	Controllability	(default)

1 Observability

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Grammian** returns the controllability or observability Grammian matrix.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Grammians Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Observability Matrix VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Calculates the **Observability Matrix** of the **State-Space Model**. You can use the observability matrix \mathbf{N} to determine if the given system is observable. A system of order *n* is observable if \mathbf{N} is full rank, meaning the rank of \mathbf{N} is equal to *n*. This VI also determines if the given system is detectable. A system is detectable if all the unstable eigenvalues are observable.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines the observability matrix.
- **Tolerance** is the threshold this VI uses to determine if the observability matrix is column rank deficient. The default is 1E–6.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Observability Matrix** is the matrix **O** computed for a system with a system matrix **A** and output matrix **C**. The matrix **O** is defined by the following equation:

$$\mathbf{O} = \begin{bmatrix} \mathbf{C} \\ \mathbf{C} \mathbf{A} \\ \vdots \\ \mathbf{C} \mathbf{A}^{n-1} \end{bmatrix}$$

- If **Is Observable?** is TRUE, the system is observable.
- If **Is Detectable?** is TRUE, the system is detectable. A system is detectable if all unstable eigenvalues are observable.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Observability Matrix Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD Observability Staircase VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Calculates the observable staircase similarity transformation of a **State-Space Model**. You can use the staircase representation to identify observable and unobservable states by simple inspection of the **A** and **C** matrices of the transformed model.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI determines staircase transformation.
- **Tolerance** determines the threshold value below which this VI considers a diagonal entry in the system matrix **A** of the **State-Space Model** zero. The default is 0.0001.
- **Observable Block Location** determines if the observable block is in the upper left or bottom right corner of the transformed matrix.

0 **up** (default)—The observable block is in the upper left corner of the transformed matrix.

1 **down**—The observable block is in the bottom right corner of the transformed matrix.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Transformed Model** returns the transformed system model in staircase form.
- **Transformation Matrix (T)** returns the matrix use for the similarity transformation.
- Observable States returns the number of observable states this VI finds per iteration of the calculation that the VI performs to get T. The sum of elements in this array is the total number of observable states.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

CD Observability Staircase Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

CD State Similarity Transform VI

Owning Palette: <u>State-Space Model Analysis VIs</u>

Installed With: Control Design and Simulation Module

Applies a similarity transformation on **State-Space Model** using the given **Transformation Matrix (T)**.

<u>Details</u>



■ Place on the block diagram ■ Find on the **Functions** palette

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about a system that this VI transforms into the specified similarity transformation.
- **Transformation Matrix (T)** is the matrix **T** that this VI uses to transform the system. **T** must be an invertible matrix.
- **Transformation** determines the type of realization this VI uses in the resulting state-space model.

0 **Direct** (default)—Specifies that this VI uses a direct transformation as defined by the following equation: $\bar{x} = \tau x$

1 **Inverse**—Specifies that this VI uses an inverse transformation as defined by the following equation: $x = \tau^{-1}\bar{x}$

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Transformed Model** returns the model that results from applying the similarity transform to the **State-Space Model**.
- **Inverse Transformation Matrix** returns the inverse of the **Transformation Matrix (T)**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD State Similarity Transform Details

This VI does not support delays unless the delays are part of the mathematical model that represents the dynamic system. To account for the delays in the synthesis of the controller, you must incorporate the delays into the mathematical model of the dynamic system using the <u>CD</u> <u>Convert Delay with Pade Approximation</u> VI (continuous models) or the <u>CD Convert Delay to Poles at Origin</u> VI (discrete models). Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays and the limitations of Pade Approximation.

Stochastic Systems VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Stochastic Systems VIs to construct, manipulate, and analyze stochastic state-space system models.

Palette Object	Description
<u>CD</u> Construct Noise Model	Constructs a first- and second-order statistical noise model. A noise model defines the statistical behavior of the process noise vector w and the measurement noise vector v . The data type you wire to the E{w} , E{v} , Q , R , or N parameter determines the polymorphic instance to use.
<u>CD</u> <u>Construct</u> <u>Stochastic</u> <u>Model</u>	Creates a <u>stochastic state-space representation</u> of a system using the matrices A , B , C , D , G , and H , and the Sampling Time . This VI also produces a stochastic state- space model that specifies the data in symbolic form. The data type you wire to the A , B , C , D , G , or H parameter determines the polymorphic instance to use.
CD Convert Continuous Stochastic to Discrete	Converts a continuous stochastic state-space model and the associated continuous noise model to a discrete stochastic state-space model and discrete noise model.
	Converts a deterministic state-space model to a stochastic state-space model. This VI appends matrices G and H to the A , B , C , and D matrices of the deterministic model.
<u>CD Convert</u> <u>Stochastic to</u> <u>Deterministic</u> <u>Model</u>	Converts a stochastic state-space model to a deterministic state-space model by removing matrices G and H from the stochastic model definition.
<u>CD</u> Correlated	Generates a sample of one or two Gaussian-distributed random vectors, which you can use as the noise vectors w

<u>Gaussian</u> <u>Random</u> <u>Noise</u>	and v . You specify the mean, auto-covariance, and cross- covariance of these vectors. You must <u>manually select the</u> <u>polymorphic instance</u> to use.
<u>CD Get Data</u> from Noise Model	Obtains data that describes the statistics of the specified noise model.
<u>CD Get Data</u> from Stochastic Model	Obtains data that describes the dynamics of a stochastic state-space system model.
<u>CD Get</u> <u>Sampling</u> <u>Time from</u> <u>Stochastic</u> <u>Model</u>	Obtains the sampling time of a stochastic state-space model. The sampling time is zero for continuous systems and greater than zero for discrete systems.
<u>CD Get</u> <u>Stochastic</u> <u>System</u> <u>Dimensions</u>	Obtains the number of states, inputs, and outputs of the stochastic state-space system model. This VI also returns the dimensions of the process noise vector w .
<u>CD Verify</u> <u>Noise Model</u>	Determines if the dimensions of the Second-Order Statistics Noise Model matrices are consistent with the dimensions of the Stochastic State-Space Model matrices. This VI also calculates the number of inputs, outputs, and states, and the dimensions of the process noise vector w .
<u>CD Verify</u> <u>Stochastic</u> <u>Model</u> <u>Properties</u>	Determines if the dimensions of the stochastic state-space system matrices are consistent with each other. This VI also calculates the number of inputs, outputs, and states, as well as the dimensions of the process noise vector w .

CD Construct Noise Model VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Constructs a first- and second-order statistical noise model. A noise model defines the statistical behavior of the process noise vector w and the measurement noise vector v. The data type you wire to the **E{w}**, **E{v}**, **Q**, **R**, or **N** parameter determines the polymorphic instance to use.



Note If you do not wire a value to the **E{w}**, **E{v}**, **Q**, **R**, **N**, or symbolic parameters, this VI replaces the unwired parameters with an appropriately-sized vector or matrix of zeros.

•

This VI uses the <u>CD Verify Noise Model VI</u> to verify that the dimensions of the noise model are consistent with the **Stochastic State-Space Model** you provide.

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Construct Noise Model (Numeric)



- Stochastic State-Space Model specifies a <u>mathematical</u> <u>representation</u> of a stochastic system.
- **E{w}** specifies the mean vector of the process noise vector w.
- **E**{**v**} specifies the mean vector of the measurement noise vector \mathbf{v} .
- \mathbf{Q} specifies the auto-covariance matrix of the process noise vector.
- **R** specifies the auto-covariance matrix of the measurement noise vector.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **N** specifies the cross-covariance matrix between the process noise and the measurement noise vectors.
 - Note If the process noise and measurement noise vectors are uncorrelated, either specify a matrix of zeros for this parameter or do not wire a value to this parameter.
- Second-Order Statistics Noise Model Out returns a <u>mathematical representation</u> of the noise model of the Stochastic State-Space Model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Noise Model (Symbolic)



- **Stochastic State-Space Model** specifies a <u>mathematical</u> <u>representation</u> of a stochastic system.
- **Symbolic E{w}** specifies the <u>symbolic representation</u> of the mean vector of the process noise vector.
- **Symbolic E{v}** specifies the symbolic representation of the mean vector of the measurement noise vector.
- **Symbolic Q** specifies the symbolic representation of the autocovariance matrix of the process noise vector.
- **Symbolic R** specifies the symbolic representation of the autocovariance matrix of the measurement noise vector.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Symbolic N** specifies the symbolic representation of the crosscovariance matrix between the process noise and measurement noise vectors.
- **Variables** contains the name and value of each variable.
 - **Name** is a variable name this VI uses to define the data of the system model. Variable names can be a combination of letters and numbers. A variable name that begins with a capital letter E can produce unpredictable errors if parts of the original string represent numbers like 1E–2. Avoid terms beginning with E in such cases.
 - **Value** is the numeric value this VI associates with the variable. The VI uses this value to evaluate the model.
- Second-Order Statistics Noise Model Out returns a <u>mathematical representation</u> of the noise model of the Stochastic State-Space Model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Construct Stochastic Model VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Creates a <u>stochastic state-space representation</u> of a system using the matrices **A**, **B**, **C**, **D**, **G**, and **H**, and the **Sampling Time**. This VI also produces a stochastic state-space model that specifies the data in symbolic form. The data type you wire to the **A**, **B**, **C**, **D**, **G**, or **H** parameter determines the polymorphic instance to use.

