Here is a list of all modules:

- License Terms and Copyright Information
- Abbreviations and Definitions
- Overview
- Architecture Description
- APP Configuration Parameters
- Enumerations
- Data structures
- Methods
- Usage
- Release History
License Terms and Copyright Information

Copyright (c) 2016, Infineon Technologies AG All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of the copyright holders nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

To improve the quality of the software, users are encouraged to share modifications, enhancements or bug fixes with Infineon Technologies AG (dave@infineon.com).
## Abbreviations and Definitions

### Abbreviations:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAVE™</td>
<td>Digital Application Virtual Engineer</td>
</tr>
<tr>
<td>APP</td>
<td>DAVE™ Application</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>MCU</td>
<td>Microcontroller Unit</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>HW</td>
<td>Hardware</td>
</tr>
<tr>
<td>LLD</td>
<td>Low Level Driver</td>
</tr>
<tr>
<td>I/O</td>
<td>Input/Output</td>
</tr>
<tr>
<td>CCU</td>
<td>Capture Compare Unit</td>
</tr>
<tr>
<td>CC</td>
<td>Capture Compare Slice</td>
</tr>
<tr>
<td>SCU</td>
<td>System Control Unit</td>
</tr>
</tbody>
</table>

### Definitions:

<table>
<thead>
<tr>
<th>Definition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Singleton</td>
<td>Only single instance of the APP is permitted</td>
</tr>
<tr>
<td>Sharable</td>
<td>Resource sharing with other APPs is permitted</td>
</tr>
<tr>
<td>initProvider</td>
<td>Provides the initialization routine</td>
</tr>
<tr>
<td>Physical connectivity</td>
<td>Hardware inter/intra peripheral (constant) signal connection</td>
</tr>
<tr>
<td>Conditional connectivity</td>
<td>Constrained hardware inter/intra peripheral signal connection</td>
</tr>
<tr>
<td>Aggregation</td>
<td>Indicates consumption of low level (dependent)</td>
</tr>
</tbody>
</table>
The primary purpose of the GLOBAL_CCU4 APP is to share the common resources in the CCU4 module among top level APPs. It provides functions to accomplish the following:

1. Enable CCU4 module
2. Start CC4y (y = [0..3]) timers synchronously
3. Shadow transfer of selected values (in the GUI) in multi-channel mode
The figure above shows the layered architecture of the GLOBAL_CCU4 APP for DAVE™. The LLD layer provides an abstraction for the CCU4 hardware module. The GLOBAL_CCU4 APP uses CCU4 and SCU LLDs for its functionality.

**Supported Devices**
The APP supports the following devices:

1. XMC4800/XMC4700 Series
2. XMC4500 Series
3. XMC4400 Series
4. XMC4300 Series
5. XMC4200 / XMC4100 Series
6. XMC1400 Series
7. XMC1300 Series
8. XMC1200 Series
9. XMC1100 Series

References

1. XMC4800 / XMC4700 Reference Manual
2. XMC4500 Reference Manual
5. XMC4200 / XMC4100 Reference Manual
7. XMC1300 Reference Manual
The diagram above represents the internal software architecture of the GLOBAL_CCU4 APP. A GLOBAL_CCU4 APP instance exists in a DAVE™ project with fixed attributes as shown. Each instance of this APP configures one CCU4 module in the MCU. In addition, the APP requires the consumption of the CLOCK APP for its configuration and functioning. The GLOBAL_CCU4 APP also provides an output signal.
for inter-peripheral connections.

An instantiated APP (after code generation) generates a specific data structure with the GUI configuration. The name of this data structure can be modified by changing the APP instance label (e.g. change label from default GLOBAL_CCU4_0 to GLOBAL_CCU4_SVM).

