STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL Private TypesDefinitions

STM32F4 DISCOVERY LOW LEVEL

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL Private Macros

STM32F4 DISCOVERY LOW LEVEL

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL Private FunctionPrototypes

STM32F4 DISCOVERY LOW LEVEL

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL Exported Macros

STM32F4 DISCOVERY LOW LEVEL

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER Private TypesDefinitions

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER Private Defines

STM32F4 DISCOVERY ACCELEROMETER

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER Private Macros

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER Private FunctionPrototypes

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO Private Types

STM32F4 DISCOVERY AUDIO

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO Private Macros

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO Private Function Prototypes

STM32F4 DISCOVERY AUDIO

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO Exported Types

STM32F4 DISCOVERY AUDIO

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

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

- _ -

- __STM32F4_DISCO_BSP_VERSION : **stm32f4_discovery.c**
- __STM32F4_DISCO_BSP_VERSION_MAIN : **stm32f4_discovery.c**
- __STM32F4_DISCO_BSP_VERSION_RC : **stm32f4_discovery.c**
- __STM32F4_DISCO_BSP_VERSION_SUB1 : **stm32f4_discovery.c**
- __STM32F4_DISCO_BSP_VERSION_SUB2 : **stm32f4_discovery.c**

STM32F4-Discovery BSP User Manual

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

- a -

- ACCELERO_CS_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- ACCELERO_CS_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- ACCELERO_CS_GPIO_PORT : **stm32f4_discovery.h**
- ACCELERO_CS_HIGH : **stm32f4_discovery.h**
- ACCELERO_CS_LOW : **stm32f4_discovery.h**
- ACCELERO_CS_PIN : **stm32f4_discovery.h**
- ACCELERO_ERROR : **stm32f4_discovery_accelerometer.h**
- ACCELERO_INT1_EXTI_IRQn : **stm32f4_discovery.h**
- ACCELERO_INT1_PIN : **stm32f4_discovery.h**
- ACCELERO_INT2_EXTI_IRQn : **stm32f4_discovery.h**
- ACCELERO_INT2_PIN : **stm32f4_discovery.h**
- ACCELERO_INT_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- ACCELERO_INT_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- ACCELERO_INT_GPIO_PORT : **stm32f4_discovery.h**
- ACCELERO_IO_Init() : **stm32f4_discovery.c**
- ACCELERO_IO_ITConfig() : **stm32f4_discovery.c**
- ACCELERO_IO_Read() : **stm32f4_discovery.c**
- ACCELERO_IO_Write() : **stm32f4_discovery.c**
- ACCELERO_OK : **stm32f4_discovery_accelerometer.h**
- ACCELERO_StatusTypeDef : **stm32f4_discovery_accelerometer.h**
- ACCELERO_TIMEOUT : **stm32f4_discovery_accelerometer.h**

- AcceleroDrv : **stm32f4_discovery_accelerometer.c**
- AUDIO_ERROR : **stm32f4_discovery_audio.h**
- AUDIO_I2C_ADDRESS : **stm32f4_discovery.h**
- AUDIO_IN_IRQ_PREPRIO : **stm32f4_discovery_audio.h**
- AUDIO_IO_DeInit() : **stm32f4_discovery.c**
- AUDIO_IO_Init() : **stm32f4_discovery.c**
- AUDIO_IO_Read() : **stm32f4_discovery.c**
- AUDIO_IO_Write() : **stm32f4_discovery.c**
- AUDIO_OK : **stm32f4_discovery_audio.h**
- AUDIO_OUT_IRQ_PREPRIO : **stm32f4_discovery_audio.h**
- AUDIO_RESET_GPIO : **stm32f4_discovery.h**
- AUDIO_RESET_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- AUDIO_RESET_PIN : **stm32f4_discovery.h**
- AUDIO_TIMEOUT : **stm32f4_discovery_audio.h**
- AUDIODATA_SIZE : **stm32f4_discovery_audio.h**
- AudioInVolume : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**

STM32F4-Discovery BSP User Manual

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

- b -

- BSP_ACCELERO_Click_ITClear() : **stm32f4_discovery_accelerometer.c** , **stm32f4_discovery_accelerometer.h**
- BSP_ACCELERO_Click_ITConfig() : **stm32f4_discovery_accelerometer.h** , **stm32f4_discovery_accelerometer.c**
- BSP_ACCELERO_GetXYZ() : **stm32f4_discovery_accelerometer.c** , **stm32f4_discovery_accelerometer.h**
- BSP_ACCELERO_Init() : **stm32f4_discovery_accelerometer.h** , **stm32f4_discovery_accelerometer.c**
- BSP_ACCELERO_ReadID() : **stm32f4_discovery_accelerometer.c** , **stm32f4_discovery_accelerometer.h**
- BSP_ACCELERO_Reset() : **stm32f4_discovery_accelerometer.c** , **stm32f4_discovery_accelerometer.h**
- BSP_AUDIO_IN_ClockConfig() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Error_Callback() : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BSP_AUDIO_IN_HalfTransfer_CallBack() :

stm32f4_discovery_audio.c , **stm32f4_discovery_audio.h**

- BSP_AUDIO_IN_Init() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_MspDeInit() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_MspInit() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Pause() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_PDMToPCM() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Record() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Resume() : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BSP_AUDIO_IN_SetVolume() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Stop() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_TransferComplete_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_ChangeBuffer() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_ClockConfig() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Error_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_HalfTransfer_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Init() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_MspDeInit() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_MspInit() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Pause() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**

- BSP_AUDIO_OUT_Play() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Resume() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_SetFrequency() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_SetMute() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_SetOutputMode() : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BSP_AUDIO_OUT_SetVolume() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Stop() : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BSP_AUDIO_OUT_TransferComplete_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_GetVersion() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_LED_Init() : **stm32f4_discovery.c** , **stm32f4_discovery.h**
- BSP_LED_Off() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_LED_On() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_LED_Toggle() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_PB_GetState() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_PB_Init() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BUTTON_IRQn : **stm32f4_discovery.c**
- BUTTON_KEY : **stm32f4_discovery.h**
- BUTTON_MODE_EXTI : **stm32f4_discovery.h**
- BUTTON_MODE_GPIO : **stm32f4_discovery.h**
- BUTTON_PIN : **stm32f4_discovery.c**
- BUTTON_PORT : **stm32f4_discovery.c**
- Button_TypeDef : **stm32f4_discovery.h**
- ButtonMode_TypeDef : **stm32f4_discovery.h**
- BUTTONn : **stm32f4_discovery.h**
- BUTTONx_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- BUTTONx_GPIO_CLK_ENABLE : **stm32f4_discovery.h**

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP
User Manual by  1.7.6.1

STM32F4-Discovery BSP User Manual

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

- c -

- CHANNEL_DEMUX_MASK : **stm32f4_discovery_audio.h**

STM32F4-Discovery BSP User Manual

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

- d -

- DEFAULT_AUDIO_IN_BIT_RESOLUTION : **stm32f4_discovery_audio.h**
- DEFAULT_AUDIO_IN_CHANNEL_NBR : **stm32f4_discovery_audio.h**
- DEFAULT_AUDIO_IN_FREQ : **stm32f4_discovery_audio.h**
- DEFAULT_AUDIO_IN_VOLUME : **stm32f4_discovery_audio.h**
- DISCOVERY_I2Cx : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_CLK_ENABLE : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_ER_IRQn : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_EV_IRQn : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_FORCE_RESET : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_RELEASE_RESET : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SCL_PIN : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SCL_SDA_AF : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SCL_SDA_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SCL_SDA_GPIO_PORT : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SDA_PIN : **stm32f4_discovery.h**
- DISCOVERY_SPIx : **stm32f4_discovery.h**
- DISCOVERY_SPIx_AF : **stm32f4_discovery.h**
- DISCOVERY_SPIx_CLK_ENABLE : **stm32f4_discovery.h**

- DISCOVERY_SPIx_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- DISCOVERY_SPIx_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- DISCOVERY_SPIx_GPIO_PORT : **stm32f4_discovery.h**
- DISCOVERY_SPIx_MISO_PIN : **stm32f4_discovery.h**
- DISCOVERY_SPIx_MOSI_PIN : **stm32f4_discovery.h**
- DISCOVERY_SPIx_SCK_PIN : **stm32f4_discovery.h**
- DMA_MAX : **stm32f4_discovery_audio.h**
- DMA_MAX_SZE : **stm32f4_discovery_audio.h**
- DUMMY_BYTE : **stm32f4_discovery.h**

STM32F4-Discovery BSP User Manual

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

- f -

- Filter : **stm32f4_discovery_audio.c**

STM32F4-Discovery BSP User Manual

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

- g -

- GPIO_PIN : **stm32f4_discovery.c**
- GPIO_PORT : **stm32f4_discovery.c**

STM32F4-Discovery BSP User Manual

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

- h -

- HAL_I2S_ErrorCallback() : **stm32f4_discovery_audio.c**
- HAL_I2S_RxCpltCallback() : **stm32f4_discovery_audio.c**
- HAL_I2S_RxHalfCpltCallback() : **stm32f4_discovery_audio.c**
- HAL_I2S_TxCpltCallback() : **stm32f4_discovery_audio.c**
- HAL_I2S_TxHalfCpltCallback() : **stm32f4_discovery_audio.c**
- hAudioInI2s : **stm32f4_discovery_audio.c**
- hAudioOutI2s : **stm32f4_discovery_audio.c**

STM32F4-Discovery BSP User Manual

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

- i -

- I2cHandle : **stm32f4_discovery.c**
- I2Cx_Error() : **stm32f4_discovery.c**
- I2Cx_Init() : **stm32f4_discovery.c**
- I2Cx_MspInit() : **stm32f4_discovery.c**
- I2Cx_ReadData() : **stm32f4_discovery.c**
- I2Cx_TIMEOUT_MAX : **stm32f4_discovery.h**
- I2Cx_WriteData() : **stm32f4_discovery.c**
- I2cxTimeout : **stm32f4_discovery.c**
- I2S2 : **stm32f4_discovery_audio.h**
- I2S2_CLK_DISABLE : **stm32f4_discovery_audio.h**
- I2S2_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_CHANNEL : **stm32f4_discovery_audio.h**
- I2S2_DMAx_CLK_DISABLE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_IRQ : **stm32f4_discovery_audio.h**
- I2S2_DMAx_MEM_DATA_SIZE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_PERIPH_DATA_SIZE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_STREAM : **stm32f4_discovery_audio.h**
- I2S2_Init() : **stm32f4_discovery_audio.c**
- I2S2_IRQHandler : **stm32f4_discovery_audio.h**
- I2S2_MOSI_AF : **stm32f4_discovery_audio.h**
- I2S2_MOSI_GPIO_CLK_ENABLE : **stm32f4_discovery_audio.h**

- I2S2_MOSI_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S2_MOSI_PIN : **stm32f4_discovery_audio.h**
- I2S2_SCK_AF : **stm32f4_discovery_audio.h**
- I2S2_SCK_GPIO_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S2_SCK_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S2_SCK_PIN : **stm32f4_discovery_audio.h**
- I2S3 : **stm32f4_discovery_audio.h**
- I2S3_CLK_DISABLE : **stm32f4_discovery_audio.h**
- I2S3_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_CHANNEL : **stm32f4_discovery_audio.h**
- I2S3_DMAx_CLK_DISABLE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_IRQ : **stm32f4_discovery_audio.h**
- I2S3_DMAx_MEM_DATA_SIZE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_PERIPH_DATA_SIZE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_STREAM : **stm32f4_discovery_audio.h**
- I2S3_Init() : **stm32f4_discovery_audio.c**
- I2S3_IRQHandler : **stm32f4_discovery_audio.h**
- I2S3_MCK_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_MCK_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S3_MCK_PIN : **stm32f4_discovery_audio.h**
- I2S3_SCK_PIN : **stm32f4_discovery_audio.h**
- I2S3_SCK_SD_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_SCK_SD_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S3_SCK_SD_WS_AF : **stm32f4_discovery_audio.h**
- I2S3_SD_PIN : **stm32f4_discovery_audio.h**
- I2S3_WS_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_WS_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S3_WS_PIN : **stm32f4_discovery_audio.h**
- I2SFreq : **stm32f4_discovery_audio.c**
- I2SPLLN : **stm32f4_discovery_audio.c**
- I2SPLLR : **stm32f4_discovery_audio.c**
- INTERNAL_BUFF_SIZE : **stm32f4_discovery_audio.h**

STM32F4-Discovery BSP User Manual

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

- k -

- KEY_BUTTON_EXTI_IRQn : **stm32f4_discovery.h**
- KEY_BUTTON_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- KEY_BUTTON_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- KEY_BUTTON_GPIO_PORT : **stm32f4_discovery.h**
- KEY_BUTTON_PIN : **stm32f4_discovery.h**

STM32F4-Discovery BSP User Manual

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

- l -

- LED3 : **stm32f4_discovery.h**
- LED3_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LED3_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- LED3_GPIO_PORT : **stm32f4_discovery.h**
- LED3_PIN : **stm32f4_discovery.h**
- LED4 : **stm32f4_discovery.h**
- LED4_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LED4_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- LED4_GPIO_PORT : **stm32f4_discovery.h**
- LED4_PIN : **stm32f4_discovery.h**
- LED5 : **stm32f4_discovery.h**
- LED5_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LED5_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- LED5_GPIO_PORT : **stm32f4_discovery.h**
- LED5_PIN : **stm32f4_discovery.h**
- LED6 : **stm32f4_discovery.h**
- LED6_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LED6_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- LED6_GPIO_PORT : **stm32f4_discovery.h**
- LED6_PIN : **stm32f4_discovery.h**
- Led_TypeDef : **stm32f4_discovery.h**
- LEDn : **stm32f4_discovery.h**

- LEDx_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LEDx_GPIO_CLK_ENABLE : **stm32f4_discovery.h**