Note If you do not wire a value to the A, B, C, D, G, H, or symbolic parameters, this VI replaces the unwired parameters with an appropriately-sized vector or matrix of zeros.

This VI uses the <u>CD Verify Stochastic Model Properties</u> VI to verify that the dimensions of the **A**, **B**, **C**, **D**, **G**, and **H** matrices are consistent with each other.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Construct Stochastic Model (Numeric)



- Sampling Time (s) specifies the sampling time of the system model and determines whether the model represents a continuous-time or discrete-time system. If the model represents a continuous-time system, **Sampling Time** must equal zero. If the model represents a discrete-time system, **Sampling Time** must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - Note If you use the inputs to create a continuous-time system, setting the **Sampling Time** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous</u> <u>Stochastic to Discrete Stochastic</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.
- A specifies the system matrix that describes the dynamics of the states of the system.
- **B** specifies the input matrix that relates the inputs to the states.
- **C** specifies the output matrix that relates the outputs to the states.
- **D** specifies the transmission matrix that relates the inputs to the outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use

exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **G** specifies the matrix that relates the process noise vector to the model states.
- **H** specifies the matrix that relates the process noise vector to the model outputs.
- Stochastic State-Space Model returns a mathematical representation of a stochastic state-space model. The data consists of matrices A, B, C, D, G, and H. To access the model data, use the <u>CD Get Data from Stochastic Model</u> VI.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

CD Construct Stochastic Model (Symbolic)



- Sampling Time (s) specifies the sampling time of the system model and determines whether the model represents a continuous-time or discrete-time system. If the model represents a continuous-time system, **Sampling Time** must equal zero. If the model represents a discrete-time system, **Sampling Time** must be greater than zero and equal to the sampling rate, in seconds, of the discrete system. The default is 0.
 - $\overline{\mathbb{N}}$

Note If you use the inputs to create a continuous-time system, setting the **Sampling Time** to a value greater than zero does not yield the discrete-time equivalent of the system. You must use the <u>CD Convert Continuous</u> <u>Stochastic to Discrete Stochastic</u> VI to convert the continuous-time system to the discrete-time equivalent of the system.

- **Symbolic A** is the <u>symbolic representation</u> of the system matrix that describes the dynamics of the states of the system.
- **Symbolic B** is the symbolic representation of the input matrix of the system that relates the inputs to the states.
- **Symbolic C** is the symbolic representation of the output matrix that relates the outputs to the states.
- **Symbolic D** is the symbolic representation of the transmission matrix that relates the inputs to the outputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error

status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Symbolic G** is the symbolic representation of the matrix that relates the process noise to the model states.
- **Symbolic H** is the symbolic representation of the matrix that relates the process noise to the model outputs.
- **Variables** contains the name and value of each variable.
 - **Name** is a variable name this VI uses to define the data of the system model. Variable names can be a combination of letters and numbers. A variable name that begins with a capital letter E can produce unpredictable errors if parts of the original string represent numbers like 1E–2. Avoid terms beginning with E in such cases.
 - **Value** is the numeric value this VI associates with the variable. The VI uses this value to evaluate the model.
- Stochastic State-Space Model returns a mathematical representation of a stochastic state-space model. The data consists of matrices Symbolic A, Symbolic B, Symbolic C, Symbolic D, Symbolic G, and Symbolic H. To access the model data, use the CD Get Data from Stochastic Model VI.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Continuous Stochastic to Discrete VI

Owning Palette: Stochastic Systems VIs

Installed With: Control Design and Simulation Module

Converts a continuous stochastic state-space model and the associated continuous noise model to a discrete stochastic state-space model and discrete noise model.

Details



■ Place on the block diagram ■ Find on the Functions palette

- **Continuous Stochastic State-Space Model** specifies a <u>mathematical representation</u> of a continuous stochastic system.
- Continuous Second-Order Statistics Noise Model specifies a continuous mathematical representation of the noise model of the Continuous Stochastic State-Space Model. A noise model defines the expected behavior of the noise vectors *w* and *v*. You can use the CD Construct Noise Model VI to construct a noise model for a given stochastic state-space system.
- Sampling Time (s) specifies the discrete sampling time of the Discrete Stochastic State-Space Model. The default value is 1 second.
- Method specifies the method this VI uses to calculate the discrete equivalent of the process noise covariance matrix **Q**.
 - 0 **Truncation of Taylor Series** Computes the discrete equivalent of the process noise covariance matrix by using the Truncation of Taylor Series Expansion method.
 - 1 **Numerical Integration** (default)—Computes the discrete equivalent of the process noise covariance matrix by using the numerical integration method as proposed by Van Loan.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Discrete Stochastic State-Space Model returns a <u>mathematical</u> representation of the discrete equivalent of the Continuous Stochastic State-Space Model.
- Discrete Second-Order Statistics Noise Model returns a mathematical representation of the discrete equivalent of the Continuous Second-Order Statistics Noise Model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Continuous Stochastic to Discrete Details

This VI assumes the noise vectors w and v are temporally uncorrelated. This VI also assumes the **Sampling Time** *T* you specify is much smaller than the Shannon period of the input signal u(t). Therefore, the following relationship is true:

 $\boldsymbol{u}(t) \approx \boldsymbol{u}(kT), \, kT \leq t < (k+1)T$

where *t* is continuous time and *k* is discrete time.

This VI assumes the **Continuous Stochastic State-Space Model** is of the following form:

 $\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{G}\boldsymbol{w}(t)$ $\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t) + \boldsymbol{H}\boldsymbol{w}(t) + \boldsymbol{v}(t)$

This VI also assumes the **Continuous Second-Order Statistics Noise Model** is of the following form:

$$E\{\boldsymbol{w}(t)\} = \boldsymbol{m}_{w}(t)$$

$$E\{\boldsymbol{w}(t) \cdot \boldsymbol{w}^{\mathsf{T}}(\tau)\} = \boldsymbol{Q}(t) \cdot \delta(t - \tau)$$

$$E\{\boldsymbol{v}(t)\} = \boldsymbol{m}_{v}(t)$$

$$E\{\boldsymbol{v}(t) \cdot \boldsymbol{v}^{\mathsf{T}}(\tau)\} = \boldsymbol{R}(t) \cdot \delta(t - \tau)$$

$$E\{\boldsymbol{w}(t) \cdot \boldsymbol{v}^{\mathsf{T}}(\tau)\} = \boldsymbol{N}(t) \cdot \delta(t - \tau)$$

where d(t) is the Dirac delta function. This function is defined as $d(t) = \infty$ when x = 0; d(t) = 0 when $x \neq 0$.

This VI returns the **Discrete Stochastic State-Space Model** in the following form:

$$x[(k+1]T] = A_{d}x(kT) + B_{d}u(kT) + n(kT)$$

$$y(kT) = C_{d}x(kT) + D_{d}u(kT) + r(kT), k = 0, 1, 2 ...$$

where

 $A_{d} = e^{AT}$ $B_{d} = \int_{0}^{T} e^{A\eta} B d\eta$ $C_{d} = C$ $D_{d} = D$

n(kT) and r(kT) are the discrete equivalents of the noise vectors.

