STM32F3-Discovery BSP User Manual

Private Types

STM32F3-DISCOVERY ACCELEROMETER

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Private Constants

STM32F3-DISCOVERY ACCELEROMETER

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Private Macros

STM32F3-DISCOVERY ACCELEROMETER

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Exported Types

STM32F3-DISCOVERY ACCELEROMETER

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Exported Macros

STM32F3-DISCOVERY ACCELEROMETER

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Private Functions

STM32F3-DISCOVERY ACCELEROMETER

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Private Types

STM32F3-DISCOVERY GYROSCOPE

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Private Constants

STM32F3-DISCOVERY GYROSCOPE

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Private Macros

STM32F3-DISCOVERY GYROSCOPE

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Private Functions

STM32F3-DISCOVERY GYROSCOPE

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Exported Constants

STM32F3-DISCOVERY GYROSCOPE

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Exported Macros

STM32F3-DISCOVERY GYROSCOPE

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- _ -

- __STM32F3_DISCO_BSP_VERSION : **stm32f3_discovery.c**
- __STM32F3_DISCO_BSP_VERSION_MAIN : **stm32f3_discovery.c**
- __STM32F3_DISCO_BSP_VERSION_RC : **stm32f3_discovery.c**
- __STM32F3_DISCO_BSP_VERSION_SUB1 : **stm32f3_discovery.c**
- __STM32F3_DISCO_BSP_VERSION_SUB2 : **stm32f3_discovery.c**

- a -

- ACCELERO_DRDY_EXTI_IRQn : **stm32f3_discovery.h**
- ACCELERO_DRDY_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- ACCELERO_DRDY_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- ACCELERO_DRDY_GPIO_PORT : **stm32f3_discovery.h**
- ACCELERO_DRDY_PIN : **stm32f3_discovery.h**
- ACCELERO_ERROR : **stm32f3_discovery_accelerometer.h**
- ACCELERO_I2C_ADDRESS : **stm32f3_discovery.h**
- ACCELERO_INT1_EXTI_IRQn : **stm32f3_discovery.h**
- ACCELERO_INT1_PIN : **stm32f3_discovery.h**
- ACCELERO_INT2_EXTI_IRQn : **stm32f3_discovery.h**

- ACCELERO_INT2_PIN : **stm32f3_discovery.h**
- ACCELERO_INT_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- ACCELERO_INT_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- ACCELERO_INT_GPIO_PORT : **stm32f3_discovery.h**
- ACCELERO_OK : **stm32f3_discovery_accelerometer.h**
- ACCELERO_StatusTypeDef : **stm32f3_discovery_accelerometer.h**
- ACCELERO_TIMEOUT : **stm32f3_discovery_accelerometer.h**
- AccelerometerDrv : **stm32f3_discovery_accelerometer.c**

- b -

- BSP_ACCELERO_GetXYZ() : **stm32f3_discovery_accelerometer.c**
- BSP_ACCELERO_Init() : **stm32f3_discovery_accelerometer.c**
- BSP_ACCELERO_Reset() : **stm32f3_discovery_accelerometer.c**
- BSP_GetVersion() : **stm32f3_discovery.c**
- BSP_GYRO_DisableIT() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_EnableIT() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_GetXYZ() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_Init() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_ITConfig() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_ReadID() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_Reset() : **stm32f3_discovery_gyroscope.c**
- BSP_LED_Init() : **stm32f3_discovery.c**
- BSP_LED_Off() : **stm32f3_discovery.c**
- BSP_LED_On() : **stm32f3_discovery.c**
- BSP_LED_Toggle() : **stm32f3_discovery.c**
- BSP_PB_GetState() : **stm32f3_discovery.c**
- BSP_PB_Init() : **stm32f3_discovery.c**
- BUTTON_IRQn : **stm32f3_discovery.c**
- BUTTON_MODE_EXTI : **stm32f3_discovery.h**
- BUTTON_MODE_GPIO : **stm32f3_discovery.h**
- BUTTON_PIN : **stm32f3_discovery.c**
- BUTTON_PORT : **stm32f3_discovery.c**
- Button_TypeDef : **stm32f3_discovery.h**

- BUTTON_USER : **stm32f3_discovery.h**
- ButtonMode_TypeDef : **stm32f3_discovery.h**
- BUTTONn : **stm32f3_discovery.h**
- BUTTONx_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- BUTTONx_GPIO_CLK_ENABLE : **stm32f3_discovery.h**

- c -

- COMPASSACCELERO_IO_Init() : **stm32f3_discovery.c**
- COMPASSACCELERO_IO_ITConfig() : **stm32f3_discovery.c**
- COMPASSACCELERO_IO_Read() : **stm32f3_discovery.c**
- COMPASSACCELERO_IO_Write() : **stm32f3_discovery.c**

- d -

- DISCOVERY_I2Cx : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_AF : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_CLK_DISABLE : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_CLK_ENABLE : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_FORCE_RESET : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_GPIO_PORT : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_RELEASE_RESET : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_SCL_PIN : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_SDA_PIN : **stm32f3_discovery.h**
- DISCOVERY_SPIx : **stm32f3_discovery.h**
- DISCOVERY_SPIx_AF : **stm32f3_discovery.h**
- DISCOVERY_SPIx_CLK_ENABLE : **stm32f3_discovery.h**
- DISCOVERY_SPIx_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- DISCOVERY_SPIx_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- DISCOVERY_SPIx_GPIO_PORT : **stm32f3_discovery.h**
- DISCOVERY_SPIx_MISO_PIN : **stm32f3_discovery.h**
- DISCOVERY_SPIx_MOSI_PIN : **stm32f3_discovery.h**
- DISCOVERY_SPIx_SCK_PIN : **stm32f3_discovery.h**
- DUMMY_BYTE : **stm32f3_discovery.h**

- g -

- GYRO_CS_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- GYRO_CS_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- GYRO_CS_GPIO_PORT : **stm32f3_discovery.h**
- GYRO_CS_HIGH : **stm32f3_discovery.h**
- GYRO_CS_LOW : **stm32f3_discovery.h**
- GYRO_CS_PIN : **stm32f3_discovery.h**
- GYRO_ERROR : **stm32f3_discovery_gyroscope.h**
- GYRO_INT1_EXTI_IRQn : **stm32f3_discovery.h**
- GYRO_INT1_PIN : **stm32f3_discovery.h**
- GYRO_INT2_EXTI_IRQn : **stm32f3_discovery.h**
- GYRO_INT2_PIN : **stm32f3_discovery.h**
- GYRO_INT_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- GYRO_INT_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- GYRO_INT_GPIO_PORT : **stm32f3_discovery.h**
- GYRO_IO_Init() : **stm32f3_discovery.c**
- GYRO_IO_Read() : **stm32f3_discovery.c**
- GYRO_IO_Write() : **stm32f3_discovery.c**
- GYRO_OK : **stm32f3_discovery_gyroscope.h**
- GYRO_StatusTypeDef : **stm32f3_discovery_gyroscope.h**
- GYRO_TIMEOUT : **stm32f3_discovery_gyroscope.h**
- GyroscopeDrv : **stm32f3_discovery_gyroscope.c**

- i -

- I2cHandle : **stm32f3_discovery.c**
- I2Cx_Error() : **stm32f3_discovery.c**
- I2Cx_Init() : **stm32f3_discovery.c**
- I2Cx_MspInit() : **stm32f3_discovery.c**
- I2Cx_ReadData() : **stm32f3_discovery.c**
- I2Cx_TIMEOUT_MAX : **stm32f3_discovery.h**
- I2Cx_WriteData() : **stm32f3_discovery.c**
- I2cxTimeout : **stm32f3_discovery.c**

- l -

- LED10 : **stm32f3_discovery.h**
- LED10_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED10_GPIO_CLK_ENABLE : **stm32f3_discovery.h**

- LED10_GPIO_PORT : **stm32f3_discovery.h**
- LED10_PIN : **stm32f3_discovery.h**
- LED3 : **stm32f3_discovery.h**
- LED3_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED3_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED3_GPIO_PORT : **stm32f3_discovery.h**
- LED3_PIN : **stm32f3_discovery.h**
- LED4 : **stm32f3_discovery.h**
- LED4_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED4_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED4_GPIO_PORT : **stm32f3_discovery.h**
- LED4_PIN : **stm32f3_discovery.h**
- LED5 : **stm32f3_discovery.h**
- LED5_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED5_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED5_GPIO_PORT : **stm32f3_discovery.h**
- LED5_PIN : **stm32f3_discovery.h**
- LED6 : **stm32f3_discovery.h**
- LED6_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED6_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED6_GPIO_PORT : **stm32f3_discovery.h**
- LED6_PIN : **stm32f3_discovery.h**
- LED7 : **stm32f3_discovery.h**
- LED7_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED7_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED7_GPIO_PORT : **stm32f3_discovery.h**
- LED7_PIN : **stm32f3_discovery.h**
- LED8 : **stm32f3_discovery.h**
- LED8_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED8_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED8_GPIO_PORT : **stm32f3_discovery.h**
- LED8_PIN : **stm32f3_discovery.h**
- LED9 : **stm32f3_discovery.h**
- LED9_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED9_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED9_GPIO_PORT : **stm32f3_discovery.h**
- LED9_PIN : **stm32f3_discovery.h**

- LED_BLUE : **stm32f3_discovery.h**
- LED_BLUE_2 : **stm32f3_discovery.h**
- LED_GREEN : **stm32f3_discovery.h**
- LED_GREEN_2 : **stm32f3_discovery.h**
- LED_ORANGE : **stm32f3_discovery.h**
- LED_ORANGE_2 : **stm32f3_discovery.h**
- LED_PIN : **stm32f3_discovery.c**
- LED_PORT : **stm32f3_discovery.c**
- LED_RED : **stm32f3_discovery.h**
- LED_RED_2 : **stm32f3_discovery.h**
- Led_TypeDef : **stm32f3_discovery.h**
- LEDn : **stm32f3_discovery.h**
- LEDx_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LEDx_GPIO_CLK_ENABLE : **stm32f3_discovery.h**

- m -

- MULTIPLEBYTE_CMD : **stm32f3_discovery.h**

- r -

- READWRITE_CMD : **stm32f3_discovery.h**

- s -

- SpiHandle : **stm32f3_discovery.c**
- SPIx_Error() : **stm32f3_discovery.c**
- SPIx_Init() : **stm32f3_discovery.c**
- SPIx_MspInit() : **stm32f3_discovery.c**
- SPIx_TIMEOUT_MAX : **stm32f3_discovery.h**
- SPIx_WriteRead() : **stm32f3_discovery.c**
- SpixTimeout : **stm32f3_discovery.c**

- u -

- USER_BUTTON_EXTI_IRQn : **stm32f3_discovery.h**
- USER_BUTTON_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- USER_BUTTON_GPIO_CLK_ENABLE : **stm32f3_discovery.h**

- USER_BUTTON_GPIO_PORT : **stm32f3_discovery.h**
- USER_BUTTON_PIN : **stm32f3_discovery.h**

STM32F3-Discovery BSP User Manual

- b -

- BSP_ACCELERO_GetXYZ() : **stm32f3_discovery_accelerometer.c**
- BSP_ACCELERO_Init() : **stm32f3_discovery_accelerometer.c**
- BSP_ACCELERO_Reset() : **stm32f3_discovery_accelerometer.c**
- BSP_GetVersion() : **stm32f3_discovery.c**
- BSP_GYRO_DisableIT() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_EnableIT() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_GetXYZ() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_Init() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_ITConfig() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_ReadID() : **stm32f3_discovery_gyroscope.c**
- BSP_GYRO_Reset() : **stm32f3_discovery_gyroscope.c**
- BSP_LED_Init() : **stm32f3_discovery.c**
- BSP_LED_Off() : **stm32f3_discovery.c**
- BSP_LED_On() : **stm32f3_discovery.c**
- BSP_LED_Toggle() : **stm32f3_discovery.c**
- BSP_PB_GetState() : **stm32f3_discovery.c**
- BSP_PB_Init() : **stm32f3_discovery.c**

- c -

- COMPASSACCELERO_IO_Init() : **stm32f3_discovery.c**

- COMPASSACCELERO_IO_ITConfig() : **stm32f3_discovery.c**
- COMPASSACCELERO_IO_Read() : **stm32f3_discovery.c**
- COMPASSACCELERO_IO_Write() : **stm32f3_discovery.c**

- g -

- GYRO_IO_Init() : **stm32f3_discovery.c**
- GYRO_IO_Read() : **stm32f3_discovery.c**
- GYRO_IO_Write() : **stm32f3_discovery.c**

- i -

- I2Cx_Error() : **stm32f3_discovery.c**
- I2Cx_Init() : **stm32f3_discovery.c**
- I2Cx_MspInit() : **stm32f3_discovery.c**
- I2Cx_ReadData() : **stm32f3_discovery.c**
- I2Cx_WriteData() : **stm32f3_discovery.c**

- s -

- SPIx_Error() : **stm32f3_discovery.c**
- SPIx_Init() : **stm32f3_discovery.c**
- SPIx_MspInit() : **stm32f3_discovery.c**
- SPIx_WriteRead() : **stm32f3_discovery.c**

STM32F3-Discovery BSP User Manual

- AccelerometerDrv : **stm32f3_discovery_accelerometer.c**
- BUTTON_IRQn : **stm32f3_discovery.c**
- BUTTON_PIN : **stm32f3_discovery.c**
- BUTTON_PORT : **stm32f3_discovery.c**
- GyroscopeDrv : **stm32f3_discovery_gyroscope.c**
- I2cHandle : **stm32f3_discovery.c**
- I2cxTimeout : **stm32f3_discovery.c**
- LED_PIN : **stm32f3_discovery.c**
- LED_PORT : **stm32f3_discovery.c**
- SpiHandle : **stm32f3_discovery.c**
- SpixTimeout : **stm32f3_discovery.c**

STM32F3-Discovery BSP User Manual

- ACCELERO_StatusTypeDef : **stm32f3_discovery_accelerometer.h**
- Button_TypeDef : **stm32f3_discovery.h**
- ButtonMode_TypeDef : **stm32f3_discovery.h**
- GYRO_StatusTypeDef : **stm32f3_discovery_gyroscope.h**
- Led_TypeDef : **stm32f3_discovery.h**

STM32F3-Discovery BSP User Manual

- ACCELERO_ERROR : **stm32f3_discovery_accelerometer.h**
- ACCELERO_OK : **stm32f3_discovery_accelerometer.h**
- ACCELERO_TIMEOUT : **stm32f3_discovery_accelerometer.h**
- BUTTON_MODE_EXTI : **stm32f3_discovery.h**
- BUTTON_MODE_GPIO : **stm32f3_discovery.h**
- BUTTON_USER : **stm32f3_discovery.h**
- GYRO_ERROR : **stm32f3_discovery_gyroscope.h**
- GYRO_OK : **stm32f3_discovery_gyroscope.h**
- GYRO_TIMEOUT : **stm32f3_discovery_gyroscope.h**
- LED10 : **stm32f3_discovery.h**
- LED3 : **stm32f3_discovery.h**
- LED4 : **stm32f3_discovery.h**
- LED5 : **stm32f3_discovery.h**
- LED6 : **stm32f3_discovery.h**
- LED7 : **stm32f3_discovery.h**
- LED8 : **stm32f3_discovery.h**
- LED9 : **stm32f3_discovery.h**
- LED_BLUE : **stm32f3_discovery.h**
- LED_BLUE_2 : **stm32f3_discovery.h**
- LED_GREEN : **stm32f3_discovery.h**
- LED_GREEN_2 : **stm32f3_discovery.h**
- LED_ORANGE : **stm32f3_discovery.h**
- LED_ORANGE_2 : **stm32f3_discovery.h**
- LED_RED : **stm32f3_discovery.h**
- LED_RED_2 : **stm32f3_discovery.h**

Generated on Wed May 31 2017 10:47:41 for STM32F3-Discovery BSP
User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

- _ -

- __STM32F3_DISCO_BSP_VERSION : **stm32f3_discovery.c**
- __STM32F3_DISCO_BSP_VERSION_MAIN : **stm32f3_discovery.c**
- __STM32F3_DISCO_BSP_VERSION_RC : **stm32f3_discovery.c**
- __STM32F3_DISCO_BSP_VERSION_SUB1 : **stm32f3_discovery.c**
- __STM32F3_DISCO_BSP_VERSION_SUB2 : **stm32f3_discovery.c**

- a -

- ACCELERO_DRDY_EXTI_IRQn : **stm32f3_discovery.h**
- ACCELERO_DRDY_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- ACCELERO_DRDY_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- ACCELERO_DRDY_GPIO_PORT : **stm32f3_discovery.h**
- ACCELERO_DRDY_PIN : **stm32f3_discovery.h**
- ACCELERO_I2C_ADDRESS : **stm32f3_discovery.h**
- ACCELERO_INT1_EXTI_IRQn : **stm32f3_discovery.h**
- ACCELERO_INT1_PIN : **stm32f3_discovery.h**
- ACCELERO_INT2_EXTI_IRQn : **stm32f3_discovery.h**
- ACCELERO_INT2_PIN : **stm32f3_discovery.h**
- ACCELERO_INT_GPIO_CLK_DISABLE : **stm32f3_discovery.h**

- ACCELERO_INT_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- ACCELERO_INT_GPIO_PORT : **stm32f3_discovery.h**

- b -

- BUTTONn : **stm32f3_discovery.h**
- BUTTONx_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- BUTTONx_GPIO_CLK_ENABLE : **stm32f3_discovery.h**

- d -

- DISCOVERY_I2Cx : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_AF : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_CLK_DISABLE : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_CLK_ENABLE : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_FORCE_RESET : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_GPIO_PORT : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_RELEASE_RESET : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_SCL_PIN : **stm32f3_discovery.h**
- DISCOVERY_I2Cx_SDA_PIN : **stm32f3_discovery.h**
- DISCOVERY_SPIx : **stm32f3_discovery.h**
- DISCOVERY_SPIx_AF : **stm32f3_discovery.h**
- DISCOVERY_SPIx_CLK_ENABLE : **stm32f3_discovery.h**
- DISCOVERY_SPIx_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- DISCOVERY_SPIx_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- DISCOVERY_SPIx_GPIO_PORT : **stm32f3_discovery.h**
- DISCOVERY_SPIx_MISO_PIN : **stm32f3_discovery.h**
- DISCOVERY_SPIx_MOSI_PIN : **stm32f3_discovery.h**
- DISCOVERY_SPIx_SCK_PIN : **stm32f3_discovery.h**
- DUMMY_BYTE : **stm32f3_discovery.h**

- g -

- GYRO_CS_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- GYRO_CS_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- GYRO_CS_GPIO_PORT : **stm32f3_discovery.h**

- GYRO_CS_HIGH : **stm32f3_discovery.h**
- GYRO_CS_LOW : **stm32f3_discovery.h**
- GYRO_CS_PIN : **stm32f3_discovery.h**
- GYRO_INT1_EXTI_IRQn : **stm32f3_discovery.h**
- GYRO_INT1_PIN : **stm32f3_discovery.h**
- GYRO_INT2_EXTI_IRQn : **stm32f3_discovery.h**
- GYRO_INT2_PIN : **stm32f3_discovery.h**
- GYRO_INT_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- GYRO_INT_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- GYRO_INT_GPIO_PORT : **stm32f3_discovery.h**

- i -

- I2Cx_TIMEOUT_MAX : **stm32f3_discovery.h**

- l -

- LED10_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED10_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED10_GPIO_PORT : **stm32f3_discovery.h**
- LED10_PIN : **stm32f3_discovery.h**
- LED3_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED3_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED3_GPIO_PORT : **stm32f3_discovery.h**
- LED3_PIN : **stm32f3_discovery.h**
- LED4_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED4_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED4_GPIO_PORT : **stm32f3_discovery.h**
- LED4_PIN : **stm32f3_discovery.h**
- LED5_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED5_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED5_GPIO_PORT : **stm32f3_discovery.h**
- LED5_PIN : **stm32f3_discovery.h**
- LED6_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED6_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED6_GPIO_PORT : **stm32f3_discovery.h**
- LED6_PIN : **stm32f3_discovery.h**
- LED7_GPIO_CLK_DISABLE : **stm32f3_discovery.h**

- LED7_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED7_GPIO_PORT : **stm32f3_discovery.h**
- LED7_PIN : **stm32f3_discovery.h**
- LED8_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED8_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED8_GPIO_PORT : **stm32f3_discovery.h**
- LED8_PIN : **stm32f3_discovery.h**
- LED9_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LED9_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- LED9_GPIO_PORT : **stm32f3_discovery.h**
- LED9_PIN : **stm32f3_discovery.h**
- LEDn : **stm32f3_discovery.h**
- LEDx_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- LEDx_GPIO_CLK_ENABLE : **stm32f3_discovery.h**

- m -

- MULTIPLEBYTE_CMD : **stm32f3_discovery.h**

- r -

- READWRITE_CMD : **stm32f3_discovery.h**

- s -

- SPIx_TIMEOUT_MAX : **stm32f3_discovery.h**

- u -

- USER_BUTTON_EXTI_IRQn : **stm32f3_discovery.h**
- USER_BUTTON_GPIO_CLK_DISABLE : **stm32f3_discovery.h**
- USER_BUTTON_GPIO_CLK_ENABLE : **stm32f3_discovery.h**
- USER_BUTTON_GPIO_PORT : **stm32f3_discovery.h**
- USER_BUTTON_PIN : **stm32f3_discovery.h**

STM32F3-Discovery BSP User Manual

Defines | Functions | Variables

stm32f3_discovery.c File Reference

This file provides set of firmware functions to manage Leds and push-button available on STM32F3-DISCOVERY Kit from STMicroelectronics. More...

```
#include "stm32f3_discovery.h"
```

Go to the source code of this file.