**Signals:**

The following table presents the signals provided by the APP for inter-peripheral connections:

**Table 1: APP I/O signals**

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Input/Output</th>
<th>Availability</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ccu4_global_start</td>
<td>Output</td>
<td>Always</td>
<td>Global Synchronous Start: This signal can be used to connect with top level APPs like PWM to start the timers synchronously.</td>
</tr>
</tbody>
</table>

**APPs Consumed:**

The following table presents the APPs consumed to support the functionality:

**Table 2: APPs Consumed**

<table>
<thead>
<tr>
<th>APP Name</th>
<th>Consumption</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOCK_XMC4</td>
<td>Conditionally consumed. For XMC4000 devices.</td>
<td>Initialise the clock settings.</td>
</tr>
<tr>
<td>CLOCK_XMC1</td>
<td>Conditionally consumed. Initialise the clock settings. For XMC1000 devices.</td>
<td></td>
</tr>
</tbody>
</table>
GLOBAL_CCU4

APP Configuration Parameters

App Configuration Parameters

<table>
<thead>
<tr>
<th>General Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock frequency [MHz]:</td>
</tr>
<tr>
<td>Multi channel mode shadow transfer:</td>
</tr>
</tbody>
</table>

Figure 1: General Settings
### Enumerations

```c
enum GLOBAL_CCU4_STATUS
{
    GLOBAL_CCU4_STATUS_SUCCESS = 0U,
    GLOBAL_CCU4_STATUS_FAILURE
};
```

Return status of the **GLOBAL_CCU4** APP. More...

```c
typedef enum GLOBAL_CCU4_STATUS
```

Return status of the **GLOBAL_CCU4** APP.
Enumeration Type Documentation

enum GLOBAL_CCU4_STATUS

Return status of the GLOBAL_CCU4 APP.

**Enumerator:**

- $GLOBAL_CCU4\_STATUS\_SUCCESS$ Status success
- $GLOBAL_CCU4\_STATUS\_FAILURE$ Status failure

Definition at line 104 of file GLOBAL_CCU4.h.
<table>
<thead>
<tr>
<th>Home</th>
<th>Data Structures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data structures</td>
<td></td>
</tr>
</tbody>
</table>
## Data Structures

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>struct</td>
<td>GLOBAL_CCU4</td>
</tr>
<tr>
<td>typedef</td>
<td>struct GLOBAL_CCU4 GLOBAL_CCU4_t</td>
</tr>
</tbody>
</table>
Typedef Documentation

typedef struct GLOBAL_CCU4 GLOBAL_CCU4_t

This saves the context of the GLOBAL_CCU4 APP.
## GLOBAL_CCU4

### Methods

<table>
<thead>
<tr>
<th>DAVE_APP_VERSION_t</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GLOBAL_CCU4_GetAppVersion</strong> (void)</td>
</tr>
<tr>
<td>Get <strong>GLOBAL_CCU4</strong> APP version.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GLOBAL_CCU4_STATUS_t</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GLOBAL_CCU4_Init</strong> (GLOBAL_CCU4_t *handle)</td>
</tr>
<tr>
<td>Initializes a <strong>GLOBAL_CCU4</strong> with generated configuration.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>__STATIC_INLINE void</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GLOBAL_CCU4_SyncStartTriggerHigh</strong> (uint32_t ccucon_msk)</td>
</tr>
<tr>
<td>Start all the timers which are configured to start externally on positive edge.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>__STATIC_INLINE void</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GLOBAL_CCU4_SyncStartTriggerLow</strong> (uint32_t ccucon_msk)</td>
</tr>
<tr>
<td>Start all the timers which are configured to start externally on negative edge.</td>
</tr>
</tbody>
</table>
Function Documentation

DAVE_APP_VERSION_t GLOBAL_CCU4_GetAppVersion (void)

Get GLOBAL_CCU4 APP version.

Returns:
DAVE_APP_VERSION_t APP version information (major, minor and patch number)

Description:
The function can be used to check application software compatibility with a specific version of the APP.

Example Usage:

```c
#include <DAVE.h>
int main(void)
{
    DAVE_STATUS_t status;
    DAVE_APP_VERSION_t app_version;

    status = DAVE_Init();       // GLOBAL_CCU4_Init() is called from DAVE_Init()
    app_version = GLOBAL_CCU4_GetAppVersion();

    if (app_version.major != 4U)
    {
        // Probably, not the right version.
    }

    while(1U)
    {
    }

    return 1;
```
GLOBAL_CCU4_STATUS_t GLOBAL_CCU4_Init (GLOBAL_CCU4_t handle)

Initializes a GLOBAL_CCU4 with generated configuration.