This VI returns the **Discrete Second-Order Statistics Noise Model** in the following form:

$$E\{\boldsymbol{n}(kT)\} = [\boldsymbol{A}_{d} - \boldsymbol{I}]\boldsymbol{A}^{-1}\boldsymbol{G}\boldsymbol{m}_{W}(kT)$$

$$E\{\boldsymbol{n}(kT) \cdot \boldsymbol{n}^{T}(lT)\} = \boldsymbol{P}(kT)\delta(kT - lT), \ k, \ l = 0, \ 1, \ 2 \ \dots$$

$$E\{\boldsymbol{r}(kT)\} = \boldsymbol{H}\boldsymbol{m}_{W} + \boldsymbol{m}_{V}(kT)$$

$$E\{\boldsymbol{r}(kT) \cdot \boldsymbol{r}^{T}(lT)\} = \boldsymbol{S}(kT)\delta(kT - lT)$$

$$\mathsf{E}\{\boldsymbol{n}(kT) \cdot \boldsymbol{r}^{\mathsf{T}}(lT)\} = \boldsymbol{O}(kT)\delta(kT - lT)$$

where

$$\boldsymbol{S}(kT) = \frac{1}{T} [\boldsymbol{H} \boldsymbol{Q}(kT) \boldsymbol{H}^{\mathsf{T}} + \boldsymbol{H} \boldsymbol{N}(kT) + \boldsymbol{N}^{\mathsf{T}}(kT) \boldsymbol{H}^{\mathsf{T}} + \boldsymbol{R}(kT)]$$

 $\boldsymbol{O}(kT) = \frac{1}{T} [\boldsymbol{A}_{d} - \boldsymbol{I}] \boldsymbol{A}^{-1} \boldsymbol{G} [\boldsymbol{Q}(kT) \boldsymbol{H}^{T} + \boldsymbol{N}(kT)]$

If you specify Numerical Integration for the Method parameter, then

 $\boldsymbol{P}(kT) = \int_{0}^{T} e^{A\eta} \boldsymbol{G} \boldsymbol{Q} \boldsymbol{G}^{\mathsf{T}} e^{AT\eta} d\eta$

If you specify Truncation of TSE for the Method parameter, then

$$\boldsymbol{P}(kT) \approx \int_{0}^{T} \boldsymbol{G} \boldsymbol{Q} \, \boldsymbol{G}^{\mathsf{T}} d\eta = \boldsymbol{G} \boldsymbol{Q} \, \boldsymbol{G}^{\mathsf{T}} \cdot \boldsymbol{T}$$

where n is the number of states

m is the number of inputs

r is the number of outputs

x is the state vector.

u is the input vector.

y is the output vector.

w is the process noise vector.

 \boldsymbol{v} is the observation noise vector.

A is an $n \times n$ state matrix of the given system.

B is an $n \times m$ input matrix of the given system.
C is an $r \times n$ output matrix of the given system.

D is an $r \times m$ direct transmission matrix of the given system.

G is a matrix relating **w** to the states.

H is a matrix relating w to the outputs.

 ${\boldsymbol{\mathsf{Q}}}$ is the auto-covariance matrix of ${\boldsymbol{\mathsf{w}}}.$

 ${f R}$ is the auto-covariance matrix of ${f v}$.

N is the cross-covariance matrix between w and v.

E{} denotes the expected mean or value of the enclosed term(s).

This approximation becomes less accurate as the value of T increases.

CD Convert Deterministic to Stochastic Model VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Converts a deterministic state-space model to a stochastic state-space model. This VI appends matrices **G** and **H** to the **A**, **B**, **C**, and **D** matrices of the deterministic model.



- Deterministic State-Space Model specifies a <u>mathematical</u> representation of and <u>information</u> about a deterministic state-space model. You can construct a deterministic state-space model using the <u>CD Construct State-Space Model</u> VI.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **G** specifies the matrix that relates the process noise vector to the model states.
- **H** specifies the matrix that relates the process noise vector to the model outputs.
- Stochastic State-Space Model returns a mathematical representation of a stochastic state-space model. The data consists of matrices A, B, C, D, G, and H. To access the model data, use the <u>CD Get Data from Stochastic Model</u> VI.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Convert Stochastic to Deterministic Model VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Converts a stochastic state-space model to a deterministic state-space model by removing matrices ${f G}$ and ${f H}$ from the stochastic model definition.

Stochastic State-Space Model	ate-Space M
error in (no error)	

- Stochastic State-Space Model In specifies a <u>mathematical</u> representation of a stochastic system.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

Deterministic State-Space Model returns a <u>mathematical</u> <u>representation</u> of and <u>information</u> about a deterministic state-space model.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Correlated Gaussian Random Noise VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Generates a sample of one or two Gaussian-distributed random vectors, which you can use as the noise vectors w and v. You specify the mean, auto-covariance, and cross-covariance of these vectors. You must manually select the polymorphic instance to use.

Use this VI to generate values for the **Process Noise w(k)** and **Measurement Noise v(k)** inputs of the <u>CD Discrete Stochastic State-Space</u> function.

•

Use the pull-down menu to select an instance of this VI.

Select a	n instance
----------	------------

CD Correlated Gaussian Random Noise (One Vector)



- E{x} specifies the mean of the Gaussian random vector x. The length of E{x} determines the length of the random vector sample x and the dimensions of the Cov{x,x} matrix.
- **Cov{x,x}** specifies the covariance of the Gaussian-distributed random vector **x**. This covariance matrix can be either diagonal or non-diagonal, which means that samples of **x** can be uncorrelated or correlated with each other. If *n* is the length of $E{x}$, the $Cov{x,x}$ must be an *n* × *n* matrix.

The **Cov{x,x}** matrix must be symmetric and positive semi-definite such that $Cov{x,x} = Cov{x,x}^T \ge 0$.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- random vector sample x returns a random sample of the Gaussian random vector x. The length of random vector sample x is equal to the length of the E{x} vector.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Correlated Gaussian Random Noise (Two Vectors)



- E{x} specifies the mean of the Gaussian random vector x. The length of E{x} determines the length of the random vector sample x and the dimensions of the Cov{x,x} matrix.
- E{y} specifies the mean of the Gaussian random vector y. The length of E{y} determines the length of the random vector sample y and the dimensions of the Cov{y,y}.
- **Cov{x,x}** specifies the covariance of the Gaussian-distributed random vector **x**. This covariance matrix can be either diagonal or non-diagonal, which means that samples of **x** can be uncorrelated or correlated with each other. If *n* is the length of **E{x}**, the **Cov{x,x}** must be an $n \times n$ matrix.

The **Cov{x,x}** matrix must be symmetric and positive semi-definite such that $Cov{x,x} = Cov{x,x}^T \ge 0$.

Cov{y,y} specifies the covariance of the Gaussian-distributed random vector **y**. This covariance matrix can be either diagonal or non-diagonal, which means that samples of **y** can be uncorrelated or correlated with each other. If *m* is the length of **E{y}**, the **Cov{y,y}** matrix must be an *m* × *m* matrix.

The **Cov{y,y}** matrix must be symmetric and positive semi-definite such that $Cov{y,y} = Cov{y,y}^{T} \ge 0$.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Cov{x,y}** specifies the cross-covariance between the Gaussian random vectors \mathbf{x} and \mathbf{y} . If *n* is the length of **E{x}** and *m* is the length of **E{y}**, the **Cov{x,y}** matrix must be an *n* × *m* matrix.

The **Cov{x,y}** matrix also must satisfy the following relationship:

```
\begin{bmatrix} \boldsymbol{C}\boldsymbol{o}\boldsymbol{v}\{\boldsymbol{x},\,\boldsymbol{x}\} & \boldsymbol{C}\boldsymbol{o}\boldsymbol{v}\{\boldsymbol{x},\,\boldsymbol{y}\} \\ \boldsymbol{C}\boldsymbol{o}\boldsymbol{v}\{\boldsymbol{x},\,\boldsymbol{y}\}^{\mathsf{T}} & \boldsymbol{C}\boldsymbol{o}\boldsymbol{v}\{\boldsymbol{y},\,\boldsymbol{y}\} \end{bmatrix} \geq \boldsymbol{0}
```

If you specify a matrix of zeros for the $Cov{x,y}$ matrix, samples of the random vectors x and y are uncorrelated.

- random vector sample x returns a random sample of the Gaussian random vector x. The length of random vector sample x is equal to the length of the E{x} vector.
- **random vector sample y** returns a random sample of the Gaussian random vector **y**. The length of **random vector sample y** is equal to the length of the **E**{**y**} vector.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Data from Noise Model VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Obtains data that describes the statistics of the specified noise model.

Place on the block diagram \blacksquare Find on the **Functions** palette

- Second-Order Statistics Noise Model specifies a <u>mathematical</u> <u>representation</u> of the noise model of a stochastic state-space model.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **E{w}** returns the mean vector of the process noise vector w.
- **E** $\{v\}$ returns the mean vector of the measurement noise vector v.

[DBL]

Q returns the auto-covariance matrix of **w**.

- **R** returns the auto-covariance matrix of v.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **N** returns the cross-covariance matrix between **w** and **v**. If **w** and **v** are uncorrelated, **N** returns a matrix of zeros.