Defines

#define	**__STM32F3_DISCO_BSP_VERSION_MAIN** (0x02) STM32F3 DISCOVERY BSP Driver version number V2.1.5.
#define	**__STM32F3_DISCO_BSP_VERSION_SUB1** (0x01)
#define	**__STM32F3_DISCO_BSP_VERSION_SUB2** (0x05)
#define	**__STM32F3_DISCO_BSP_VERSION_RC** (0x00)
#define	**__STM32F3_DISCO_BSP_VERSION**

Functions

static void	**I2Cx_Init** (void) Discovery I2Cx Bus initialization.
static void	**I2Cx_WriteData** (uint16_t Addr, uint8_t Reg, uint8_t Value) Write a value in a register of the device through BUS.
static uint8_t	**I2Cx_ReadData** (uint16_t Addr, uint8_t Reg) Read a value in a register of the device through BUS.
static void	**I2Cx_Error** (void) I2C3 error treatment function.
static void	**I2Cx_MspInit** (I2C_HandleTypeDef *hi2c) Discovery I2Cx MSP Initialization.
static void	**SPIx_Init** (void) SPIx Bus initialization.
static uint8_t	**SPIx_WriteRead** (uint8_t Byte) Sends a Byte through the SPI interface and return the Byte received from the SPI bus.
static void	**SPIx_Error** (void) SPIx error treatment function.
static void	**SPIx_MspInit** (SPI_HandleTypeDef *hspi) SPI MSP Init.
void	**GYRO_IO_Init** (void) Configures the GYROSCOPE SPI interface.
void	**GYRO_IO_Write** (uint8_t *pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite) Writes one byte to the GYROSCOPE.
void	**GYRO_IO_Read** (uint8_t *pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead) Reads a block of data from the GYROSCOPE.
void	**COMPASSACCELERO_IO_Init** (void) Configures COMPASS / ACCELEROMETER I2C interface.

void	**COMPASSACCELERO_IO_ITConfig** (void) Configures COMPASS / ACCELERO click IT.
void	**COMPASSACCELERO_IO_Write** (uint16_t DeviceAddr, uint8_t RegisterAddr, uint8_t Value) Writes one byte to the COMPASS / ACCELEROMETER.
uint8_t	**COMPASSACCELERO_IO_Read** (uint16_t DeviceAddr, uint8_t RegisterAddr) Reads a block of data from the COMPASS / ACCELEROMETER.
uint32_t	**BSP_GetVersion** (void) This method returns the STM32F3-DISCOVERY BSP Driver revision.
void	**BSP_LED_Init** (**Led_TypeDef** Led) Configures LED GPIO.
void	**BSP_LED_On** (**Led_TypeDef** Led) Turns selected LED On.
void	**BSP_LED_Off** (**Led_TypeDef** Led) Turns selected LED Off.
void	**BSP_LED_Toggle** (**Led_TypeDef** Led) Toggles the selected LED.
void	**BSP_PB_Init** (**Button_TypeDef** Button, **ButtonMode_TypeDef** ButtonMode) Configures Push Button GPIO and EXTI Line.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Returns the selected Push Button state.

Variables

GPIO_TypeDef *	**LED_PORT** [**LEDn**] LED variables.
const uint16_t	**LED_PIN** [**LEDn**]
GPIO_TypeDef *	**BUTTON_PORT** [**BUTTONn**] = {**USER_BUTTON_GPIO_PORT**} BUTTON variables.
const uint16_t	**BUTTON_PIN** [**BUTTONn**] = {**USER_BUTTON_PIN**}
const uint8_t	**BUTTON_IRQn** [**BUTTONn**] = {**USER_BUTTON_EXTI_IRQn**}
uint32_t	**SpixTimeout** = **SPIx_TIMEOUT_MAX** BUS variables.
static SPI_HandleTypeDef	**SpiHandle**
static I2C_HandleTypeDef	**I2cHandle**
uint32_t	**I2cxTimeout** = **I2Cx_TIMEOUT_MAX**

Detailed Description

This file provides set of firmware functions to manage Leds and push-button available on STM32F3-DISCOVERY Kit from STMicroelectronics.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2016 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. 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. 3. Neither the name of STMicroelectronics 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.

Definition in file **stm32f3_discovery.c**.

STM32F3-Discovery BSP User Manual

Defines | Enumerations | Functions

stm32f3_discovery.h File Reference

This file contains definitions for STM32F3-Discovery's Leds, push-buttons hardware resources. More...

```
#include "stm32f3xx_hal.h"
```

Go to the source code of this file.

Defines

```
#define LEDn 8
#define LED6_PIN GPIO_PIN_15
#define LED6_GPIO_PORT GPIOE
#define LED6_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED6_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED8_PIN GPIO_PIN_14
#define LED8_GPIO_PORT GPIOE
#define LED8_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED8_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED10_PIN GPIO_PIN_13
#define LED10_GPIO_PORT GPIOE
#define LED10_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED10_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_
#define LED9_PIN GPIO_PIN_12
#define LED9_GPIO_PORT GPIOE
#define LED9_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED9_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED7_PIN GPIO_PIN_11
#define LED7_GPIO_PORT GPIOE
#define LED7_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED7_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED5_PIN GPIO_PIN_10
#define LED5_GPIO_PORT GPIOE
#define LED5_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED5_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED3_PIN GPIO_PIN_9
#define LED3_GPIO_PORT GPIOE
#define LED3_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED3_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED4_PIN GPIO_PIN_8
#define LED4_GPIO_PORT GPIOE
```

#define	**LED4_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOE_CLK_E
#define	**LED4_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOE_CLK_D
#define	**LEDx_GPIO_CLK_ENABLE**(__LED__)
#define	**LEDx_GPIO_CLK_DISABLE**(__LED__)
#define	**BUTTONn** 1
#define	**USER_BUTTON_PIN** GPIO_PIN_0 push-button
#define	**USER_BUTTON_GPIO_PORT** GPIOA
#define	**USER_BUTTON_GPIO_CLK_ENABLE**() __HAL_RCC_GPIO
#define	**USER_BUTTON_GPIO_CLK_DISABLE**() __HAL_RCC_GPI
#define	**USER_BUTTON_EXTI_IRQn** EXTI0_IRQn
#define	**BUTTONx_GPIO_CLK_ENABLE**(__BUTTON__) do { if ((__B **BUTTON_USER**) **USER_BUTTON_GPIO_CLK_ENABLE**();} w
#define	**BUTTONx_GPIO_CLK_DISABLE**(__BUTTON__) (((__BUTT **BUTTON_USER**) ? **USER_BUTTON_GPIO_CLK_DISABLE**()
#define	**DISCOVERY_SPIx** SPI1 Definition for SPI Interface pins (SPI1 used)
#define	**DISCOVERY_SPIx_CLK_ENABLE**() __HAL_RCC_SPI1_CL
#define	**DISCOVERY_SPIx_GPIO_PORT** GPIOA /* GPIOA */
#define	**DISCOVERY_SPIx_AF** GPIO_AF5_SPI1
#define	**DISCOVERY_SPIx_GPIO_CLK_ENABLE**() __HAL_RCC_GI
#define	**DISCOVERY_SPIx_GPIO_CLK_DISABLE**() __HAL_RCC_G
#define	**DISCOVERY_SPIx_SCK_PIN** GPIO_PIN_5 /* PA.05 */
#define	**DISCOVERY_SPIx_MISO_PIN** GPIO_PIN_6 /* PA.06 */
#define	**DISCOVERY_SPIx_MOSI_PIN** GPIO_PIN_7 /* PA.07 */
#define	**SPIx_TIMEOUT_MAX** ((uint32_t)0x1000)
#define	**DISCOVERY_I2Cx** I2C1 Definition for I2C Interface pins (I2C1 used)
#define	**DISCOVERY_I2Cx_CLK_ENABLE**() __HAL_RCC_I2C1_CLI
#define	**DISCOVERY_I2Cx_CLK_DISABLE**() __HAL_RCC_I2C1_CL
#define	**DISCOVERY_I2Cx_FORCE_RESET**() __HAL_RCC_I2C1_F
#define	**DISCOVERY_I2Cx_RELEASE_RESET**() __HAL_RCC_I2C1_
#define	**DISCOVERY_I2Cx_SCL_PIN** GPIO_PIN_6 /* PB.06 */
#define	**DISCOVERY_I2Cx_SDA_PIN** GPIO_PIN_7 /* PB.07 */

#define	**DISCOVERY_I2Cx_GPIO_PORT** GPIOB /* GPIOB */
#define	**DISCOVERY_I2Cx_GPIO_CLK_ENABLE**() __HAL_RCC_GP
#define	**DISCOVERY_I2Cx_GPIO_CLK_DISABLE**() __HAL_RCC_G
#define	**DISCOVERY_I2Cx_AF** GPIO_AF4_I2C1
#define	**I2Cx_TIMEOUT_MAX** 0x10000
#define	**READWRITE_CMD** ((uint8_t)0x80)
#define	**MULTIPLEBYTE_CMD** ((uint8_t)0x40)
#define	**DUMMY_BYTE** ((uint8_t)0x00)
#define	**GYRO_CS_LOW**() HAL_GPIO_WritePin(**GYRO_CS_GPIO_P** GPIO_PIN_RESET)
#define	**GYRO_CS_HIGH**() HAL_GPIO_WritePin(**GYRO_CS_GPIO_** GPIO_PIN_SET)
#define	**GYRO_CS_GPIO_PORT** GPIOE /* GPIOE */ GYRO SPI Interface pins.
#define	**GYRO_CS_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOE_CL
#define	**GYRO_CS_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOE_C
#define	**GYRO_CS_PIN** GPIO_PIN_3 /* PE.03 */
#define	**GYRO_INT_GPIO_PORT** GPIOE /* GPIOE */
#define	**GYRO_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOE_C
#define	**GYRO_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOE_C
#define	**GYRO_INT1_PIN** GPIO_PIN_0 /* PE.00 */
#define	**GYRO_INT1_EXTI_IRQn** EXTI0_IRQn
#define	**GYRO_INT2_PIN** GPIO_PIN_1 /* PE.01 */
#define	**GYRO_INT2_EXTI_IRQn** EXTI1_IRQn
#define	**ACCELERO_I2C_ADDRESS** 0x32 ACCELEROMETER I2C1 Interface pins.
#define	**ACCELERO_DRDY_PIN** GPIO_PIN_2 /* PE.02 */
#define	**ACCELERO_DRDY_GPIO_PORT** GPIOE /* GPIOE */
#define	**ACCELERO_DRDY_GPIO_CLK_ENABLE**() __HAL_RCC_G
#define	**ACCELERO_DRDY_GPIO_CLK_DISABLE**() __HAL_RCC_C
#define	**ACCELERO_DRDY_EXTI_IRQn** EXTI2_TSC_IRQn /*TAMP_
#define	**ACCELERO_INT_GPIO_PORT** GPIOE /* GPIOE */
#define	**ACCELERO_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPI
#define	**ACCELERO_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GP

```
#define ACCELERO_INT1_PIN GPIO_PIN_4 /* PE.04 */
#define ACCELERO_INT1_EXTI_IRQn EXTI4_IRQn
#define ACCELERO_INT2_PIN GPIO_PIN_5 /* PE.05 */
#define ACCELERO_INT2_EXTI_IRQn EXTI9_5_IRQn
```

Enumerations

enum	**Led_TypeDef** {
	LED3 = 0, **LED4** = 1, **LED5** = 2, **LED6** = 3,
	LED7 = 4, **LED8** = 5, **LED9** = 6, **LED10** = 7,
	LED_GREEN = LED6, **LED_ORANGE** = LED5, **LED_RED** = LED3, **LED_BLUE** = LED4,
	LED_GREEN_2 = LED7, **LED_ORANGE_2** = LED8, **LED_RED_2** = LED10, **LED_BLUE_2** = LED9
	}
	LED Types Definition. More...

enum	**Button_TypeDef** { **BUTTON_USER** = 0 } BUTTON Types Definition. More...
enum	**ButtonMode_TypeDef** { **BUTTON_MODE_GPIO** = 0, **BUTTON_MODE_EXTI** = 1 }

Functions

uint32_t	**BSP_GetVersion** (void) This method returns the STM32F3-DISCOVERY BSP Driver revision.
void	**BSP_LED_Init** (**Led_TypeDef** Led) Configures LED GPIO.
void	**BSP_LED_On** (**Led_TypeDef** Led) Turns selected LED On.
void	**BSP_LED_Off** (**Led_TypeDef** Led) Turns selected LED Off.
void	**BSP_LED_Toggle** (**Led_TypeDef** Led) Toggles the selected LED.
void	**BSP_PB_Init** (**Button_TypeDef** Button, **ButtonMode_TypeDef** ButtonMode) Configures Push Button GPIO and EXTI Line.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Returns the selected Push Button state.

Detailed Description

This file contains definitions for STM32F3-Discovery's Leds, push-buttons hardware resources.

Author:
MCD Application Team

Attention:

© COPYRIGHT(c) 2016 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. 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. 3. Neither the name of STMicroelectronics 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.

Definition in file **stm32f3_discovery.h**.

STM32F3-Discovery BSP User Manual

Functions | Variables

stm32f3_discovery_accelerometer.c File Reference

This file provides a set of functions needed to manage the ACCELEROMETER MEMS available on STM32F3-Discovery Kit. More...

```
#include "stm32f3_discovery_accelerometer.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_ACCELERO_Init** (void)
void	**BSP_ACCELERO_Reset** (void) Reboot memory content of ACCELEROMETER.
void	**BSP_ACCELERO_GetXYZ** (int16_t *pDataXYZ) Get XYZ acceleration.

Variables

static ACCELERO_DrvTypeDef *	**AccelerometerDrv**

Detailed Description

This file provides a set of functions needed to manage the ACCELEROMETER MEMS available on STM32F3-Discovery Kit.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2016 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. 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. 3. Neither the name of STMicroelectronics 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.

Definition in file **stm32f3_discovery_accelerometer.c**.

STM32F3-Discovery BSP User Manual

Enumerations | Functions

stm32f3_discovery_accelerometer.h File Reference

This file contains definitions for **stm32f3_discovery_accelerometer.c** firmware driver. More...

```
#include "stm32f3_discovery.h" #include
"../Components/lsm303dlhc/lsm303dlhc.h"
```

Go to the source code of this file.

Enumerations

enum	**ACCELERO_StatusTypeDef** { **ACCELERO_OK** = 0, **ACCELERO_ERROR** = 1, **ACCELERO_TIMEOUT** = 2 }

Functions

uint8_t	**BSP_ACCELERO_Init** (void)
void	**BSP_ACCELERO_Reset** (void) Reboot memory content of ACCELEROMETER.
void	**BSP_ACCELERO_GetXYZ** (int16_t *pDataXYZ) Get XYZ acceleration.

Detailed Description

This file contains definitions for **stm32f3_discovery_accelerometer.c** firmware driver.

Author:
MCD Application Team

Attention:

© COPYRIGHT(c) 2016 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. 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. 3. Neither the name of STMicroelectronics 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.

Definition in file **stm32f3_discovery_accelerometer.h**.

stm32f3_discovery_gyroscope.c File Reference

This file provides a set of functions needed to manage the l3gd20 MEMS accelerometer available on STM32F3-Discovery Kit. More...

```
#include "stm32f3_discovery_gyroscope.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_GYRO_Init** (void) Set GYROSCOPE Initialization.
uint8_t	**BSP_GYRO_ReadID** (void) Read ID of Gyroscope component.
void	**BSP_GYRO_Reset** (void) Reboot memory content of GYROSCOPE.
void	**BSP_GYRO_ITConfig** (GYRO_InterruptConfigTypeDef *pIntConfig) Configure INT1 interrupt.
void	**BSP_GYRO_EnableIT** (uint8_t IntPin) Enable INT1 or INT2 interrupt.
void	**BSP_GYRO_DisableIT** (uint8_t IntPin) Disable INT1 or INT2 interrupt.
void	**BSP_GYRO_GetXYZ** (float *pfData) Get XYZ angular acceleration.

Variables

static GYRO_DrvTypeDef *	**GyroscopeDrv**

Detailed Description

This file provides a set of functions needed to manage the l3gd20 MEMS accelerometer available on STM32F3-Discovery Kit.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2016 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. 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. 3. Neither the name of STMicroelectronics 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.

Definition in file **stm32f3_discovery_gyroscope.c**.

STM32F3-Discovery BSP User Manual

Enumerations | Functions

stm32f3_discovery_gyroscope.h File Reference

This file contains definitions for **stm32f3_discovery_gyroscope.c** firmware driver. More...

```
#include "stm32f3_discovery.h" #include
"../Components/l3gd20/l3gd20.h"
```

Go to the source code of this file.

Enumerations

enum	**GYRO_StatusTypeDef** { **GYRO_OK** = 0, **GYRO_ERROR** = 1, **GYRO_TIMEOUT** = 2 }

Functions

uint8_t	**BSP_GYRO_Init** (void) Set GYROSCOPE Initialization.
void	**BSP_GYRO_Reset** (void) Reboot memory content of GYROSCOPE.
uint8_t	**BSP_GYRO_ReadID** (void) Read ID of Gyroscope component.
void	**BSP_GYRO_ITConfig** (GYRO_InterruptConfigTypeDef *pIntConfig) Configure INT1 interrupt.
void	**BSP_GYRO_EnableIT** (uint8_t IntPin) Enable INT1 or INT2 interrupt.
void	**BSP_GYRO_DisableIT** (uint8_t IntPin) Disable INT1 or INT2 interrupt.
void	**BSP_GYRO_GetXYZ** (float *pfData) Get XYZ angular acceleration.

Detailed Description

This file contains definitions for **stm32f3_discovery_gyroscope.c** firmware driver.

Author:
MCD Application Team

Attention:

© COPYRIGHT(c) 2016 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. 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. 3. Neither the name of STMicroelectronics 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.

Definition in file **stm32f3_discovery_gyroscope.h**.

STM32F3-Discovery BSP User Manual

Modules

Here is a list of all modules:

- BSP
  - STM32F3-DISCOVERY
    - STM32F3-DISCOVERY Common
      - Bus Operation functions
      - Link Operation functions
      - Private Constants
      - Private Variables
      - Exported Types
      - Exported Constants
        - STM32F3-DISCOVERY LED
        - STM32F3-DISCOVERY BUTTON
        - STM32F3-DISCOVERY COM
        - STM32F3-DISCOVERY COMPONENT
      - Exported Functions
    - STM32F3-DISCOVERY ACCELEROMETER
      - Private Types
      - Private Constants
      - Private Macros
      - Private Variables
      - Exported Types
      - Exported Constants
      - Exported Macros
      - Exported Functions
      - Private Functions
    - STM32F3-DISCOVERY GYROSCOPE
      - Private Types

- **Private Constants**
- **Private Macros**
- **Private Variables**
- **Private Functions**
- **Exported Types**
- **Exported Constants**
- **Exported Macros**
- **Exported Functions**

STM32F3-Discovery BSP User Manual

File List

Here is a list of all files with brief descriptions:

stm32f3_discovery.c [code]	This file provides set of functions to manage Le button available on STI DISCOVERY Kit from STMicroelectronics
stm32f3_discovery.h [code]	This file contains defini STM32F3-Discovery's buttons hardware reso
stm32f3_discovery_accelerometer.c [code]	This file provides a set needed to manage the ACCELEROMETER M on STM32F3-Discover
stm32f3_discovery_accelerometer.h [code]	This file contains defini **stm32f3_discovery_a** firmware driver
stm32f3_discovery_gyroscope.c [code]	This file provides a set needed to manage the accelerometer availabl Discovery Kit
stm32f3_discovery_gyroscope.h [code]	This file contains defini **stm32f3_discovery_g** firmware driver

User Manual by doxygen 1.7.6.1

STM32F3-Discovery BSP User Manual

Directories

This directory hierarchy is sorted roughly, but not completely, alphabetically:

- Firmware
  - Drivers
    - BSP
      - STM32F3-Discovery

STM32F3-Discovery BSP User Manual

STM32F3-DISCOVERY ACCELEROMETER

STM32F3-DISCOVERY

Modules

Private Types
Private Constants
Private Macros
Private Variables
Exported Types
Exported Constants
Exported Macros
Exported Functions
Private Functions

STM32F3-Discovery BSP User Manual

STM32F3-DISCOVERY GYROSCOPE

STM32F3-DISCOVERY

Modules

Private Types
Private Constants
Private Macros
Private Variables
Private Functions
Exported Types
Exported Constants
Exported Macros
Exported Functions

Defines

Private Constants

STM32F3-DISCOVERY Common

Defines

#define	**__STM32F3_DISCO_BSP_VERSION_MAIN** (0x02) STM32F3 DISCOVERY BSP Driver version number V2.1.5.
#define	**__STM32F3_DISCO_BSP_VERSION_SUB1** (0x01)
#define	**__STM32F3_DISCO_BSP_VERSION_SUB2** (0x05)
#define	**__STM32F3_DISCO_BSP_VERSION_RC** (0x00)
#define	**__STM32F3_DISCO_BSP_VERSION**

Define Documentation

#define __STM32F3_DISCO_BSP_VERSION

Value:

```
((__STM32F3_DISCO_BSP_VERSION_MAIN << 24)\
                                          |(__S
TM32F3_DISCO_BSP_VERSION_SUB1 << 16)\
                                          |(__S
TM32F3_DISCO_BSP_VERSION_SUB2 << 8 )\
                                          |(__S
TM32F3_DISCO_BSP_VERSION_RC))
```

Definition at line **64** of file **stm32f3_discovery.c**.