STM32F4-Discovery BSP User Manual

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

- m -

- MULTIPLEBYTE_CMD : **stm32f4_discovery.h**

STM32F4-Discovery BSP User Manual

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

- p -

- pAudioDrv : **stm32f4_discovery_audio.c**
- PCM_OUT_SIZE : **stm32f4_discovery_audio.h**
- PDMDecoder_Init() : **stm32f4_discovery_audio.c**

STM32F4-Discovery BSP User Manual

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

- r -

- READWRITE_CMD : **stm32f4_discovery.h**

STM32F4-Discovery BSP User Manual

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

- s -

- SpiHandle : **stm32f4_discovery.c**
- SPIx_Error() : **stm32f4_discovery.c**
- SPIx_Init() : **stm32f4_discovery.c**
- SPIx_MspInit() : **stm32f4_discovery.c**
- SPIx_TIMEOUT_MAX : **stm32f4_discovery.h**
- SPIx_WriteRead() : **stm32f4_discovery.c**
- SpixTimeout : **stm32f4_discovery.c**

STM32F4-Discovery BSP User Manual

- a -

- ACCELERO_IO_Init() : **stm32f4_discovery.c**
- ACCELERO_IO_ITConfig() : **stm32f4_discovery.c**
- ACCELERO_IO_Read() : **stm32f4_discovery.c**
- ACCELERO_IO_Write() : **stm32f4_discovery.c**
- AUDIO_IO_DeInit() : **stm32f4_discovery.c**
- AUDIO_IO_Init() : **stm32f4_discovery.c**
- AUDIO_IO_Read() : **stm32f4_discovery.c**
- AUDIO_IO_Write() : **stm32f4_discovery.c**

- b -

- BSP_ACCELERO_Click_ITClear() : **stm32f4_discovery_accelerometer.c** , **stm32f4_discovery_accelerometer.h**
- BSP_ACCELERO_Click_ITConfig() : **stm32f4_discovery_accelerometer.h** , **stm32f4_discovery_accelerometer.c**
- BSP_ACCELERO_GetXYZ() : **stm32f4_discovery_accelerometer.c** , **stm32f4_discovery_accelerometer.h**
- BSP_ACCELERO_Init() : **stm32f4_discovery_accelerometer.h** , **stm32f4_discovery_accelerometer.c**
- BSP_ACCELERO_ReadID() :

stm32f4_discovery_accelerometer.c , **stm32f4_discovery_accelerometer.h**

- BSP_ACCELERO_Reset() : **stm32f4_discovery_accelerometer.c** , **stm32f4_discovery_accelerometer.h**
- BSP_AUDIO_IN_ClockConfig() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Error_Callback() : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BSP_AUDIO_IN_HalfTransfer_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Init() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_MspDeInit() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_MspInit() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Pause() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_PDMToPCM() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Record() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Resume() : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BSP_AUDIO_IN_SetVolume() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_Stop() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_IN_TransferComplete_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_ChangeBuffer() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_ClockConfig() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Error_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**

- BSP_AUDIO_OUT_HalfTransfer_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Init() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_MspDeInit() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_MspInit() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Pause() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Play() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Resume() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_SetFrequency() : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BSP_AUDIO_OUT_SetMute() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_SetOutputMode() : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BSP_AUDIO_OUT_SetVolume() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_Stop() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_AUDIO_OUT_TransferComplete_CallBack() : **stm32f4_discovery_audio.c** , **stm32f4_discovery_audio.h**
- BSP_GetVersion() : **stm32f4_discovery.c** , **stm32f4_discovery.h**
- BSP_LED_Init() : **stm32f4_discovery.c** , **stm32f4_discovery.h**
- BSP_LED_Off() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_LED_On() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_LED_Toggle() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_PB_GetState() : **stm32f4_discovery.h** , **stm32f4_discovery.c**
- BSP_PB_Init() : **stm32f4_discovery.h** , **stm32f4_discovery.c**

- h -

- HAL_I2S_ErrorCallback() : **stm32f4_discovery_audio.c**
- HAL_I2S_RxCpltCallback() : **stm32f4_discovery_audio.c**
- HAL_I2S_RxHalfCpltCallback() : **stm32f4_discovery_audio.c**
- HAL_I2S_TxCpltCallback() : **stm32f4_discovery_audio.c**
- HAL_I2S_TxHalfCpltCallback() : **stm32f4_discovery_audio.c**

- i -

- I2Cx_Error() : **stm32f4_discovery.c**
- I2Cx_Init() : **stm32f4_discovery.c**
- I2Cx_MspInit() : **stm32f4_discovery.c**
- I2Cx_ReadData() : **stm32f4_discovery.c**
- I2Cx_WriteData() : **stm32f4_discovery.c**
- I2S2_Init() : **stm32f4_discovery_audio.c**
- I2S3_Init() : **stm32f4_discovery_audio.c**

- p -

- PDMDecoder_Init() : **stm32f4_discovery_audio.c**

- s -

- SPIx_Error() : **stm32f4_discovery.c**
- SPIx_Init() : **stm32f4_discovery.c**
- SPIx_MspInit() : **stm32f4_discovery.c**
- SPIx_WriteRead() : **stm32f4_discovery.c**

STM32F4-Discovery BSP User Manual

- AcceleroDrv : **stm32f4_discovery_accelerometer.c**
- AudioInVolume : **stm32f4_discovery_audio.h** , **stm32f4_discovery_audio.c**
- BUTTON_IRQn : **stm32f4_discovery.c**
- BUTTON_PIN : **stm32f4_discovery.c**
- BUTTON_PORT : **stm32f4_discovery.c**
- Filter : **stm32f4_discovery_audio.c**
- GPIO_PIN : **stm32f4_discovery.c**
- GPIO_PORT : **stm32f4_discovery.c**
- hAudioInI2s : **stm32f4_discovery_audio.c**
- hAudioOutI2s : **stm32f4_discovery_audio.c**
- I2cHandle : **stm32f4_discovery.c**
- I2cxTimeout : **stm32f4_discovery.c**
- I2SFreq : **stm32f4_discovery_audio.c**
- I2SPLLN : **stm32f4_discovery_audio.c**
- I2SPLLR : **stm32f4_discovery_audio.c**
- pAudioDrv : **stm32f4_discovery_audio.c**
- SpiHandle : **stm32f4_discovery.c**
- SpixTimeout : **stm32f4_discovery.c**

STM32F4-Discovery BSP User Manual

- ACCELERO_StatusTypeDef : **stm32f4_discovery_accelerometer.h**
- Button_TypeDef : **stm32f4_discovery.h**
- ButtonMode_TypeDef : **stm32f4_discovery.h**
- Led_TypeDef : **stm32f4_discovery.h**

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

- ACCELERO_ERROR : **stm32f4_discovery_accelerometer.h**
- ACCELERO_OK : **stm32f4_discovery_accelerometer.h**
- ACCELERO_TIMEOUT : **stm32f4_discovery_accelerometer.h**
- BUTTON_KEY : **stm32f4_discovery.h**
- BUTTON_MODE_EXTI : **stm32f4_discovery.h**
- BUTTON_MODE_GPIO : **stm32f4_discovery.h**
- LED3 : **stm32f4_discovery.h**
- LED4 : **stm32f4_discovery.h**
- LED5 : **stm32f4_discovery.h**
- LED6 : **stm32f4_discovery.h**

STM32F4-Discovery BSP User Manual

- _ -

- __STM32F4_DISCO_BSP_VERSION : **stm32f4_discovery.c**
- __STM32F4_DISCO_BSP_VERSION_MAIN : **stm32f4_discovery.c**
- __STM32F4_DISCO_BSP_VERSION_RC : **stm32f4_discovery.c**
- __STM32F4_DISCO_BSP_VERSION_SUB1 : **stm32f4_discovery.c**
- __STM32F4_DISCO_BSP_VERSION_SUB2 : **stm32f4_discovery.c**

- a -

- ACCELERO_CS_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- ACCELERO_CS_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- ACCELERO_CS_GPIO_PORT : **stm32f4_discovery.h**
- ACCELERO_CS_HIGH : **stm32f4_discovery.h**
- ACCELERO_CS_LOW : **stm32f4_discovery.h**
- ACCELERO_CS_PIN : **stm32f4_discovery.h**
- ACCELERO_INT1_EXTI_IRQn : **stm32f4_discovery.h**
- ACCELERO_INT1_PIN : **stm32f4_discovery.h**
- ACCELERO_INT2_EXTI_IRQn : **stm32f4_discovery.h**
- ACCELERO_INT2_PIN : **stm32f4_discovery.h**
- ACCELERO_INT_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- ACCELERO_INT_GPIO_CLK_ENABLE : **stm32f4_discovery.h**

- ACCELERO_INT_GPIO_PORT : **stm32f4_discovery.h**
- AUDIO_ERROR : **stm32f4_discovery_audio.h**
- AUDIO_I2C_ADDRESS : **stm32f4_discovery.h**
- AUDIO_IN_IRQ_PREPRIO : **stm32f4_discovery_audio.h**
- AUDIO_OK : **stm32f4_discovery_audio.h**
- AUDIO_OUT_IRQ_PREPRIO : **stm32f4_discovery_audio.h**
- AUDIO_RESET_GPIO : **stm32f4_discovery.h**
- AUDIO_RESET_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- AUDIO_RESET_PIN : **stm32f4_discovery.h**
- AUDIO_TIMEOUT : **stm32f4_discovery_audio.h**
- AUDIODATA_SIZE : **stm32f4_discovery_audio.h**

- b -

- BUTTONn : **stm32f4_discovery.h**
- BUTTONx_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- BUTTONx_GPIO_CLK_ENABLE : **stm32f4_discovery.h**

- c -

- CHANNEL_DEMUX_MASK : **stm32f4_discovery_audio.h**

- d -

- DEFAULT_AUDIO_IN_BIT_RESOLUTION : **stm32f4_discovery_audio.h**
- DEFAULT_AUDIO_IN_CHANNEL_NBR : **stm32f4_discovery_audio.h**
- DEFAULT_AUDIO_IN_FREQ : **stm32f4_discovery_audio.h**
- DEFAULT_AUDIO_IN_VOLUME : **stm32f4_discovery_audio.h**
- DISCOVERY_I2Cx : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_CLK_ENABLE : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_ER_IRQn : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_EV_IRQn : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_FORCE_RESET : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_RELEASE_RESET : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SCL_PIN : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SCL_SDA_AF : **stm32f4_discovery.h**

- DISCOVERY_I2Cx_SCL_SDA_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SCL_SDA_GPIO_PORT : **stm32f4_discovery.h**
- DISCOVERY_I2Cx_SDA_PIN : **stm32f4_discovery.h**
- DISCOVERY_SPIx : **stm32f4_discovery.h**
- DISCOVERY_SPIx_AF : **stm32f4_discovery.h**
- DISCOVERY_SPIx_CLK_ENABLE : **stm32f4_discovery.h**
- DISCOVERY_SPIx_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- DISCOVERY_SPIx_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- DISCOVERY_SPIx_GPIO_PORT : **stm32f4_discovery.h**
- DISCOVERY_SPIx_MISO_PIN : **stm32f4_discovery.h**
- DISCOVERY_SPIx_MOSI_PIN : **stm32f4_discovery.h**
- DISCOVERY_SPIx_SCK_PIN : **stm32f4_discovery.h**
- DMA_MAX : **stm32f4_discovery_audio.h**
- DMA_MAX_SZE : **stm32f4_discovery_audio.h**
- DUMMY_BYTE : **stm32f4_discovery.h**

- i -

- I2Cx_TIMEOUT_MAX : **stm32f4_discovery.h**
- I2S2 : **stm32f4_discovery_audio.h**
- I2S2_CLK_DISABLE : **stm32f4_discovery_audio.h**
- I2S2_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_CHANNEL : **stm32f4_discovery_audio.h**
- I2S2_DMAx_CLK_DISABLE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_IRQ : **stm32f4_discovery_audio.h**
- I2S2_DMAx_MEM_DATA_SIZE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_PERIPH_DATA_SIZE : **stm32f4_discovery_audio.h**
- I2S2_DMAx_STREAM : **stm32f4_discovery_audio.h**
- I2S2_IRQHandler : **stm32f4_discovery_audio.h**
- I2S2_MOSI_AF : **stm32f4_discovery_audio.h**
- I2S2_MOSI_GPIO_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S2_MOSI_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S2_MOSI_PIN : **stm32f4_discovery_audio.h**
- I2S2_SCK_AF : **stm32f4_discovery_audio.h**

- I2S2_SCK_GPIO_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S2_SCK_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S2_SCK_PIN : **stm32f4_discovery_audio.h**
- I2S3 : **stm32f4_discovery_audio.h**
- I2S3_CLK_DISABLE : **stm32f4_discovery_audio.h**
- I2S3_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_CHANNEL : **stm32f4_discovery_audio.h**
- I2S3_DMAx_CLK_DISABLE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_IRQ : **stm32f4_discovery_audio.h**
- I2S3_DMAx_MEM_DATA_SIZE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_PERIPH_DATA_SIZE : **stm32f4_discovery_audio.h**
- I2S3_DMAx_STREAM : **stm32f4_discovery_audio.h**
- I2S3_IRQHandler : **stm32f4_discovery_audio.h**
- I2S3_MCK_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_MCK_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S3_MCK_PIN : **stm32f4_discovery_audio.h**
- I2S3_SCK_PIN : **stm32f4_discovery_audio.h**
- I2S3_SCK_SD_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_SCK_SD_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S3_SCK_SD_WS_AF : **stm32f4_discovery_audio.h**
- I2S3_SD_PIN : **stm32f4_discovery_audio.h**
- I2S3_WS_CLK_ENABLE : **stm32f4_discovery_audio.h**
- I2S3_WS_GPIO_PORT : **stm32f4_discovery_audio.h**
- I2S3_WS_PIN : **stm32f4_discovery_audio.h**
- INTERNAL_BUFF_SIZE : **stm32f4_discovery_audio.h**