CD Get Data from Stochastic Model VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Obtains data that describes the dynamics of a stochastic state-space system model.



- **Stochastic State-Space Model In** specifies a <u>mathematical</u> <u>representation</u> of a stochastic system.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **A** returns the system matrix that describes the dynamic of the states of the system.
- **B** returns the input matrix that relates the inputs to the states.
- **C** returns the output matrix that relates the outputs to the states.
- **D** returns the transmission matrix that relates the inputs to the outputs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **G** returns a matrix that relates the process noise vector to the states.
- **H** returns a matrix that relates the process noise vector to the outputs.

CD Get Sampling Time from Stochastic Model VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Obtains the sampling time of a stochastic state-space model. The sampling time is zero for continuous systems and greater than zero for discrete systems.

Stochastic State-Space Model	Sampling Time (s)
error in (no error)	error out

■ Place on the block diagram ■ Find on the **Functions** palette

- Stochastic State-Space Model In specifies a <u>mathematical</u> representation of a stochastic system.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

DBL

Sampling Time returns the sampling time this VI associates with the system model. **Sampling Time** defines whether the model represents a continuous-time system or a discrete-time system. If the model represents a continuous-time system, **Sampling Time** equals zero. If the model represents a discrete-time system, **Sampling Time** is greater than zero and equal to the sampling rate, in seconds, of the discrete system.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Get Stochastic System Dimensions VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Obtains the number of states, inputs, and outputs of the stochastic statespace system model. This VI also returns the dimensions of the process noise vector w.



Note This VI does not verify the properties of the stochastic statespace model. You can use the <u>CD Verify Stochastic Model</u> <u>Properties</u> VI to perform this verification.



- Stochastic State-Space Model In specifies a <u>mathematical</u> representation of a stochastic system.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Number of States** returns the number of states in the system model.
- **Number of Inputs** returns the number of inputs in the system model.
- **Number of Outputs** returns the number of outputs in the system model.
- **Dimension of Process Noise** returns the length of the process noise vector **w**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Verify Noise Model VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Determines if the dimensions of the **Second-Order Statistics Noise Model** matrices are consistent with the dimensions of the **Stochastic State-Space Model** matrices. This VI also calculates the number of inputs, outputs, and states, and the dimensions of the process noise vector *w*.



- Second-Order Statistics Noise Model In specifies a mathematical representation of the noise model of the Stochastic State-Space Model.
- Stochastic State-Space Model In specifies a <u>mathematical</u> representation of a stochastic system.
- **Required Matrices** specifies the system matrices that this VI requires to continue execution.

0	None (default)
1	Α
2	В
3	А, В
4	С
5	A, C
6	B, C
7	A, B, C
8	D
9	A, D

10	B, D
11	A, B, D
12	C, D
13	A, C, D
14	B, C, D
15	A, B, C, D

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- TFI status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- 132 code is the error or warning code. The default is 0. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
- abc **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Second-Order Statistics Noise Model Out returns a mathematical representation of the noise model of the Stochastic State-Space Model.
- Stochastic State-Space Model Out returns the model after this VI verifies the stochastic properties.
- 132 Number of States returns the number of states in the system

model.

- **Number of Inputs** returns the number of inputs in the system model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Number of Outputs** returns the number of outputs in the system model.
- **Dimension of Process Noise** returns the length of the process noise vector **w**.

CD Verify Stochastic Model Properties VI

Owning Palette: <u>Stochastic Systems VIs</u>

Installed With: Control Design and Simulation Module

Determines if the dimensions of the stochastic state-space system matrices are consistent with each other. This VI also calculates the number of inputs, outputs, and states, as well as the dimensions of the process noise vector w.



- **Stochastic State-Space Model In** specifies a <u>mathematical</u> <u>representation</u> of a stochastic system.
- **Required Matrices** specifies the system matrices that this VI requires to continue execution.

0	None (default)
1	Α
2	В
3	А, В
4	С
5	A, C
6	B, C
7	A, B, C
8	D
9	A, D
10	B, D
11	A, B, D
12	C, D
13	A, C, D

14	B, C, D
15	A, B, C, D

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Stochastic State-Space Model Out returns the model after this VI verifies the stochastic properties.
- **Number of States** returns the number of states in the system model.
- **Number of Inputs** returns the number of inputs in the system model.
- Number of Outputs returns the number of outputs in the system model.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Dimension of Process Noise** returns the length of the process noise vector **w**.

Time Response VIs

Owning Palette: Control Design VIs and Functions

Installed With: Control Design and Simulation Module. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Time Response VIs to create generic linear simulations and time domain plots for step inputs, impulse inputs, and initial condition responses.

The VIs on this palette can return <u>general LabVIEW error codes</u> or specific <u>control design error codes</u>.

Palette Object	Description
<u>CD Get</u> <u>Time</u> <u>Response</u> <u>Data</u>	Obtains the time response information from the time response data of a system model. You must <u>manually select</u> <u>the polymorphic instance</u> to use.
<u>CD</u> Impulse Response	Calculates the output of the system when an impulse excites it. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD Initial</u> <u>Response</u>	Calculates the natural or zero-input response of the system. This VI returns the output of the system due to the initial states only. You can use polymorphic VI to calculate the initial response of state-space, transfer function, and zero- pole-gain models. The data type you wire to the State- Space Model input determines the polymorphic instance to use.
	Calculates the output when the Inputs excite the given system using discrete simulation. The data type you wire to the State-Space Model input determines the polymorphic instance to use.
<u>CD</u> <u>Parametric</u> <u>Time</u> <u>Response</u>	Calculates parametric information, such as rise time, peak time, settling time, steady-state gain, overshoot, and peak value of an input model based on a time response you specify. The data type you wire to the State-Space Model or Time Range parameter determines the polymorphic

	instance to use.
	Calculates the output of the system when a step input
Response	excites it. This VI assumes the initial states of the system to
	be zero. The data type you wire to the State-Space Model
	input determines the polymorphic instance to use.

CD Get Time Response Data VI

Owning Palette: <u>Time Response VIs</u>

Installed With: Control Design and Simulation Module

Obtains the time response information from the time response data of a system model. You must <u>manually select the polymorphic instance</u> to use.

You also can use this polymorphic VI to return the time response data in a <u>waveform data type</u>.

•

The <u>Input-Output Pair</u> instance returns the time response data for a specific input-output pair.

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

Get Time Response Data (Input-Output Pair)



- **Time Response Data** contains information about the time response of a model. Refer to the <u>Details</u> section for more information about the time response data.
 - **Time** is the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** contains data about the time response of the outputs to the inputs.
 - **States Data** contains data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty.
- **Input** determines the index number of each input for which you want to obtain data.
- **Output** determines the index number of each output for which you want to obtain data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Type of Response Data** specifies the type of response data you obtain.

0 **Outputs** (default)—Specifies to obtain the time response of the input-output pair(s).

- 1 **State**—Specifies to obtain the time response of the input-state pair(s) for a state-space model. If you specify this **Type of Response Data** for a transfer function or zero-pole-gain model, this VI returns the time response data as an empty array.
- **Time** returns the time at which this VI generates the response data. **Time** is the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
- **Response Data** returns a vector that provides the time response data for the input-output pair you select.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

Get Time Response Data (Input-Output List)

The Input-Output List instance returns the time response data for multiple input-output pairs.



- **Time Response Data** contains information about the time response of a model. Refer to the <u>Details</u> section for more information about the time response data.
 - **Time** is the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** contains data about the time response of the outputs to the inputs.
 - **States Data** contains data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty.
- **Inputs** specifies a list of index numbers of the inputs for which you want to obtain data. The index is zero-based.
- **Outputs** specifies a list of index numbers of the outputs for which you want to obtain data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or

function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Type of Response Data** specifies the type of response data you obtain.
 - 0 **Outputs** (default)—Specifies to obtain the time response of the input-output pair(s).
 - 1 **State**—Specifies to obtain the time response of the input-state pair(s) for a state-space model. If you specify this **Type of Response Data** for a transfer function or zero-pole-gain model, this VI returns the time response data as an empty array.
- **Time** returns the time at which this VI generates the response data. **Time** is the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
- **Response Data** returns a matrix that provides the time response data for all the input-output pairs you select. The first element in the matrix is the response to the first input and first output. The second element is the response to the first input and second output. After listing all input-output pairs this VI associates with the first input, the matrix lists the input-output pairs that this VI associates with the second input, then third input, and so on until the matrix lists all responses.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Get Time Response Data (Waveform Datatype)

The Waveform Datatype instance returns the time response data for an input-output pair in a waveform data type. You then can use the time response data with the <u>Waveform</u> VIs and functions.