**Parameters:**
- `handle` pointer to the GLOBAL_CCU4 APP handle structure.

**Returns:**
- `GLOBAL_CCU4_STATUS_t`
- `GLOBAL_CCU4_STATUS_SUCCESS` : if initialization is successful
- `GLOBAL_CCU4_STATUS_FAILURE` : if initialization is failed

**Description:**
- Enable the module.
- Start the prescaler.

**Example Usage:**

```c
#include <DAVE.h>
int main(void)
{
    DAVE_STATUS_t init_status;
    init_status = DAVE_Init(); // GLOBAL_CCU4_Init(&GLOBAL_CCU4_0) will be called from DAVE_Init()

    while(1)
    {
```
Definition at line 82 of file GLOBAL_CCU4.c.

References GLOBAL_CCU4_STATUS_SUCCESS, GLOBAL_CCU4::is_initialized, GLOBAL_CCU4::mcs_action, and GLOBAL_CCU4::module_ptr.

```c
__STATIC_INLINE void GLOBAL_CCU4_SyncStartTriggerHigh ( uint32_t ccucon_msk ) {
    // Start all the timers which are configured to start externally on positive edge.

    Parameters:
        ccucon_msk  mask for which kernels sync start has to be applied.

    Note:
        This mask has been generated in the APP handle and as a macro in global_ccu4_conf.h file. 1. The variable from the APP handle is useful while starting the specific kernel/s 2. GLOBAL_CCU4_CCUCON_Msk Macro from global_ccu4_conf.h file can be used to start all the selected kernels at a time.

    Return values:
        none

    Description:
        The top level APPs have to be enabled, to start the timer externally with positive trigger edge.
```
Example Usage:

```c
#include <DAVE.h>
int main(void) {
    DAVE_STATUS_t status;

    status = DAVE_Init(); // GLOBAL_CCU4_Init() is called from DAVE_Init()

    // Below can be used to start the specific kernels, by generating two instance of APP
    // GLOBAL_CCU4.SyncStartTriggerHigh((uint32_t)(GLOBAL_CCU4_0.syncstart_trigger_msk | GLOBAL_CCU4_1.syncstart_trigger_msk));
    // Below can be used to start all the kernels simultaneously
    GLOBAL_CCU4.SyncStartTriggerHigh(GLOBAL_CCU4_CUCON_Msk);

    while(1) {
    
    }

    return 1;
}
```

Definition at line 245 of file GLOBAL_CCU4.h.

__STATIC_INLINE void GLOBAL_CCU4.SyncStartTriggerLow ( uint

Start all the timers which are configured to start externally on negative edge.
Parameters:

**ccucon_msk** mask for which kernels sync start has to be applied.

Note:
This mask has been generated in the APP handle and a macro in global_ccu4_conf.h file. 1. The variable from the APP handle is useful while starting the specific kernel/s 2. GLOBAL_CCU4_CCUCON_Msk Macro from global_ccu4_conf.h file can be used to start all the selected kernels at a time.

Return values:

none

Description:
The top level APPs have to be enabled, to start the timer externally with negative trigger edge.

Example Usage:

```c
#include <DAVE.h>
int main(void) {
    DAVE_STATUS_t status;

    status = DAVE_Init(); // GLOBAL_CCU4_Init() is called from DAVE_Init()

    // Below can be used to start the specific kernels, by generating two instance of APP
    // GLOBAL_CCU4_SyncStartTriggerLow((uint32_t)(GLOBAL_CCU4_0.syncstart_trigger_msk | GLOBAL_CCU4_1.syncstart_trigger_msk));
    // Below can be used to start all the kernels simultaneously
    GLOBAL_CCU4_SyncStartTriggerLow(GLOBAL_CCU4_CC
```
UCON_Msk);

    while(1)
    {
    }

    return 1;
    }

Definition at line 284 of file GLOBAL_CCU4.h.
GLOBAL_CCU4

Usage

GLOBAL_CCU4 is a global DAVE™ APP. It is consumed by the PWM and other (top level) APPs. For information on how GLOBAL_CCU4 is being used, refer to the PWM related APPs help documentation. e.g.: PWM, PWM_CCU4
# GLOBAL_CCU4