- k -

- KEY_BUTTON_EXTI_IRQn : **stm32f4_discovery.h**
- KEY_BUTTON_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- KEY_BUTTON_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- KEY_BUTTON_GPIO_PORT : **stm32f4_discovery.h**
- KEY_BUTTON_PIN : **stm32f4_discovery.h**

- l -

- LED3_GPIO_CLK_DISABLE : **stm32f4_discovery.h**

- LED3_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- LED3_GPIO_PORT : **stm32f4_discovery.h**
- LED3_PIN : **stm32f4_discovery.h**
- LED4_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LED4_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- LED4_GPIO_PORT : **stm32f4_discovery.h**
- LED4_PIN : **stm32f4_discovery.h**
- LED5_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LED5_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- LED5_GPIO_PORT : **stm32f4_discovery.h**
- LED5_PIN : **stm32f4_discovery.h**
- LED6_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LED6_GPIO_CLK_ENABLE : **stm32f4_discovery.h**
- LED6_GPIO_PORT : **stm32f4_discovery.h**
- LED6_PIN : **stm32f4_discovery.h**
- LEDn : **stm32f4_discovery.h**
- LEDx_GPIO_CLK_DISABLE : **stm32f4_discovery.h**
- LEDx_GPIO_CLK_ENABLE : **stm32f4_discovery.h**

- m -

- MULTIPLEBYTE_CMD : **stm32f4_discovery.h**

- p -

- PCM_OUT_SIZE : **stm32f4_discovery_audio.h**

- r -

- READWRITE_CMD : **stm32f4_discovery.h**

- s -

- SPIx_TIMEOUT_MAX : **stm32f4_discovery.h**

STM32F4-Discovery BSP User Manual

Defines | Functions | Variables

stm32f4_discovery.c File Reference

This file provides set of firmware functions to manage Leds and push-button available on STM32F4-Discovery Kit from STMicroelectronics. More...

```
#include "stm32f4_discovery.h"
```

Go to the source code of this file.

Defines

#define	**__STM32F4_DISCO_BSP_VERSION_MAIN** (0x02) STM32F4 DISCO BSP Driver version number V2.1.2.
#define	**__STM32F4_DISCO_BSP_VERSION_SUB1** (0x01)
#define	**__STM32F4_DISCO_BSP_VERSION_SUB2** (0x02)
#define	**__STM32F4_DISCO_BSP_VERSION_RC** (0x00)
#define	**__STM32F4_DISCO_BSP_VERSION**

Functions

static void	**I2Cx_Init** (void) Configures I2C interface.
static void	**I2Cx_WriteData** (uint8_t Addr, uint8_t Reg, uint8_t Value) Write a value in a register of the device through BUS.
static uint8_t	**I2Cx_ReadData** (uint8_t Addr, uint8_t Reg) Read a register of the device through BUS.
static void	**I2Cx_MspInit** (void) I2C MSP Initialization.
static void	**I2Cx_Error** (uint8_t Addr) Manages error callback by re-initializing I2C.
static void	**SPIx_Init** (void) SPIx Bus initialization.
static void	**SPIx_MspInit** (void) SPI MSP Init.
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.
void	**ACCELERO_IO_Init** (void) Configures the Accelerometer SPI interface.
void	**ACCELERO_IO_ITConfig** (void) Configures the Accelerometer INT2.
void	**ACCELERO_IO_Write** (uint8_t *pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite) Writes one byte to the Accelerometer.
void	**ACCELERO_IO_Read** (uint8_t *pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead) Reads a block of data from the Accelerometer.
void	**AUDIO_IO_Init** (void)

	Initializes Audio low level.
void	**AUDIO_IO_DeInit** (void) DeInitializes Audio low level.
void	**AUDIO_IO_Write** (uint8_t Addr, uint8_t Reg, uint8_t Value) Writes a single data.
uint8_t	**AUDIO_IO_Read** (uint8_t Addr, uint8_t Reg) Reads a single data.
uint32_t	**BSP_GetVersion** (void) This method returns the STM32F4 DISCO 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** Mode) Configures Button GPIO and EXTI Line.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Returns the selected Button state.

Variables

GPIO_TypeDef *	**GPIO_PORT** [**LEDn**]
const uint16_t	**GPIO_PIN** [**LEDn**]
GPIO_TypeDef *	**BUTTON_PORT** [**BUTTONn**] = {**KEY_BUTTON_GPIO_PORT**}
const uint16_t	**BUTTON_PIN** [**BUTTONn**] = {**KEY_BUTTON_PIN**}
const uint8_t	**BUTTON_IRQn** [**BUTTONn**] = {**KEY_BUTTON_EXTI_IRQn**}
uint32_t	**I2cxTimeout** = **I2Cx_TIMEOUT_MAX**
uint32_t	**SpixTimeout** = **SPIx_TIMEOUT_MAX**
static SPI_HandleTypeDef	**SpiHandle**
static I2C_HandleTypeDef	**I2cHandle**

Detailed Description

This file provides set of firmware functions to manage Leds and push-button available on STM32F4-Discovery Kit from STMicroelectronics.

Author:
MCD Application Team

Version:
V2.1.2

Date:
31-January-2017

Attention:

Definition in file **stm32f4_discovery.c**.

STM32F4-Discovery BSP User Manual

Defines | Enumerations | Functions

stm32f4_discovery.h File Reference

This file contains definitions for STM32F4-Discovery Kit's Leds and push-button hardware resources. More...

```
#include "stm32f4xx_hal.h"
```

Go to the source code of this file.

Defines

#define **LEDn** 4

#define **LED4_PIN** GPIO_PIN_12

#define **LED4_GPIO_PORT** GPIOD

#define **LED4_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOD_CLK_E

#define **LED4_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOD_CLK_D

#define **LED3_PIN** GPIO_PIN_13

#define **LED3_GPIO_PORT** GPIOD

#define **LED3_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOD_CLK_E

#define **LED3_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOD_CLK_D

#define **LED5_PIN** GPIO_PIN_14

#define **LED5_GPIO_PORT** GPIOD

#define **LED5_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOD_CLK_E

#define **LED5_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOD_CLK_D

#define **LED6_PIN** GPIO_PIN_15

#define **LED6_GPIO_PORT** GPIOD

#define **LED6_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOD_CLK_E

#define **LED6_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOD_CLK_D

#define **LEDx_GPIO_CLK_ENABLE**(__INDEX__)

#define **LEDx_GPIO_CLK_DISABLE**(__INDEX__)

#define **BUTTONn** 1

#define **KEY_BUTTON_PIN** GPIO_PIN_0

Wakeup push-button.

#define **KEY_BUTTON_GPIO_PORT** GPIOA

#define **KEY_BUTTON_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOA

#define **KEY_BUTTON_GPIO_CLK_DISABLE**() __HAL_RCC_GPIO

#define **KEY_BUTTON_EXTI_IRQn** EXTI0_IRQn

#define **BUTTONx_GPIO_CLK_ENABLE**(__INDEX__)

#define **BUTTONx_GPIO_CLK_DISABLE**(__INDEX__)

#define **DISCOVERY_SPIx** SPI1

#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** 0x1000 /*<! The value of the maximal

#define **DISCOVERY_I2Cx** I2C1

#define **DISCOVERY_I2Cx_CLK_ENABLE**() __HAL_RCC_I2C1_CL

#define **DISCOVERY_I2Cx_SCL_SDA_GPIO_CLK_ENABLE**() __HA

#define **DISCOVERY_I2Cx_SCL_SDA_AF** GPIO_AF4_I2C1

#define **DISCOVERY_I2Cx_SCL_SDA_GPIO_PORT** GPIOB

#define **DISCOVERY_I2Cx_SCL_PIN** GPIO_PIN_6

#define **DISCOVERY_I2Cx_SDA_PIN** GPIO_PIN_9

#define **DISCOVERY_I2Cx_FORCE_RESET**() __HAL_RCC_I2C1_F

#define **DISCOVERY_I2Cx_RELEASE_RESET**() __HAL_RCC_I2C1_

#define **DISCOVERY_I2Cx_EV_IRQn** I2C1_EV_IRQn

#define **DISCOVERY_I2Cx_ER_IRQn** I2C1_ER_IRQn

#define **I2Cx_TIMEOUT_MAX** 0x1000 /*<! The value of the maximal

#define **READWRITE_CMD** ((uint8_t)0x80)

#define **MULTIPLEBYTE_CMD** ((uint8_t)0x40)

#define **DUMMY_BYTE** ((uint8_t)0x00)

#define **ACCELERO_CS_LOW**() HAL_GPIO_WritePin(**ACCELERO_**
ACCELERO_CS_PIN, GPIO_PIN_RESET)

#define **ACCELERO_CS_HIGH**() HAL_GPIO_WritePin(**ACCELERO_**
ACCELERO_CS_PIN, GPIO_PIN_SET)

#define **ACCELERO_CS_PIN** GPIO_PIN_3 /* PE.03 */
ACCELEROMETER Interface pins.

#define **ACCELERO_CS_GPIO_PORT** GPIOE /* GPIOE */

#define **ACCELERO_CS_GPIO_CLK_ENABLE**() __HAL_RCC_GPI

#define **ACCELERO_CS_GPIO_CLK_DISABLE**() __HAL_RCC_GPI

#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_0 /* PE.00 */

#define **ACCELERO_INT1_EXTI_IRQn** EXTI0_IRQn

#define **ACCELERO_INT2_PIN** GPIO_PIN_1 /* PE.01 */

#define **ACCELERO_INT2_EXTI_IRQn** EXTI1_IRQn

#define **AUDIO_I2C_ADDRESS** 0x94

AUDIO I2C Interface pins.

#define **AUDIO_RESET_GPIO_CLK_ENABLE**() __HAL_RCC_GPIO

#define **AUDIO_RESET_PIN** GPIO_PIN_4

#define **AUDIO_RESET_GPIO** GPIOD

Enumerations

enum	**Led_TypeDef** { **LED4** = 0, **LED3** = 1, **LED5** = 2, **LED6** = 3 }
enum	**Button_TypeDef** { **BUTTON_KEY** = 0 }
enum	**ButtonMode_TypeDef** { **BUTTON_MODE_GPIO** = 0, **BUTTON_MODE_EXTI** = 1 }

Functions

uint32_t	**BSP_GetVersion** (void) This method returns the STM32F4 DISCO 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** Mode) Configures Button GPIO and EXTI Line.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Returns the selected Button state.

Detailed Description

This file contains definitions for STM32F4-Discovery Kit's Leds and push-button hardware resources.

Author:
MCD Application Team

Version:
V2.1.2

Date:
31-January-2017

Attention:

Definition in file **stm32f4_discovery.h**.

STM32F4-Discovery BSP User Manual

stm32f4_discovery_accelerometer.c File Reference

This file provides a set of functions needed to manage the MEMS accelerometers available on STM32F4-Discovery Kit. More...

```
#include "stm32f4_discovery_accelerometer.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_ACCELERO_Init** (void) Setx Accelerometer Initialization.
uint8_t	**BSP_ACCELERO_ReadID** (void) Read ID of Accelerometer component.
void	**BSP_ACCELERO_Reset** (void) Reboot memory content of Accelerometer.
void	**BSP_ACCELERO_Click_ITConfig** (void) Configure Accelerometer click IT.
void	**BSP_ACCELERO_Click_ITClear** (void) Clear Accelerometer click IT.
void	**BSP_ACCELERO_GetXYZ** (int16_t *pDataXYZ) Get XYZ axes acceleration.

Variables

static ACCELERO_DrvTypeDef * **AcceleroDrv**

Detailed Description

This file provides a set of functions needed to manage the MEMS accelerometers available on STM32F4-Discovery Kit.

Author:
MCD Application Team

Version:
V2.1.2

Date:
31-January-2017

Attention:

Definition in file **stm32f4_discovery_accelerometer.c**.

STM32F4-Discovery BSP User Manual

Enumerations | Functions

stm32f4_discovery_accelerometer.h File Reference

This file contains all the functions prototypes for the **stm32f4_discovery_accelerometer.c** firmware driver. More...

```
#include "stm32f4_discovery.h" #include
"../Components/lis302dl/lis302dl.h"
#include "../Components/lis3dsh/lis3dsh.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) Setx Accelerometer Initialization.
uint8_t	**BSP_ACCELERO_ReadID** (void) Read ID of Accelerometer component.
void	**BSP_ACCELERO_Reset** (void) Reboot memory content of Accelerometer.
void	**BSP_ACCELERO_Click_ITConfig** (void) Configure Accelerometer click IT.
void	**BSP_ACCELERO_Click_ITClear** (void) Clear Accelerometer click IT.
void	**BSP_ACCELERO_GetXYZ** (int16_t *pDataXYZ) Get XYZ axes acceleration.

Detailed Description

This file contains all the functions prototypes for the **stm32f4_discovery_accelerometer.c** firmware driver.

Author:
MCD Application Team

Version:
V2.1.2

Date:
31-January-2017

Attention:

Definition in file **stm32f4_discovery_accelerometer.h**.

STM32F4-Discovery BSP User Manual

stm32f4_discovery_audio.c File Reference

This file provides the Audio driver for the STM32F4-Discovery board. More...

```
#include "stm32f4_discovery_audio.h"
```

Go to the source code of this file.