- **Time Response Data** contains information about the time response of a model. Refer to the <u>Details</u> section for more information about the time response data.
 - **Time** is the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** contains data about the time response of the outputs to the inputs.
 - **States Data** contains data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty.
- **Input** determines the index number of each input for which you want to obtain data.
- **Output** determines the index number of each output for which you want to obtain data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Type of Response Data** specifies the type of response data you obtain.

0 **Outputs** (default)—Specifies to obtain the time response of the input-output pair(s).

1 **State**—Specifies to obtain the time response of the input-state pair(s) for a state-space model. If you specify this **Type of Response Data** for a transfer function or zero-pole-gain model, this VI returns the time response data as an empty array.

- **WDT Data** returns the time response data for the input-output pair you select. This VI starts the time response Y at time t_0 and evaluates it at a sample time dt.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in
most cases, the name of the VI or function that produced the error or warning.

Get Time Response Data (All Input-Output)

The All Input-Output instance returns the time response data for all inputoutput pairs.



- **Time Response Data** contains information about the time response of a model. Refer to the <u>Details</u> section for more information about the time response data.
 - **Time** is the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** contains data about the time response of the outputs to the inputs.
 - **States Data** contains data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Time** returns the time at which this VI generates the response data. **Time** is the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
- **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
- **States Data** returns data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty. Refer to the <u>Details</u> section for more information about the **States Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **Response Data** returns the type of response data you obtain.



2	nichols
3	nyquist
4	singular values
5	step
6	impulse
7	initial
8	simulation

CD Get Time Response Data Details

The **Time Response Data** data type contains information about the time response of all the input-output pairs in the model. You can use this VI to customize how you want to view this information. The following explanation describes how to view and understand the time response data.

The **Time** is an array of timestamps this VI records at a certain interval. The **Outputs Data** is a three-dimensional array with three index displays. The first index display is the index number of the inputs. The third index display is the index number of the outputs. The data that **Outputs Data** displays is based on the input-output pair these two index displays specify. The second index display is the index number of the response data. This index display is relative to the index display of the **Time**. The indexes are zero-based.

For example, consider the following **Time Response Data**.



If you want to know the time response at t = 1, set the index display for **Outputs Data** equal to the index display for **Time**. In this example, the index display is equal to 5. The **Outputs Data** then indicates that for the input-output pair (1, 0), the response at that time is 0.74.

This VI organizes the **States Data** the same as **Outputs Data**. The only difference is the third index display, which is the index number of the

states. The data that **States Data** displays is based on the input-state pair the first and third index displays specify.

CD Impulse Response VI

Owning Palette: <u>Time Response VIs</u>

Installed With: Control Design and Simulation Module

Calculates the output of the system when an impulse excites it. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Note In the <u>state-space</u> version of the CD Impulse Response VI, the **State Trajectory Graph** is not available when the model contains transport delays. You can reduce the nonzero delay entries in the transport delay matrix with the <u>CD Distribute Delay</u> VI. However, distributing the transport delay does not accurately reflect the effect of the delay in the states. Therefore, when transport delays are present in state-space models, the CD Impulse Response VI eliminates the state trajectories.

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

Impulse Response (State-Space)



- States Graph Ref is a reference to the State Trajectories graph. States Graph Ref configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Response Graph Ref** is a reference to the **Impulse Response** graph. **Response Graph Ref** configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI calculates impulse response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is –1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Response Plots Index specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Initial Response graph. The index is zero-based.
 - **Output** # is the index number of the specific output of the system that this VI displays on the **Initial Response** graph. The index is zero-based.
- **State Plots Index** specifies the index number of the inputs and outputs of the system.
 - Input # is the number of the specific input to the system due to which this VI displays the state trajectories in the State Trajectories graph. The index is zero-based.
 - **State** # is the index number of the specific state of the system that is displayed in the **Impulse Response** graph. The index is zero-based.
- **Impulse Response Graph** displays an XY graph that shows the forced response of the system when the forcing function is an

impulse. For MIMO systems, this VI determines the impulse response by applying an impulse on one input at a time and letting other inputs to the system equal zero.

- **State Trajectory Graph** displays an XY graph that shows the value of each state as a function of time. For MIMO systems, this VI calculates the states for each input at a time, where one input is the impulse and all the other inputs to the system are zero.
- Impulse Response Data returns information about the impulse response. To access the Impulse Response Data, use the CD Get Time Response Data VI.
 - **Time** is the uniformly-spaced time vector against which this VI plots the impulse response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
 - **States Data** returns data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty. Refer to the <u>Details</u> section for more information about the **States Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Impulse Response (Transfer Function)



- Response Graph Ref is a reference to the Impulse Response graph. Response Graph Ref configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI calculates impulse response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is -1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Response Plots Index** specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Initial Response graph. The index is zero-based.
 - **Output** *#* is the index number of the specific output of the system that this VI displays on the **Initial Response** graph. The index is zero-based.
- **Impulse Response Graph** displays an XY graph that shows the forced response of the system when the forcing function is an impulse. For MIMO systems, this VI determines the impulse response by applying an impulse on one input at a time and letting other inputs to the system equal zero.
- Impulse Response Data returns information about the impulse response. To access the Impulse Response Data, use the <u>CD</u> <u>Get Time Response Data</u> VI.
 - **Time** is the uniformly-spaced time vector against which this VI plots the impulse response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Impulse Response (Zero-Pole-Gain)



- **Response Graph Ref** is a reference to the **Impulse Response** graph. **Response Graph Ref** configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI calculates impulse response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is -1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Response Plots Index** specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Initial Response graph. The index is zero-based.
 - **Output** *#* is the index number of the specific output of the system that this VI displays on the **Initial Response** graph. The index is zero-based.
- **Impulse Response Graph** displays an XY graph that shows the forced response of the system when the forcing function is an impulse. For MIMO systems, this VI determines the impulse response by applying an impulse on one input at a time and letting other inputs to the system equal zero.
- Impulse Response Data returns information about the impulse response. To access the Impulse Response Data, use the <u>CD</u> <u>Get Time Response Data</u> VI.
 - **Time** is the uniformly-spaced time vector against which this VI plots the impulse response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Impulse Response Details

This VI supports input and output delays. This VI ignores transport delay information. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Initial Response VI

Owning Palette: <u>Time Response VIs</u>

Installed With: Control Design and Simulation Module

Calculates the natural or zero-input response of the system. This VI returns the output of the system due to the initial states only. You can use polymorphic VI to calculate the initial response of <u>state-space</u>, <u>transfer</u> <u>function</u>, and <u>zero-pole-gain</u> models. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Note In the <u>state-space</u> version of the CD Initial Response VI, the **State Trajectory Graph** is not available when the model contains transport delays. You can reduce the nonzero delay entries in the transport delay matrix with the <u>CD Distribute Delay</u> VI. However, distributing the transport delay does not accurately reflect the effect of the delay in the states. Therefore, when transport delays are present in state-space models, the CD Initial Response VI eliminates the state trajectories.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

Place on the block diagram Find on the Functions palette

Initial Response (State-Space)



- States Graph Ref is a reference to the State Trajectories graph. States Graph Ref configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Response Graph Ref is a reference to the Initial Response graph. Response Graph Ref customizes the x-scale, y-scale, and legend properties. If you want to use the default settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI calculates initial response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is -1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error

status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Response Plots Index specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Initial Response graph. The index is zero-based.
 - **Output** # is the index number of the specific output of the system that this VI displays on the **Initial Response** graph. The index is zero-based.
- **State Plots Index** specifies the index number of the inputs and outputs of the system.
 - Input # is the number of the specific input to the system due to which this VI displays the state trajectories in the State Trajectories graph. The index is zero-based.
 - **State** # is the index number of the specific state of the system that is displayed in the **Impulse Response** graph. The index is zero-based.
- **Initial Response Graph** returns an XY graph that shows the unforced response of the system when the initial conditions are X0.

- **State Trajectory Graph** is an XY graph that shows the value of each state as a function of time. This VI computes the magnitude by solving the first order difference equation of the system.
- Initial Response Data returns information about the initial response. To access the Initial Response Data, use the <u>CD Get</u> <u>Time Response Data</u> VI. When you use the CD Get Time Response Data VI to access the Initial Response Data, you do not need to specify an input value.
 - **Time** is the uniformly-spaced time vector against which this VI plots the initial response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
 - **States Data** returns data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Initial Response (Transfer Function)

This VI calculates the unforced response of a transfer function system given by **Transfer Function Model** and **Initial Condition**.