Referenced by **BSP_GetVersion()**.

#define __STM32F3_DISCO_BSP_VERSION_MAIN (0x02)

STM32F3 DISCOVERY BSP Driver version number V2.1.5.

[31:24] main version

Definition at line **60** of file **stm32f3_discovery.c**.

#define __STM32F3_DISCO_BSP_VERSION_RC (0x00)

[7:0] release candidate

Definition at line **63** of file **stm32f3_discovery.c**.

#define __STM32F3_DISCO_BSP_VERSION_SUB1 (0x01)

[23:16] sub1 version

Definition at line **61** of file **stm32f3_discovery.c**.

#define __STM32F3_DISCO_BSP_VERSION_SUB2 (0x05)

[15:8] sub2 version

Definition at line **62** of file **stm32f3_discovery.c**.

STM32F3-DISCOVERY COMPONENT

Exported Constants

Defines

#define	**READWRITE_CMD** ((uint8_t)0x80)
#define	**MULTIPLEBYTE_CMD** ((uint8_t)0x40)
#define	**DUMMY_BYTE** ((uint8_t)0x00)
#define	**GYRO_CS_LOW**() HAL_GPIO_WritePin(**GYRO_CS_GPIO_P** GPIO_PIN_RESET)
#define	**GYRO_CS_HIGH**() HAL_GPIO_WritePin(**GYRO_CS_GPIO_** GPIO_PIN_SET)
#define	**GYRO_CS_GPIO_PORT** GPIOE /* GPIOE */ GYRO SPI Interface pins.
#define	**GYRO_CS_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOE_CL
#define	**GYRO_CS_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOE_C
#define	**GYRO_CS_PIN** GPIO_PIN_3 /* PE.03 */
#define	**GYRO_INT_GPIO_PORT** GPIOE /* GPIOE */
#define	**GYRO_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOE_C
#define	**GYRO_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOE_C
#define	**GYRO_INT1_PIN** GPIO_PIN_0 /* PE.00 */
#define	**GYRO_INT1_EXTI_IRQn** EXTI0_IRQn
#define	**GYRO_INT2_PIN** GPIO_PIN_1 /* PE.01 */
#define	**GYRO_INT2_EXTI_IRQn** EXTI1_IRQn
#define	**ACCELERO_I2C_ADDRESS** 0x32 ACCELEROMETER I2C1 Interface pins.
#define	**ACCELERO_DRDY_PIN** GPIO_PIN_2 /* PE.02 */
#define	**ACCELERO_DRDY_GPIO_PORT** GPIOE /* GPIOE */
#define	**ACCELERO_DRDY_GPIO_CLK_ENABLE**() __HAL_RCC_G
#define	**ACCELERO_DRDY_GPIO_CLK_DISABLE**() __HAL_RCC_C
#define	**ACCELERO_DRDY_EXTI_IRQn** EXTI2_TSC_IRQn /*TAMP_
#define	**ACCELERO_INT_GPIO_PORT** GPIOE /* GPIOE */
#define	**ACCELERO_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPI
#define	**ACCELERO_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GP
#define	**ACCELERO_INT1_PIN** GPIO_PIN_4 /* PE.04 */
#define	**ACCELERO_INT1_EXTI_IRQn** EXTI4_IRQn

```
#define ACCELERO_INT2_PIN   GPIO_PIN_5 /* PE.05 */
#define ACCELERO_INT2_EXTI_IRQn   EXTI9_5_IRQn
```

Define Documentation

#define ACCELERO_DRDY_EXTI_IRQn EXTI2_TSC_IRQn /*TAMP_

Definition at line 320 of file stm32f3_discovery.h.

Referenced by COMPASSACCELERO_IO_Init().

#define ACCELERO_DRDY_GPIO_CLK_DISABLE ( ) __HAL_RCC

Definition at line 319 of file stm32f3_discovery.h.

#define ACCELERO_DRDY_GPIO_CLK_ENABLE ( ) __HAL_RCC_

Definition at line 318 of file stm32f3_discovery.h.

Referenced by COMPASSACCELERO_IO_Init().

#define ACCELERO_DRDY_GPIO_PORT GPIOE /* GPIOE */

Definition at line 317 of file stm32f3_discovery.h.

Referenced by COMPASSACCELERO_IO_Init().

#define ACCELERO_DRDY_PIN GPIO_PIN_2 /* PE.02 */

Definition at line 316 of file stm32f3_discovery.h.

Referenced by COMPASSACCELERO_IO_Init().

#define ACCELERO_I2C_ADDRESS 0x32

ACCELEROMETER I2C1 Interface pins.

Definition at line **314** of file **stm32f3_discovery.h**.

Referenced by **I2Cx_Init()**.

#define ACCELERO_INT1_EXTI_IRQn EXTI4_IRQn

Definition at line **326** of file **stm32f3_discovery.h**.

Referenced by **COMPASSACCELERO_IO_ITConfig()**.

#define ACCELERO_INT1_PIN GPIO_PIN_4 /* PE.04 */

Definition at line **325** of file **stm32f3_discovery.h**.

Referenced by **COMPASSACCELERO_IO_Init()**, and **COMPASSACCELERO_IO_ITConfig()**.

#define ACCELERO_INT2_EXTI_IRQn EXTI9_5_IRQn

Definition at line **328** of file **stm32f3_discovery.h**.

#define ACCELERO_INT2_PIN GPIO_PIN_5 /* PE.05 */

Definition at line **327** of file **stm32f3_discovery.h**.

Referenced by **COMPASSACCELERO_IO_Init()**, and **COMPASSACCELERO_IO_ITConfig()**.

#define ACCELERO_INT_GPIO_CLK_DISABLE ( ) __HAL_RCC_G

Definition at line 324 of file stm32f3_discovery.h.

#define ACCELERO_INT_GPIO_CLK_ENABLE ( ) __HAL_RCC_GI

Definition at line 323 of file stm32f3_discovery.h.

Referenced by COMPASSACCELERO_IO_Init(), and COMPASSACCELERO_IO_ITConfig().

#define ACCELERO_INT_GPIO_PORT GPIOE /* GPIOE */

Definition at line 322 of file stm32f3_discovery.h.

Referenced by COMPASSACCELERO_IO_Init(), and COMPASSACCELERO_IO_ITConfig().

#define DUMMY_BYTE ((uint8_t)0x00)

Definition at line 288 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Read().

#define GYRO_CS_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_

Definition at line 299 of file stm32f3_discovery.h.

#define GYRO_CS_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_

Definition at line 298 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Init().

#define GYRO_CS_GPIO_PORT GPIOE /* GPIOE */

GYRO SPI Interface pins.

Definition at line 297 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Init().

#define GYRO_CS_HIGH ( ) HAL_GPIO_WritePin(GYRO_CS_GPIC

Definition at line 292 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Init(), GYRO_IO_Read(), and GYRO_IO_Write().

#define GYRO_CS_LOW ( ) HAL_GPIO_WritePin(GYRO_CS_GPIC

Definition at line 291 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Read(), and GYRO_IO_Write().

#define GYRO_CS_PIN GPIO_PIN_3 /* PE.03 */

Definition at line 300 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Init().

#define GYRO_INT1_EXTI_IRQn EXTI0_IRQn

Definition at line 306 of file stm32f3_discovery.h.

#define GYRO_INT1_PIN GPIO_PIN_0 /* PE.00 */

Definition at line 305 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Init().

#define GYRO_INT2_EXTI_IRQn EXTI1_IRQn

Definition at line 308 of file stm32f3_discovery.h.

#define GYRO_INT2_PIN GPIO_PIN_1 /* PE.01 */

Definition at line 307 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Init().

#define GYRO_INT_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_

Definition at line 304 of file stm32f3_discovery.h.

#define GYRO_INT_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_

Definition at line 303 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Init().

#define GYRO_INT_GPIO_PORT GPIOE /* GPIOE */

Definition at line 302 of file stm32f3_discovery.h.

Referenced by GYRO_IO_Init().

#define MULTIPLEBYTE_CMD ((uint8_t)0x40)

Definition at line **286** of file **stm32f3_discovery.h**.

Referenced by **GYRO_IO_Read()**, and **GYRO_IO_Write()**.

#define READWRITE_CMD ((uint8_t)0x80)

Definition at line **284** of file **stm32f3_discovery.h**.

Referenced by **GYRO_IO_Read()**.

Exported Constants

STM32F3-DISCOVERY ACCELEROMETER

Enumerations

enum	**ACCELERO_StatusTypeDef** { **ACCELERO_OK** = 0, **ACCELERO_ERROR** = 1, **ACCELERO_TIMEOUT** = 2 }

Enumeration Type Documentation

enum ACCELERO_StatusTypeDef

Enumerator:

ACCELERO_OK

ACCELERO_ERROR

ACCELERO_TIMEOUT

Definition at line **74** of file **stm32f3_discovery_accelerometer.h**.

Private Variables

STM32F3-DISCOVERY ACCELEROMETER

Variables

static ACCELERO_DrvTypeDef *	**AccelerometerDrv**

Variable Documentation

ACCELERO_DrvTypeDef* AccelerometerDrv [static]

Definition at line **80** of file **stm32f3_discovery_accelerometer.c**.

Referenced by **BSP_ACCELERO_GetXYZ()**, **BSP_ACCELERO_Init()**, and **BSP_ACCELERO_Reset()**.

Exported Functions

STM32F3-DISCOVERY ACCELEROMETER

Functions

uint8_t	**BSP_ACCELERO_Init** (void)
void	**BSP_ACCELERO_Reset** (void) Reboot memory content of ACCELEROMETER.
void	**BSP_ACCELERO_GetXYZ** (int16_t *pDataXYZ) Get XYZ acceleration.

Function Documentation

void BSP_ACCELERO_GetXYZ (int16_t * pDataXYZ)

Get XYZ acceleration.

Parameters:

pDataXYZ	Pointeur on 3 angular accelerations pDataXYZ[0] = X axis, pDataXYZ[1] = Y axis, pDataXYZ[2] = Z axis

Return values:

None

Definition at line **174** of file **stm32f3_discovery_accelerometer.c**.

References **AccelerometerDrv**.

uint8_t BSP_ACCELERO_Init (void)

Definition at line **99** of file **stm32f3_discovery_accelerometer.c**.

References **ACCELERO_ERROR**, **ACCELERO_OK**, and **AccelerometerDrv**.

void BSP_ACCELERO_Reset (void)

Reboot memory content of ACCELEROMETER.

Return values:

None

Definition at line **160** of file **stm32f3_discovery_accelerometer.c**.

References **AccelerometerDrv**.

STM32F3-Discovery BSP User Manual

Exported Functions

STM32F3-DISCOVERY Common

Functions

uint32_t	**BSP_GetVersion** (void) This method returns the STM32F3-DISCOVERY BSP Driver revision.
void	**BSP_LED_Init** (**Led_TypeDef** Led) Configures LED GPIO.
void	**BSP_LED_On** (**Led_TypeDef** Led) Turns selected LED On.
void	**BSP_LED_Off** (**Led_TypeDef** Led) Turns selected LED Off.
void	**BSP_LED_Toggle** (**Led_TypeDef** Led) Toggles the selected LED.
void	**BSP_PB_Init** (**Button_TypeDef** Button, **ButtonMode_TypeDef** ButtonMode) Configures Push Button GPIO and EXTI Line.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Returns the selected Push Button state.

Function Documentation

uint32_t BSP_GetVersion (void)

This method returns the STM32F3-DISCOVERY BSP Driver revision.

Return values:

version : 0xXYZR (8bits for each decimal, R for RC)

Definition at line **155** of file **stm32f3_discovery.c**.

References **__STM32F3_DISCO_BSP_VERSION**.

void BSP_LED_Init (Led_TypeDef Led)

Configures LED GPIO.

Parameters:

Led Specifies the Led to be configured. This parameter can be one of following parameters:

- LED_RED
- LED_BLUE
- LED_ORANGE
- LED_GREEN
- LED_GREEN2
- LED_ORANGE2
- LED_BLUE2
- LED_RED2

Return values:

None

Definition at line **174** of file **stm32f3_discovery.c**.

References **LED_PIN**, **LED_PORT**, and

LEDx_GPIO_CLK_ENABLE.

void BSP_LED_Off (Led_TypeDef Led)

Turns selected LED Off.

Parameters:

Led Specifies the Led to be set off. This parameter can be one of following parameters:

- LED_RED
- LED_BLUE
- LED_ORANGE
- LED_GREEN
- LED_GREEN2
- LED_ORANGE2
- LED_BLUE2
- LED_RED2

Return values:

None

Definition at line **225** of file **stm32f3_discovery.c**.

References **LED_PIN**, and **LED_PORT**.

void BSP_LED_On (Led_TypeDef Led)

Turns selected LED On.

Parameters:

Led Specifies the Led to be set on. This parameter can be one of following parameters:

- LED_RED
- LED4
- LED5

- LED6
- LED7
- LED8
- LED9
- LED10

Return values:

None

Definition at line **206** of file **stm32f3_discovery.c**.

References **LED_PIN**, and **LED_PORT**.

void BSP_LED_Toggle (Led_TypeDef Led)

Toggles the selected LED.

Parameters:

Led Specifies the Led to be toggled. This parameter can be one of following parameters:

- LED_RED
- LED_BLUE
- LED_ORANGE
- LED_GREEN
- LED_GREEN2
- LED_ORANGE2
- LED_BLUE2
- LED_RED2

Return values:

None

Definition at line **244** of file **stm32f3_discovery.c**.

References **LED_PIN**, and **LED_PORT**.

uint32_t BSP_PB_GetState (Button_TypeDef Button)

Returns the selected Push Button state.

Parameters:

Button Specifies the Button to be checked. This parameter should be: BUTTON_USER

Return values:

The Button GPIO pin value.

Definition at line **300** of file **stm32f3_discovery.c**.

References **BUTTON_PIN**, and **BUTTON_PORT**.

void BSP_PB_Init (Button_TypeDef Button, ButtonMode_TypeDef ButtonMode)

Configures Push Button GPIO and EXTI Line.

Parameters:

Button Specifies the Button to be configured. This parameter should be: BUTTON_USER

ButtonMode Specifies Button mode. This parameter can be one of following parameters:

- BUTTON_MODE_GPIO: Button will be used as simple IO
- BUTTON_MODE_EXTI: Button will be connected to EXTI line with interrupt generation capability

Return values:

None

Definition at line **261** of file **stm32f3_discovery.c**.

References **BUTTON_IRQn**, **BUTTON_MODE_EXTI**, **BUTTON_MODE_GPIO**, **BUTTON_PIN**, **BUTTON_PORT**, and **BUTTONx_GPIO_CLK_ENABLE**.

Exported Functions

STM32F3-DISCOVERY GYROSCOPE

Functions

uint8_t	**BSP_GYRO_Init** (void) Set GYROSCOPE Initialization.
uint8_t	**BSP_GYRO_ReadID** (void) Read ID of Gyroscope component.
void	**BSP_GYRO_Reset** (void) Reboot memory content of GYROSCOPE.
void	**BSP_GYRO_ITConfig** (GYRO_InterruptConfigTypeDef *pIntConfig) Configure INT1 interrupt.
void	**BSP_GYRO_EnableIT** (uint8_t IntPin) Enable INT1 or INT2 interrupt.
void	**BSP_GYRO_DisableIT** (uint8_t IntPin) Disable INT1 or INT2 interrupt.
void	**BSP_GYRO_GetXYZ** (float *pfData) Get XYZ angular acceleration.

Function Documentation

void BSP_GYRO_DisableIT (uint8_t IntPin)

Disable INT1 or INT2 interrupt.

Parameters:

IntPin Interrupt pin This parameter can be:

- L3GD20_INT1
- L3GD20_INT2

Return values:

None

Definition at line **227** of file **stm32f3_discovery_gyroscope.c**.

References **GyroscopeDrv**.

void BSP_GYRO_EnableIT (uint8_t IntPin)

Enable INT1 or INT2 interrupt.

Parameters:

IntPin Interrupt pin This parameter can be:

- L3GD20_INT1
- L3GD20_INT2

Return values:

None

Definition at line **211** of file **stm32f3_discovery_gyroscope.c**.

References **GyroscopeDrv**.

void BSP_GYRO_GetXYZ (float * pfData)

Get XYZ angular acceleration.

Parameters:

pfData pointer on floating array

Return values:

None

Definition at line **240** of file **stm32f3_discovery_gyroscope.c**.

References **GyroscopeDrv**.

uint8_t BSP_GYRO_Init (void)

Set GYROSCOPE Initialization.

Return values:

GYRO_OK if no problem during initialization

Definition at line **102** of file **stm32f3_discovery_gyroscope.c**.

References **GYRO_ERROR**, **GYRO_OK**, and **GyroscopeDrv**.

void BSP_GYRO_ITConfig (GYRO_InterruptConfigTypeDef * pIntC

Configure INT1 interrupt.

Parameters:

pIntConfig pointer to a L3GD20_InterruptConfig_TypeDef structure that contains the configuration setting for the L3GD20 Interrupt.

Return values:

None

Definition at line **187** of file **stm32f3_discovery_gyroscope.c**.

References **GyroscopeDrv**.

uint8_t BSP_GYRO_ReadID (void)

Read ID of Gyroscope component.

Return values:

ID

Definition at line **157** of file **stm32f3_discovery_gyroscope.c**.

References **GyroscopeDrv**.

void BSP_GYRO_Reset (void)

Reboot memory content of GYROSCOPE.

Return values:

None

Definition at line **172** of file **stm32f3_discovery_gyroscope.c**.

References **GyroscopeDrv**.

Private Variables

STM32F3-DISCOVERY Common

Variables

GPIO_TypeDef *	**LED_PORT** [**LEDn**] LED variables.
const uint16_t	**LED_PIN** [**LEDn**]
GPIO_TypeDef *	**BUTTON_PORT** [**BUTTONn**] = {**USER_BUTTON_GPIO_PORT**} BUTTON variables.
const uint16_t	**BUTTON_PIN** [**BUTTONn**] = {**USER_BUTTON_PIN**}
const uint8_t	**BUTTON_IRQn** [**BUTTONn**] = {**USER_BUTTON_EXTI_IRQn**}
uint32_t	**SpixTimeout** = **SPIx_TIMEOUT_MAX** BUS variables.
static SPI_HandleTypeDef	**SpiHandle**
static I2C_HandleTypeDef	**I2cHandle**
uint32_t	**I2cxTimeout** = **I2Cx_TIMEOUT_MAX**

Variable Documentation

const uint8_t BUTTON_IRQn[BUTTONn] = {USER_BUTTON_EXTI_I

Definition at line 89 of file stm32f3_discovery.c.

Referenced by BSP_PB_Init().

const uint16_t BUTTON_PIN[BUTTONn] = {USER_BUTTON_PIN}

Definition at line 88 of file stm32f3_discovery.c.

Referenced by BSP_PB_GetState(), and BSP_PB_Init().

GPIO_TypeDef* BUTTON_PORT[BUTTONn] = {USER_BUTTON_GPI

BUTTON variables.