Functions

static uint8_t	**I2S3_Init** (uint32_t AudioFreq) Initializes the Audio Codec audio interface (I2S).
static uint8_t	**I2S2_Init** (uint32_t AudioFreq) Initializes the Audio Codec audio interface (I2S)
static void	**PDMDecoder_Init** (uint32_t AudioFreq, uint32_t ChnlNbr) Initialize the PDM library.
uint8_t	**BSP_AUDIO_OUT_Init** (uint16_t OutputDevice, uint8_t Volume, uint32_t AudioFreq) Configures the audio peripherals.
uint8_t	**BSP_AUDIO_OUT_Play** (uint16_t *pBuffer, uint32_t Size) Starts playing audio stream from a data buffer for a determined size.
void	**BSP_AUDIO_OUT_ChangeBuffer** (uint16_t *pData, uint16_t Size) Sends n-Bytes on the I2S interface.
uint8_t	**BSP_AUDIO_OUT_Pause** (void) Pauses the audio file stream.
uint8_t	**BSP_AUDIO_OUT_Resume** (void) Resumes the audio file streaming.
uint8_t	**BSP_AUDIO_OUT_Stop** (uint32_t Option) Stops audio playing and Power down the Audio Codec.
uint8_t	**BSP_AUDIO_OUT_SetVolume** (uint8_t Volume) Controls the current audio volume level.
uint8_t	**BSP_AUDIO_OUT_SetMute** (uint32_t Cmd) Enables or disables the MUTE mode by software.
uint8_t	**BSP_AUDIO_OUT_SetOutputMode** (uint8_t Output) Switch dynamically (while audio file is played) the output target (speaker or headphone).
	BSP_AUDIO_OUT_SetFrequency (uint32_t

void	AudioFreq) Update the audio frequency.
void	**HAL_I2S_TxCpltCallback** (I2S_HandleTypeDef *hi2s) Tx Transfer completed callbacks.
void	**HAL_I2S_TxHalfCpltCallback** (I2S_HandleTypeDef *hi2s) Tx Half Transfer completed callbacks.
__weak void	**BSP_AUDIO_OUT_ClockConfig** (I2S_HandleTypeDef *hi2s, uint32_t AudioFreq, void *Params) Clock Config.
__weak void	**BSP_AUDIO_OUT_MspInit** (I2S_HandleTypeDef *hi2s, void *Params) AUDIO OUT I2S MSP Init.
__weak void	**BSP_AUDIO_OUT_MspDeInit** (I2S_HandleTypeDef *hi2s, void *Params) De-Initializes BSP_AUDIO_OUT MSP.
__weak void	**BSP_AUDIO_OUT_TransferComplete_CallBack** (void) Manages the DMA full Transfer complete event.
__weak void	**BSP_AUDIO_OUT_HalfTransfer_CallBack** (void) Manages the DMA Half Transfer complete event.
__weak void	**BSP_AUDIO_OUT_Error_CallBack** (void) Manages the DMA FIFO error event.
uint8_t	**BSP_AUDIO_IN_Init** (uint32_t AudioFreq, uint32_t BitRes, uint32_t ChnlNbr) Initializes wave recording.
uint8_t	**BSP_AUDIO_IN_Record** (uint16_t *pbuf, uint32_t size) Starts audio recording.
uint8_t	**BSP_AUDIO_IN_Stop** (void) Stops audio recording.
uint8_t	**BSP_AUDIO_IN_Pause** (void) Pauses the audio file stream.

uint8_t	**BSP_AUDIO_IN_Resume** (void) Resumes the audio file stream.
uint8_t	**BSP_AUDIO_IN_SetVolume** (uint8_t Volume) Controls the audio in volume level.
uint8_t	**BSP_AUDIO_IN_PDMToPCM** (uint16_t *PDMBuf, uint16_t *PCMBuf) Converts audio format from PDM to PCM.
void	**HAL_I2S_RxCpltCallback** (I2S_HandleTypeDef *hi2s) Rx Transfer completed callbacks.
void	**HAL_I2S_RxHalfCpltCallback** (I2S_HandleTypeDef *hi2s) Rx Half Transfer completed callbacks.
__weak void	**BSP_AUDIO_IN_ClockConfig** (I2S_HandleTypeDef *hi2s, uint32_t AudioFreq, void *Params) Audio In Clock Config.
__weak void	**BSP_AUDIO_IN_MspInit** (I2S_HandleTypeDef *hi2s, void *Params) BSP AUDIO IN MSP Init.
__weak void	**BSP_AUDIO_IN_MspDeInit** (I2S_HandleTypeDef *hi2s, void *Params) DeInitializes BSP_AUDIO_IN MSP.
__weak void	**BSP_AUDIO_IN_TransferComplete_CallBack** (void) User callback when record buffer is filled.
__weak void	**BSP_AUDIO_IN_HalfTransfer_CallBack** (void) Manages the DMA Half Transfer complete event.
__weak void	**BSP_AUDIO_IN_Error_Callback** (void) Audio IN Error callback function.
void	**HAL_I2S_ErrorCallback** (I2S_HandleTypeDef *hi2s) I2S error callbacks.

Variables

const uint32_t	**I2SFreq** [8] = {8000, 11025, 16000, 22050, 32000, 44100, 48000, 96000}
const uint32_t	**I2SPLLN** [8] = {256, 429, 213, 429, 426, 271, 258, 344}
const uint32_t	**I2SPLLR** [8] = {5, 4, 4, 4, 4, 6, 3, 1}
static AUDIO_DrvTypeDef *	**pAudioDrv**
I2S_HandleTypeDef	**hAudioOutI2s**
I2S_HandleTypeDef	**hAudioInI2s**
PDMFilter_InitStruct	**Filter** [**DEFAULT_AUDIO_IN_CHANNEL_NBR**]
__IO uint16_t	**AudioInVolume** = **DEFAULT_AUDIO_IN_VOLUME**

Detailed Description

This file provides the Audio driver for the STM32F4-Discovery board.

Author:
MCD Application Team

Version:
V2.1.2

Date:
31-January-2017

Attention:

Definition in file **stm32f4_discovery_audio.c**.

STM32F4-Discovery BSP User Manual

stm32f4_discovery_audio.h File Reference

This file contains the common defines and functions prototypes for **stm32f4_discovery_audio.c** driver. More...

```
#include "../Components/cs43l22/cs43l22.h" #include
"stm32f4_discovery.h"
#include
"../../../Middlewares/ST/STM32_Audio/Addons/PDM/pdm_filter.h"
```

Go to the source code of this file.

Defines

```
#define I2S3 SPI3
#define I2S3_CLK_ENABLE() __HAL_RCC_SPI3_CLK_ENABLE()
#define I2S3_CLK_DISABLE() __HAL_RCC_SPI3_CLK_DISABLE()
#define I2S3_SCK_SD_WS_AF GPIO_AF6_SPI3
#define I2S3_SCK_SD_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_
#define I2S3_MCK_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_EN
#define I2S3_WS_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENA
#define I2S3_WS_PIN GPIO_PIN_4
#define I2S3_SCK_PIN GPIO_PIN_10
#define I2S3_SD_PIN GPIO_PIN_12
#define I2S3_MCK_PIN GPIO_PIN_7
#define I2S3_SCK_SD_GPIO_PORT GPIOC
#define I2S3_WS_GPIO_PORT GPIOA
#define I2S3_MCK_GPIO_PORT GPIOC
#define I2S3_DMAx_CLK_ENABLE() __HAL_RCC_DMA1_CLK_EN
#define I2S3_DMAx_CLK_DISABLE() __HAL_RCC_DMA1_CLK_DI
#define I2S3_DMAx_STREAM DMA1_Stream7
#define I2S3_DMAx_CHANNEL DMA_CHANNEL_0
#define I2S3_DMAx_IRQ DMA1_Stream7_IRQn
#define I2S3_DMAx_PERIPH_DATA_SIZE DMA_PDATAALIGN_HAL
#define I2S3_DMAx_MEM_DATA_SIZE DMA_MDATAALIGN_HALFW
#define DMA_MAX_SZE 0xFFFF
#define I2S3_IRQHandler DMA1_Stream7_IRQHandler
#define AUDIO_OUT_IRQ_PREPRIO 0x0E /* Select the preemption
#define I2S2 SPI2
#define I2S2_CLK_ENABLE() __HAL_RCC_SPI2_CLK_ENABLE()
#define I2S2_CLK_DISABLE() __HAL_RCC_SPI2_CLK_DISABLE()
#define I2S2_SCK_PIN GPIO_PIN_10
#define I2S2_SCK_GPIO_PORT GPIOB
#define I2S2_SCK_GPIO_CLK_ENABLE() __HAL_RCC_GPIOB_CL
#define I2S2_SCK_AF GPIO_AF5_SPI2
```

```
#define I2S2_MOSI_PIN GPIO_PIN_3
#define I2S2_MOSI_GPIO_PORT GPIOC
#define I2S2_MOSI_GPIO_CLK_ENABLE() __HAL_RCC_GPIOC_C
#define I2S2_MOSI_AF GPIO_AF5_SPI2
#define I2S2_DMAx_CLK_ENABLE() __HAL_RCC_DMA1_CLK_EN
#define I2S2_DMAx_CLK_DISABLE() __HAL_RCC_DMA1_CLK_DI
#define I2S2_DMAx_STREAM DMA1_Stream3
#define I2S2_DMAx_CHANNEL DMA_CHANNEL_0
#define I2S2_DMAx_IRQ DMA1_Stream3_IRQn
#define I2S2_DMAx_PERIPH_DATA_SIZE DMA_PDATAALIGN_HAL
#define I2S2_DMAx_MEM_DATA_SIZE DMA_MDATAALIGN_HALFW
#define I2S2_IRQHandler DMA1_Stream3_IRQHandler
#define AUDIO_IN_IRQ_PREPRIO 0x0F /* Select the preemption pri
#define AUDIODATA_SIZE 2 /* 16-bits audio data size */
#define AUDIO_OK 0
#define AUDIO_ERROR 1
#define AUDIO_TIMEOUT 2
#define DEFAULT_AUDIO_IN_FREQ I2S_AUDIOFREQ_16K
#define DEFAULT_AUDIO_IN_BIT_RESOLUTION 16
#define DEFAULT_AUDIO_IN_CHANNEL_NBR 1 /* Mono = 1, Stere
#define DEFAULT_AUDIO_IN_VOLUME 64
#define INTERNAL_BUFF_SIZE 128*DEFAULT_AUDIO_IN_FREQ/1
#define PCM_OUT_SIZE DEFAULT_AUDIO_IN_FREQ/1000
#define CHANNEL_DEMUX_MASK 0x55
#define DMA_MAX(_X_) (((_X_) <= DMA_MAX_SZE)? (_X_):DMA_M
```

Functions

uint8_t	**BSP_AUDIO_OUT_Init** (uint16_t OutputDevice, uint8_t Volume, uint32_t AudioFreq) Configures the audio peripherals.
uint8_t	**BSP_AUDIO_OUT_Play** (uint16_t *pBuffer, uint32_t Size) Starts playing audio stream from a data buffer for a determined size.
void	**BSP_AUDIO_OUT_ChangeBuffer** (uint16_t *pData, uint16_t Size) Sends n-Bytes on the I2S interface.
uint8_t	**BSP_AUDIO_OUT_Pause** (void) Pauses the audio file stream.
uint8_t	**BSP_AUDIO_OUT_Resume** (void) Resumes the audio file streaming.
uint8_t	**BSP_AUDIO_OUT_Stop** (uint32_t Option) Stops audio playing and Power down the Audio Codec.
uint8_t	**BSP_AUDIO_OUT_SetVolume** (uint8_t Volume) Controls the current audio volume level.
void	**BSP_AUDIO_OUT_SetFrequency** (uint32_t AudioFreq) Update the audio frequency.
uint8_t	**BSP_AUDIO_OUT_SetMute** (uint32_t Cmd) Enables or disables the MUTE mode by software.
uint8_t	**BSP_AUDIO_OUT_SetOutputMode** (uint8_t Output) Switch dynamically (while audio file is played) the output target (speaker or headphone).
void	**BSP_AUDIO_OUT_TransferComplete_CallBack** (void) Manages the DMA full Transfer complete event.
void	**BSP_AUDIO_OUT_HalfTransfer_CallBack** (void) Manages the DMA Half Transfer complete event.
void	**BSP_AUDIO_OUT_Error_CallBack** (void) Manages the DMA FIFO error event.
	BSP_AUDIO_OUT_ClockConfig (I2S_HandleTypeDef *hi2s,

void	uint32_t AudioFreq, void *Params) Clock Config.
void	**BSP_AUDIO_OUT_MspInit** (I2S_HandleTypeDef *hi2s, void *Params) AUDIO OUT I2S MSP Init.
void	**BSP_AUDIO_OUT_MspDeInit** (I2S_HandleTypeDef *hi2s, void *Params) De-Initializes BSP_AUDIO_OUT MSP.
uint8_t	**BSP_AUDIO_IN_Init** (uint32_t AudioFreq, uint32_t BitRes, uint32_t ChnlNbr) Initializes wave recording.
uint8_t	**BSP_AUDIO_IN_Record** (uint16_t *pData, uint32_t Size) Starts audio recording.
uint8_t	**BSP_AUDIO_IN_Stop** (void) Stops audio recording.
uint8_t	**BSP_AUDIO_IN_Pause** (void) Pauses the audio file stream.
uint8_t	**BSP_AUDIO_IN_Resume** (void) Resumes the audio file stream.
uint8_t	**BSP_AUDIO_IN_SetVolume** (uint8_t Volume) Controls the audio in volume level.
uint8_t	**BSP_AUDIO_IN_PDMToPCM** (uint16_t *PDMBuf, uint16_t *PCMBuf) Converts audio format from PDM to PCM.
void	**BSP_AUDIO_IN_TransferComplete_CallBack** (void) User callback when record buffer is filled.
void	**BSP_AUDIO_IN_HalfTransfer_CallBack** (void) Manages the DMA Half Transfer complete event.
void	**BSP_AUDIO_IN_Error_Callback** (void) Audio IN Error callback function.
void	**BSP_AUDIO_IN_ClockConfig** (I2S_HandleTypeDef *hi2s, uint32_t AudioFreq, void *Params) Audio In Clock Config.

void	**BSP_AUDIO_IN_MspInit** (I2S_HandleTypeDef *hi2s, void *Params) BSP AUDIO IN MSP Init.
void	**BSP_AUDIO_IN_MspDeInit** (I2S_HandleTypeDef *hi2s, void *Params) DeInitializes BSP_AUDIO_IN MSP.