- **Response Graph Ref** is a reference to the **Initial Response** graph. **Response Graph Ref** customizes the x-scale, y-scale, and legend properties. If you want to use the default settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI calculates initial response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is -1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check

errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Response Plots Index specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Initial Response graph. The index is zero-based.
 - **Output** # is the index number of the specific output of the system that this VI displays on the **Initial Response** graph. The index is zero-based.
- **Initial Response Graph** returns an XY graph that shows the unforced response of the system when the initial conditions are X0.
- Initial Response Data returns information about the initial response. To access the Initial Response Data, use the <u>CD Get</u> <u>Time Response Data</u> VI. When you use the CD Get Time Response Data VI to access the Initial Response Data, you do not need to specify an input value.
 - **Time** is the uniformly-spaced time vector against which this VI plots the initial response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Initial Response (Zero-Pole-Gain)

This VI calculates the unforced response of a zero-pole-gain system given by **Zero-Pole-Gain Model** and **Initial Condition**.



- **Response Graph Ref** is a reference to the **Initial Response** graph. **Response Graph Ref** customizes the x-scale, y-scale, and legend properties. If you want to use the default settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system for which this VI calculates initial response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is –1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check

errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Response Plots Index specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Initial Response graph. The index is zero-based.
 - **Output** # is the index number of the specific output of the system that this VI displays on the **Initial Response** graph. The index is zero-based.
- **Initial Response Graph** returns an XY graph that shows the unforced response of the system when the initial conditions are X0.
- Initial Response Data returns information about the initial response. To access the Initial Response Data, use the <u>CD Get</u> <u>Time Response Data</u> VI. When you use the CD Get Time Response Data VI to access the Initial Response Data, you do not need to specify an input value.
 - **Time** is the uniformly-spaced time vector against which this VI plots the initial response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Initial Response Details

This VI supports input and output delays. This VI ignores transport delay information. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Linear Simulation VI

Owning Palette: <u>Time Response VIs</u>

Installed With: Control Design and Simulation Module

Calculates the output when the **Inputs** excite the given system using discrete simulation. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.



Note The CD Linear Simulation VI ignores the transport delay in a system. You can reduce the nonzero delay entries in the transport delay matrix with the <u>CD Distribute Delay</u> VI. However, the system inputs and outputs cannot accommodate all delay elements. Therefore, the CD Linear Simulation VI ignores the residue, or transport delay.

•

Details

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

Linear Simulation (State-Space)



Delta t is the integration time step.

- **Initial Time** represents the time at the beginning of the simulation.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates a response.
- **Inputs** is the stimulus signal this VI applies to the system model. When you use this VI with a discrete model, you must ensure that the time step of the model matches the time step of this input signal.

If you have only one input signal, specify **Inputs** as a 2D array with only one row or column. If you have two or more input signals, this VI considers the smaller dimension of the 2D array to be the number of channels and the larger dimension to be the number of points in each channel.

- **Initial Conditions** is the initial state vector. The *i*th element of the array corresponds to the *i*th initial state. If **Initial Conditions** is empty, then this VI uses zero initial conditions to determine the time response.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check

errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Method specifies the method this VI uses to convert a continuous system to a discrete system before simulating the output. This VI uses the sampling interval (dt) from the **Inputs** signal.

0 **ZOH** (default)—Zero-Order-Hold

1 **FOH**—First-Order-Hold

- **Simulation Graph** displays all the outputs resulting from inputs to the system.
- **State Trajectory Graph** displays the individual trajectories of each state of the system.
- Simulation Data returns the outputs and state data from the linear simulation. To access the Simulation Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
 - **States Data** returns data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Linear Simulation (Transfer Function)



Delta t is the integration time step.

- **Initial Time** represents the time at the beginning of the simulation.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates a response.
- **Inputs** is the stimulus signal this VI applies to the system model. When you use this VI with a discrete model, you must ensure that the time step of the model matches the time step of this input signal.

If you have only one input signal, specify **Inputs** as a 2D array with only one row or column. If you have two or more input signals, this VI considers the smaller dimension of the 2D array to be the number of channels and the larger dimension to be the number of points in each channel.

- **Initial Conditions** is the initial state vector. The *i*th element of the array corresponds to the *i*th initial state. If **Initial Conditions** is empty, then this VI uses zero initial conditions to determine the time response.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check

errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Method specifies the method this VI uses to convert a continuous system to a discrete system before simulating the output. This VI uses the sampling interval (dt) from the **Inputs** signal.

0 **ZOH** (default)—Zero-Order-Hold

1 **FOH**—First-Order-Hold

- **Simulation Graph** displays all the outputs resulting from inputs to the system.
- Simulation Data returns the outputs and state data from the linear simulation. To access the Simulation Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

TF

status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Linear Simulation (Zero-Pole-Gain)



Delta t is the integration time step.

- **Initial Time** represents the time at the beginning of the simulation.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates a response.
- **Inputs** is the stimulus signal this VI applies to the system model. When you use this VI with a discrete model, you must ensure that the time step of the model matches the time step of this input signal.

If you have only one input signal, specify **Inputs** as a 2D array with only one row or column. If you have two or more input signals, this VI considers the smaller dimension of the 2D array to be the number of channels and the larger dimension to be the number of points in each channel.

- **Initial Conditions** is the initial state vector. The *i*th element of the array corresponds to the *i*th initial state. If **Initial Conditions** is empty, then this VI uses zero initial conditions to determine the time response.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check
errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Method specifies the method this VI uses to convert a continuous system to a discrete system before simulating the output. This VI uses the sampling interval (dt) from the **Inputs** signal.

0 **ZOH** (default)—Zero-Order-Hold

1 **FOH**—First-Order-Hold

- **Simulation Graph** displays all the outputs resulting from inputs to the system.
- Simulation Data returns the outputs and state data from the linear simulation. To access the Simulation Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

TF

status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Linear Simulation (State-Space with Waveforms)

State-Space Model	
Inputs	State Trajectories
Initial Conditions	Simulation Data
error in	error out
Method	

- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates a response.
- Inputs is an array of waveforms. There must be as many waveforms in the array as the number of inputs to the system. The size of Inputs must equal the number of columns of matrices B and D of the State-Space Model. When you use this VI with a discrete model, you must ensure that the time step of the model matches the time step of this input signal.
- **Initial Conditions** is the initial state vector. The *i*th element of the array corresponds to the *i*th initial state. If **Initial Conditions** is empty, then this VI uses zero initial conditions to determine the time response.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Method specifies the method this VI uses to convert a continuous system to a discrete system before simulating the output. This VI uses the sampling interval (dt) from the **Inputs** signal.

0 **ZOH** (default)—Zero-Order-Hold 1 **FOH**—First-Order-Hold

- •••• **Outputs** is a waveform graph that plots all the outputs resulting from **Inputs** to the system as the model represents.
- **State Trajectories** is a waveform graph that shows the individual trajectories of each state of the system.
- Simulation Data returns the outputs and state data from the linear simulation. To access the Simulation Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
 - **States Data** returns data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **Status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Linear Simulation (Transfer Function with Waveforms)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates a response.
- Inputs is an array of waveforms. There must be as many waveforms in the array as the number of inputs to the system. The size of Inputs must equal the number of columns of matrices B and D of the State-Space Model. When you use this VI with a discrete model, you must ensure that the time step of the model matches the time step of this input signal.
- **Initial Conditions** is the initial state vector. The *i*th element of the array corresponds to the *i*th initial state. If **Initial Conditions** is empty, then this VI uses zero initial conditions to determine the time response.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Method specifies the method this VI uses to convert a continuous system to a discrete system before simulating the output. This VI uses the sampling interval (dt) from the **Inputs** signal.

```
0 ZOH (default)—Zero-Order-Hold
1 FOH—First-Order-Hold
```

- •••• **Outputs** is a waveform graph that plots all the outputs resulting from **Inputs** to the system as the model represents.
- Simulation Data returns the outputs and state data from the linear simulation. To access the Simulation Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

Linear Simulation (Zero-Pole-Gain with Waveforms)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates a response.
- **Inputs** is the stimulus signal this VI applies to the system model. When you use this VI with a discrete model, you must ensure that the time step of the model matches the time step of this input signal.

If you have only one input signal, specify **Inputs** as a 2D array with only one row or column. If you have two or more input signals, this VI considers the smaller dimension of the 2D array to be the number of channels and the larger dimension to be the number of points in each channel.

- **Initial Conditions** is the initial state vector. The *i*th element of the array corresponds to the *i*th initial state. If **Initial Conditions** is empty, then this VI uses zero initial conditions to determine the time response.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Method specifies the method this VI uses to convert a continuous system to a discrete system before simulating the output. This VI uses the sampling interval (dt) from the **Inputs** signal.