Here are the data structures with brief descriptions:

<table>
<thead>
<tr>
<th>GLOBAL_CCU4</th>
<th></th>
</tr>
</thead>
</table>

---
## GLOBAL_CCU4

<table>
<thead>
<tr>
<th>Home</th>
<th>Data Structures</th>
<th>Data Structure Index</th>
<th>Data Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Data Fields</td>
</tr>
</tbody>
</table>

### GLOBAL_CCU4 Struct Reference

Data structures

---
Detailed Description

This saves the context of the `GLOBAL_CCU4` APP.

Definition at line 124 of file `GLOBAL_CCU4.h`.

```
#include <GLOBAL_CCU4.h>
```
### Data Fields

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Field Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>const uint32_t</td>
<td>module_frequency</td>
</tr>
<tr>
<td>const XMC_SCU_CCU_TRIGGER_t</td>
<td>syncstart_trigger_msk</td>
</tr>
<tr>
<td>XMC_CCU4_MODULE_t *const</td>
<td>module_ptr</td>
</tr>
<tr>
<td>XMC_CCU4_SLICE_MCMS_ACTION_t const</td>
<td>mcs_action</td>
</tr>
<tr>
<td>bool</td>
<td>is_initialized</td>
</tr>
</tbody>
</table>
### Field Documentation

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>GLOBAL_CCU4::is_initialized</code></td>
<td>Indicates initialized state of particular instance of the APP. Definition at line 130 of file <code>GLOBAL_CCU4.h</code>. Referenced by <code>GLOBAL_CCU4_Init()</code>.</td>
</tr>
<tr>
<td>XMC_CCU4_SLICE_MCMS_ACTION_t const</td>
<td><code>GLOBAL_CCU4::mcs_action</code></td>
<td>Shadow transfer of selected values in multi-channel mode. Definition at line 129 of file <code>GLOBAL_CCU4.h</code>. Referenced by <code>GLOBAL_CCU4_Init()</code>.</td>
</tr>
<tr>
<td>const uint32_t</td>
<td><code>GLOBAL_CCU4::module_frequency</code></td>
<td>fccu frequency. Definition at line 126 of file <code>GLOBAL_CCU4.h</code>.</td>
</tr>
<tr>
<td>XMC_CCU4_MODULE_t* const</td>
<td><code>GLOBAL_CCU4::module_ptr</code></td>
<td>Reference to module handle. Definition at line 128 of file <code>GLOBAL_CCU4.h</code>. Referenced by <code>GLOBAL_CCU4_Init()</code>.</td>
</tr>
<tr>
<td>const XMC_SCU_CCU_TRIGGER_t</td>
<td><code>GLOBAL_CCU4::syncstart_trig</code></td>
<td></td>
</tr>
</tbody>
</table>

---

**Note:** Lines 126, 127, 128, 129, 130 refer to specific lines in the `GLOBAL_CCU4.h` file.
Mask to start the timers synchronously

Definition at line 127 of file GLOBAL_CCU4.h.

The documentation for this struct was generated from the following file:

- GLOBAL_CCU4.h
<table>
<thead>
<tr>
<th>Data Structure Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLOBAL_CCU4</td>
</tr>
</tbody>
</table>

GLOBAL_CCU4
Here is a list of all documented struct and union fields with links to the struct/union documentation for each field:

- `is_initialized`: `GLOBAL_CCU4`
- `mcs_action`: `GLOBAL_CCU4`
- `module_frequency`: `GLOBAL_CCU4`
- `module_ptr`: `GLOBAL_CCU4`
- `syncstart_trigger_msk`: `GLOBAL_CCU4`
**GLOBAL_CCU4**