Variables

__IO uint16_t	**AudioInVolume**

Detailed Description

This file contains the common defines and functions prototypes for **stm32f4_discovery_audio.c** driver.

Author:
MCD Application Team

Version:
V2.1.2

Date:
31-January-2017

Attention:

Definition in file **stm32f4_discovery_audio.h**.

STM32F4-Discovery BSP User Manual

Modules

Here is a list of all modules:

- **STM32F4 DISCOVERY AUDIO OUT Exported Constants**
- **STM32F4 DISCOVERY AUDIO Exported Variables**
- **STM32F4 DISCOVERY AUDIO Exported Macros**
- **STM32F4 DISCOVERY AUDIO OUT Exported Functions**
- **STM32F4 DISCOVERY AUDIO IN Exported Functions**

STM32F4-Discovery BSP User Manual

File List

Here is a list of all files with brief descriptions:

stm32f4_discovery.c [code]	This file provides set of functions to manage Le button available on STI Discovery Kit from STM
stm32f4_discovery.h [code]	This file contains defini STM32F4-Discovery K push-button hardware
stm32f4_discovery_accelerometer.c [code]	This file provides a set needed to manage the accelerometers availab STM32F4-Discovery K
stm32f4_discovery_accelerometer.h [code]	This file contains all the prototypes for the **stm32f4_discovery_a** firmware driver
stm32f4_discovery_audio.c [code]	This file provides the A the STM32F4-Discover
stm32f4_discovery_audio.h [code]	This file contains the c and functions prototype **stm32f4_discovery_a**

STM32F4-Discovery BSP User Manual

Directories

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

- Drivers
  - BSP
    - STM32F4-Discovery

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL

STM32F4 DISCOVERY

This file provides set of firmware functions to manage Leds and push-button available on STM32F4-Discovery Kit from STMicroelectronics.
More...

Modules

Detailed Description

This file provides set of firmware functions to manage Leds and push-button available on STM32F4-Discovery Kit from STMicroelectronics.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER

STM32F4 DISCOVERY

This file includes the motion sensor driver for ACCELEROMETER motion sensor devices. More...

Modules

Detailed Description

This file includes the motion sensor driver for ACCELEROMETER motion sensor devices.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO

STM32F4 DISCOVERY

This file includes the low layer audio driver available on STM32F4-Discovery discovery board. More...

Modules

Detailed Description

This file includes the low layer audio driver available on STM32F4-Discovery discovery board.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
LOW LEVEL Private Defines

STM32F4 DISCOVERY LOW LEVEL

Defines

#define	**__STM32F4_DISCO_BSP_VERSION_MAIN** (0x02) STM32F4 DISCO BSP Driver version number V2.1.2.
#define	**__STM32F4_DISCO_BSP_VERSION_SUB1** (0x01)
#define	**__STM32F4_DISCO_BSP_VERSION_SUB2** (0x02)
#define	**__STM32F4_DISCO_BSP_VERSION_RC** (0x00)
#define	**__STM32F4_DISCO_BSP_VERSION**

Define Documentation

#define __STM32F4_DISCO_BSP_VERSION

Value:

```
((__STM32F4_DISCO_BSP_VERSION_MAIN << 24)\
                                                |(__
STM32F4_DISCO_BSP_VERSION_SUB1 << 16)\
                                                |(__
STM32F4_DISCO_BSP_VERSION_SUB2 << 8 )\
                                                |(__
STM32F4_DISCO_BSP_VERSION_RC))
```

Definition at line **73** of file **stm32f4_discovery.c**.

Referenced by **BSP_GetVersion()**.

#define __STM32F4_DISCO_BSP_VERSION_MAIN (0x02)

STM32F4 DISCO BSP Driver version number V2.1.2.

[31:24] main version

Definition at line **69** of file **stm32f4_discovery.c**.

#define __STM32F4_DISCO_BSP_VERSION_RC (0x00)

[7:0] release candidate

Definition at line **72** of file **stm32f4_discovery.c**.

#define __STM32F4_DISCO_BSP_VERSION_SUB1 (0x01)

[23:16] sub1 version

Definition at line **70** of file **stm32f4_discovery.c**.

#define __STM32F4_DISCO_BSP_VERSION_SUB2 (0x02)

[15:8] sub2 version

Definition at line **71** of file **stm32f4_discovery.c**.

STM32F4-Discovery BSP User Manual

Defines

STM32F4 DISCOVERY LOW LEVEL BUS

STM32F4 DISCOVERY LOW LEVEL Exported Constants

Defines

#define	**DISCOVERY_SPIx** SPI1
#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** 0x1000 /*<! The value of the maximal
#define	**DISCOVERY_I2Cx** I2C1
#define	**DISCOVERY_I2Cx_CLK_ENABLE**() __HAL_RCC_I2C1_CLI
#define	**DISCOVERY_I2Cx_SCL_SDA_GPIO_CLK_ENABLE**() __HA
#define	**DISCOVERY_I2Cx_SCL_SDA_AF** GPIO_AF4_I2C1
#define	**DISCOVERY_I2Cx_SCL_SDA_GPIO_PORT** GPIOB
#define	**DISCOVERY_I2Cx_SCL_PIN** GPIO_PIN_6
#define	**DISCOVERY_I2Cx_SDA_PIN** GPIO_PIN_9
#define	**DISCOVERY_I2Cx_FORCE_RESET**() __HAL_RCC_I2C1_F
#define	**DISCOVERY_I2Cx_RELEASE_RESET**() __HAL_RCC_I2C1
#define	**DISCOVERY_I2Cx_EV_IRQn** I2C1_EV_IRQn
#define	**DISCOVERY_I2Cx_ER_IRQn** I2C1_ER_IRQn
#define	**I2Cx_TIMEOUT_MAX** 0x1000 /*<! The value of the maximal
#define	**READWRITE_CMD** ((uint8_t)0x80)
#define	**MULTIPLEBYTE_CMD** ((uint8_t)0x40)
#define	**DUMMY_BYTE** ((uint8_t)0x00)
#define	**ACCELERO_CS_LOW**() HAL_GPIO_WritePin(**ACCELERO_** **ACCELERO_CS_PIN**, GPIO_PIN_RESET)
#define	**ACCELERO_CS_HIGH**() HAL_GPIO_WritePin(**ACCELERO_** **ACCELERO_CS_PIN**, GPIO_PIN_SET)
#define	**ACCELERO_CS_PIN** GPIO_PIN_3 /* PE.03 */ ACCELEROMETER Interface pins.

```
#define ACCELERO_CS_GPIO_PORT   GPIOE /* GPIOE */
#define ACCELERO_CS_GPIO_CLK_ENABLE()   __HAL_RCC_GPIC
#define ACCELERO_CS_GPIO_CLK_DISABLE()   __HAL_RCC_GPI
#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_0 /* PE.00 */
#define ACCELERO_INT1_EXTI_IRQn   EXTI0_IRQn
#define ACCELERO_INT2_PIN   GPIO_PIN_1 /* PE.01 */
#define ACCELERO_INT2_EXTI_IRQn   EXTI1_IRQn
```

Define Documentation

#define ACCELERO_CS_GPIO_CLK_DISABLE ( ) __HAL_RCC_GI

Definition at line **232** of file **stm32f4_discovery.h**.

#define ACCELERO_CS_GPIO_CLK_ENABLE ( ) __HAL_RCC_GP

Definition at line **231** of file **stm32f4_discovery.h**.

Referenced by **ACCELERO_IO_Init()**.

#define ACCELERO_CS_GPIO_PORT GPIOE /* GPIOE */

Definition at line **230** of file **stm32f4_discovery.h**.

Referenced by **ACCELERO_IO_Init()**.

#define ACCELERO_CS_HIGH ( ) HAL_GPIO_WritePin(ACCELER

Definition at line **224** of file **stm32f4_discovery.h**.

Referenced by **ACCELERO_IO_Init()**, **ACCELERO_IO_Read()**, and **ACCELERO_IO_Write()**.

#define ACCELERO_CS_LOW ( ) HAL_GPIO_WritePin(ACCELERO

Definition at line **223** of file **stm32f4_discovery.h**.

Referenced by **ACCELERO_IO_Read()**, and **ACCELERO_IO_Write()**.

#define ACCELERO_CS_PIN GPIO_PIN_3 /* PE.03 */

ACCELEROMETER Interface pins.

Definition at line 229 of file stm32f4_discovery.h.

Referenced by ACCELERO_IO_Init().

#define ACCELERO_INT1_EXTI_IRQn EXTI0_IRQn

Definition at line 237 of file stm32f4_discovery.h.

#define ACCELERO_INT1_PIN GPIO_PIN_0 /* PE.00 */

Definition at line 236 of file stm32f4_discovery.h.

#define ACCELERO_INT2_EXTI_IRQn EXTI1_IRQn

Definition at line 239 of file stm32f4_discovery.h.

Referenced by ACCELERO_IO_ITConfig().

#define ACCELERO_INT2_PIN GPIO_PIN_1 /* PE.01 */

Definition at line 238 of file stm32f4_discovery.h.

Referenced by ACCELERO_IO_ITConfig().

#define ACCELERO_INT_GPIO_CLK_DISABLE ( ) __HAL_RCC_G

Definition at line 235 of file stm32f4_discovery.h.

#define ACCELERO_INT_GPIO_CLK_ENABLE ( ) __HAL_RCC_GI

Definition at line 234 of file stm32f4_discovery.h.

Referenced by ACCELERO_IO_ITConfig().

#define ACCELERO_INT_GPIO_PORT GPIOE /* GPIOE */

Definition at line 233 of file stm32f4_discovery.h.

Referenced by ACCELERO_IO_ITConfig().

#define DISCOVERY_I2Cx I2C1

Definition at line 191 of file stm32f4_discovery.h.

Referenced by I2Cx_Init().

#define DISCOVERY_I2Cx_CLK_ENABLE ( ) __HAL_RCC_I2C1_C

Definition at line 192 of file stm32f4_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_ER_IRQn I2C1_ER_IRQn

Definition at line 204 of file stm32f4_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_EV_IRQn I2C1_EV_IRQn

Definition at line **203** of file **stm32f4_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_FORCE_RESET ( ) __HAL_RCC_I2C1_

Definition at line **199** of file **stm32f4_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_RELEASE_RESET ( ) __HAL_RCC_I2C

Definition at line **200** of file **stm32f4_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_SCL_PIN GPIO_PIN_6

Definition at line **196** of file **stm32f4_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_SCL_SDA_AF GPIO_AF4_I2C1

Definition at line **194** of file **stm32f4_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_SCL_SDA_GPIO_CLK_ENABLE ( ) __H

Definition at line **193** of file **stm32f4_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_SCL_SDA_GPIO_PORT GPIOB

Definition at line **195** of file **stm32f4_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_I2Cx_SDA_PIN GPIO_PIN_9

Definition at line **197** of file **stm32f4_discovery.h**.

Referenced by **I2Cx_MspInit()**.

#define DISCOVERY_SPIx SPI1

Definition at line **166** of file **stm32f4_discovery.h**.

Referenced by **SPIx_Init()**.

#define DISCOVERY_SPIx_AF GPIO_AF5_SPI1

Definition at line **169** of file **stm32f4_discovery.h**.

Referenced by **SPIx_MspInit()**.

#define DISCOVERY_SPIx_CLK_ENABLE ( ) __HAL_RCC_SPI1_C

Definition at line **167** of file **stm32f4_discovery.h**.

Referenced by **SPIx_MspInit()**.

#define DISCOVERY_SPIx_GPIO_CLK_DISABLE ( ) __HAL_RCC_

Definition at line 171 of file stm32f4_discovery.h.

#define DISCOVERY_SPIx_GPIO_CLK_ENABLE ( ) __HAL_RCC_(

Definition at line 170 of file stm32f4_discovery.h.

Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_GPIO_PORT GPIOA /* GPIOA */

Definition at line 168 of file stm32f4_discovery.h.

Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_MISO_PIN GPIO_PIN_6 /* PA.06 */

Definition at line 173 of file stm32f4_discovery.h.

Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_MOSI_PIN GPIO_PIN_7 /* PA.07 */

Definition at line 174 of file stm32f4_discovery.h.

Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_SCK_PIN GPIO_PIN_5 /* PA.05 */

Definition at line 172 of file stm32f4_discovery.h.

Referenced by **SPIx_MspInit()**.

#define DUMMY_BYTE ((uint8_t)0x00)

Definition at line **220** of file **stm32f4_discovery.h**.

Referenced by **ACCELERO_IO_Read()**.

#define I2Cx_TIMEOUT_MAX 0x1000 /*<! The value of the maxima

Definition at line **211** of file **stm32f4_discovery.h**.

#define MULTIPLEBYTE_CMD ((uint8_t)0x40)

Definition at line **218** of file **stm32f4_discovery.h**.

Referenced by **ACCELERO_IO_Read()**, and **ACCELERO_IO_Write()**.

#define READWRITE_CMD ((uint8_t)0x80)

Definition at line **216** of file **stm32f4_discovery.h**.

Referenced by **ACCELERO_IO_Read()**.