0 **ZOH** (default)—Zero-Order-Hold 1 **FOH**—First-Order-Hold

- •••• Outputs is a waveform graph that plots all the outputs resulting from Inputs to the system as the model represents.
- Simulation Data returns the outputs and state data from the linear simulation. To access the Simulation Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a

warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Linear Simulation Details

This VI supports input and output delays. This VI ignores the transport delay information. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Parametric Time Response VI

Owning Palette: <u>Time Response VIs</u>

Installed With: Control Design and Simulation Module

Calculates parametric information, such as rise time, peak time, settling time, steady-state gain, overshoot, and peak value of an input model based on a time response you specify. The data type you wire to the **State-Space Model** or **Time Range** parameter determines the polymorphic instance to use.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

CD Parametric Time Response (State-Space Internal)



Type of Response Data specifies whether you want this VI to return the time response data of the model states or model outputs.

0 **Outputs** (default)—Specifies that you want this VI to return the time response data of the model outputs.

1 **State trajectories**— Specifies that you want this VI to return the time response data of the model states.

- **Type of Analysis** specifies the type of time response analysis this VI performs on the model.
 - 0 **Step Response** (default)—Specifies this VI uses a step response to obtain the parametric information.
 - 1 **Impulse Response**—Specifies this VI uses an impulse response to obtain the parametric information.
 - 2 **Initial Response**—Specifies this VI uses an initial response to obtain the parametric information.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates parametric information.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is –1.
 - **tf** is the final time in seconds up to which this VI calculates

the step response. The default is -1.

- **Initial Conditions** are the initial values the parametric response uses.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Rise Time Thresholds (%)** specifies the lower and upper thresholds that define the rise time this VI returns. By default, the rise time is the time required for the system response to rise from 10% to 90% of the settling point.
 - **Lower** specifies the lower limit of the rise time threshold. The default value is 10%.
 - **Upper** specifies the upper limit of the rise time threshold. The default value is 90%.
- **Settling Time Threshold (%)** defines the percentage in which the signal must fall to be within range of its steady state value. The

default is 1%. Therefore, the settling time is the time required for the signal to fall within a 1% range of its steady state value.

- **Time Response Graph** displays an XY graph containing the time response of the model.
- **Time Response Parametric Data** returns the parametric time response data this VI measures from the **Time Response Data**.
 - **Rise Time (s)** is the time required for the dynamic system response to rise from 10% of its final value to 90% of its final value.
 - **Peak Time (s)** is the time required for the dynamic system response to reach the peak value of its first overshoot.
 - **Settling Time (s)** is the time required for the response to reach 1% of its final value.
 - **Overshoot** is the dynamic system response value that most exceeds unity, expressed as a percent.
 - **Steady-State Gain** is the final value of the signal after transient responses decay.
 - **Peak value** returns the value at which the maximum absolute value of the time response occurs.
- Time Response Data returns the data before this VI parameterizes it. To access the Time Response Data, use the <u>CD</u> <u>Get Time Response Data</u> VI.
 - **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
 - **States Data** returns data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty. Refer to the <u>Details</u> section for more information about the **States Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parametric Time Response (Transfer Function Internal)



Type of Response Data specifies whether you want this VI to return the time response data of the model states or model outputs.

0 **Outputs** (default)—Specifies that you want this VI to return the time response data of the model outputs.

1 **State trajectories**— Specifies that you want this VI to return the time response data of the model states.

- **Type of Analysis** specifies the type of time response analysis this VI performs on the model.
 - 0 **Step Response** (default)—Specifies this VI uses a step response to obtain the parametric information.
 - 1 **Impulse Response**—Specifies this VI uses an impulse response to obtain the parametric information.
 - 2 **Initial Response**—Specifies this VI uses an initial response to obtain the parametric information.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates parametric information.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is –1.
 - **tf** is the final time in seconds up to which this VI calculates

the step response. The default is -1.

- **Initial Conditions** are the initial values the parametric response uses.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Rise Time Thresholds (%)** specifies the lower and upper thresholds that define the rise time this VI returns. By default, the rise time is the time required for the system response to rise from 10% to 90% of the settling point.
 - **Lower** specifies the lower limit of the rise time threshold. The default value is 10%.
 - **Upper** specifies the upper limit of the rise time threshold. The default value is 90%.
- **Settling Time Threshold (%)** defines the percentage in which the signal must fall to be within range of its steady state value. The

default is 1%. Therefore, the settling time is the time required for the signal to fall within a 1% range of its steady state value.

- **Time Response Graph** displays an XY graph containing the time response of the model.
- **Time Response Parametric Data** returns the parametric time response data this VI measures from the **Time Response Data**.
 - **Rise Time (s)** is the time required for the dynamic system response to rise from 10% of its final value to 90% of its final value.
 - **Peak Time (s)** is the time required for the dynamic system response to reach the peak value of its first overshoot.
 - **Settling Time (s)** is the time required for the response to reach 1% of its final value.
 - **Overshoot** is the dynamic system response value that most exceeds unity, expressed as a percent.
 - **Steady-State Gain** is the final value of the signal after transient responses decay.
 - **Peak value** returns the value at which the maximum absolute value of the time response occurs.
- Time Response Data returns the data before this VI parameterizes it. To access the Time Response Data, use the <u>CD</u> <u>Get Time</u>

Response Data VI.

- **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
- **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parametric Time Response (Zero-Pole-Gain Internal)



Type of Response Data specifies whether you want this VI to return the time response data of the model states or model outputs.

0 **Outputs** (default)—Specifies that you want this VI to return the time response data of the model outputs.

1 **State trajectories**— Specifies that you want this VI to return the time response data of the model states.

- **Type of Analysis** specifies the type of time response analysis this VI performs on the model.
 - 0 **Step Response** (default)—Specifies this VI uses a step response to obtain the parametric information.
 - 1 **Impulse Response**—Specifies this VI uses an impulse response to obtain the parametric information.
 - 2 **Initial Response**—Specifies this VI uses an initial response to obtain the parametric information.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates parametric information.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is -1.
 - **tf** is the final time in seconds up to which this VI calculates

the step response. The default is -1.

- **Initial Conditions** are the initial values the parametric response uses.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Rise Time Thresholds (%)** specifies the lower and upper thresholds that define the rise time this VI returns. By default, the rise time is the time required for the system response to rise from 10% to 90% of the settling point.
 - **Lower** specifies the lower limit of the rise time threshold. The default value is 10%.
 - **Upper** specifies the upper limit of the rise time threshold. The default value is 90%.
- **Settling Time Threshold (%)** defines the percentage in which the signal must fall to be within range of its steady state value. The

default is 1%. Therefore, the settling time is the time required for the signal to fall within a 1% range of its steady state value.

- **Time Response Graph** displays an XY graph containing the time response of the model.
- **Time Response Parametric Data** returns the parametric time response data this VI measures from the **Time Response Data**.
 - **Rise Time (s)** is the time required for the dynamic system response to rise from 10% of its final value to 90% of its final value.
 - **Peak Time (s)** is the time required for the dynamic system response to reach the peak value of its first overshoot.
 - **Settling Time (s)** is the time required for the response to reach 1% of its final value.
 - **Overshoot** is the dynamic system response value that most exceeds unity, expressed as a percent.
 - **Steady-State Gain** is the final value of the signal after transient responses decay.
 - **Peak value** returns the value at which the maximum absolute value of the time response occurs.
- Time Response Data returns the data before this VI parameterizes it. To access the Time Response Data, use the <u>CD</u> <u>Get Time</u>

Response Data VI.

- **Time** returns the uniformly-spaced time vector against which this VI plots the impulse, initial, or step response and the state trajectories.
- •••• Outputs Data returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parametric Time Response (State-Space External)



Type of Response Data specifies whether you want this VI to return the time response data of the model states or model outputs.

0 **Outputs** (default)—Specifies that you want this VI to return the time response data of the model outputs.