Definition at line 87 of file stm32f3_discovery.c.

Referenced by BSP_PB_GetState(), and BSP_PB_Init().

I2C_HandleTypeDef I2cHandle `[static]`

Definition at line 100 of file stm32f3_discovery.c.

Referenced by I2Cx_Error(), I2Cx_Init(), I2Cx_ReadData(), and I2Cx_WriteData().

uint32_t I2cxTimeout = I2Cx_TIMEOUT_MAX

Definition at line 101 of file stm32f3_discovery.c.

Referenced by **I2Cx_ReadData()**, and **I2Cx_WriteData()**.

const uint16_t LED_PIN[LEDn]

Initial value:

```
 {LED3_PIN, LED4_PIN, LED5_PIN, LED6_PIN,
                                LED7_PIN, LED8_P
IN, LED9_PIN, LED10_PIN}
```

Definition at line **81** of file **stm32f3_discovery.c**.

Referenced by **BSP_LED_Init()**, **BSP_LED_Off()**, **BSP_LED_On()**, and **BSP_LED_Toggle()**.

GPIO_TypeDef* LED_PORT[LEDn]

Initial value:

```
 {LED3_GPIO_PORT, LED4_GPIO_PORT, LED5_GPIO_PORT,
LED6_GPIO_PORT,
                                LED7_GPIO_PORT,
LED8_GPIO_PORT, LED9_GPIO_PORT, LED10_GPIO_PORT}
```

LED variables.

Definition at line **78** of file **stm32f3_discovery.c**.

Referenced by **BSP_LED_Init()**, **BSP_LED_Off()**, **BSP_LED_On()**, and **BSP_LED_Toggle()**.

SPI_HandleTypeDef SpiHandle [static]

Definition at line **96** of file **stm32f3_discovery.c**.

Referenced by **SPIx_Error()**, **SPIx_Init()**, and **SPIx_WriteRead()**.

uint32_t SpixTimeout = SPIx_TIMEOUT_MAX

BUS variables.

Definition at line **95** of file **stm32f3_discovery.c**.

Referenced by **SPIx_WriteRead()**.

STM32F3-Discovery BSP User Manual

Exported Types

STM32F3-DISCOVERY Common

Enumerations

enum	**Led_TypeDef** {
	LED3 = 0, **LED4** = 1, **LED5** = 2, **LED6** = 3,
	LED7 = 4, **LED8** = 5, **LED9** = 6, **LED10** = 7,
	LED_GREEN = LED6, **LED_ORANGE** = LED5, **LED_RED** = LED3, **LED_BLUE** = LED4,
	LED_GREEN_2 = LED7, **LED_ORANGE_2** = LED8, **LED_RED_2** = LED10, **LED_BLUE_2** = LED9
	}
	LED Types Definition. More...

enum	**Button_TypeDef** { **BUTTON_USER** = 0 } BUTTON Types Definition. More...
enum	**ButtonMode_TypeDef** { **BUTTON_MODE_GPIO** = 0, **BUTTON_MODE_EXTI** = 1 }

Enumeration Type Documentation

enum Button_TypeDef

BUTTON Types Definition.

Enumerator:

BUTTON_USER

Definition at line **105** of file **stm32f3_discovery.h**.

enum ButtonMode_TypeDef

Enumerator:

BUTTON_MODE_GPIO

BUTTON_MODE_EXTI

Definition at line **111** of file **stm32f3_discovery.h**.

enum Led_TypeDef

LED Types Definition.

Enumerator:

LED3

LED4

LED5

LED6

LED7

LED8

LED9

LED10

LED_GREEN
LED_ORANGE
LED_RED
LED_BLUE
LED_GREEN_2
LED_ORANGE_2
LED_RED_2
LED_BLUE_2

Definition at line **81** of file **stm32f3_discovery.h**.

STM32F3-DISCOVERY BUTTON

Exported Constants

Defines

#define	**BUTTONn** 1
#define	**USER_BUTTON_PIN** GPIO_PIN_0 push-button
#define	**USER_BUTTON_GPIO_PORT** GPIOA
#define	**USER_BUTTON_GPIO_CLK_ENABLE**() __HAL_RCC_GPIO
#define	**USER_BUTTON_GPIO_CLK_DISABLE**() __HAL_RCC_GPI
#define	**USER_BUTTON_EXTI_IRQn** EXTI0_IRQn
#define	**BUTTONx_GPIO_CLK_ENABLE**(__BUTTON__) do { if ((__B **BUTTON_USER**) **USER_BUTTON_GPIO_CLK_ENABLE**();} v
#define	**BUTTONx_GPIO_CLK_DISABLE**(__BUTTON__) (((__BUTT **BUTTON_USER**) ? **USER_BUTTON_GPIO_CLK_DISABLE**()

Define Documentation

#define BUTTONn 1

Definition at line 206 of file stm32f3_discovery.h.

#define BUTTONx_GPIO_CLK_DISABLE (__BUTTON__) (((__BU

Definition at line 219 of file stm32f3_discovery.h.

#define BUTTONx_GPIO_CLK_ENABLE (__BUTTON__) do { if (

Definition at line 217 of file stm32f3_discovery.h.

Referenced by BSP_PB_Init().

#define USER_BUTTON_EXTI_IRQn EXTI0_IRQn

Definition at line 215 of file stm32f3_discovery.h.

#define USER_BUTTON_GPIO_CLK_DISABLE ( ) __HAL_RCC_GI

Definition at line 214 of file stm32f3_discovery.h.

#define USER_BUTTON_GPIO_CLK_ENABLE ( ) __HAL_RCC_GF

Definition at line 213 of file stm32f3_discovery.h.

#define USER_BUTTON_GPIO_PORT GPIOA

Definition at line **212** of file **stm32f3_discovery.h**.

#define USER_BUTTON_PIN GPIO_PIN_0

push-button

Definition at line **211** of file **stm32f3_discovery.h**.

Link Operation functions

STM32F3-DISCOVERY Common

Functions

void	**GYRO_IO_Init** (void) Configures the GYROSCOPE SPI interface.
void	**GYRO_IO_Write** (uint8_t *pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite) Writes one byte to the GYROSCOPE.
void	**GYRO_IO_Read** (uint8_t *pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead) Reads a block of data from the GYROSCOPE.
void	**COMPASSACCELERO_IO_Init** (void) Configures COMPASS / ACCELEROMETER I2C interface.
void	**COMPASSACCELERO_IO_ITConfig** (void) Configures COMPASS / ACCELERO click IT.
void	**COMPASSACCELERO_IO_Write** (uint16_t DeviceAddr, uint8_t RegisterAddr, uint8_t Value) Writes one byte to the COMPASS / ACCELEROMETER.
uint8_t	**COMPASSACCELERO_IO_Read** (uint16_t DeviceAddr, uint8_t RegisterAddr) Reads a block of data from the COMPASS / ACCELEROMETER.

Function Documentation

void COMPASSACCELERO_IO_Init (void)

Configures COMPASS / ACCELEROMETER I2C interface.

Return values:

None

Definition at line 640 of file stm32f3_discovery.c.

References ACCELERO_DRDY_EXTI_IRQn, ACCELERO_DRDY_GPIO_CLK_ENABLE, ACCELERO_DRDY_GPIO_PORT, ACCELERO_DRDY_PIN, ACCELERO_INT1_PIN, ACCELERO_INT2_PIN, ACCELERO_INT_GPIO_CLK_ENABLE, ACCELERO_INT_GPIO_PORT, and I2Cx_Init().

void COMPASSACCELERO_IO_ITConfig (void)

Configures COMPASS / ACCELERO click IT.

Return values:

None

Definition at line 675 of file stm32f3_discovery.c.

References ACCELERO_INT1_EXTI_IRQn, ACCELERO_INT1_PIN, ACCELERO_INT2_PIN, ACCELERO_INT_GPIO_CLK_ENABLE, and ACCELERO_INT_GPIO_PORT.

uint8_t COMPASSACCELERO_IO_Read (uint16_t DeviceAddr, uint8_t RegisterAddr)

Reads a block of data from the COMPASS / ACCELEROMETER.

Parameters:

DeviceAddr	specifies the slave address to be programmed(ACC_I2C_ADDRESS or MAG_I2C_ADDRESS).
RegisterAddr	specifies the COMPASS / ACCELEROMETER internal address register to read from

Return values:

ACCELEROMETER	register value

Definition at line **714** of file **stm32f3_discovery.c**.

References **I2Cx_ReadData()**.

```
void COMPASSACCELERO_IO_Write ( uint16_t DeviceAddr,
                                uint8_t  RegisterAddr,
                                uint8_t  Value
                              )
```

Writes one byte to the COMPASS / ACCELEROMETER.

Parameters:

DeviceAddr	specifies the slave address to be programmed.
RegisterAddr	specifies the COMPASS / ACCELEROMETER register to be written.
Value	Data to be written

Return values:

None	

Definition at line **702** of file **stm32f3_discovery.c**.

References **I2Cx_WriteData()**.

void GYRO_IO_Init (void)

Configures the GYROSCOPE SPI interface.

Return values:

None

Definition at line **535** of file **stm32f3_discovery.c**.

References **GYRO_CS_GPIO_CLK_ENABLE**, **GYRO_CS_GPIO_PORT**, **GYRO_CS_HIGH**, **GYRO_CS_PIN**, **GYRO_INT1_PIN**, **GYRO_INT2_PIN**, **GYRO_INT_GPIO_CLK_ENABLE**, **GYRO_INT_GPIO_PORT**, and **SPIx_Init()**.

void GYRO_IO_Read (uint8_t * pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead)

Reads a block of data from the GYROSCOPE.

Parameters:

pBuffer	pointer to the buffer that receives the data read from the GYROSCOPE.
ReadAddr	GYROSCOPE's internal address to read from.
NumByteToRead	number of bytes to read from the GYROSCOPE.

Return values:

None

Definition at line **604** of file **stm32f3_discovery.c**.

References **DUMMY_BYTE**, **GYRO_CS_HIGH**, **GYRO_CS_LOW**, **MULTIPLEBYTE_CMD**, **READWRITE_CMD**, and **SPIx_WriteRead()**.

```
void GYRO_IO_Write ( uint8_t *  pBuffer,
                     uint8_t    WriteAddr,
                     uint16_t   NumByteToWrite
                   )
```

Writes one byte to the GYROSCOPE.

Parameters:

pBuffer	pointer to the buffer containing the data to be written to the GYROSCOPE.
WriteAddr	GYROSCOPE's internal address to write to.
NumByteToWrite	Number of bytes to write.

Return values:

None

Definition at line **569** of file **stm32f3_discovery.c**.

References **GYRO_CS_HIGH**, **GYRO_CS_LOW**, **MULTIPLEBYTE_CMD**, and **SPIx_WriteRead()**.

STM32F3-DISCOVERY COM

Exported Constants

Defines

#define	**DISCOVERY_SPIx** SPI1 Definition for SPI Interface pins (SPI1 used)
#define	**DISCOVERY_SPIx_CLK_ENABLE**() __HAL_RCC_SPI1_CL
#define	**DISCOVERY_SPIx_GPIO_PORT** GPIOA /* GPIOA */
#define	**DISCOVERY_SPIx_AF** GPIO_AF5_SPI1
#define	**DISCOVERY_SPIx_GPIO_CLK_ENABLE**() __HAL_RCC_GI
#define	**DISCOVERY_SPIx_GPIO_CLK_DISABLE**() __HAL_RCC_G
#define	**DISCOVERY_SPIx_SCK_PIN** GPIO_PIN_5 /* PA.05 */
#define	**DISCOVERY_SPIx_MISO_PIN** GPIO_PIN_6 /* PA.06 */
#define	**DISCOVERY_SPIx_MOSI_PIN** GPIO_PIN_7 /* PA.07 */
#define	**SPIx_TIMEOUT_MAX** ((uint32_t)0x1000)
#define	**DISCOVERY_I2Cx** I2C1 Definition for I2C Interface pins (I2C1 used)
#define	**DISCOVERY_I2Cx_CLK_ENABLE**() __HAL_RCC_I2C1_CLI
#define	**DISCOVERY_I2Cx_CLK_DISABLE**() __HAL_RCC_I2C1_CL
#define	**DISCOVERY_I2Cx_FORCE_RESET**() __HAL_RCC_I2C1_F
#define	**DISCOVERY_I2Cx_RELEASE_RESET**() __HAL_RCC_I2C1_
#define	**DISCOVERY_I2Cx_SCL_PIN** GPIO_PIN_6 /* PB.06 */
#define	**DISCOVERY_I2Cx_SDA_PIN** GPIO_PIN_7 /* PB.07 */
#define	**DISCOVERY_I2Cx_GPIO_PORT** GPIOB /* GPIOB */
#define	**DISCOVERY_I2Cx_GPIO_CLK_ENABLE**() __HAL_RCC_GI
#define	**DISCOVERY_I2Cx_GPIO_CLK_DISABLE**() __HAL_RCC_G
#define	**DISCOVERY_I2Cx_AF** GPIO_AF4_I2C1
#define	**I2Cx_TIMEOUT_MAX** 0x10000

Define Documentation

#define DISCOVERY_I2Cx I2C1

Definition for I2C Interface pins (I2C1 used)

Definition at line 253 of file stm32f3_discovery.h.

Referenced by I2Cx_Init().

#define DISCOVERY_I2Cx_AF GPIO_AF4_I2C1

Definition at line 265 of file stm32f3_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_CLK_DISABLE ( ) __HAL_RCC_I2C1_C

Definition at line 255 of file stm32f3_discovery.h.

#define DISCOVERY_I2Cx_CLK_ENABLE ( ) __HAL_RCC_I2C1_C

Definition at line 254 of file stm32f3_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_FORCE_RESET ( ) __HAL_RCC_I2C1_

Definition at line 256 of file stm32f3_discovery.h.

#define DISCOVERY_I2Cx_GPIO_CLK_DISABLE ( ) __HAL_RCC_

Definition at line **264** of file **stm32f3_discovery.h**.

#define DISCOVERY_I2Cx_GPIO_CLK_ENABLE ( ) __HAL_RCC_(

Definition at line **263** of file **stm32f3_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_GPIO_PORT GPIOB /* GPIOB */

Definition at line **262** of file **stm32f3_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_RELEASE_RESET ( ) __HAL_RCC_I2C

Definition at line **257** of file **stm32f3_discovery.h**.

#define DISCOVERY_I2Cx_SCL_PIN GPIO_PIN_6 /* PB.06 */

Definition at line **259** of file **stm32f3_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_SDA_PIN GPIO_PIN_7 /* PB.07 */

Definition at line **260** of file **stm32f3_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_SPIx SPI1

Definition for SPI Interface pins (SPI1 used)

Definition at line **233** of file **stm32f3_discovery.h**.

Referenced by **SPIx_Init()**.

#define DISCOVERY_SPIx_AF GPIO_AF5_SPI1

Definition at line **236** of file **stm32f3_discovery.h**.

Referenced by **SPIx_MspInit()**.

#define DISCOVERY_SPIx_CLK_ENABLE ( ) __HAL_RCC_SPI1_C

Definition at line **234** of file **stm32f3_discovery.h**.

Referenced by **SPIx_MspInit()**.

#define DISCOVERY_SPIx_GPIO_CLK_DISABLE ( ) __HAL_RCC_

Definition at line **238** of file **stm32f3_discovery.h**.

#define DISCOVERY_SPIx_GPIO_CLK_ENABLE ( ) __HAL_RCC_(

Definition at line **237** of file **stm32f3_discovery.h**.

Referenced by **SPIx_MspInit()**.

#define DISCOVERY_SPIx_GPIO_PORT GPIOA /* GPIOA */

Definition at line **235** of file **stm32f3_discovery.h**.

Referenced by **SPIx_MspInit()**.

#define DISCOVERY_SPIx_MISO_PIN GPIO_PIN_6 /* PA.06 */

Definition at line 240 of file stm32f3_discovery.h.

Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_MOSI_PIN GPIO_PIN_7 /* PA.07 */

Definition at line 241 of file stm32f3_discovery.h.

Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_SCK_PIN GPIO_PIN_5 /* PA.05 */

Definition at line 239 of file stm32f3_discovery.h.

Referenced by SPIx_MspInit().

#define I2Cx_TIMEOUT_MAX 0x10000

Definition at line 272 of file stm32f3_discovery.h.

#define SPIx_TIMEOUT_MAX ((uint32_t)0x1000)

Definition at line 247 of file stm32f3_discovery.h.

Enumerations

Exported Types

STM32F3-DISCOVERY GYROSCOPE

Enumerations

enum	**GYRO_StatusTypeDef** { **GYRO_OK** = 0, **GYRO_ERROR** = 1, **GYRO_TIMEOUT** = 2 }

Enumeration Type Documentation

enum GYRO_StatusTypeDef

Enumerator:

GYRO_OK

GYRO_ERROR

GYRO_TIMEOUT

Definition at line **66** of file **stm32f3_discovery_gyroscope.h**.

Private Variables

STM32F3-DISCOVERY GYROSCOPE

Variables

static GYRO_DrvTypeDef *	**GyroscopeDrv**

Variable Documentation

GYRO_DrvTypeDef* GyroscopeDrv [static]

Definition at line 79 of file stm32f3_discovery_gyroscope.c.

Referenced by BSP_GYRO_DisableIT(), BSP_GYRO_EnableIT(), BSP_GYRO_GetXYZ(), BSP_GYRO_Init(), BSP_GYRO_ITConfig(), BSP_GYRO_ReadID(), and BSP_GYRO_Reset().

Bus Operation functions

STM32F3-DISCOVERY Common

Functions

static void	**I2Cx_Init** (void) Discovery I2Cx Bus initialization.
static void	**I2Cx_WriteData** (uint16_t Addr, uint8_t Reg, uint8_t Value) Write a value in a register of the device through BUS.
static uint8_t	**I2Cx_ReadData** (uint16_t Addr, uint8_t Reg) Read a value in a register of the device through BUS.
static void	**I2Cx_Error** (void) I2C3 error treatment function.
static void	**I2Cx_MspInit** (I2C_HandleTypeDef *hi2c) Discovery I2Cx MSP Initialization.
static void	**SPIx_Init** (void) SPIx Bus initialization.
static uint8_t	**SPIx_WriteRead** (uint8_t Byte) Sends a Byte through the SPI interface and return the Byte received from the SPI bus.
static void	**SPIx_Error** (void) SPIx error treatment function.
static void	**SPIx_MspInit** (SPI_HandleTypeDef *hspi) SPI MSP Init.

Function Documentation

static void I2Cx_Error (void) [static]

I2C3 error treatment function.

Return values:

None

Definition at line 413 of file stm32f3_discovery.c.

References I2cHandle, and I2Cx_Init().

Referenced by I2Cx_ReadData(), and I2Cx_WriteData().

static void I2Cx_Init (void) [static]

Discovery I2Cx Bus initialization.

Return values:

None

Definition at line 347 of file stm32f3_discovery.c.

References ACCELERO_I2C_ADDRESS, DISCOVERY_I2Cx, I2cHandle, and I2Cx_MspInit().

Referenced by COMPASSACCELERO_IO_Init(), and I2Cx_Error().

static void I2Cx_MspInit (I2C_HandleTypeDef * hi2c) [static]

Discovery I2Cx MSP Initialization.

Parameters:

hi2c I2C handle

Return values:

None

Definition at line **323** of file **stm32f3_discovery.c**.

References **DISCOVERY_I2Cx_AF**, **DISCOVERY_I2Cx_CLK_ENABLE**, **DISCOVERY_I2Cx_GPIO_CLK_ENABLE**, **DISCOVERY_I2Cx_GPIO_PORT**, **DISCOVERY_I2Cx_SCL_PIN**, and **DISCOVERY_I2Cx_SDA_PIN**.

Referenced by **I2Cx_Init()**.

static uint8_t I2Cx_ReadData (uint16_t Addr, uint8_t Reg) `[static]`

Read a value in a register of the device through BUS.

Parameters:

Addr Device address on BUS Bus.

Reg The target register address to write

Return values:

Data read at register @

Definition at line **392** of file **stm32f3_discovery.c**.

References **I2cHandle**, **I2Cx_Error()**, and **I2cxTimeout**.

Referenced by **COMPASSACCELERO_IO_Read()**.

static void I2Cx_WriteData (uint16_t Addr,

	uint8_t	Reg,
	uint8_t	Value
)		[static]

Write a value in a register of the device through BUS.

Parameters:

- **Addr** Device address on BUS Bus.
- **Reg** The target register address to write
- **Value** The target register value to be written

Return values:

- **None**

Definition at line **372** of file **stm32f3_discovery.c**.

References **I2cHandle**, **I2Cx_Error()**, and **I2cxTimeout**.

Referenced by **COMPASSACCELERO_IO_Write()**.

static void SPIx_Error (void) [static]

SPIx error treatment function.