#define SPIx_TIMEOUT_MAX 0x1000 /*<! The value of the maxima

Definition at line **181** of file **stm32f4_discovery.h**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER Exported Types

STM32F4 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 **69** of file **stm32f4_discovery_accelerometer.h**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
LOW LEVEL Private Functions

STM32F4 DISCOVERY LOW LEVEL

Functions

static void	**I2Cx_Init** (void) Configures I2C interface.
static void	**I2Cx_MspInit** (void) I2C MSP Initialization.
static void	**SPIx_Init** (void) SPIx Bus initialization.
static void	**SPIx_MspInit** (void) SPI MSP Init.
static void	**SPIx_Error** (void) SPIx error treatment function.
void	**ACCELERO_IO_Init** (void) Configures the Accelerometer SPI interface.
void	**AUDIO_IO_Init** (void) Initializes Audio low level.
void	**AUDIO_IO_DeInit** (void) DeInitializes Audio low level.

Function Documentation

void ACCELERO_IO_Init (void)

Configures the Accelerometer SPI interface.

Definition at line 510 of file stm32f4_discovery.c.

References ACCELERO_CS_GPIO_CLK_ENABLE, ACCELERO_CS_GPIO_PORT, ACCELERO_CS_HIGH, ACCELERO_CS_PIN, and SPIx_Init().

void AUDIO_IO_DeInit (void)

DeInitializes Audio low level.

Definition at line 660 of file stm32f4_discovery.c.

void AUDIO_IO_Init (void)

Initializes Audio low level.

Definition at line 628 of file stm32f4_discovery.c.

References AUDIO_RESET_GPIO, AUDIO_RESET_GPIO_CLK_ENABLE, AUDIO_RESET_PIN, and I2Cx_Init().

static void I2Cx_Init (void) [static]

Configures I2C interface.

Definition at line 392 of file stm32f4_discovery.c.

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

Referenced by **AUDIO_IO_Init()**, and **I2Cx_Error()**.

static void I2Cx_MspInit (void) [static]

I2C MSP Initialization.

Definition at line **468** of file **stm32f4_discovery.c**.

References **DISCOVERY_I2Cx_CLK_ENABLE**, **DISCOVERY_I2Cx_ER_IRQn**, **DISCOVERY_I2Cx_EV_IRQn**, **DISCOVERY_I2Cx_FORCE_RESET**, **DISCOVERY_I2Cx_RELEASE_RESET**, **DISCOVERY_I2Cx_SCL_PIN**, **DISCOVERY_I2Cx_SCL_SDA_AF**, **DISCOVERY_I2Cx_SCL_SDA_GPIO_CLK_ENABLE**, **DISCOVERY_I2Cx_SCL_SDA_GPIO_PORT**, and **DISCOVERY_I2Cx_SDA_PIN**.

Referenced by **I2Cx_Init()**.

static void SPIx_Error (void) [static]

SPIx error treatment function.

Definition at line **357** of file **stm32f4_discovery.c**.

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

Referenced by **SPIx_WriteRead()**.

static void SPIx_Init (void) [static]

SPIx Bus initialization.

Definition at line **311** of file **stm32f4_discovery.c**.

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

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

static void SPIx_MspInit (void) [static]

SPI MSP Init.

Definition at line **369** of file **stm32f4_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()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL BUS Functions

STM32F4 DISCOVERY LOW LEVEL

Functions

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	**I2Cx_WriteData** (uint8_t Addr, uint8_t Reg, uint8_t Value) Write a value in a register of the device through BUS.
static uint8_t	**I2Cx_ReadData** (uint8_t Addr, uint8_t Reg) Read a register of the device through BUS.
static void	**I2Cx_Error** (uint8_t Addr) Manages error callback by re-initializing I2C.
void	**ACCELERO_IO_ITConfig** (void) Configures the Accelerometer INT2.
void	**ACCELERO_IO_Write** (uint8_t *pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite) Writes one byte to the Accelerometer.
void	**ACCELERO_IO_Read** (uint8_t *pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead) Reads a block of data from the Accelerometer.
void	**AUDIO_IO_Write** (uint8_t Addr, uint8_t Reg, uint8_t Value) Writes a single data.
uint8_t	**AUDIO_IO_Read** (uint8_t Addr, uint8_t Reg) Reads a single data.

Function Documentation

void ACCELERO_IO_ITConfig (void)

Configures the Accelerometer INT2.

EXTI0 is already used by user button so INT1 is not configured here.

Definition at line 535 of file stm32f4_discovery.c.

References ACCELERO_INT2_EXTI_IRQn, ACCELERO_INT2_PIN, ACCELERO_INT_GPIO_CLK_ENABLE, and ACCELERO_INT_GPIO_PORT.

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

Reads a block of data from the Accelerometer.

Parameters:

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

Definition at line 594 of file stm32f4_discovery.c.

References ACCELERO_CS_HIGH, ACCELERO_CS_LOW, DUMMY_BYTE, MULTIPLEBYTE_CMD, READWRITE_CMD, and SPIx_WriteRead().

void ACCELERO_IO_Write (uint8_t * pBuffer,
uint8_t WriteAddr,
uint16_t NumByteToWrite
)

Writes one byte to the Accelerometer.

Parameters:

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

Definition at line **560** of file **stm32f4_discovery.c**.

References **ACCELERO_CS_HIGH**, **ACCELERO_CS_LOW**, **MULTIPLEBYTE_CMD**, and **SPIx_WriteRead()**.

uint8_t AUDIO_IO_Read (uint8_t Addr,
uint8_t Reg
)

Reads a single data.

Parameters:

Addr,:	I2C address
Reg,:	Reg address

Return values:

Data	to be read

Definition at line **682** of file **stm32f4_discovery.c**.

References I2Cx_ReadData().

void AUDIO_IO_Write (uint8_t Addr,
uint8_t Reg,
uint8_t Value
)

Writes a single data.

Parameters:

Addr,: I2C address
Reg,: Reg address
Value,: Data to be written

Definition at line 671 of file stm32f4_discovery.c.

References I2Cx_WriteData().

static void I2Cx_Error (uint8_t Addr) [static]

Manages error callback by re-initializing I2C.

Parameters:

Addr,: I2C Address

Definition at line 456 of file stm32f4_discovery.c.

References I2cHandle, and I2Cx_Init().

Referenced by I2Cx_ReadData(), and I2Cx_WriteData().

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

)

Read a register of the device through BUS.

Parameters:

Addr,: Device address on BUS

Reg,: The target register address to read

Return values:

HAL status

Definition at line **436** of file **stm32f4_discovery.c**.

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

Referenced by **AUDIO_IO_Read()**.

static void I2Cx_WriteData (uint8_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:

HAL status

Definition at line **416** of file **stm32f4_discovery.c**.

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

Referenced by **AUDIO_IO_Write()**.

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 **340** of file **stm32f4_discovery.c**.

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

Referenced by **ACCELERO_IO_Read()**, and **ACCELERO_IO_Write()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER Private Variables

STM32F4 DISCOVERY ACCELEROMETER

Variables

static ACCELERO_DrvTypeDef * **AcceleroDrv**

Variable Documentation

ACCELERO_DrvTypeDef* AcceleroDrv [static]

Definition at line 80 of file stm32f4_discovery_accelerometer.c.

Referenced by BSP_ACCELERO_Click_ITClear(), BSP_ACCELERO_Click_ITConfig(), BSP_ACCELERO_GetXYZ(), BSP_ACCELERO_Init(), BSP_ACCELERO_ReadID(), and BSP_ACCELERO_Reset().

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
AUDIO OUT Exported Constants

STM32F4 DISCOVERY AUDIO

Defines

```
#define I2S3 SPI3
#define I2S3_CLK_ENABLE() __HAL_RCC_SPI3_CLK_ENABLE()
#define I2S3_CLK_DISABLE() __HAL_RCC_SPI3_CLK_DISABLE()
#define I2S3_SCK_SD_WS_AF GPIO_AF6_SPI3
#define I2S3_SCK_SD_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_
#define I2S3_MCK_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_EN
#define I2S3_WS_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENA
#define I2S3_WS_PIN GPIO_PIN_4
#define I2S3_SCK_PIN GPIO_PIN_10
#define I2S3_SD_PIN GPIO_PIN_12
#define I2S3_MCK_PIN GPIO_PIN_7
#define I2S3_SCK_SD_GPIO_PORT GPIOC
#define I2S3_WS_GPIO_PORT GPIOA
#define I2S3_MCK_GPIO_PORT GPIOC
#define I2S3_DMAx_CLK_ENABLE() __HAL_RCC_DMA1_CLK_EN
#define I2S3_DMAx_CLK_DISABLE() __HAL_RCC_DMA1_CLK_DI
#define I2S3_DMAx_STREAM DMA1_Stream7
#define I2S3_DMAx_CHANNEL DMA_CHANNEL_0
#define I2S3_DMAx_IRQ DMA1_Stream7_IRQn
#define I2S3_DMAx_PERIPH_DATA_SIZE DMA_PDATAALIGN_HAL
#define I2S3_DMAx_MEM_DATA_SIZE DMA_MDATAALIGN_HALFW
#define DMA_MAX_SZE 0xFFFF
#define I2S3_IRQHandler DMA1_Stream7_IRQHandler
#define AUDIO_OUT_IRQ_PREPRIO 0x0E /* Select the preemption
#define I2S2 SPI2
#define I2S2_CLK_ENABLE() __HAL_RCC_SPI2_CLK_ENABLE()
#define I2S2_CLK_DISABLE() __HAL_RCC_SPI2_CLK_DISABLE()
#define I2S2_SCK_PIN GPIO_PIN_10
#define I2S2_SCK_GPIO_PORT GPIOB
#define I2S2_SCK_GPIO_CLK_ENABLE() __HAL_RCC_GPIOB_CL
#define I2S2_SCK_AF GPIO_AF5_SPI2
```

```
#define I2S2_MOSI_PIN GPIO_PIN_3
#define I2S2_MOSI_GPIO_PORT GPIOC
#define I2S2_MOSI_GPIO_CLK_ENABLE() __HAL_RCC_GPIOC_C
#define I2S2_MOSI_AF GPIO_AF5_SPI2
#define I2S2_DMAx_CLK_ENABLE() __HAL_RCC_DMA1_CLK_EN
#define I2S2_DMAx_CLK_DISABLE() __HAL_RCC_DMA1_CLK_DI
#define I2S2_DMAx_STREAM DMA1_Stream3
#define I2S2_DMAx_CHANNEL DMA_CHANNEL_0
#define I2S2_DMAx_IRQ DMA1_Stream3_IRQn
#define I2S2_DMAx_PERIPH_DATA_SIZE DMA_PDATAALIGN_HAL
#define I2S2_DMAx_MEM_DATA_SIZE DMA_MDATAALIGN_HALFW
#define I2S2_IRQHandler DMA1_Stream3_IRQHandler
#define AUDIO_IN_IRQ_PREPRIO 0x0F /* Select the preemption pri
#define AUDIODATA_SIZE 2 /* 16-bits audio data size */
#define AUDIO_OK 0
#define AUDIO_ERROR 1
#define AUDIO_TIMEOUT 2
#define DEFAULT_AUDIO_IN_FREQ I2S_AUDIOFREQ_16K
#define DEFAULT_AUDIO_IN_BIT_RESOLUTION 16
#define DEFAULT_AUDIO_IN_CHANNEL_NBR 1 /* Mono = 1, Stere
#define DEFAULT_AUDIO_IN_VOLUME 64
#define INTERNAL_BUFF_SIZE 128*DEFAULT_AUDIO_IN_FREQ/1
#define PCM_OUT_SIZE DEFAULT_AUDIO_IN_FREQ/1000
#define CHANNEL_DEMUX_MASK 0x55
```

Define Documentation

#define AUDIO_ERROR 1

Definition at line 152 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_Record(), BSP_AUDIO_IN_Stop(), BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_Pause(), BSP_AUDIO_OUT_Play(), BSP_AUDIO_OUT_Resume(), BSP_AUDIO_OUT_SetMute(), BSP_AUDIO_OUT_SetOutputMode(), BSP_AUDIO_OUT_SetVolume(), BSP_AUDIO_OUT_Stop(), I2S2_Init(), and I2S3_Init().

#define AUDIO_IN_IRQ_PREPRIO 0x0F /* Select the preemption p

Definition at line 142 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspInit().

#define AUDIO_OK 0

Definition at line 151 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_Init(), BSP_AUDIO_IN_Pause(), BSP_AUDIO_IN_PDMToPCM(), BSP_AUDIO_IN_Record(), BSP_AUDIO_IN_Resume(), BSP_AUDIO_IN_SetVolume(), BSP_AUDIO_IN_Stop(), BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_Pause(), BSP_AUDIO_OUT_Play(), BSP_AUDIO_OUT_Resume(), BSP_AUDIO_OUT_SetMute(), BSP_AUDIO_OUT_SetOutputMode(), BSP_AUDIO_OUT_SetVolume(), BSP_AUDIO_OUT_Stop(), I2S2_Init(), and I2S3_Init().

#define AUDIO_OUT_IRQ_PREPRIO 0x0E /* Select the preemption

Definition at line 111 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspInit().

#define AUDIO_TIMEOUT 2

Definition at line 153 of file stm32f4_discovery_audio.h.

#define AUDIODATA_SIZE 2 /* 16-bits audio data size */

Definition at line 148 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_Play().

#define CHANNEL_DEMUX_MASK 0x55

Definition at line 165 of file stm32f4_discovery_audio.h.

#define DEFAULT_AUDIO_IN_BIT_RESOLUTION 16

Definition at line 157 of file stm32f4_discovery_audio.h.

#define DEFAULT_AUDIO_IN_CHANNEL_NBR 1 /* Mono = 1, Stere

Definition at line 158 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_PDMToPCM().

#define DEFAULT_AUDIO_IN_FREQ I2S_AUDIOFREQ_16K

Definition at line **156** of file **stm32f4_discovery_audio.h**.

#define DEFAULT_AUDIO_IN_VOLUME 64

Definition at line **159** of file **stm32f4_discovery_audio.h**.

#define DMA_MAX_SZE 0xFFFF

Definition at line **106** of file **stm32f4_discovery_audio.h**.