- 1 **State trajectories** Specifies that you want this VI to return the time response data of the model states.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates parametric information.
- **Time Response Data** specifies the time response data for which this VI calculates the parametric information.
 - **Time** specifies the uniformly-spaced time vector.
 - **Outputs Data** specifies data about the time response of the outputs to the inputs.
 - **States Data** specifies information about the time response of the states to the inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one

node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **Rise Time Thresholds (%)** specifies the lower and upper thresholds that define the rise time this VI returns. By default, the rise time is the time required for the system response to rise from 10% to 90% of the settling point.
 - **Lower** specifies the lower limit of the rise time threshold. The default value is 10%.
 - **Upper** specifies the upper limit of the rise time threshold. The default value is 90%.
- Settling Time Threshold (%) defines the percentage in which the signal must fall to be within range of its steady state value. The default is 1%. Therefore, the settling time is the time required for the signal to fall within a 1% range of its steady state value.
- **Time Response Parametric Data** returns the parametric time response data this VI measures from the **Time Response Data**.
 - **Rise Time (s)** is the time required for the dynamic system response to rise from 10% of its final value to 90% of its final value.
 - **Peak Time (s)** is the time required for the dynamic system response to reach the peak value of its first overshoot.
 - **Settling Time (s)** is the time required for the response to reach 1% of its final value.
 - **Overshoot** is the dynamic system response value that most exceeds unity, expressed as a percent.

- **Steady-State Gain** is the final value of the signal after transient responses decay.
- **Peak value** returns the value at which the maximum absolute value of the time response occurs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parametric Time Response (Transfer Function External)



- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates parametric information.
- **Time Response Data** specifies the time response data for which this VI calculates the parametric information.
 - **Time** specifies the uniformly-spaced time vector.
 - **Outputs Data** specifies data about the time response of the outputs to the inputs.
 - **States Data** specifies information about the time response of the states to the inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Rise Time Thresholds (%)** specifies the lower and upper thresholds that define the rise time this VI returns. By default, the rise time is the time required for the system response to rise from 10% to 90% of the settling point.
 - **Lower** specifies the lower limit of the rise time threshold. The default value is 10%.
 - **Upper** specifies the upper limit of the rise time threshold. The default value is 90%.
- Settling Time Threshold (%) defines the percentage in which the signal must fall to be within range of its steady state value. The default is 1%. Therefore, the settling time is the time required for the signal to fall within a 1% range of its steady state value.
- **Time Response Parametric Data** returns the parametric time response data this VI measures from the **Time Response Data**.
 - **Rise Time (s)** is the time required for the dynamic system response to rise from 10% of its final value to 90% of its final value.
 - **Peak Time (s)** is the time required for the dynamic system response to reach the peak value of its first overshoot.
 - **Settling Time (s)** is the time required for the response to reach 1% of its final value.
 - **Overshoot** is the dynamic system response value that most exceeds unity, expressed as a percent.
 - **Steady-State Gain** is the final value of the signal after transient responses decay.
 - **Peak value** returns the value at which the maximum absolute value of the time response occurs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parametric Time Response (Zero-Pole-Gain External)



- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates parametric information.
- **Time Response Data** specifies the time response data for which this VI calculates the parametric information.
 - **Time** specifies the uniformly-spaced time vector.
 - **Outputs Data** specifies data about the time response of the outputs to the inputs.
 - **States Data** specifies information about the time response of the states to the inputs.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

abc

source specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **Rise Time Thresholds (%)** specifies the lower and upper thresholds that define the rise time this VI returns. By default, the rise time is the time required for the system response to rise from 10% to 90% of the settling point.
 - **Lower** specifies the lower limit of the rise time threshold. The default value is 10%.
 - **Upper** specifies the upper limit of the rise time threshold. The default value is 90%.
- Settling Time Threshold (%) defines the percentage in which the signal must fall to be within range of its steady state value. The default is 1%. Therefore, the settling time is the time required for the signal to fall within a 1% range of its steady state value.
- **Time Response Parametric Data** returns the parametric time response data this VI measures from the **Time Response Data**.
 - **Rise Time (s)** is the time required for the dynamic system response to rise from 10% of its final value to 90% of its final value.
 - **Peak Time (s)** is the time required for the dynamic system response to reach the peak value of its first overshoot.
 - **Settling Time (s)** is the time required for the response to reach 1% of its final value.
 - **Overshoot** is the dynamic system response value that most exceeds unity, expressed as a percent.
 - **Steady-State Gain** is the final value of the signal after transient responses decay.
 - **Peak value** returns the value at which the maximum absolute value of the time response occurs.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Parametric Time Response Details

This VI supports delays when calculating the dynamic characteristics of a system. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.

CD Step Response VI

Owning Palette: <u>Time Response VIs</u>

Installed With: Control Design and Simulation Module

Calculates the output of the system when a step input excites it. This VI assumes the initial states of the system to be zero. The data type you wire to the **State-Space Model** input determines the polymorphic instance to use.

Note In the <u>state-space</u> version of the CD Step Response VI, the State Trajectory Graph is not available when the model contains transport delays. You can reduce the nonzero delay entries in the transport delay matrix with the <u>CD Distribute Delay</u> VI. However, distributing the transport delay does not accurately reflect the effect of the delay in the states. Therefore, when transport delays are present in state-space models, the CD Step Response VI eliminates the state trajectories.

•

<u>Details</u>

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

Step Response (State-Space)



- States Graph Ref is a reference to the State Trajectories graph. States Graph Ref configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- Response Graph Ref is a reference to the Step Response graph. Response Graph Ref configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **State-Space Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates step response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is -1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error

status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use <u>exception control</u> to treat what is normally an error as no error or to treat a warning as an error. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Response Plots Index specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Step Response graph. The index is zero-based.
 - **Output** # is the index number of the specific output of the system that this VI displays on the **Step Response** graph. The index is zero-based.
- **State Plots Index** specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Step Response graph. The index is zero-based.
 - **State** # is the index number of the specific state of the system that this VI displays in the **Step Response** graph. The index is zero-based.
- **Step Response Graph** displays a graph that shows the forced response of the system when the forcing function is a step. For MIMO systems, this VI determines the step response by applying a

step on one input at a time and letting other inputs to the system be zero.

- **State Trajectory Graph** displays an XY graph that shows the value of each state as a function of time. For MIMO systems, this VI calculates the states for each input at a time, where one input is the impulse and all the other inputs to the system are zero.
- Step Response Data returns information about the step response. To access the Step Response Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** is the uniformly-spaced time vector against which this VI plots the step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
 - **States Data** returns data about the time response of the states to the inputs. For transfer function and zero-pole-gain models, this array is empty. Refer to the <u>Details</u> section for more information about the **States Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Step Response (Transfer Function)



- Response Graph Ref is a reference to the Step Response graph. Response Graph Ref configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Transfer Function Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates step response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is -1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Response Plots Index specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Step Response graph. The index is zero-based.
 - **Output** # is the index number of the specific output of the system that this VI displays on the **Step Response** graph. The index is zero-based.
- **Step Response Graph** displays a graph that shows the forced response of the system when the forcing function is a step. For MIMO systems, this VI determines the step response by applying a step on one input at a time and letting other inputs to the system be zero.
- Step Response Data returns information about the step response. To access the Step Response Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** is the uniformly-spaced time vector against which this VI plots the step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Step Response (Zero-Pole-Gain)



- Response Graph Ref is a reference to the Step Response graph. Response Graph Ref configures the x-scale, y-scale, and legend properties. If you want to use the default settings or customize the settings for these properties, do not wire a value to this input.
- **Zero-Pole-Gain Model** contains a <u>mathematical representation</u> of and <u>information</u> about the system of which this VI calculates step response.
- **Time Range** contains information about the initial time, final time, and time step.
 - **t0** is the initial time in seconds in the **Step Response** graph. The default is –1.
 - **dt** is the constant interval between successive values in the time vector. The default is -1.
 - **tf** is the final time in seconds up to which this VI calculates the step response. The default is –1.
- **Initial Conditions** specifies the initial values of the states or outputs. The default is 0.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use exception control to treat what is normally an error as no error or to treat a warning as an error. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- Response Plots Index specifies the index number of the inputs and outputs of the system.
 - Input # is the index number of the specific input to the system. This VI displays the response to this input on the Step Response graph. The index is zero-based.
 - **Output** # is the index number of the specific output of the system that this VI displays on the **Step Response** graph. The index is zero-based.
- **Impulse Response Graph** displays an XY graph that shows the forced response of the system when the forcing function is an impulse. For MIMO systems, this VI determines the impulse response by applying an impulse on one input at a time and letting other inputs to the system equal zero.
- Step Response Data returns information about the step response. To access the Step Response Data, use the <u>CD Get Time</u> <u>Response Data</u> VI.
 - **Time** is the uniformly-spaced time vector against which this VI plots the step response and the state trajectories.
 - **Outputs Data** returns data about the time response of the outputs to the inputs. Refer to the <u>Details</u> section for more information about the **Outputs Data**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

CD Step Response Details

This VI supports input and output delays. This VI ignores transport delay information. Refer to the <u>LabVIEW Control Design User Manual</u> for more information about delays.