Return values:

- **None**

Definition at line **485** of file **stm32f3_discovery.c**.

References **SpiHandle**, and **SPIx_Init()**.

Referenced by **SPIx_WriteRead()**.

static void SPIx_Init (void) [static]

SPIx Bus initialization.

Return values:

None

Definition at line **430** of file **stm32f3_discovery.c**.

References **DISCOVERY_SPIx**, **SpiHandle**, and **SPIx_MspInit()**.

Referenced by **GYRO_IO_Init()**, and **SPIx_Error()**.

static void SPIx_MspInit (SPI_HandleTypeDef * hspi) [static]

SPI MSP Init.

Parameters:

hspi SPI handle

Return values:

None

Definition at line **500** of file **stm32f3_discovery.c**.

References **DISCOVERY_SPIx_AF**, **DISCOVERY_SPIx_CLK_ENABLE**, **DISCOVERY_SPIx_GPIO_CLK_ENABLE**, **DISCOVERY_SPIx_GPIO_PORT**, **DISCOVERY_SPIx_MISO_PIN**, **DISCOVERY_SPIx_MOSI_PIN**, and **DISCOVERY_SPIx_SCK_PIN**.

Referenced by **SPIx_Init()**.

static uint8_t SPIx_WriteRead (uint8_t Byte) [static]

Sends a Byte through the SPI interface and return the Byte received from the SPI bus.

Parameters:

Byte Byte send.

Return values:

The received byte value

Definition at line **465** of file **stm32f3_discovery.c**.

References **SpiHandle**, **SPIx_Error()**, and **SpixTimeout**.

Referenced by **GYRO_IO_Read()**, and **GYRO_IO_Write()**.

STM32F3-Discovery BSP User Manual

STM32F3-DISCOVERY LED

Exported Constants

Define for STM32F3_DISCOVERY board. More...

Defines

```
#define LEDn 8
#define LED6_PIN GPIO_PIN_15
#define LED6_GPIO_PORT GPIOE
#define LED6_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED6_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED8_PIN GPIO_PIN_14
#define LED8_GPIO_PORT GPIOE
#define LED8_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED8_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED10_PIN GPIO_PIN_13
#define LED10_GPIO_PORT GPIOE
#define LED10_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED10_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_
#define LED9_PIN GPIO_PIN_12
#define LED9_GPIO_PORT GPIOE
#define LED9_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED9_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED7_PIN GPIO_PIN_11
#define LED7_GPIO_PORT GPIOE
#define LED7_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED7_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED5_PIN GPIO_PIN_10
#define LED5_GPIO_PORT GPIOE
#define LED5_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED5_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED3_PIN GPIO_PIN_9
#define LED3_GPIO_PORT GPIOE
#define LED3_GPIO_CLK_ENABLE() __HAL_RCC_GPIOE_CLK_E
#define LED3_GPIO_CLK_DISABLE() __HAL_RCC_GPIOE_CLK_D
#define LED4_PIN GPIO_PIN_8
#define LED4_GPIO_PORT GPIOE
```

```
#define LED4_GPIO_CLK_ENABLE()   __HAL_RCC_GPIOE_CLK_E
#define LED4_GPIO_CLK_DISABLE()   __HAL_RCC_GPIOE_CLK_D
#define LEDx_GPIO_CLK_ENABLE(__LED__)
#define LEDx_GPIO_CLK_DISABLE(__LED__)
```

Detailed Description

Define for STM32F3_DISCOVERY board.

Define Documentation

#define LED10_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_CLI

Definition at line 154 of file stm32f3_discovery.h.

#define LED10_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_CLK

Definition at line 153 of file stm32f3_discovery.h.

#define LED10_GPIO_PORT GPIOE

Definition at line 152 of file stm32f3_discovery.h.

#define LED10_PIN GPIO_PIN_13

Definition at line 151 of file stm32f3_discovery.h.

#define LED3_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 174 of file stm32f3_discovery.h.

#define LED3_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 173 of file stm32f3_discovery.h.

#define LED3_GPIO_PORT GPIOE

Definition at line 172 of file stm32f3_discovery.h.

#define LED3_PIN GPIO_PIN_9

Definition at line 171 of file stm32f3_discovery.h.

#define LED4_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 179 of file stm32f3_discovery.h.

#define LED4_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 178 of file stm32f3_discovery.h.

#define LED4_GPIO_PORT GPIOE

Definition at line 177 of file stm32f3_discovery.h.

#define LED4_PIN GPIO_PIN_8

Definition at line 176 of file stm32f3_discovery.h.

#define LED5_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 169 of file stm32f3_discovery.h.

#define LED5_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 168 of file stm32f3_discovery.h.

#define LED5_GPIO_PORT GPIOE

Definition at line 167 of file stm32f3_discovery.h.

#define LED5_PIN GPIO_PIN_10

Definition at line 166 of file stm32f3_discovery.h.

#define LED6_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 144 of file stm32f3_discovery.h.

#define LED6_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 143 of file stm32f3_discovery.h.

#define LED6_GPIO_PORT GPIOE

Definition at line 142 of file stm32f3_discovery.h.

#define LED6_PIN GPIO_PIN_15

Definition at line 141 of file stm32f3_discovery.h.

#define LED7_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 164 of file stm32f3_discovery.h.

#define LED7_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 163 of file stm32f3_discovery.h.

#define LED7_GPIO_PORT GPIOE

Definition at line 162 of file stm32f3_discovery.h.

#define LED7_PIN GPIO_PIN_11

Definition at line 161 of file stm32f3_discovery.h.

#define LED8_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 149 of file stm32f3_discovery.h.

#define LED8_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 148 of file stm32f3_discovery.h.

#define LED8_GPIO_PORT GPIOE

Definition at line 147 of file stm32f3_discovery.h.

#define LED8_PIN GPIO_PIN_14

Definition at line 146 of file stm32f3_discovery.h.

#define LED9_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line 159 of file stm32f3_discovery.h.

#define LED9_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOE_CLK_

Definition at line **158** of file **stm32f3_discovery.h**.

#define LED9_GPIO_PORT GPIOE

Definition at line **157** of file **stm32f3_discovery.h**.

#define LED9_PIN GPIO_PIN_12

Definition at line **156** of file **stm32f3_discovery.h**.

#define LEDn 8

Definition at line **139** of file **stm32f3_discovery.h**.

#define LEDx_GPIO_CLK_DISABLE (__LED__)

Value:

```
(((__LED__) == LED3) ? LED3_GPIO_CLK_DISABLE() :\
                                                ((__LED_
_) == LED4) ? LED4_GPIO_CLK_DISABLE() :\
                                                ((__LED_
_) == LED5) ? LED5_GPIO_CLK_DISABLE() :\
                                                ((__LED_
_) == LED6) ? LED6_GPIO_CLK_DISABLE() :\
                                                ((__LED_
_) == LED7) ? LED7_GPIO_CLK_DISABLE() :\
                                                ((__LED_
_) == LED8) ? LED8_GPIO_CLK_DISABLE() :\
                                                ((__LED_
_) == LED9) ? LED9_GPIO_CLK_DISABLE() :\
```

```
                                                        ((__LED_
_) == LED10) ? LED10_GPIO_CLK_DISABLE() : 0 )
```

Definition at line **190** of file **stm32f3_discovery.h**.

#define LEDx_GPIO_CLK_ENABLE (__LED__)

Value:

```
do { if ((__LED__) == LED3) LED3_GPIO_CLK_ENABLE(
); else\
                                                  if (
(__LED__) == LED4) LED4_GPIO_CLK_ENABLE(); else\
                                                  if (
(__LED__) == LED5) LED5_GPIO_CLK_ENABLE(); else\
                                                  if (
(__LED__) == LED6) LED6_GPIO_CLK_ENABLE(); else\
                                                  if (
(__LED__) == LED7) LED7_GPIO_CLK_ENABLE(); else\
                                                  if (
(__LED__) == LED8) LED8_GPIO_CLK_ENABLE(); else\
                                                  if (
(__LED__) == LED9) LED9_GPIO_CLK_ENABLE(); else\
                                                  if (
(__LED__) == LED10) LED10_GPIO_CLK_ENABLE();} whi
le(0)
```

Definition at line **181** of file **stm32f3_discovery.h**.

Referenced by **BSP_LED_Init()**.

STM32F3-Discovery BSP User Manual

Firmware Directory Reference

Directories

directory	**Drivers**

STM32F3-Discovery BSP User Manual

Drivers Directory Reference

Directories

directory	**BSP**

STM32F3-Discovery BSP User Manual

BSP Directory Reference

Directories

directory	**STM32F3-Discovery**

STM32F3-Discovery BSP User Manual

STM32F3-Discovery Directory Reference

Files

file	**stm32f3_discovery.c** [code] This file provides set of firmware functions to manage Leds and push-button available on STM32F3-DISCOVERY Kit from STMicroelectronics.
file	**stm32f3_discovery.h** [code] This file contains definitions for STM32F3-Discovery's Leds, push- buttons hardware resources.
file	**stm32f3_discovery_accelerometer.c** [code] This file provides a set of functions needed to manage the ACCELEROMETER MEMS available on STM32F3-Discovery Kit.
file	**stm32f3_discovery_accelerometer.h** [code] This file contains definitions for **stm32f3_discovery_accelerometer.c** firmware driver.
file	**stm32f3_discovery_gyroscope.c** [code] This file provides a set of functions needed to manage the l3gd20 MEMS accelerometer available on STM32F3-Discovery Kit.
file	**stm32f3_discovery_gyroscope.h** [code]

This file contains definitions for **stm32f3_discovery_gyroscope.c** firmware driver.

STM32F3-Discovery BSP User Manual

stm32f3_discovery.h

Go to the documentation of this file.

```
/**
  ******************************************************************************
  * @file    stm32f3_discovery.h
  * @author  MCD Application Team
  * @brief   This file contains definitions for STM32F3-Discovery's Leds, push-
  *          buttons hardware resources.
  ******************************************************************************
  * @attention
  *
  * <h2><center>&copy; COPYRIGHT(c) 2016 STMicroelectronics</center></h2>
  *
  * Redistribution and use in source and binary forms, with or without modification,
  * are permitted provided that the following conditions are met:
  *   1. Redistributions of source code must retain the above copyright notice,
  *      this list of conditions and the following disclaimer.
  *   2. Redistributions in binary form must
```

```
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ******************************************
**********************************
00035   */
00036
```

```
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32F3_DISCOVERY_H
00039 #define __STM32F3_DISCOVERY_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044

00045 /** @addtogroup BSP
00046   * @{
00047   */
00048
00049 /** @defgroup STM32F3_DISCOVERY STM32F3-DISC
OVERY
00050   * @{
00051   */
00052
00053 /* Includes ---------------------------------
-----------------------------------*/
00054 #include "stm32f3xx_hal.h"
00055
00056 /** @defgroup STM32F3_DISCOVERY_Common STM32
F3-DISCOVERY Common
00057   * @{
00058   */
00059
00060 /** @defgroup STM32F3_DISCOVERY_Private_Cons
tants Private Constants
00061   * @{
00062   */
00063 /**
00064 * @}
00065 */
00066
00067 /** @defgroup STM32F3_DISCOVERY_Private_Vari
```

```
ables Private Variables
00068   * @{
00069   */
00070 /**
00071 * @}
00072   */
00073
00074 /** @defgroup STM32F3_DISCOVERY_Exported_Typ
es Exported Types
00075   * @{
00076   */
00077
00078 /**
00079  * @brief LED Types Definition
00080  */
00081 typedef enum
00082 {
00083   LED3 = 0,
00084   LED4 = 1,
00085   LED5 = 2,
00086   LED6 = 3,
00087   LED7 = 4,
00088   LED8 = 5,
00089   LED9 = 6,
00090   LED10 = 7,
00091
00092   LED_GREEN  = LED6,
00093   LED_ORANGE = LED5,
00094   LED_RED    = LED3,
00095   LED_BLUE   = LED4,
00096   LED_GREEN_2  = LED7,
00097   LED_ORANGE_2 = LED8,
00098   LED_RED_2    = LED10,
00099   LED_BLUE_2   = LED9
00100 }Led_TypeDef;
00101
00102 /**
```

```
00103  * @brief BUTTON Types Definition
00104  */
00105 typedef enum
00106 {
00107   BUTTON_USER = 0
00108
00109 }Button_TypeDef;
00110
00111 typedef enum
00112 {
00113   BUTTON_MODE_GPIO = 0,
00114   BUTTON_MODE_EXTI = 1
00115
00116 }ButtonMode_TypeDef;
00117
00118
00119 /**
00120   * @}
00121   */
00122
00123 /** @defgroup STM32F3_DISCOVERY_Exported_Con
stants Exported Constants
00124   * @{
00125   */
00126
00127 /**
00128   * @brief  Define for STM32F3_DISCOVERY boa
rd
00129   */
00130
00131 #if !defined (USE_STM32F3_DISCO)
00132  #define USE_STM32F3_DISCO
00133 #endif
00134
00135
00136 /** @defgroup STM32F3_DISCOVERY_LED STM32F3-
DISCOVERY LED
```

```
00137   * @{
00138   */
00139 #define LEDn                                8
00140
00141 #define LED6_PIN                            GPI
O_PIN_15
00142 #define LED6_GPIO_PORT                      GPI
OE
00143 #define LED6_GPIO_CLK_ENABLE()              __H
AL_RCC_GPIOE_CLK_ENABLE()
00144 #define LED6_GPIO_CLK_DISABLE()             __H
AL_RCC_GPIOE_CLK_DISABLE()
00145
00146 #define LED8_PIN                            GPI
O_PIN_14
00147 #define LED8_GPIO_PORT                      GPI
OE
00148 #define LED8_GPIO_CLK_ENABLE()              __H
AL_RCC_GPIOE_CLK_ENABLE()
00149 #define LED8_GPIO_CLK_DISABLE()             __H
AL_RCC_GPIOE_CLK_DISABLE()
00150
00151 #define LED10_PIN                           GPI
O_PIN_13
00152 #define LED10_GPIO_PORT                     GPI
OE
00153 #define LED10_GPIO_CLK_ENABLE()              __
HAL_RCC_GPIOE_CLK_ENABLE()
00154 #define LED10_GPIO_CLK_DISABLE()             __
HAL_RCC_GPIOE_CLK_DISABLE()
00155
00156 #define LED9_PIN                            GPI
O_PIN_12
00157 #define LED9_GPIO_PORT                      GPI
OE
00158 #define LED9_GPIO_CLK_ENABLE()              __H
AL_RCC_GPIOE_CLK_ENABLE()
```

```
00159 #define LED9_GPIO_CLK_DISABLE()          __H
AL_RCC_GPIOE_CLK_DISABLE()
00160
00161 #define LED7_PIN                         GPI
O_PIN_11
00162 #define LED7_GPIO_PORT                   GPI
OE
00163 #define LED7_GPIO_CLK_ENABLE()           __H
AL_RCC_GPIOE_CLK_ENABLE()
00164 #define LED7_GPIO_CLK_DISABLE()          __H
AL_RCC_GPIOE_CLK_DISABLE()
00165
00166 #define LED5_PIN                         GPI
O_PIN_10
00167 #define LED5_GPIO_PORT                   GPI
OE
00168 #define LED5_GPIO_CLK_ENABLE()           __H
AL_RCC_GPIOE_CLK_ENABLE()
00169 #define LED5_GPIO_CLK_DISABLE()          __H
AL_RCC_GPIOE_CLK_DISABLE()
00170
00171 #define LED3_PIN                         GPI
O_PIN_9
00172 #define LED3_GPIO_PORT                   GPI
OE
00173 #define LED3_GPIO_CLK_ENABLE()           __H
AL_RCC_GPIOE_CLK_ENABLE()
00174 #define LED3_GPIO_CLK_DISABLE()          __H
AL_RCC_GPIOE_CLK_DISABLE()
00175
00176 #define LED4_PIN                         GPI
O_PIN_8
00177 #define LED4_GPIO_PORT                   GPI
OE
00178 #define LED4_GPIO_CLK_ENABLE()           __H
AL_RCC_GPIOE_CLK_ENABLE()
00179 #define LED4_GPIO_CLK_DISABLE()          __H
```

```
AL_RCC_GPIOE_CLK_DISABLE()
00180
00181 #define LEDx_GPIO_CLK_ENABLE(__LED__)   do {
 if ((__LED__) == LED3) LED3_GPIO_CLK_ENABLE(); el
se\
00182
 if ((__LED__) == LED4) LED4_GPIO_CLK_ENABLE(); el
se\
00183
 if ((__LED__) == LED5) LED5_GPIO_CLK_ENABLE(); el
se\
00184
 if ((__LED__) == LED6) LED6_GPIO_CLK_ENABLE(); el
se\
00185
 if ((__LED__) == LED7) LED7_GPIO_CLK_ENABLE(); el
se\
00186
 if ((__LED__) == LED8) LED8_GPIO_CLK_ENABLE(); el
se\
00187
 if ((__LED__) == LED9) LED9_GPIO_CLK_ENABLE(); el
se\
00188
 if ((__LED__) == LED10) LED10_GPIO_CLK_ENABLE();}
 while(0)
00189
00190 #define LEDx_GPIO_CLK_DISABLE(__LED__)  (((_
_LED__) == LED3) ? LED3_GPIO_CLK_DISABLE() :\
00191                                          ((_
_LED__) == LED4) ? LED4_GPIO_CLK_DISABLE() :\
00192                                          ((_
_LED__) == LED5) ? LED5_GPIO_CLK_DISABLE() :\
00193                                          ((_
_LED__) == LED6) ? LED6_GPIO_CLK_DISABLE() :\
00194                                          ((_
_LED__) == LED7) ? LED7_GPIO_CLK_DISABLE() :\
```

```
00195                                                   ((__LED__) == LED8) ? LED8_GPIO_CLK_DISABLE() :\
00196                                                   ((__LED__) == LED9) ? LED9_GPIO_CLK_DISABLE() :\
00197                                                   ((__LED__) == LED10) ? LED10_GPIO_CLK_DISABLE() : 0 )
00198
00199 /**
00200   * @}
00201   */
00202
00203 /** @defgroup STM32F3_DISCOVERY_BUTTON STM32F3-DISCOVERY BUTTON
00204   * @{
00205   */
00206 #define BUTTONn                                 1
00207
00208 /**
00209  * @brief push-button
00210  */
00211 #define USER_BUTTON_PIN                         GPIO_PIN_0
00212 #define USER_BUTTON_GPIO_PORT                   GPIOA
00213 #define USER_BUTTON_GPIO_CLK_ENABLE()           __HAL_RCC_GPIOA_CLK_ENABLE()
00214 #define USER_BUTTON_GPIO_CLK_DISABLE()          __HAL_RCC_GPIOA_CLK_DISABLE()
00215 #define USER_BUTTON_EXTI_IRQn                   EXTI0_IRQn
00216
00217 #define BUTTONx_GPIO_CLK_ENABLE(__BUTTON__)  do { if ((__BUTTON__) == BUTTON_USER) USER_BUTTON_GPIO_CLK_ENABLE();} while(0)
00218
00219 #define BUTTONx_GPIO_CLK_DISABLE(__BUTTON__)  (((__BUTTON__) == BUTTON_USER) ? USER_BUTTON_GPIO
```

```
_CLK_DISABLE() : 0 )
00220
00221 /**
00222   * @}
00223   */
00224
00225 /** @defgroup STM32F3_DISCOVERY_COM STM32F3-
DISCOVERY COM
00226   * @{
00227   */
00228
00229 /*##################### SPIx ###############
####################*/
00230 /**
00231   * @brief  Definition for SPI Interface pin
s (SPI1 used)
00232   */
00233 #define DISCOVERY_SPIx
  SPI1
00234 #define DISCOVERY_SPIx_CLK_ENABLE()
  __HAL_RCC_SPI1_CLK_ENABLE()
00235 #define DISCOVERY_SPIx_GPIO_PORT
  GPIOA                       /* GPIOA */
00236 #define DISCOVERY_SPIx_AF
  GPIO_AF5_SPI1
00237 #define DISCOVERY_SPIx_GPIO_CLK_ENABLE()
  __HAL_RCC_GPIOA_CLK_ENABLE()
00238 #define DISCOVERY_SPIx_GPIO_CLK_DISABLE()
  __HAL_RCC_GPIOA_CLK_DISABLE()
00239 #define DISCOVERY_SPIx_SCK_PIN
  GPIO_PIN_5                  /* PA.05 */
00240 #define DISCOVERY_SPIx_MISO_PIN
  GPIO_PIN_6                  /* PA.06 */
00241 #define DISCOVERY_SPIx_MOSI_PIN
  GPIO_PIN_7                  /* PA.07 */
00242 /* Maximum Timeout values for flags waiting
loops. These timeouts are not based
```

```
00243    on accurate values, they just guarantee t
hat the application will not remain
00244    stuck if the SPI communication is corrupt
ed.
00245    You may modify these timeout values depen
ding on CPU frequency and application
00246    conditions (interrupts routines ...). */

00247 #define SPIx_TIMEOUT_MAX
  ((uint32_t)0x1000)
00248
00249 /*##################### I2Cx ###############
#####################*/
00250 /**
00251   * @brief  Definition for I2C Interface pin
s (I2C1 used)
00252   */
00253 #define DISCOVERY_I2Cx
  I2C1
00254 #define DISCOVERY_I2Cx_CLK_ENABLE()
  __HAL_RCC_I2C1_CLK_ENABLE()
00255 #define DISCOVERY_I2Cx_CLK_DISABLE()
  __HAL_RCC_I2C1_CLK_DISABLE()
00256 #define DISCOVERY_I2Cx_FORCE_RESET()
  __HAL_RCC_I2C1_FORCE_RESET()
00257 #define DISCOVERY_I2Cx_RELEASE_RESET()
  __HAL_RCC_I2C1_RELEASE_RESET()
00258
00259 #define DISCOVERY_I2Cx_SCL_PIN
  GPIO_PIN_6                  /* PB.06 */
00260 #define DISCOVERY_I2Cx_SDA_PIN
  GPIO_PIN_7                  /* PB.07 */
00261
00262 #define DISCOVERY_I2Cx_GPIO_PORT
  GPIOB                       /* GPIOB */
00263 #define DISCOVERY_I2Cx_GPIO_CLK_ENABLE()
  __HAL_RCC_GPIOB_CLK_ENABLE()
```

```
00264 #define DISCOVERY_I2Cx_GPIO_CLK_DISABLE()
  __HAL_RCC_GPIOB_CLK_DISABLE()
00265 #define DISCOVERY_I2Cx_AF
  GPIO_AF4_I2C1
00266
00267 /* Maximum Timeout values for flags waiting
loops. These timeouts are not based
00268    on accurate values, they just guarantee t
hat the application will not remain
00269    stuck if the I2C communication is corrupt
ed.
00270    You may modify these timeout values depen
ding on CPU frequency and application
00271    conditions (interrupts routines ...). */

00272 #define I2Cx_TIMEOUT_MAX
  0x10000
00273
00274 /**
00275   * @}
00276   */
00277
00278
00279 /** @defgroup STM32F3_DISCOVERY_COMPONENT ST
M32F3-DISCOVERY COMPONENT
00280   * @{
00281   */
00282 /*##################### GYRO ###############
###########*/
00283 /* Read/Write command */
00284 #define READWRITE_CMD              ((uint8_t
)0x80)
00285 /* Multiple byte read/write command */
00286 #define MULTIPLEBYTE_CMD           ((uint8_t
)0x40)
00287 /* Dummy Byte Send by the SPI Master device
in order to generate the Clock to the Slave device
```

```
 */
00288 #define DUMMY_BYTE                ((uint8_t
)0x00)
00289
00290 /* Chip Select macro definition */
00291 #define GYRO_CS_LOW()       HAL_GPIO_WritePi
n(GYRO_CS_GPIO_PORT, GYRO_CS_PIN, GPIO_PIN_RESET)
00292 #define GYRO_CS_HIGH()      HAL_GPIO_WritePi
n(GYRO_CS_GPIO_PORT, GYRO_CS_PIN, GPIO_PIN_SET)
00293
00294 /**
00295   * @brief  GYRO SPI Interface pins
00296   */
00297 #define GYRO_CS_GPIO_PORT           GPIOE
                    /* GPIOE */
00298 #define GYRO_CS_GPIO_CLK_ENABLE()    __HAL_R
CC_GPIOE_CLK_ENABLE()
00299 #define GYRO_CS_GPIO_CLK_DISABLE()   __HAL_R
CC_GPIOE_CLK_DISABLE()
00300 #define GYRO_CS_PIN                 GPIO_PI
N_3                 /* PE.03 */
00301
00302 #define GYRO_INT_GPIO_PORT          GPIOE
                    /* GPIOE */
00303 #define GYRO_INT_GPIO_CLK_ENABLE()   __HAL_R
CC_GPIOE_CLK_ENABLE()
00304 #define GYRO_INT_GPIO_CLK_DISABLE()  __HAL_R
CC_GPIOE_CLK_DISABLE()
00305 #define GYRO_INT1_PIN               GPIO_PI
N_0                 /* PE.00 */
00306 #define GYRO_INT1_EXTI_IRQn         EXTI0_I
RQn
00307 #define GYRO_INT2_PIN               GPIO_PI
N_1                 /* PE.01 */
00308 #define GYRO_INT2_EXTI_IRQn         EXTI1_I
RQn
00309
```

```
/*##################### ACCELEROMETER ######
#####################*/
/**
  * @brief  ACCELEROMETER I2C1 Interface pins

  */
#define ACCELERO_I2C_ADDRESS            0x32

#define ACCELERO_DRDY_PIN               GPI
O_PIN_2                  /* PE.02 */
#define ACCELERO_DRDY_GPIO_PORT         GPI
OE                       /* GPIOE */
#define ACCELERO_DRDY_GPIO_CLK_ENABLE()  __H
AL_RCC_GPIOE_CLK_ENABLE()
#define ACCELERO_DRDY_GPIO_CLK_DISABLE() __H
AL_RCC_GPIOE_CLK_DISABLE()
#define ACCELERO_DRDY_EXTI_IRQn         EXT
I2_TSC_IRQn              /*TAMP_STAMP_IRQn*/

#define ACCELERO_INT_GPIO_PORT          GPI
OE                       /* GPIOE */
#define ACCELERO_INT_GPIO_CLK_ENABLE()   __H
AL_RCC_GPIOE_CLK_ENABLE()
#define ACCELERO_INT_GPIO_CLK_DISABLE()  __H
AL_RCC_GPIOE_CLK_DISABLE()
#define ACCELERO_INT1_PIN               GPI
O_PIN_4                  /* PE.04 */
#define ACCELERO_INT1_EXTI_IRQn         EXT
I4_IRQn
#define ACCELERO_INT2_PIN               GPI
O_PIN_5                  /* PE.05 */
#define ACCELERO_INT2_EXTI_IRQn         EXT
I9_5_IRQn

/**
  * @}
```

```
00332   */
00333
00334 /**
00335   * @}
00336   */
00337
00338 /** @defgroup STM32F3_DISCOVERY_Exported_Fun
ctions Exported Functions
00339   * @{
00340   */
00341 uint32_t  BSP_GetVersion(void);
00342 void      BSP_LED_Init(Led_TypeDef Led);
00343 void      BSP_LED_On(Led_TypeDef Led);
00344 void      BSP_LED_Off(Led_TypeDef Led);
00345 void      BSP_LED_Toggle(Led_TypeDef Led);
00346 void      BSP_PB_Init(Button_TypeDef Button,
ButtonMode_TypeDef ButtonMode);
00347 uint32_t  BSP_PB_GetState(Button_TypeDef But
ton);
00348
00349 /**
00350   * @}
00351   */
00352
00353 /**
00354   * @}
00355   */
00356
00357 /**
00358   * @}
00359   */
00360
00361 /**
00362   * @}
00363   */
00364
00365 #ifdef __cplusplus
```

```
}
#endif

#endif /* __STM32F3_DISCOVERY_H */

/********************** (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F3-Discovery BSP User Manual

stm32f3_discovery.c

Go to the documentation of this file.