#define I2S2 SPI2

Definition at line **117** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_IN_Init()**, **BSP_AUDIO_IN_MspDeInit()**, **BSP_AUDIO_IN_MspInit()**, **HAL_I2S_ErrorCallback()**, and **I2S2_Init()**.

#define I2S2_CLK_DISABLE ( ) __HAL_RCC_SPI2_CLK_DISABLE

Definition at line **119** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_IN_MspDeInit()**.

#define I2S2_CLK_ENABLE ( ) __HAL_RCC_SPI2_CLK_ENABLE(

Definition at line **118** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_IN_MspInit()**.

#define I2S2_DMAx_CHANNEL DMA_CHANNEL_0

Definition at line 134 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspInit().

#define I2S2_DMAx_CLK_DISABLE ( ) __HAL_RCC_DMA1_CLK_

Definition at line 132 of file stm32f4_discovery_audio.h.

#define I2S2_DMAx_CLK_ENABLE ( ) __HAL_RCC_DMA1_CLK_E

Definition at line 131 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspInit().

#define I2S2_DMAx_IRQ DMA1_Stream3_IRQn

Definition at line 135 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspDeInit(), and BSP_AUDIO_IN_MspInit().

#define I2S2_DMAx_MEM_DATA_SIZE DMA_MDATAALIGN_HALF

Definition at line 137 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspInit().

#define I2S2_DMAx_PERIPH_DATA_SIZE DMA_PDATAALIGN_HA

Definition at line 136 of file stm32f4_discovery_audio.h.

Referenced by **BSP_AUDIO_IN_MspInit()**.

#define I2S2_DMAx_STREAM DMA1_Stream3

Definition at line **133** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_IN_MspInit()**.

#define I2S2_IRQHandler DMA1_Stream3_IRQHandler

Definition at line **139** of file **stm32f4_discovery_audio.h**.

#define I2S2_MOSI_AF GPIO_AF5_SPI2

Definition at line **128** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_IN_MspInit()**.

#define I2S2_MOSI_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOC_

Definition at line **127** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_IN_MspInit()**.

#define I2S2_MOSI_GPIO_PORT GPIOC

Definition at line **126** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_IN_MspDeInit()**, and **BSP_AUDIO_IN_MspInit()**.

#define I2S2_MOSI_PIN GPIO_PIN_3

Definition at line 125 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspDeInit(), and BSP_AUDIO_IN_MspInit().

#define I2S2_SCK_AF GPIO_AF5_SPI2

Definition at line 123 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspInit().

#define I2S2_SCK_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOB_(

Definition at line 122 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspInit().

#define I2S2_SCK_GPIO_PORT GPIOB

Definition at line 121 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspDeInit(), and BSP_AUDIO_IN_MspInit().

#define I2S2_SCK_PIN GPIO_PIN_10

Definition at line 120 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_MspDeInit(), and BSP_AUDIO_IN_MspInit().

#define I2S3 SPI3

Definition at line 83 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_MspDeInit(), BSP_AUDIO_OUT_MspInit(), HAL_I2S_ErrorCallback(), HAL_I2S_TxCpltCallback(), HAL_I2S_TxHalfCpltCallback(), and I2S3_Init().

#define I2S3_CLK_DISABLE ( ) __HAL_RCC_SPI3_CLK_DISABLE

Definition at line 85 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspDeInit().

#define I2S3_CLK_ENABLE ( ) __HAL_RCC_SPI3_CLK_ENABLE(

Definition at line 84 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspInit().

#define I2S3_DMAx_CHANNEL DMA_CHANNEL_0

Definition at line 102 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspInit().

#define I2S3_DMAx_CLK_DISABLE ( ) __HAL_RCC_DMA1_CLK_

Definition at line 100 of file stm32f4_discovery_audio.h.

#define I2S3_DMAx_CLK_ENABLE ( ) __HAL_RCC_DMA1_CLK_E

Definition at line **99** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_OUT_MspInit()**.

#define I2S3_DMAx_IRQ DMA1_Stream7_IRQn

Definition at line **103** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_OUT_MspDeInit()**, and **BSP_AUDIO_OUT_MspInit()**.

#define I2S3_DMAx_MEM_DATA_SIZE DMA_MDATAALIGN_HALF

Definition at line **105** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_OUT_MspInit()**.

#define I2S3_DMAx_PERIPH_DATA_SIZE DMA_PDATAALIGN_HA

Definition at line **104** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_OUT_MspInit()**.

#define I2S3_DMAx_STREAM DMA1_Stream7

Definition at line **101** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_OUT_MspInit()**.

#define I2S3_IRQHandler DMA1_Stream7_IRQHandler

Definition at line **108** of file **stm32f4_discovery_audio.h**.

#define I2S3_MCK_CLK_ENABLE ( ) __HAL_RCC_GPIOC_CLK_E

Definition at line 88 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspInit().

#define I2S3_MCK_GPIO_PORT GPIOC

Definition at line 96 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspDeInit(), and BSP_AUDIO_OUT_MspInit().

#define I2S3_MCK_PIN GPIO_PIN_7

Definition at line 93 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspDeInit(), and BSP_AUDIO_OUT_MspInit().

#define I2S3_SCK_PIN GPIO_PIN_10

Definition at line 91 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspDeInit(), and BSP_AUDIO_OUT_MspInit().

#define I2S3_SCK_SD_CLK_ENABLE ( ) __HAL_RCC_GPIOC_CL

Definition at line 87 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspInit().

#define I2S3_SCK_SD_GPIO_PORT GPIOC

Definition at line 94 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspDeInit(), and BSP_AUDIO_OUT_MspInit().

#define I2S3_SCK_SD_WS_AF GPIO_AF6_SPI3

Definition at line 86 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspInit().

#define I2S3_SD_PIN GPIO_PIN_12

Definition at line 92 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspDeInit(), and BSP_AUDIO_OUT_MspInit().

#define I2S3_WS_CLK_ENABLE ( ) __HAL_RCC_GPIOA_CLK_EN

Definition at line 89 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspInit().

#define I2S3_WS_GPIO_PORT GPIOA

Definition at line 95 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspDeInit(), and BSP_AUDIO_OUT_MspInit().

#define I2S3_WS_PIN GPIO_PIN_4

Definition at line 90 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_OUT_MspDeInit(), and BSP_AUDIO_OUT_MspInit().

#define INTERNAL_BUFF_SIZE 128*DEFAULT_AUDIO_IN_FREQ/1

Definition at line 162 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_PDMToPCM().

#define PCM_OUT_SIZE DEFAULT_AUDIO_IN_FREQ/1000

Definition at line 164 of file stm32f4_discovery_audio.h.

Referenced by BSP_AUDIO_IN_PDMToPCM().

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
LOW LEVEL Exported Constants

STM32F4 DISCOVERY LOW LEVEL

Modules

STM32F4 DISCOVERY LOW LEVEL LED

Define for STM32F4_DISCOVERY board.

STM32F4 DISCOVERY LOW LEVEL BUTTON

STM32F4 DISCOVERY LOW LEVEL BUS

Defines

#define	**AUDIO_I2C_ADDRESS** 0x94 AUDIO I2C Interface pins.
#define	**AUDIO_RESET_GPIO_CLK_ENABLE**() __HAL_RCC_GPIO
#define	**AUDIO_RESET_PIN** GPIO_PIN_4
#define	**AUDIO_RESET_GPIO** GPIOD

Define Documentation

#define AUDIO_I2C_ADDRESS 0x94

AUDIO I2C Interface pins.

Definition at line 249 of file stm32f4_discovery.h.

Referenced by BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_Pause(), BSP_AUDIO_OUT_Play(), BSP_AUDIO_OUT_Resume(), BSP_AUDIO_OUT_SetMute(), BSP_AUDIO_OUT_SetOutputMode(), BSP_AUDIO_OUT_SetVolume(), and BSP_AUDIO_OUT_Stop().

#define AUDIO_RESET_GPIO GPIOD

Definition at line 254 of file stm32f4_discovery.h.

Referenced by AUDIO_IO_Init(), and BSP_AUDIO_OUT_Stop().

#define AUDIO_RESET_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPI

Definition at line 252 of file stm32f4_discovery.h.

Referenced by AUDIO_IO_Init().

#define AUDIO_RESET_PIN GPIO_PIN_4

Definition at line 253 of file stm32f4_discovery.h.

Referenced by AUDIO_IO_Init(), and BSP_AUDIO_OUT_Stop().

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP
User Manual by  1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO Private Variables

STM32F4 DISCOVERY AUDIO

Variables

static AUDIO_DrvTypeDef *	**pAudioDrv**
I2S_HandleTypeDef	**hAudioOutI2s**
I2S_HandleTypeDef	**hAudioInI2s**
PDMFilter_InitStruct	**Filter** [**DEFAULT_AUDIO_IN_CHANNEL_NBR**]
__IO uint16_t	**AudioInVolume** = **DEFAULT_AUDIO_IN_VOLUME**

Variable Documentation

__IO uint16_t AudioInVolume = DEFAULT_AUDIO_IN_VOLUME

Definition at line 186 of file stm32f4_discovery_audio.c.

Referenced by BSP_AUDIO_IN_PDMToPCM(), and BSP_AUDIO_IN_SetVolume().

PDMFilter_InitStruct Filter[DEFAULT_AUDIO_IN_CHANNEL_NBR]

Definition at line 185 of file stm32f4_discovery_audio.c.

Referenced by BSP_AUDIO_IN_PDMToPCM(), and PDMDecoder_Init().

I2S_HandleTypeDef hAudioInI2s

Definition at line 183 of file stm32f4_discovery_audio.c.

Referenced by BSP_AUDIO_IN_Init(), BSP_AUDIO_IN_Pause(), BSP_AUDIO_IN_Record(), BSP_AUDIO_IN_Resume(), BSP_AUDIO_IN_Stop(), and I2S2_Init().

I2S_HandleTypeDef hAudioOutI2s

Definition at line 180 of file stm32f4_discovery_audio.c.

Referenced by BSP_AUDIO_OUT_ChangeBuffer(), BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_Pause(), BSP_AUDIO_OUT_Play(), BSP_AUDIO_OUT_Resume(), BSP_AUDIO_OUT_SetFrequency(), BSP_AUDIO_OUT_Stop(), and I2S3_Init().

AUDIO_DrvTypeDef* pAudioDrv `[static]`

Definition at line **179** of file **stm32f4_discovery_audio.c**.

Referenced by **BSP_AUDIO_OUT_Init()**, **BSP_AUDIO_OUT_Pause()**, **BSP_AUDIO_OUT_Play()**, **BSP_AUDIO_OUT_Resume()**, **BSP_AUDIO_OUT_SetMute()**, **BSP_AUDIO_OUT_SetOutputMode()**, **BSP_AUDIO_OUT_SetVolume()**, and **BSP_AUDIO_OUT_Stop()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO Exported Variables

STM32F4 DISCOVERY AUDIO

Variables

__IO uint16_t	**AudioInVolume**

Variable Documentation

__IO uint16_t AudioInVolume

Definition at line **186** of file **stm32f4_discovery_audio.c**.

Referenced by **BSP_AUDIO_IN_PDMToPCM()**, and **BSP_AUDIO_IN_SetVolume()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER Private Functions

STM32F4 DISCOVERY ACCELEROMETER

Functions

uint8_t	**BSP_ACCELERO_Init** (void) Setx Accelerometer Initialization.
uint8_t	**BSP_ACCELERO_ReadID** (void) Read ID of Accelerometer component.
void	**BSP_ACCELERO_Reset** (void) Reboot memory content of Accelerometer.
void	**BSP_ACCELERO_Click_ITConfig** (void) Configure Accelerometer click IT.
void	**BSP_ACCELERO_Click_ITClear** (void) Clear Accelerometer click IT.
void	**BSP_ACCELERO_GetXYZ** (int16_t *pDataXYZ) Get XYZ axes acceleration.

Function Documentation

void BSP_ACCELERO_Click_ITClear (void)

Clear Accelerometer click IT.

Definition at line 219 of file stm32f4_discovery_accelerometer.c.

References AcceleroDrv.

void BSP_ACCELERO_Click_ITConfig (void)

Configure Accelerometer click IT.

Definition at line 208 of file stm32f4_discovery_accelerometer.c.

References AcceleroDrv.

void BSP_ACCELERO_GetXYZ (int16_t * pDataXYZ)

Get XYZ axes acceleration.

Parameters:

pDataXYZ,:	Pointer to 3 angular acceleration axes. pDataXYZ[0] = X axis, pDataXYZ[1] = Y axis, pDataXYZ[2] = Z axis

Definition at line 232 of file stm32f4_discovery_accelerometer.c.

References AcceleroDrv.

uint8_t BSP_ACCELERO_Init (void)

Setx Accelerometer Initialization.

Return values:

ACCELERO_OK if no problem during initialization

Definition at line **100** of file **stm32f4_discovery_accelerometer.c**.

References **ACCELERO_ERROR**, **ACCELERO_OK**, and **AcceleroDrv**.

uint8_t BSP_ACCELERO_ReadID (void)

Read ID of Accelerometer component.

Return values:

ID

Definition at line **183** of file **stm32f4_discovery_accelerometer.c**.

References **AcceleroDrv**.

void BSP_ACCELERO_Reset (void)

Reboot memory content of Accelerometer.

Definition at line **197** of file **stm32f4_discovery_accelerometer.c**.

References **AcceleroDrv**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY ACCELEROMETER Exported Functions

STM32F4 DISCOVERY ACCELEROMETER

Functions

uint8_t	**BSP_ACCELERO_Init** (void) Setx Accelerometer Initialization.
uint8_t	**BSP_ACCELERO_ReadID** (void) Read ID of Accelerometer component.
void	**BSP_ACCELERO_Reset** (void) Reboot memory content of Accelerometer.
void	**BSP_ACCELERO_Click_ITConfig** (void) Configure Accelerometer click IT.
void	**BSP_ACCELERO_Click_ITClear** (void) Clear Accelerometer click IT.
void	**BSP_ACCELERO_GetXYZ** (int16_t *pDataXYZ) Get XYZ axes acceleration.