<table>
<thead>
<tr>
<th>Home</th>
<th>Data Structures</th>
<th>Data Structure Index</th>
<th>Data Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Variables</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- `is_initialized` : `GLOBAL_CCU4`
- `mcs_action` : `GLOBAL_CCU4`
- `module_frequency` : `GLOBAL_CCU4`
- `module_ptr` : `GLOBAL_CCU4`
- `syncstart_trigger_msk` : `GLOBAL_CCU4`
GLOBAL_CCU4

Here is a list of all documented files with brief descriptions:

- GLOBAL_CCU4.c
- GLOBAL_CCU4.h
GLOBAL_CCU4

<table>
<thead>
<tr>
<th>Home</th>
<th>File List</th>
<th>Globals</th>
<th>Functions</th>
</tr>
</thead>
</table>

GLOBAL_CCU4.c File Reference
Detailed Description

Date:
   2016-02-10

NOTE: This file is generated by DAVE. Any manual modification done to this file will be lost when the code is regenerated.

Definition in file **GLOBAL_CCU4.c**.

#include "global_ccu4.h"
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAVE_APP_VERSION_t</td>
<td>GLOBAL_CCU4_GetAppVersion</td>
<td>(void) Get <strong>GLOBAL_CCU4</strong> APP version.</td>
</tr>
<tr>
<td>GLOBAL_CCU4_STATUS_t</td>
<td>GLOBAL_CCU4_Init</td>
<td>(GLOBAL_CCU4_t *handle) Initializes a <strong>GLOBAL_CCU4</strong> with generated configuration.</td>
</tr>
</tbody>
</table>
Function Documentation

GLOBAL_CCU4_STATUS_t GLOBAL_CCU4_Init (GLOBAL_CCU4_t

Initializes a GLOBAL_CCU4 with generated configuration.

Parameters:
  handle pointer to the GLOBAL_CCU4 APP handle structure.

Returns:
  GLOBAL_CCU4_STATUS_t
  GLOBAL_CCU4_STATUS_SUCCESS : if initialization is successful
  GLOBAL_CCU4_STATUS_FAILURE : if initialization is failed

Description:
  • Enable the module.
  • Start the prescaler.

Example Usage:

```c
#include <DAVE.h>
int main(void)
{
    DAVE_STATUS_t init_status;
    init_status = DAVE_Init();       // GLOBAL_CCU4_Init
    GLOBAL_CCU4_Init(&GLOBAL_CCU4_0) will be called from DAVE_Init()
    while(1)
    {
    }
    return 1;
}
```
Definition at line 82 of file GLOBAL_CCU4.c.

References GLOBAL_CCU4_STATUS_SUCCESS, GLOBAL_CCU4::is_initialized, GLOBAL_CCU4::mcs_action, and GLOBAL_CCU4::module_ptr.

Go to the source code of this file.
# GLOBAL_CCU4

<table>
<thead>
<tr>
<th>Home</th>
<th>File List</th>
<th>Globals</th>
<th>Data Structures</th>
</tr>
</thead>
</table>

GLOBAL_CCU4.h File Reference
Detailed Description

Date:
  2016-02-10

NOTE: This file is generated by DAVE. Any manual modification done to this file will be lost when the code is regenerated.

Definition in file GLOBAL_CCU4.h.

#include <xmc_ccu4.h> #include <xmc_scu.h>
#include <DAVE_Common.h>
#include "global_ccu4_conf.h"
#include ".CLOCK_XMC4/clock_xmc4.h"
#include "global_ccu4Extern.h"
Data Structures