```
/**
  ******************************************************************************
  * @file    stm32f3_discovery.c
  * @author  MCD Application Team
  * @brief   This file provides set of firmware functions to manage Leds and
  *          push-button available on STM32F3-DISCOVERY Kit from STMicroelectronics.
  ******************************************************************************
  * @attention
  *
  * <h2><center>&copy; COPYRIGHT(c) 2016 STMicroelectronics</center></h2>
  *
  * Redistribution and use in source and binary forms, with or without modification,
  * are permitted provided that the following conditions are met:
  *   1. Redistributions of source code must retain the above copyright notice,
  *      this list of conditions and the following disclaimer.
```

```
*   2. 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.
*   3. Neither the name of STMicroelectronics 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.
*
*****************************************************************************
*/
```

```
00036
00037 /* Includes ------------------------------------------------------------------------------*/
00038 #include "stm32f3_discovery.h"
00039
00040 /** @addtogroup BSP
00041   * @{
00042   */
00043
00044 /** @addtogroup STM32F3_DISCOVERY
00045   * @brief This file provides set of firmware functions to manage Leds and push-button
00046   *        available on STM32F3-Discovery Kit from STMicroelectronics.
00047   * @{
00048   */
00049 /** @addtogroup STM32F3_DISCOVERY_Common
00050   * @{
00051   */
00052
00053 /** @addtogroup STM32F3_DISCOVERY_Private_Constants
00054   * @{
00055   */
00056
00057 /**
00058  * @brief STM32F3 DISCOVERY BSP Driver version number V2.1.5
00059    */
00060 #define __STM32F3_DISCO_BSP_VERSION_MAIN   (0x02) /*!< [31:24] main version */
00061 #define __STM32F3_DISCO_BSP_VERSION_SUB1   (0x01) /*!< [23:16] sub1 version */
00062 #define __STM32F3_DISCO_BSP_VERSION_SUB2   (0x05) /*!< [15:8]  sub2 version */
00063 #define __STM32F3_DISCO_BSP_VERSION_RC     (0x00) /*!< [7:0]  release candidate */
```

```
00064 #define __STM32F3_DISCO_BSP_VERSION        (
(__STM32F3_DISCO_BSP_VERSION_MAIN << 24)\
00065
|(__STM32F3_DISCO_BSP_VERSION_SUB1 << 16)\
00066
|(__STM32F3_DISCO_BSP_VERSION_SUB2 << 8 )\
00067
|(__STM32F3_DISCO_BSP_VERSION_RC))
00068 /**
00069   * @}
00070   */
00071
00072   /** @addtogroup STM32F3_DISCOVERY_Private_
Variables
00073   * @{
00074   */
00075 /**
00076  * @brief LED variables
00077   */
00078 GPIO_TypeDef* LED_PORT[LEDn] = {LED3_GPIO_PO
RT, LED4_GPIO_PORT, LED5_GPIO_PORT, LED6_GPIO_PORT
,
00079                                   LED7_GPIO_P
ORT, LED8_GPIO_PORT, LED9_GPIO_PORT, LED10_GPIO_PO
RT};
00080
00081 const uint16_t LED_PIN[LEDn] = {LED3_PIN, LE
D4_PIN, LED5_PIN, LED6_PIN,
00082                                   LED7_PIN, L
ED8_PIN, LED9_PIN, LED10_PIN};
00083
00084 /**
00085  * @brief BUTTON variables
00086  */
00087 GPIO_TypeDef* BUTTON_PORT[BUTTONn] = {USER_B
UTTON_GPIO_PORT};
00088 const uint16_t BUTTON_PIN[BUTTONn] = {USER_B
```

```
UTTON_PIN};
00089 const uint8_t BUTTON_IRQn[BUTTONn] = {USER_B
UTTON_EXTI_IRQn};
00090
00091 /**
00092  * @brief BUS variables
00093  */
00094 #ifdef HAL_SPI_MODULE_ENABLED
00095 uint32_t SpixTimeout = SPIx_TIMEOUT_MAX;
/*<! Value of Timeout when SPI communication fails
 */
00096 static SPI_HandleTypeDef SpiHandle;
00097 #endif
00098
00099 #ifdef HAL_I2C_MODULE_ENABLED
00100 static I2C_HandleTypeDef I2cHandle;
00101 uint32_t I2cxTimeout = I2Cx_TIMEOUT_MAX;
/*<! Value of Timeout when I2C communication fails
 */
00102 #endif
00103
00104 /**
00105   * @}
00106   */
00107
00108 /** @defgroup STM32F3_DISCOVERY_BUS Bus Oper
ation functions
00109   * @{
00110   */
00111 #ifdef HAL_I2C_MODULE_ENABLED
00112 /* I2Cx bus function */
00113 static void     I2Cx_Init(void);
00114 static void     I2Cx_WriteData(uint16_t Addr
, uint8_t Reg, uint8_t Value);
00115 static uint8_t  I2Cx_ReadData(uint16_t Addr,
 uint8_t Reg);
00116 static void     I2Cx_Error (void);
```

```
00117 static void     I2Cx_MspInit(I2C_HandleTypeD
ef *hi2c);
00118 #endif
00119
00120 #ifdef HAL_SPI_MODULE_ENABLED
00121 /* SPIx bus function */
00122 static void     SPIx_Init(void);
00123 static uint8_t  SPIx_WriteRead(uint8_t byte)
;
00124 static void     SPIx_Error (void);
00125 static void     SPIx_MspInit(SPI_HandleTypeD
ef *hspi);
00126 #endif
00127
00128 #ifdef HAL_SPI_MODULE_ENABLED
00129 /* Link function for GYRO peripheral */
00130 void            GYRO_IO_Init(void);
00131 void            GYRO_IO_Write(uint8_t* pBuff
er, uint8_t WriteAddr, uint16_t NumByteToWrite);
00132 void            GYRO_IO_Read(uint8_t* pBuffe
r, uint8_t ReadAddr, uint16_t NumByteToRead);
00133 #endif
00134
00135 #ifdef HAL_I2C_MODULE_ENABLED
00136 /* Link function for COMPASS / ACCELEROMETER
 peripheral */
00137 void      COMPASSACCELERO_IO_Init(void);
00138 void      COMPASSACCELERO_IO_ITConfig(void);
00139 void      COMPASSACCELERO_IO_Write(uint16_t
DeviceAddr, uint8_t RegisterAddr, uint8_t Value);
00140 uint8_t   COMPASSACCELERO_IO_Read(uint16_t D
eviceAddr, uint8_t RegisterAddr);
00141 #endif
00142
00143 /**
00144   * @}
00145   */
```

```
00146
00147 /** @addtogroup STM32F3_DISCOVERY_Exported_F
unctions
00148   * @{
00149   */
00150
00151 /**
00152   * @brief  This method returns the STM32F3-
DISCOVERY BSP Driver revision
00153   * @retval version : 0xXYZR (8bits for each
 decimal, R for RC)
00154   */
00155 uint32_t BSP_GetVersion(void)
00156 {
00157   return __STM32F3_DISCO_BSP_VERSION;
00158 }
00159
00160 /**
00161   * @brief  Configures LED GPIO.
00162   * @param  Led Specifies the Led to be conf
igured.
00163   *   This parameter can be one of following
 parameters:
00164   *     @arg LED_RED
00165   *     @arg LED_BLUE
00166   *     @arg LED_ORANGE
00167   *     @arg LED_GREEN
00168   *     @arg LED_GREEN2
00169   *     @arg LED_ORANGE2
00170   *     @arg LED_BLUE2
00171   *     @arg LED_RED2
00172   * @retval None
00173   */
00174 void BSP_LED_Init(Led_TypeDef Led)
00175 {
00176   GPIO_InitTypeDef  GPIO_InitStruct;
00177
```

```
00178   /* Enable the GPIO_LED Clock */
00179   LEDx_GPIO_CLK_ENABLE(Led);
00180
00181   /* Configure the GPIO_LED pin */
00182   GPIO_InitStruct.Pin = LED_PIN[Led];
00183   GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP
;
00184   GPIO_InitStruct.Pull = GPIO_PULLUP;
00185   GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HI
GH;
00186
00187   HAL_GPIO_Init(LED_PORT[Led], &GPIO_InitStr
uct);
00188
00189   HAL_GPIO_WritePin(LED_PORT[Led], LED_PIN[L
ed], GPIO_PIN_RESET);
00190 }
00191
00192 /**
00193   * @brief  Turns selected LED On.
00194   * @param  Led Specifies the Led to be set
on.
00195   *   This parameter can be one of following
 parameters:
00196   *     @arg LED_RED
00197   *     @arg LED4
00198   *     @arg LED5
00199   *     @arg LED6
00200   *     @arg LED7
00201   *     @arg LED8
00202   *     @arg LED9
00203   *     @arg LED10
00204   * @retval None
00205   */
00206 void BSP_LED_On(Led_TypeDef Led)
00207 {
00208   HAL_GPIO_WritePin(LED_PORT[Led], LED_PIN[L
```

```
ed], GPIO_PIN_SET);
00209 }
00210
00211 /**
00212   * @brief  Turns selected LED Off.
00213   * @param  Led Specifies the Led to be set
off.
00214   *   This parameter can be one of following
 parameters:
00215   *     @arg LED_RED
00216   *     @arg LED_BLUE
00217   *     @arg LED_ORANGE
00218   *     @arg LED_GREEN
00219   *     @arg LED_GREEN2
00220   *     @arg LED_ORANGE2
00221   *     @arg LED_BLUE2
00222   *     @arg LED_RED2
00223   * @retval None
00224   */
00225 void BSP_LED_Off(Led_TypeDef Led)
00226 {
00227   HAL_GPIO_WritePin(LED_PORT[Led], LED_PIN[L
ed], GPIO_PIN_RESET);
00228 }
00229
00230 /**
00231   * @brief  Toggles the selected LED.
00232   * @param  Led Specifies the Led to be togg
led.
00233   *   This parameter can be one of following
 parameters:
00234   *     @arg LED_RED
00235   *     @arg LED_BLUE
00236   *     @arg LED_ORANGE
00237   *     @arg LED_GREEN
00238   *     @arg LED_GREEN2
00239   *     @arg LED_ORANGE2
```

```
00240   *     @arg LED_BLUE2
00241   *     @arg LED_RED2
00242   * @retval None
00243   */
00244 void BSP_LED_Toggle(Led_TypeDef Led)
00245 {
00246   HAL_GPIO_TogglePin(LED_PORT[Led], LED_PIN[
Led]);
00247 }
00248
00249
00250 /**
00251   * @brief  Configures Push Button GPIO and
EXTI Line.
00252   * @param  Button Specifies the Button to b
e configured.
00253   *   This parameter should be: BUTTON_USER
00254   * @param  ButtonMode Specifies Button mode.

00255   *   This parameter can be one of following
 parameters:
00256   *     @arg BUTTON_MODE_GPIO: Button will b
e used as simple IO
00257   *     @arg BUTTON_MODE_EXTI: Button will b
e connected to EXTI line with interrupt
00258   *                              generation ca
pability
00259   * @retval None
00260   */
00261 void BSP_PB_Init(Button_TypeDef Button, Butt
onMode_TypeDef ButtonMode)
00262 {
00263   GPIO_InitTypeDef GPIO_InitStruct;
00264
00265   /* Enable the BUTTON Clock */
00266   BUTTONx_GPIO_CLK_ENABLE(Button);
00267   __HAL_RCC_SYSCFG_CLK_ENABLE();
```

```
00268
00269   if (ButtonMode == BUTTON_MODE_GPIO)
00270   {
00271     /* Configure Button pin as input */
00272     GPIO_InitStruct.Pin = BUTTON_PIN[Button]
;
00273     GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
00274     GPIO_InitStruct.Pull = GPIO_PULLDOWN;
00275     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_
HIGH;
00276     HAL_GPIO_Init(BUTTON_PORT[Button], &GPIO
_InitStruct);
00277   }
00278
00279   if (ButtonMode == BUTTON_MODE_EXTI)
00280   {
00281     /* Configure Button pin as input with Ex
ternal interrupt */
00282     GPIO_InitStruct.Pin = BUTTON_PIN[Button]
;
00283     GPIO_InitStruct.Pull = GPIO_NOPULL;
00284     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_
HIGH;
00285     GPIO_InitStruct.Mode = GPIO_MODE_IT_RISI
NG;
00286     HAL_GPIO_Init(BUTTON_PORT[Button], &GPIO
_InitStruct);
00287
00288     /* Enable and set Button EXTI Interrupt
to the lowest priority */
00289     HAL_NVIC_SetPriority((IRQn_Type)(BUTTON_
IRQn[Button]), 0x0F, 0x00);
00290     HAL_NVIC_EnableIRQ((IRQn_Type)(BUTTON_IR
Qn[Button]));
00291   }
00292 }
00293
```

```
00294 /**
00295   * @brief  Returns the selected Push Button
 state.
00296   * @param  Button Specifies the Button to b
e checked.
00297   *   This parameter should be: BUTTON_USER

00298   * @retval The Button GPIO pin value.
00299   */
00300 uint32_t BSP_PB_GetState(Button_TypeDef Butt
on)
00301 {
00302   return HAL_GPIO_ReadPin(BUTTON_PORT[Button
], BUTTON_PIN[Button]);
00303 }
00304
00305 /**
00306   * @}
00307   */
00308
00309 /** @addtogroup STM32F3_DISCOVERY_BUS
00310   * @{
00311   */
00312 /*****************************************
**********************************
00313                                BUS OPERATIONS
00314 *****************************************
**********************************/
00315 #ifdef HAL_I2C_MODULE_ENABLED
00316 /****************************** I2C Routine
s*********************************/
00317
00318 /**
00319   * @brief Discovery I2Cx MSP Initialization
00320   * @param hi2c I2C handle
00321   * @retval None
00322   */
```

```
00323 static void I2Cx_MspInit(I2C_HandleTypeDef *
hi2c)
00324 {
00325
00326   GPIO_InitTypeDef GPIO_InitStructure;
00327
00328   /* Enable SCK and SDA GPIO clocks */
00329   DISCOVERY_I2Cx_GPIO_CLK_ENABLE();
00330
00331   /* I2Cx SD1 & SCK pin configuration */
00332   GPIO_InitStructure.Pin = (DISCOVERY_I2Cx_S
DA_PIN | DISCOVERY_I2Cx_SCL_PIN);
00333   GPIO_InitStructure.Mode = GPIO_MODE_AF_PP;
00334   GPIO_InitStructure.Pull = GPIO_PULLDOWN;
00335   GPIO_InitStructure.Speed = GPIO_SPEED_FREQ
_HIGH;
00336   GPIO_InitStructure.Alternate = DISCOVERY_I
2Cx_AF;
00337   HAL_GPIO_Init(DISCOVERY_I2Cx_GPIO_PORT, &G
PIO_InitStructure);
00338
00339   /* Enable the I2C clock */
00340   DISCOVERY_I2Cx_CLK_ENABLE();
00341 }
00342
00343 /**
00344   * @brief Discovery I2Cx Bus initialization
00345   * @retval None
00346   */
00347 static void I2Cx_Init(void)
00348 {
00349   if(HAL_I2C_GetState(&I2cHandle) == HAL_I2C
_STATE_RESET)
00350   {
00351     I2cHandle.Instance = DISCOVERY_I2Cx;
00352     I2cHandle.Init.OwnAddress1 =  ACCELERO_I
2C_ADDRESS;
```

```
00353     I2cHandle.Init.AddressingMode = I2C_ADDR
ESSINGMODE_7BIT;
00354     I2cHandle.Init.DualAddressMode = I2C_DUA
LADDRESS_DISABLE;
00355     I2cHandle.Init.OwnAddress2 = 0;
00356     I2cHandle.Init.GeneralCallMode = I2C_GEN
ERALCALL_DISABLE;
00357     I2cHandle.Init.NoStretchMode = I2C_NOSTR
ETCH_DISABLE;
00358
00359     /* Init the I2C */
00360     I2Cx_MspInit(&I2cHandle);
00361     HAL_I2C_Init(&I2cHandle);
00362   }
00363 }
00364
00365 /**
00366   * @brief  Write a value in a register of t
he device through BUS.
00367   * @param  Addr Device address on BUS Bus.