struct GLOBAL_CCU4
typedef struct GLOBAL_CCU4 GLOBAL_CCU4_t
## Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DAVE_APP_VERSION_t</code></td>
<td>GLOBAL_CCU4_GetAppVersion                                               Get GLOBAL_CCU4 APP version.</td>
</tr>
<tr>
<td><code>GLOBAL_CCU4_STATUS_t</code></td>
<td><strong>GLOBAL_CCU4_Init</strong> (GLOBAL_CCU4_t *handle)                                     Initializes a GLOBAL_CCU4 generated configuration.</td>
</tr>
<tr>
<td><strong>__STATIC_INLINE</strong> void</td>
<td><code>GLOBAL_CCU4_SyncStartTriggerHigh</code> (uint32_t ccucon_msk)                                            Start all the timers which are configured to start externally on positive edge.</td>
</tr>
<tr>
<td><strong>__STATIC_INLINE</strong> void</td>
<td><code>GLOBAL_CCU4_SyncStartTriggerLow</code> (uint32_t ccucon_msk)                                             Start all the timers which are configured to start externally on negative edge.</td>
</tr>
<tr>
<td><code>enum</code></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td><code>GLOBAL_CCU4_STATUS {</code></td>
</tr>
<tr>
<td></td>
<td><code>GLOBAL_CCU4_STATUS_SUCCESS 0U,</code></td>
</tr>
<tr>
<td></td>
<td><code>GLOBAL_CCU4_STATUS_FAILURE</code></td>
</tr>
<tr>
<td></td>
<td><code>}</code></td>
</tr>
<tr>
<td><code>typedef enum</code> <code>GLOBAL_CCU4_STATUS_t</code></td>
<td>Return status of the GLOBAL_CCU4 APP. More...</td>
</tr>
<tr>
<td><code>GLOBAL_CCU4_STATUS_t</code></td>
<td>Return status of the GLOBAL_CCU4 APP.</td>
</tr>
</tbody>
</table>

Go to the source code of this file.
Here is a list of all documented functions, variables, defines, enums, and typedefs with links to the documentation:

- `GLOBAL_CCU4_GetAppVersion()`: [GLOBAL_CCU4.c](GLOBAL_CCU4.c), [GLOBAL_CCU4.h](GLOBAL_CCU4.h)
- `GLOBAL_CCU4_Init()`: [GLOBAL_CCU4.h](GLOBAL_CCU4.h), [GLOBAL_CCU4.c](GLOBAL_CCU4.c)
- `GLOBAL_CCU4_STATUS`: [GLOBAL_CCU4.h](GLOBAL_CCU4.h)
- `GLOBAL_CCU4_STATUS_FAILURE`: [GLOBAL_CCU4.h](GLOBAL_CCU4.h)
- `GLOBAL_CCU4_STATUS_SUCCESS`: [GLOBAL_CCU4.h](GLOBAL_CCU4.h)
- `GLOBAL_CCU4_STATUS_t`: [GLOBAL_CCU4.h](GLOBAL_CCU4.h)
- `GLOBAL_CCU4_SyncStartTriggerHigh()`: [GLOBAL_CCU4.h](GLOBAL_CCU4.h)
- `GLOBAL_CCU4_SyncStartTriggerLow()`: [GLOBAL_CCU4.h](GLOBAL_CCU4.h)
- `GLOBAL_CCU4_t`: [GLOBAL_CCU4.h](GLOBAL_CCU4.h)
GLOBAL_CCU4

- GLOBAL_CCU4_GetAppVersion() : GLOBAL_CCU4.c, GLOBAL_CCU4.h
- GLOBAL_CCU4_Init() : GLOBAL_CCU4.h, GLOBAL_CCU4.c
- GLOBAL_CCU4_SyncStartTriggerHigh() : GLOBAL_CCU4.h
- GLOBAL_CCU4_SyncStartTriggerLow() : GLOBAL_CCU4.h
GLOBAL_CCU4