00368   * @param  Reg The target register address
to write
00369   * @param  Value The target register value
to be written
00370   * @retval  None
00371   */
00372 static void I2Cx_WriteData(uint16_t Addr, ui
nt8_t Reg, uint8_t Value)
00373 {
00374   HAL_StatusTypeDef status = HAL_OK;
00375
00376   status = HAL_I2C_Mem_Write(&I2cHandle, Add
r, (uint16_t)Reg, I2C_MEMADD_SIZE_8BIT, &Value, 1,
I2cxTimeout);
00377
00378   /* Check the communication status */
```

```
00379   if(status != HAL_OK)
00380   {
00381     /* Execute user timeout callback */
00382     I2Cx_Error();
00383   }
00384 }
00385
00386 /**
00387   * @brief  Read a value in a register of th
e device through BUS.
00388   * @param  Addr Device address on BUS Bus.

00389   * @param  Reg The target register address
to write
00390   * @retval Data read at register @
00391   */
00392 static uint8_t I2Cx_ReadData(uint16_t Addr,
uint8_t Reg)
00393 {
00394   HAL_StatusTypeDef status = HAL_OK;
00395   uint8_t value = 0;
00396
00397   status = HAL_I2C_Mem_Read(&I2cHandle, Addr
, Reg, I2C_MEMADD_SIZE_8BIT, &value, 1, I2cxTimeout
);
00398
00399   /* Check the communication status */
00400   if(status != HAL_OK)
00401   {
00402     /* Execute user timeout callback */
00403     I2Cx_Error();
00404
00405   }
00406   return value;
00407 }
00408
00409 /**
```

```
00410   * @brief I2C3 error treatment function
00411   * @retval None
00412   */
00413 static void I2Cx_Error (void)
00414 {
00415   /* De-initialize the I2C comunication BUS
*/
00416   HAL_I2C_DeInit(&I2cHandle);
00417
00418   /* Re- Initiaize the I2C comunication BUS
*/
00419   I2Cx_Init();
00420 }
00421 #endif
00422
00423
00424 #ifdef HAL_SPI_MODULE_ENABLED
00425 /****************************** SPI Routine
s*********************************/
00426 /**
00427   * @brief SPIx Bus initialization
00428   * @retval None
00429   */
00430 static void SPIx_Init(void)
00431 {
00432   if(HAL_SPI_GetState(&SpiHandle) == HAL_SPI
_STATE_RESET)
00433   {
00434     /* SPI Config */
00435     SpiHandle.Instance = DISCOVERY_SPIx;
00436     /* SPI baudrate is set to 5.6 MHz (PCLK2
/SPI_BaudRatePrescaler = 90/16 = 5.625 MHz)
00437       to verify these constraints:
00438       ILI9341 LCD SPI interface max baudrate
 is 10MHz for write and 6.66MHz for read
00439       l3gd20 SPI interface max baudrate is 1
0MHz for write/read
```

```
00440         PCLK2 frequency is set to 90 MHz
00441    */
00442      SpiHandle.Init.BaudRatePrescaler = SPI_B
AUDRATEPRESCALER_16;
00443      SpiHandle.Init.Direction = SPI_DIRECTION
_2LINES;
00444      SpiHandle.Init.CLKPhase = SPI_PHASE_1EDG
E;
00445      SpiHandle.Init.CLKPolarity = SPI_POLARIT
Y_LOW;
00446      SpiHandle.Init.CRCCalculation = SPI_CRCC
ALCULATION_DISABLE;
00447      SpiHandle.Init.CRCPolynomial = 7;
00448      SpiHandle.Init.DataSize = SPI_DATASIZE_8
BIT;
00449      SpiHandle.Init.FirstBit = SPI_FIRSTBIT_M
SB;
00450      SpiHandle.Init.NSS = SPI_NSS_SOFT;
00451      SpiHandle.Init.TIMode = SPI_TIMODE_DISAB
LE;
00452      SpiHandle.Init.Mode = SPI_MODE_MASTER;
00453
00454      SPIx_MspInit(&SpiHandle);
00455      HAL_SPI_Init(&SpiHandle);
00456    }
00457 }
00458
00459 /**
00460   * @brief  Sends a Byte through the SPI int
erface and return the Byte received
00461   *         from the SPI bus.
00462   * @param  Byte Byte send.
00463   * @retval The received byte value
00464   */
00465 static uint8_t SPIx_WriteRead(uint8_t Byte)
00466 {
00467
```

```
00468   uint8_t receivedbyte = 0;
00469
00470   /* Send a Byte through the SPI peripheral
*/
00471   /* Read byte from the SPI bus */
00472   if(HAL_SPI_TransmitReceive(&SpiHandle, (ui
nt8_t*) &Byte, (uint8_t*) &receivedbyte, 1, SpixTi
meout) != HAL_OK)
00473   {
00474     SPIx_Error();
00475   }
00476
00477   return receivedbyte;
00478 }
00479
00480
00481 /**
00482   * @brief SPIx error treatment function
00483   * @retval None
00484   */
00485 static void SPIx_Error (void)
00486 {
00487   /* De-initialize the SPI comunication BUS
*/
00488   HAL_SPI_DeInit(&SpiHandle);
00489
00490   /* Re- Initiaize the SPI comunication BUS
*/
00491   SPIx_Init();
00492 }
00493
00494
00495 /**
00496   * @brief SPI MSP Init
00497   * @param hspi SPI handle
00498   * @retval None
00499   */
```

```
00500 static void SPIx_MspInit(SPI_HandleTypeDef *
hspi)
00501 {
00502   GPIO_InitTypeDef   GPIO_InitStructure;
00503
00504   /* Enable SPI1 clock  */
00505   DISCOVERY_SPIx_CLK_ENABLE();
00506
00507   /* enable SPI1 gpio clock */
00508   DISCOVERY_SPIx_GPIO_CLK_ENABLE();
00509
00510   /* configure SPI1 SCK, MOSI and MISO */
00511   GPIO_InitStructure.Pin = (DISCOVERY_SPIx_S
CK_PIN | DISCOVERY_SPIx_MOSI_PIN | DISCOVERY_SPIx_
MISO_PIN);
00512   GPIO_InitStructure.Mode = GPIO_MODE_AF_PP;
00513   GPIO_InitStructure.Pull  = GPIO_NOPULL; /*
 or GPIO_PULLDOWN */
00514   GPIO_InitStructure.Speed = GPIO_SPEED_FREQ
_HIGH;
00515   GPIO_InitStructure.Alternate = DISCOVERY_S
PIx_AF;
00516   HAL_GPIO_Init(DISCOVERY_SPIx_GPIO_PORT, &G
PIO_InitStructure);
00517 }
00518 /**
00519   * @}
00520   */
00521
00522 /** @defgroup STM32F3_DISCOVERY_LINK_OPERATI
ONS Link Operation functions
00523   * @{
00524   */
00525
00526 /*****************************************
*********************************
00527                                 LINK OPERATIONS
```

```
00528 ****************************************
*********************************/
00529
00530 /******************************** LINK GYRO
SCOPE ****************************/
00531 /**
00532   * @brief  Configures the GYROSCOPE SPI int
erface.
00533   * @retval None
00534   */
00535 void GYRO_IO_Init(void)
00536 {
00537     GPIO_InitTypeDef GPIO_InitStructure;
00538
00539   /* Configure the Gyroscope Control pins --
----------------------------------------*/
00540   /* Enable CS GPIO clock and  Configure GPI
O PIN for Gyroscope Chip select */
00541   GYRO_CS_GPIO_CLK_ENABLE();
00542   GPIO_InitStructure.Pin = GYRO_CS_PIN;
00543   GPIO_InitStructure.Mode = GPIO_MODE_OUTPUT
_PP;
00544   GPIO_InitStructure.Pull  = GPIO_NOPULL;
00545   GPIO_InitStructure.Speed = GPIO_SPEED_FREQ
_HIGH;
00546   HAL_GPIO_Init(GYRO_CS_GPIO_PORT, &GPIO_Ini
tStructure);
00547
00548   /* Deselect : Chip Select high */
00549   GYRO_CS_HIGH();
00550
00551   /* Enable INT1, INT2 GPIO clock and Config
ure GPIO PINs to detect Interrupts */
00552   GYRO_INT_GPIO_CLK_ENABLE();
00553   GPIO_InitStructure.Pin = GYRO_INT1_PIN | G
YRO_INT2_PIN;
00554   GPIO_InitStructure.Mode = GPIO_MODE_INPUT;
```

```
00555   GPIO_InitStructure.Speed = GPIO_SPEED_FREQ
_HIGH;
00556   GPIO_InitStructure.Pull= GPIO_NOPULL;
00557   HAL_GPIO_Init(GYRO_INT_GPIO_PORT, &GPIO_In
itStructure);
00558
00559   SPIx_Init();
00560 }
00561
00562 /**
00563   * @brief  Writes one byte to the GYROSCOPE.

00564   * @param  pBuffer pointer to the buffer  c
ontaining the data to be written to the GYROSCOPE.
00565   * @param  WriteAddr GYROSCOPE's internal a
ddress to write to.
00566   * @param  NumByteToWrite Number of bytes t
o write.
00567   * @retval None
00568   */
00569 void GYRO_IO_Write(uint8_t* pBuffer, uint8_t
 WriteAddr, uint16_t NumByteToWrite)
00570 {
00571   /* Configure the MS bit:
00572        - When 0, the address will remain unc
hanged in multiple read/write commands.
00573        - When 1, the address will be auto in
cremented in multiple read/write commands.
00574   */
00575   if(NumByteToWrite > 0x01)
00576   {
00577     WriteAddr |= (uint8_t)MULTIPLEBYTE_CMD;
00578   }
00579   /* Set chip select Low at the start of the
 transmission */
00580   GYRO_CS_LOW();
00581
```

```
  /* Send the Address of the indexed register */
  SPIx_WriteRead(WriteAddr);

  /* Send the data that will be written into the device (MSB First) */
  while(NumByteToWrite >= 0x01)
  {
    SPIx_WriteRead(*pBuffer);
    NumByteToWrite--;
    pBuffer++;
  }

  /* Set chip select High at the end of the transmission */
  GYRO_CS_HIGH();
}

/**
  * @brief  Reads a block of data from the GYROSCOPE.
  * @param  pBuffer pointer to the buffer that receives the data read from the GYROSCOPE.
  * @param  ReadAddr GYROSCOPE's internal address to read from.
  * @param  NumByteToRead number of bytes to read from the GYROSCOPE.
  * @retval None
  */
void GYRO_IO_Read(uint8_t* pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead)
{
  if(NumByteToRead > 0x01)
  {
    ReadAddr |= (uint8_t)(READWRITE_CMD | MULTIPLEBYTE_CMD);
  }
```

```
  else
  {
    ReadAddr |= (uint8_t)READWRITE_CMD;
  }
  /* Set chip select Low at the start of the transmission */
  GYRO_CS_LOW();

  /* Send the Address of the indexed register */
  SPIx_WriteRead(ReadAddr);

  /* Receive the data that will be read from the device (MSB First) */
  while(NumByteToRead > 0x00)
  {
    /* Send dummy byte (0x00) to generate the SPI clock to GYROSCOPE (Slave device) */
    *pBuffer = SPIx_WriteRead(DUMMY_BYTE);
    NumByteToRead--;
    pBuffer++;
  }

  /* Set chip select High at the end of the transmission */
  GYRO_CS_HIGH();
}
#endif /* HAL_SPI_MODULE_ENABLED */

#ifdef HAL_I2C_MODULE_ENABLED
/****************************** LINK ACCELEROMETER ******************************/
/**
  * @brief  Configures COMPASS / ACCELEROMETER I2C interface.
  * @retval None
  */
```

```
00640 void COMPASSACCELERO_IO_Init(void)
00641 {
00642   GPIO_InitTypeDef GPIO_InitStructure;
00643
00644   /* Enable DRDY clock */
00645   ACCELERO_DRDY_GPIO_CLK_ENABLE();
00646
00647   /* Enable INT1 & INT2 GPIO clock */
00648   ACCELERO_INT_GPIO_CLK_ENABLE();
00649
00650   /* Mems DRDY pin configuration */
00651   GPIO_InitStructure.Pin = ACCELERO_DRDY_PIN
;
00652   GPIO_InitStructure.Mode = GPIO_MODE_INPUT;
00653   GPIO_InitStructure.Pull  = GPIO_NOPULL;
00654   GPIO_InitStructure.Speed = GPIO_SPEED_FREQ
_HIGH;
00655   HAL_GPIO_Init(ACCELERO_DRDY_GPIO_PORT, &GP
IO_InitStructure);
00656
00657   /* Enable and set Button EXTI Interrupt to
 the lowest priority */
00658   HAL_NVIC_SetPriority(ACCELERO_DRDY_EXTI_IR
Qn, 0x0F, 0x00);
00659   HAL_NVIC_EnableIRQ(ACCELERO_DRDY_EXTI_IRQn
);
00660
00661   /* Configure GPIO PINs to detect Interrupt
s */
00662   GPIO_InitStructure.Pin = ACCELERO_INT1_PIN
 | ACCELERO_INT2_PIN;
00663   GPIO_InitStructure.Mode = GPIO_MODE_INPUT;
00664   GPIO_InitStructure.Speed = GPIO_SPEED_FREQ
_HIGH;
00665   GPIO_InitStructure.Pull  = GPIO_NOPULL;
00666   HAL_GPIO_Init(ACCELERO_INT_GPIO_PORT, &GPI
O_InitStructure);
```

```
00667
00668   I2Cx_Init();
00669 }
00670
00671 /**
00672   * @brief  Configures COMPASS / ACCELERO cl
ick IT
00673   * @retval None
00674   */
00675 void COMPASSACCELERO_IO_ITConfig(void)
00676 {
00677   GPIO_InitTypeDef GPIO_InitStructure;
00678
00679   /* Enable INT1 & INT2 GPIO clock */
00680   ACCELERO_INT_GPIO_CLK_ENABLE();
00681
00682   /* Configure GPIO PINs to detect Interrupt
s */
00683   GPIO_InitStructure.Pin = ACCELERO_INT1_PIN
 | ACCELERO_INT2_PIN;
00684   GPIO_InitStructure.Mode = GPIO_MODE_IT_RIS
ING;
00685   GPIO_InitStructure.Speed = GPIO_SPEED_FREQ
_HIGH;
00686   GPIO_InitStructure.Pull  = GPIO_NOPULL;
00687   HAL_GPIO_Init(ACCELERO_INT_GPIO_PORT, &GPI
O_InitStructure);
00688
00689   /* Enable and set Button EXTI Interrupt to
 the lowest priority */
00690   HAL_NVIC_SetPriority(ACCELERO_INT1_EXTI_IR
Qn, 0x0F, 0x00);
00691   HAL_NVIC_EnableIRQ(ACCELERO_INT1_EXTI_IRQn
);
00692
00693 }
00694
```

```
00695 /**
00696   * @brief  Writes one byte to the COMPASS /
 ACCELEROMETER.
00697   * @param  DeviceAddr specifies the slave a
ddress to be programmed.
00698   * @param  RegisterAddr specifies the COMPA
SS / ACCELEROMETER register to be written.
00699   * @param  Value Data to be written
00700   * @retval   None
00701  */
00702 void COMPASSACCELERO_IO_Write(uint16_t Devic
eAddr, uint8_t RegisterAddr, uint8_t Value)
00703 {
00704   /* call I2Cx Read data bus function */
00705   I2Cx_WriteData(DeviceAddr, RegisterAddr, V
alue);
00706 }
00707
00708 /**
00709   * @brief  Reads a block of data from the C
OMPASS / ACCELEROMETER.
00710   * @param  DeviceAddr specifies the slave a
ddress to be programmed(ACC_I2C_ADDRESS or MAG_I2C
_ADDRESS).
00711   * @param  RegisterAddr specifies the COMPA
SS / ACCELEROMETER internal address register to re
ad from
00712   * @retval ACCELEROMETER register value
00713   */
00714 uint8_t COMPASSACCELERO_IO_Read(uint16_t Dev
iceAddr, uint8_t RegisterAddr)
00715 {
00716   /* call I2Cx Read data bus function */
00717   return I2Cx_ReadData(DeviceAddr, RegisterA
ddr);
00718 }
00719 #endif /* HAL_I2C_MODULE_ENABLED */
```

```
/**
  * @}
  */

/**
  * @}
  */

/**
  * @}
  */

/**
  * @}
  */

/****************** (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F3-Discovery BSP User Manual

stm32f3_discovery_accelerometer.h

Go to the documentation of this file.

```
/**
  ******************************************************************************
  * @file    stm32f3_discovery_accelerometer.h
  * @author  MCD Application Team
  * @brief   This file contains definitions for stm32f3_discovery_accelerometer.c
  *          firmware driver.
  ******************************************************************************
  * @attention
  *
  * <h2><center>&copy; COPYRIGHT(c) 2016 STMicroelectronics</center></h2>
  *
  * Redistribution and use in source and binary forms, with or without modification,
  * are permitted provided that the following conditions are met:
  *   1. Redistributions of source code must retain the above copyright notice,
  *      this list of conditions and the following disclaimer.
```

```
*   2. 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.
*   3. Neither the name of STMicroelectronics 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.
*
*****************************************************************************
*/
```

```
00036
00037
00038 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00039 #ifndef __STM32F3_DISCOVERY_ACCELERO_H
00040 #define __STM32F3_DISCOVERY_ACCELERO_H
00041
00042 #ifdef __cplusplus
00043  extern "C" {
00044 #endif
00045
00046 /* Includes ---------------------------------
-----------------------------------*/
00047 #include "stm32f3_discovery.h"
00048 /* Include Gyroscope component driver */
00049 #include "../Components/lsm303dlhc/lsm303dlh
c.h"
00050
00051 /** @addtogroup BSP
00052   * @{
00053   */
00054
00055 /** @addtogroup STM32F3_DISCOVERY
00056   * @{
00057   */
00058
00059 /** @addtogroup STM32F3_DISCOVERY_ACCELEROME
TER
00060   * @{
00061   */
00062
00063 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETE
R_Exported_Types Exported Types
00064   * @{
00065   */
00066
00067 /**
```

```
00068   * @}
00069   */
00070
00071 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETE
R_Exported_Constants Exported Constants
00072   * @{
00073   */
00074 typedef enum
00075 {
00076   ACCELERO_OK = 0,
00077   ACCELERO_ERROR = 1,
00078   ACCELERO_TIMEOUT = 2
00079 }
00080 ACCELERO_StatusTypeDef;
00081
00082 /**
00083   * @}
00084   */
00085
00086 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETE
R_Exported_Macros Exported Macros
00087   * @{
00088   */
00089
00090 /**
00091   * @}
00092   */
00093
00094 /* Exported functions ----------------------
-----------------------------------*/
00095 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETE
R_Exported_Functions Exported Functions
00096   * @{
00097   */
00098 /* Acc functions */
00099 uint8_t   BSP_ACCELERO_Init(void);
00100 void      BSP_ACCELERO_Reset(void);
```

```
void      BSP_ACCELERO_GetXYZ(int16_t *pDataXYZ);

/**
  * @}
  */

/**
  * @}
  */

/**
  * @}
  */

/**
  * @}
  */
#ifdef __cplusplus
}
#endif

#endif /* __STM32F3_DISCOVERY_ACCELERO_H */
/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F3-Discovery BSP User Manual

stm32f3_discovery_accelerometer.c

Go to the documentation of this file.