- GLOBAL_CCU4_STATUS_t : GLOBAL_CCU4.h
- GLOBAL_CCU4_t : GLOBAL_CCU4.h
GLOBAL_CCU4

- GLOBAL_CCU4_STATUS : GLOBAL_CCU4.h
GLOBAL_CCU4

- GLOBAL_CCU4_STATUS_FAILURE : GLOBAL_CCU4.h
- GLOBAL_CCU4_STATUS_SUCCESS : GLOBAL_CCU4.h
GLOBAL_CCU4

Go to the documentation of this file.

```c
#ifndef GLOBAL_CCU4_H
#define GLOBAL_CCU4_H

/*******************************************
**************************************************
**************************
* HEADER FILES
*******************************************
**************************************************
**************************
#include <xmc_ccu4.h>
#include <xmc_scu.h>
#include <DAVE_Common.h>
#include "global_ccu4_conf.h"

#if (UC_FAMILY == XMC4)
#include "../CLOCK_XMC4/clock_xmc4.h"
#endif

#ifndef CLOCK_XMC4_CCUCLK_ENABLED
#error "Error: GLOBAL_CCU4 APP missing clock settings. Please review CLOCK_XMC4 settings"
#endif

#endif

/***************************************************/
```
#if (!((XMC_LIB_MAJOR_VERSION == 2U) && (XMC_LIB_MINOR_VERSION >= 0U) && (XMC_LIB_PATCH_VERSION >= 0U)))
#error "GLOBAL_CCU4 requires XMC Peripheral Library v2.0.0 or higher"
#endif

typedef enum GLOBAL_CCU4_STATUS
{
  GLOBAL_CCU4_STATUS_SUCCESS = 0U,
  GLOBAL_CCU4_STATUS_FAILURE
} GLOBAL_CCU4_STATUS_t;

typedef struct GLOBAL_CCU4
{
  const uint32_t module_frequency;
  const XMC_SCU_CCU_TRIGGER_t syncstart_trig
}
ger_msk;
00128  XMC_CCU4_MODULE_t* const module_ptr;
00129  XMC_CCU4_SLICE_MCMS_ACTION_t const mcs_action;
00130  bool isinitialized;
00131  } GLOBAL_CCU4_t;
00132
00136  /******************************* */
00137  /
00138  API Prototypes
00139  /******************************* */
00140  /******************************* */
00141  #ifdef __cplusplus
00142  extern "C" {
00143  #endif
00144
00149  DAVE_APP_VERSION_t GLOBAL_CCU4_GetAppVersion(void);
00150
00155  GLOBAL_CCU4_STATUS_t GLOBAL_CCU4_Init(GLOBAL_CCU4_t* handle);
00156
00161  __STATIC_INLINE void GLOBAL_CCU4_SyncStartTriggerHigh(uint32_t ccucon_msk)
00162  {
00163      XMC_SCU_SetCcuTriggerHigh(ccucon_msk);
00164  }
00165
00170  __STATIC_INLINE void GLOBAL_CCU4_SyncStartTriggerLow(uint32_t ccucon_msk)
00171  {
00172      XMC_SCU_SetCcuTriggerLow(ccucon_msk);
00173  }
00174
00179  #include "global_ccu4_extern.h"
#ifdef __cplusplus
}
#endif
#endif /*CCUGLOBAL_H*/
GLOBAL_CCU4

Go to the documentation of this file.

```c
#include "global_ccu4.h"
```

---

Home | File List | Globals
---|---|---

**GLOBAL_CCU4.c**
/* Returns the version of the GLOBAL_CCU4 APP. */

DAVE_APP_VERSION_t GLOBAL_CCU4_GetAppVersion(void)
{
    DAVE_APP_VERSION_t version;

    version.major = GLOBAL_CCU4_MAJOR_VERSION;
    version.minor = GLOBAL_CCU4_MINOR_VERSION;
    version.patch = GLOBAL_CCU4_PATCH_VERSION;

    return version;
}

/* Initializes the slice with the generated configuration */
GLOBAL_CCU4_STATUS_t GLOBAL_CCU4_Init(GLOBAL_CCU4_t* handle)
{
    XMC_ASSERT("GLOBAL_CCU4_Init:NULL handler", (NULL != handle));
}
if (false == handle->is Initialized)
{
  /* Enable CCU4 module */
  XMC_CCU4_Init(handle->module_ptr, handle->mcs_action);
  /* Start the prescaler */
  XMC_CCU4_StartPrescaler(handle->module_ptr);
  /* Restricts multiple initializations */
  handle->is Initialized = true;
}
return (GLOBAL_CCU4_STATUS_SUCCESS);