```
/**
  ******************************************************************************
  * @file    stm32f3_discovery_accelerometer.c
  * @author  MCD Application Team
  * @brief   This file provides a set of functions needed to manage the ACCELEROMETER
  *          MEMS available on STM32F3-Discovery Kit.
  ******************************************************************************
  * @attention
  *
  * <h2><center>&copy; COPYRIGHT(c) 2016 STMicroelectronics</center></h2>
  *
  * Redistribution and use in source and binary forms, with or without modification,
  * are permitted provided that the following conditions are met:
  *   1. Redistributions of source code must retain the above copyright notice,
  *      this list of conditions and the fol
```

```
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ******************************************
***********************************
```

```
00035   */
00036
00037 /* Includes ------------------------------------------------------------------------------------*/
00038 #include "stm32f3_discovery_accelerometer.h"
00039
00040 /** @addtogroup BSP
00041   * @{
00042   */
00043
00044 /** @addtogroup STM32F3_DISCOVERY
00045   * @{
00046   */
00047
00048 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETER STM32F3-DISCOVERY ACCELEROMETER
00049   * @{
00050   */
00051
00052 /* Private typedef -----------------------------------------------------------------------------*/
00053 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETER_Private_Types Private Types
00054   * @{
00055   */
00056 /**
00057   * @}
00058   */
00059
00060 /* Private defines -----------------------------------------------------------------------------*/
00061 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETER_Private_Constants Private Constants
00062   * @{
00063   */
00064 /**
00065   * @}
```

```
00066   */
00067
00068 /* Private macros ---------------------------
-----------------------------------*/
00069 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETE
R_Private_Macros Private Macros
00070   * @{
00071   */
00072 /**
00073   * @}
00074   */
00075
00076 /* Private variables -----------------------
-----------------------------------*/
00077 /** @defgroup STM32F3_DISCOVERY_ACCELEROMETE
R_Private_Variables Private Variables
00078   * @{
00079   */
00080 static ACCELERO_DrvTypeDef *AccelerometerDrv
;
00081
00082
00083 /**
00084   * @}
00085   */
00086
00087 /* Private function prototypes -------------
-----------------------------------*/
00088 /** @addtogroup STM32F3_DISCOVERY_ACCELEROME
TER_Private_FunctionPrototypes Private Functions
00089   * @{
00090   */
00091 /**
00092   * @}
00093   */
00094
00095 /* Exported functions ----------------------
```

```
-------------------------------------*/
00096 /** @addtogroup STM32F3_DISCOVERY_ACCELEROME
TER_Exported_Functions
00097   * @{
00098   */
00099 uint8_t BSP_ACCELERO_Init(void)
00100 {
00101   uint8_t ret = ACCELERO_ERROR;
00102   uint16_t ctrl = 0x0000;
00103   ACCELERO_InitTypeDef LSM303DLHC_InitStruct
ure;
00104   ACCELERO_FilterConfigTypeDef LSM303DLHC_Fi
lterStructure;
00105
00106   if(Lsm303dlhcDrv.ReadID() == I_AM_LMS303DL
HC)
00107   {
00108     /* Initialize the gyroscope driver struc
ture */
00109     AccelerometerDrv = &Lsm303dlhcDrv;
00110
00111   /* MEMS configuration --------------------
-----------------------------------*/
00112    /* Fill the accelerometer structure */
00113     LSM303DLHC_InitStructure.Power_Mode = LS
M303DLHC_NORMAL_MODE;
00114     LSM303DLHC_InitStructure.AccOutput_DataR
ate = LSM303DLHC_ODR_50_HZ;
00115     LSM303DLHC_InitStructure.Axes_Enable= LS
M303DLHC_AXES_ENABLE;
00116     LSM303DLHC_InitStructure.AccFull_Scale =
 LSM303DLHC_FULLSCALE_2G;
00117     LSM303DLHC_InitStructure.BlockData_Updat
e = LSM303DLHC_BlockUpdate_Continous;
00118     LSM303DLHC_InitStructure.Endianness=LSM3
03DLHC_BLE_LSB;
00119     LSM303DLHC_InitStructure.High_Resolution
```

```
=LSM303DLHC_HR_ENABLE;
00120
00121     /* Configure MEMS: data rate, power mode
, full scale and axes */
00122     ctrl |= (LSM303DLHC_InitStructure.Power_
Mode | LSM303DLHC_InitStructure.AccOutput_DataRate
 | \
00123                          LSM303DLHC_InitStruct
ure.Axes_Enable);
00124
00125     ctrl |= ((LSM303DLHC_InitStructure.Block
Data_Update | LSM303DLHC_InitStructure.Endianness
| \
00126                         LSM303DLHC_InitStructu
re.AccFull_Scale | LSM303DLHC_InitStructure.High_R
esolution) << 8);
00127
00128   /* Configure the accelerometer main parame
ters */
00129     AccelerometerDrv->Init(ctrl);
00130
00131   /* Fill the accelerometer LPF structure */
00132     LSM303DLHC_FilterStructure.HighPassFilte
r_Mode_Selection =LSM303DLHC_HPM_NORMAL_MODE;
00133     LSM303DLHC_FilterStructure.HighPassFilte
r_CutOff_Frequency = LSM303DLHC_HPFCF_16;
00134     LSM303DLHC_FilterStructure.HighPassFilte
r_AOI1 = LSM303DLHC_HPF_AOI1_DISABLE;
00135     LSM303DLHC_FilterStructure.HighPassFilte
r_AOI2 = LSM303DLHC_HPF_AOI2_DISABLE;
00136
00137     /* Configure MEMS: mode, cutoff frquency
, Filter status, Click, AOI1 and AOI2 */
00138     ctrl = (uint8_t) (LSM303DLHC_FilterStruc
ture.HighPassFilter_Mode_Selection |\
00139                         LSM303DLHC_FilterStruc
ture.HighPassFilter_CutOff_Frequency|\
```

```
                                LSM303DLHC_FilterStructure.HighPassFilter_AOI1|\
                                LSM303DLHC_FilterStructure.HighPassFilter_AOI2);

  /* Configure the accelerometer LPF main parameters */
    AccelerometerDrv->FilterConfig(ctrl);

    ret = ACCELERO_OK;
  }
  else
  {
    ret = ACCELERO_ERROR;
  }

  return ret;
}

/**
  * @brief  Reboot memory content of ACCELEROMETER
  * @retval None
  */
void BSP_ACCELERO_Reset(void)
  {
  if(AccelerometerDrv->Reset != NULL)
{
    AccelerometerDrv->Reset();
}
}

/**
  * @brief  Get XYZ acceleration
  * @param pDataXYZ Pointeur on 3 angular accelerations
  *                 pDataXYZ[0] = X axis, pD
```

```
ataXYZ[1] = Y axis, pDataXYZ[2] = Z axis
00172 * @retval None
00173 */
00174 void BSP_ACCELERO_GetXYZ(int16_t *pDataXYZ)
00175 {
00176   if(AccelerometerDrv->GetXYZ!= NULL)
00177   {
00178     AccelerometerDrv->GetXYZ(pDataXYZ);
00179   }
00180 }
00181
00182
00183 /**
00184   * @}
00185   */
00186
00187 /**
00188   * @}
00189   */
00190
00191 /**
00192   * @}
00193   */
00194
00195 /**
00196   * @}
00197   */
00198
00199
00200 /********************** (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32F3-Discovery BSP User Manual

stm32f3_discovery_gyroscope.h

Go to the documentation of this file.

```
/**
  ******************************************************************************
  * @file    stm32f3_discovery_gyroscope.h
  * @author  MCD Application Team
  * @brief   This file contains definitions for stm32f3_discovery_gyroscope.c
  *          firmware driver.
  ******************************************************************************
  * @attention
  *
  * <h2><center>&copy; COPYRIGHT(c) 2016 STMicroelectronics</center></h2>
  *
  * Redistribution and use in source and binary forms, with or without modification,
  * are permitted provided that the following conditions are met:
  *   1. Redistributions of source code must retain the above copyright notice,
  *      this list of conditions and the following disclaimer.
  *   2. Redistributions in binary form must
```

```
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ******************************************
**********************************
00035   */
00036
```

```
00037
00038 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00039 #ifndef __STM32F3_DISCOVERY_GYRO_H
00040 #define __STM32F3_DISCOVERY_GYRO_H
00041
00042 #ifdef __cplusplus
00043  extern "C" {
00044 #endif
00045
00046 /* Includes ---------------------------------
-----------------------------------*/
00047 #include "stm32f3_discovery.h"
00048 /* Include Gyroscope component driver */
00049 #include "../Components/l3gd20/l3gd20.h"
00050
00051 /** @addtogroup BSP
00052   * @{
00053   */
00054
00055 /** @addtogroup STM32F3_DISCOVERY
00056   * @{
00057   */
00058
00059 /** @addtogroup STM32F3_DISCOVERY_GYROSCOPE
00060   * @{
00061   */
00062
00063 /** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Ex
ported_Types Exported Types
00064   * @{
00065   */
00066 typedef enum
00067 {
00068   GYRO_OK = 0,
00069   GYRO_ERROR = 1,
00070   GYRO_TIMEOUT = 2
```

```
}
GYRO_StatusTypeDef;

/**
  * @}
  */

/** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Exported_Constants Exported Constants
  * @{
  */

/**
  * @}
  */

/** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Exported_Macros Exported Macros
  * @{
  */

/**
  * @}
  */

/** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Exported_Functions  Exported Functions
  * @{
  */
/* Sensor Configuration Functions */
uint8_t BSP_GYRO_Init(void);
void BSP_GYRO_Reset(void);
uint8_t BSP_GYRO_ReadID(void);
void    BSP_GYRO_ITConfig(GYRO_InterruptConfigTypeDef *pIntConfigStruct);
void BSP_GYRO_EnableIT(uint8_t IntPin);
void BSP_GYRO_DisableIT(uint8_t IntPin);
```

```
void BSP_GYRO_GetXYZ(float* pfData);

/**
  * @}
  */

/**
  * @}
  */

/**
  * @}
  */

/**
  * @}
  */

#ifdef __cplusplus
}
#endif

#endif /* __STM32F3_DISCOVERY_GYRO_H */

/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F3-Discovery BSP User Manual

stm32f3_discovery_gyroscope.c

Go to the documentation of this file.

```
/**
  ******************************************************************************
  * @file    stm32f3_discovery_gyroscope.c
  * @author  MCD Application Team
  * @brief   This file provides a set of functions needed to manage the l3gd20
  *          MEMS accelerometer available on STM32F3-Discovery Kit.
  ******************************************************************************
  * @attention
  *
  * <h2><center>&copy; COPYRIGHT(c) 2016 STMicroelectronics</center></h2>
  *
  * Redistribution and use in source and binary forms, with or without modification,
  * are permitted provided that the following conditions are met:
  *   1. Redistributions of source code must retain the above copyright notice,
  *      this list of conditions and the following disclaimer.
```

```
*   2. 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.
*   3. Neither the name of STMicroelectronics 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.
*
*****************************************************************************
*/
```

```
/* Includes ------------------------------------------------------------------------------------*/
#include "stm32f3_discovery_gyroscope.h"

/** @addtogroup BSP
  * @{
  */

/** @addtogroup STM32F3_DISCOVERY
  * @{
  */

/** @defgroup STM32F3_DISCOVERY_GYROSCOPE STM32F3-DISCOVERY GYROSCOPE
  * @{
  */

/* Private typedef -----------------------------------------------------------------------------*/
/** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Private_Types Private Types
  * @{
  */
/**
  * @}
  */

/* Private defines -----------------------------------------------------------------------------*/
/** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Private_Constants Private Constants
  * @{
  */
/**
  * @}
  */

```

```
/* Private macros ------------------------------------------------------------*/
/** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Private_Macros Private Macros
  * @{
  */
/**
  * @}
  */

/* Private variables ---------------------------------------------------------*/
/** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Private_Variables Private Variables
  * @{
  */
static GYRO_DrvTypeDef *GyroscopeDrv;

/**
  * @}
  */

/* Private function prototypes -----------------------------------------------*/
/** @defgroup STM32F3_DISCOVERY_GYROSCOPE_Private_FunctionPrototypes Private Functions
  * @{
  */
/**
  * @}
  */

/* Exported functions --------------------------------------------------------*/
/** @addtogroup STM32F3_DISCOVERY_GYROSCOPE_Exported_Functions
  * @{
```

```
00096   */
00097
00098 /**
00099   * @brief  Set GYROSCOPE Initialization.
00100   * @retval GYRO_OK if no problem during ini
tialization
00101   */
00102 uint8_t BSP_GYRO_Init(void)
00103 {
00104   uint8_t ret = GYRO_ERROR;
00105   uint16_t ctrl = 0x0000;
00106   GYRO_InitTypeDef L3GD20_InitStructure;
00107   GYRO_FilterConfigTypeDef L3GD20_FilterStru
cture;
00108
00109   if((L3gd20Drv.ReadID() == I_AM_L3GD20) ||
(L3gd20Drv.ReadID() == I_AM_L3GD20_TR))
00110   {
00111     /* Initialize the gyroscope driver struc
ture */
00112     GyroscopeDrv = &L3gd20Drv;
00113
00114     /* Configure Mems : data rate, power mod
e, full scale and axes */
00115     L3GD20_InitStructure.Power_Mode = L3GD20
_MODE_ACTIVE;
00116     L3GD20_InitStructure.Output_DataRate = L
3GD20_OUTPUT_DATARATE_1;
00117     L3GD20_InitStructure.Axes_Enable = L3GD2
0_AXES_ENABLE;
00118     L3GD20_InitStructure.Band_Width = L3GD20
_BANDWIDTH_4;
00119     L3GD20_InitStructure.BlockData_Update =
L3GD20_BlockDataUpdate_Continous;
00120     L3GD20_InitStructure.Endianness = L3GD20
_BLE_LSB;
00121     L3GD20_InitStructure.Full_Scale = L3GD20
```

```
_FULLSCALE_500;

    /* Configure MEMS: data rate, power mode, full scale and axes */
    ctrl = (uint16_t) (L3GD20_InitStructure.Power_Mode | L3GD20_InitStructure.Output_DataRate | \
                       L3GD20_InitStructure.Axes_Enable | L3GD20_InitStructure.Band_Width);

    ctrl |= (uint16_t) ((L3GD20_InitStructure.BlockData_Update | L3GD20_InitStructure.Endianness | \
                         L3GD20_InitStructure.Full_Scale) << 8);

    /* L3gd20 Init */
    GyroscopeDrv->Init(ctrl);

    L3GD20_FilterStructure.HighPassFilter_Mode_Selection =L3GD20_HPM_NORMAL_MODE_RES;
    L3GD20_FilterStructure.HighPassFilter_CutOff_Frequency = L3GD20_HPFCF_0;

    ctrl = (uint8_t) ((L3GD20_FilterStructure.HighPassFilter_Mode_Selection |\
                       L3GD20_FilterStructure.HighPassFilter_CutOff_Frequency));

    GyroscopeDrv->FilterConfig(ctrl) ;

    GyroscopeDrv->FilterCmd(L3GD20_HIGHPASSFILTER_ENABLE);

    ret = GYRO_OK;
  }
```

```
  else
  {
    ret = GYRO_ERROR;
  }

  return ret;
}

/**
  * @brief  Read ID of Gyroscope component
  * @retval ID
  */
uint8_t BSP_GYRO_ReadID(void)
{
  uint8_t id = 0x00;

  if(GyroscopeDrv->ReadID != NULL)
  {
    id = GyroscopeDrv->ReadID();
  }
  return id;
}

/**
  * @brief  Reboot memory content of GYROSCOPE
  * @retval None
  */
void BSP_GYRO_Reset(void)
{
  if(GyroscopeDrv->Reset != NULL)
  {
    GyroscopeDrv->Reset();
  }
}


```

```
00181 /**
00182   * @brief  Configure INT1 interrupt
00183   * @param  pIntConfig pointer to a L3GD20_I
nterruptConfig_TypeDef
00184   *         structure that contains the conf
iguration setting for the L3GD20 Interrupt.
00185   * @retval None
00186   */
00187 void BSP_GYRO_ITConfig(GYRO_InterruptConfigT
ypeDef *pIntConfig)
00188 {
00189 uint16_t interruptconfig = 0x0000;
00190
00191   if(GyroscopeDrv->ConfigIT != NULL)
00192   {
00193     /* Configure latch Interrupt request and
 axe interrupts */
00194     interruptconfig |= ((uint8_t)(pIntConfig
->Latch_Request| \
00195                                   pIntConfig
->Interrupt_Axes) << 8);
00196
00197     interruptconfig |= (uint8_t)(pIntConfig-
>Interrupt_ActiveEdge);
00198
00199         GyroscopeDrv->ConfigIT(interruptconf
ig);
00200   }
00201 }
00202
00203 /**
00204   * @brief  Enable INT1 or INT2 interrupt
00205   * @param  IntPin Interrupt pin
00206   *      This parameter can be:
00207   *        @arg L3GD20_INT1
00208   *        @arg L3GD20_INT2
00209   * @retval None
```

```
00210   */
00211 void BSP_GYRO_EnableIT(uint8_t IntPin)
00212 {
00213   if(GyroscopeDrv->EnableIT != NULL)
00214   {
00215         GyroscopeDrv->EnableIT(IntPin);
00216   }
00217 }
00218
00219 /**
00220   * @brief  Disable INT1 or INT2 interrupt
00221   * @param  IntPin Interrupt pin
00222   *      This parameter can be:
00223   *        @arg L3GD20_INT1
00224   *        @arg L3GD20_INT2
00225   * @retval None
00226   */
00227 void BSP_GYRO_DisableIT(uint8_t IntPin)
00228 {
00229   if(GyroscopeDrv->DisableIT != NULL)
00230   {
00231     GyroscopeDrv->DisableIT(IntPin);
00232   }
00233 }
00234
00235 /**
00236   * @brief  Get XYZ angular acceleration
00237   * @param pfData pointer on floating array

00238   * @retval None
00239   */
00240 void BSP_GYRO_GetXYZ(float* pfData)
00241 {
00242   if(GyroscopeDrv->GetXYZ!= NULL)
00243   {
00244         GyroscopeDrv->GetXYZ(pfData);
00245   }
```

```
}

/**
  * @}
  */

/**
  * @}
  */

/**
  * @}
  */

/**
  * @}
  */

/********************** (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F3-Discovery BSP User Manual

BSP

Modules

STM32F3-DISCOVERY

This file provides set of firmware functions to manage Leds and push-button available on STM32F3-Discovery Kit from STMicroelectronics.

STM32F3-Discovery BSP User Manual

STM32F3-DISCOVERY

BSP

This file provides set of firmware functions to manage Leds and push-button available on STM32F3-Discovery Kit from STMicroelectronics.
More...

Modules

STM32F3-DISCOVERY Common
STM32F3-DISCOVERY ACCELEROMETER
STM32F3-DISCOVERY GYROSCOPE

Detailed Description

This file provides set of firmware functions to manage Leds and push-button available on STM32F3-Discovery Kit from STMicroelectronics.

STM32F3-Discovery BSP User Manual

STM32F3-DISCOVERY Common

STM32F3-DISCOVERY

Modules

Bus Operation functions
Link Operation functions
Private Constants
Private Variables
Exported Types
Exported Constants
Exported Functions

STM32F3-Discovery BSP User Manual

Exported Constants

STM32F3-DISCOVERY Common

Modules

STM32F3-DISCOVERY LED Define for STM32F3_DISCOVERY board.
STM32F3-DISCOVERY BUTTON
STM32F3-DISCOVERY COM
STM32F3-DISCOVERY COMPONENT