Function Documentation

void BSP_ACCELERO_Click_ITClear (void)

Clear Accelerometer click IT.

Definition at line 219 of file stm32f4_discovery_accelerometer.c.

References AcceleroDrv.

void BSP_ACCELERO_Click_ITConfig (void)

Configure Accelerometer click IT.

Definition at line 208 of file stm32f4_discovery_accelerometer.c.

References AcceleroDrv.

void BSP_ACCELERO_GetXYZ (int16_t * pDataXYZ)

Get XYZ axes acceleration.

Parameters:

pDataXYZ,:	Pointer to 3 angular acceleration axes. pDataXYZ[0] = X axis, pDataXYZ[1] = Y axis, pDataXYZ[2] = Z axis

Definition at line 232 of file stm32f4_discovery_accelerometer.c.

References AcceleroDrv.

uint8_t BSP_ACCELERO_Init (void)

Setx Accelerometer Initialization.

Return values:

ACCELERO_OK if no problem during initialization

Definition at line **100** of file **stm32f4_discovery_accelerometer.c**.

References **ACCELERO_ERROR**, **ACCELERO_OK**, and **AcceleroDrv**.

uint8_t BSP_ACCELERO_ReadID (void)

Read ID of Accelerometer component.

Return values:

ID

Definition at line **183** of file **stm32f4_discovery_accelerometer.c**.

References **AcceleroDrv**.

void BSP_ACCELERO_Reset (void)

Reboot memory content of Accelerometer.

Definition at line **197** of file **stm32f4_discovery_accelerometer.c**.

References **AcceleroDrv**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
AUDIO IN Private Functions

STM32F4 DISCOVERY AUDIO

Functions

uint8_t	**BSP_AUDIO_IN_Init** (uint32_t AudioFreq, uint32_t BitRes, uint32_t ChnlNbr) Initializes wave recording.
uint8_t	**BSP_AUDIO_IN_Record** (uint16_t *pbuf, uint32_t size) Starts audio recording.
uint8_t	**BSP_AUDIO_IN_Stop** (void) Stops audio recording.
uint8_t	**BSP_AUDIO_IN_Pause** (void) Pauses the audio file stream.
uint8_t	**BSP_AUDIO_IN_Resume** (void) Resumes the audio file stream.
uint8_t	**BSP_AUDIO_IN_SetVolume** (uint8_t Volume) Controls the audio in volume level.
uint8_t	**BSP_AUDIO_IN_PDMToPCM** (uint16_t *PDMBuf, uint16_t *PCMBuf) Converts audio format from PDM to PCM.
void	**HAL_I2S_RxCpltCallback** (I2S_HandleTypeDef *hi2s) Rx Transfer completed callbacks.
void	**HAL_I2S_RxHalfCpltCallback** (I2S_HandleTypeDef *hi2s) Rx Half Transfer completed callbacks.
__weak void	**BSP_AUDIO_IN_ClockConfig** (I2S_HandleTypeDef *hi2s, uint32_t AudioFreq, void *Params) Audio In Clock Config.
__weak void	**BSP_AUDIO_IN_MspInit** (I2S_HandleTypeDef *hi2s, void *Params) BSP AUDIO IN MSP Init.
__weak void	**BSP_AUDIO_IN_MspDeInit** (I2S_HandleTypeDef *hi2s, void *Params) DeInitializes BSP_AUDIO_IN MSP.

__weak void	**BSP_AUDIO_IN_TransferComplete_CallBack** (void) User callback when record buffer is filled.
__weak void	**BSP_AUDIO_IN_HalfTransfer_CallBack** (void) Manages the DMA Half Transfer complete event.
__weak void	**BSP_AUDIO_IN_Error_Callback** (void) Audio IN Error callback function.
static void	**PDMDecoder_Init** (uint32_t AudioFreq, uint32_t ChnlNbr) Initialize the PDM library.
static uint8_t	**I2S2_Init** (uint32_t AudioFreq) Initializes the Audio Codec audio interface (I2S)

Function Documentation

__weak void BSP_AUDIO_IN_ClockConfig (I2S_HandleTypeDef * h
uint32_t A
void * P
)

Audio In Clock Config.

Parameters:

hi2s,:	I2S handle
AudioFreq,:	Audio frequency used to record the audio stream.
Params	: pointer on additional configuration parameters, can be NULL.

Note:
This API is called by BSP_AUDIO_IN_Init() Being __weak it can be overwritten by the application

Definition at line 866 of file stm32f4_discovery_audio.c.

Referenced by BSP_AUDIO_IN_Init().

__weak void BSP_AUDIO_IN_Error_Callback (void)

Audio IN Error callback function.

Definition at line 1018 of file stm32f4_discovery_audio.c.

Referenced by HAL_I2S_ErrorCallback().

__weak void BSP_AUDIO_IN_HalfTransfer_CallBack (void)

Manages the DMA Half Transfer complete event.

Definition at line **1008** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_RxHalfCpltCallback()**.

uint8_t BSP_AUDIO_IN_Init (uint32_t AudioFreq,
uint32_t BitRes,
uint32_t ChnlNbr
)

Initializes wave recording.

Parameters:

AudioFreq,:	Audio frequency to be configured for the I2S peripheral.
BitRes,:	Audio frequency to be configured for the I2S peripheral.
ChnlNbr,:	Audio frequency to be configured for the I2S peripheral.

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **704** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, **BSP_AUDIO_IN_ClockConfig()**, **BSP_AUDIO_IN_MspInit()**, **hAudioInI2s**, **I2S2**, **I2S2_Init()**, and **PDMDecoder_Init()**.

__weak void BSP_AUDIO_IN_MspDeInit (I2S_HandleTypeDef * hi2s
void * Par
)

DeInitializes BSP_AUDIO_IN MSP.

Parameters:

hi2s,: I2S handle

Params : pointer on additional configuration parameters, can be NULL.

Definition at line **966** of file **stm32f4_discovery_audio.c**.

References **I2S2**, **I2S2_CLK_DISABLE**, **I2S2_DMAx_IRQ**, **I2S2_MOSI_GPIO_PORT**, **I2S2_MOSI_PIN**, **I2S2_SCK_GPIO_PORT**, and **I2S2_SCK_PIN**.

__weak void BSP_AUDIO_IN_MspInit (I2S_HandleTypeDef * hi2s, void * Param)

BSP AUDIO IN MSP Init.

Parameters:

hi2s,: I2S handle

Params : pointer on additional configuration parameters, can be NULL.

Definition at line **900** of file **stm32f4_discovery_audio.c**.

References **AUDIO_IN_IRQ_PREPRIO**, **I2S2**, **I2S2_CLK_ENABLE**, **I2S2_DMAx_CHANNEL**, **I2S2_DMAx_CLK_ENABLE**, **I2S2_DMAx_IRQ**, **I2S2_DMAx_MEM_DATA_SIZE**, **I2S2_DMAx_PERIPH_DATA_SIZE**, **I2S2_DMAx_STREAM**, **I2S2_MOSI_AF**, **I2S2_MOSI_GPIO_CLK_ENABLE**, **I2S2_MOSI_GPIO_PORT**, **I2S2_MOSI_PIN**, **I2S2_SCK_AF**, **I2S2_SCK_GPIO_CLK_ENABLE**, **I2S2_SCK_GPIO_PORT**, and **I2S2_SCK_PIN**.

Referenced by **BSP_AUDIO_IN_Init()**.

uint8_t BSP_AUDIO_IN_Pause (void)

Pauses the audio file stream.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **768** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, and **hAudioInI2s**.

uint8_t BSP_AUDIO_IN_PDMToPCM (uint16_t * PDMBuf,
uint16_t * PCMBuf
)

Converts audio format from PDM to PCM.

Parameters:

PDMBuf,: Pointer to data PDM buffer

PCMBuf,: Pointer to data PCM buffer

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **811** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, **AudioInVolume**, **DEFAULT_AUDIO_IN_CHANNEL_NBR**, **Filter**, **INTERNAL_BUFF_SIZE**, and **PCM_OUT_SIZE**.

uint8_t BSP_AUDIO_IN_Record (uint16_t * pbuf,

uint32_t size
)

Starts audio recording.

Parameters:

pbuf,: Main buffer pointer for the recorded data storing

size,: Current size of the recorded buffer

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **734** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, and **hAudioInI2s**.

uint8_t BSP_AUDIO_IN_Resume (void)

Resumes the audio file stream.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **781** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, and **hAudioInI2s**.

uint8_t BSP_AUDIO_IN_SetVolume (uint8_t Volume)

Controls the audio in volume level.

Parameters:

Volume,: Volume level to be set in percentage from 0% to

100% (0 for Mute and 100 for Max volume level).

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **796** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, and **AudioInVolume**.

uint8_t BSP_AUDIO_IN_Stop (void)

Stops audio recording.

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **751** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, and **hAudioInI2s**.

__weak void BSP_AUDIO_IN_TransferComplete_CallBack (void)

User callback when record buffer is filled.

Definition at line **998** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_RxCpltCallback()**.

void HAL_I2S_RxCpltCallback (I2S_HandleTypeDef * hi2s)

Rx Transfer completed callbacks.

Parameters:

hi2s,: I2S handle

Definition at line **841** of file **stm32f4_discovery_audio.c**.

References **BSP_AUDIO_IN_TransferComplete_CallBack()**.

void HAL_I2S_RxHalfCpltCallback (I2S_HandleTypeDef * hi2s)

Rx Half Transfer completed callbacks.

Parameters:

hi2s,: I2S handle

Definition at line **851** of file **stm32f4_discovery_audio.c**.

References **BSP_AUDIO_IN_HalfTransfer_CallBack()**.

static uint8_t I2S2_Init (uint32_t AudioFreq) [static]

Initializes the Audio Codec audio interface (I2S)

Note:

This function assumes that the I2S input clock (through PLL_R in Devices RevA/Z and through dedicated PLLI2S_R in Devices RevB/Y) is already configured and ready to be used.

Parameters:

AudioFreq,: Audio frequency to be configured for the I2S peripheral.

Definition at line **1061** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **hAudioInI2s**, and **I2S2**.

Referenced by **BSP_AUDIO_IN_Init()**.

static void PDMDecoder_Init (uint32_t AudioFreq,
uint32_t ChnlNbr
) [static]

Initialize the PDM library.

Parameters:

AudioFreq,: Audio sampling frequency

ChnlNbr,: Number of audio channels (1: mono; 2: stereo)

Definition at line **1033** of file **stm32f4_discovery_audio.c**.

References **Filter**.

Referenced by **BSP_AUDIO_IN_Init()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
AUDIO IN Exported Functions

STM32F4 DISCOVERY AUDIO

Functions

uint8_t	**BSP_AUDIO_IN_Init** (uint32_t AudioFreq, uint32_t BitRes, uint32_t ChnlNbr) Initializes wave recording.
uint8_t	**BSP_AUDIO_IN_Record** (uint16_t *pData, uint32_t Size) Starts audio recording.
uint8_t	**BSP_AUDIO_IN_Stop** (void) Stops audio recording.
uint8_t	**BSP_AUDIO_IN_Pause** (void) Pauses the audio file stream.
uint8_t	**BSP_AUDIO_IN_Resume** (void) Resumes the audio file stream.
uint8_t	**BSP_AUDIO_IN_SetVolume** (uint8_t Volume) Controls the audio in volume level.
uint8_t	**BSP_AUDIO_IN_PDMToPCM** (uint16_t *PDMBuf, uint16_t *PCMBuf) Converts audio format from PDM to PCM.
void	**BSP_AUDIO_IN_TransferComplete_CallBack** (void) User callback when record buffer is filled.
void	**BSP_AUDIO_IN_HalfTransfer_CallBack** (void) Manages the DMA Half Transfer complete event.
void	**BSP_AUDIO_IN_Error_Callback** (void) Audio IN Error callback function.
void	**BSP_AUDIO_IN_ClockConfig** (I2S_HandleTypeDef *hi2s, uint32_t AudioFreq, void *Params) Audio In Clock Config.
void	**BSP_AUDIO_IN_MspInit** (I2S_HandleTypeDef *hi2s, void *Params) BSP AUDIO IN MSP Init.
void	**BSP_AUDIO_IN_MspDeInit** (I2S_HandleTypeDef *hi2s, void *Params) DeInitializes BSP_AUDIO_IN MSP.

Function Documentation

void BSP_AUDIO_IN_ClockConfig (I2S_HandleTypeDef * hi2s,
uint32_t AudioFre
void * Params
)

Audio In Clock Config.

Parameters:

hi2s,:	I2S handle
AudioFreq,:	Audio frequency used to record the audio stream.
Params	: pointer on additional configuration parameters, can be NULL.

Note:
This API is called by **BSP_AUDIO_IN_Init()** Being __weak it can be overwritten by the application

Definition at line **866** of file **stm32f4_discovery_audio.c**.

Referenced by **BSP_AUDIO_IN_Init()**.

void BSP_AUDIO_IN_Error_Callback (void)

Audio IN Error callback function.

Definition at line **1018** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_ErrorCallback()**.

void BSP_AUDIO_IN_HalfTransfer_CallBack (void)

Manages the DMA Half Transfer complete event.

Definition at line **1008** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_RxHalfCpltCallback()**.

uint8_t BSP_AUDIO_IN_Init (uint32_t AudioFreq,
uint32_t BitRes,
uint32_t ChnlNbr
)

Initializes wave recording.

Parameters:

AudioFreq,:	Audio frequency to be configured for the I2S peripheral.
BitRes,:	Audio frequency to be configured for the I2S peripheral.
ChnlNbr,:	Audio frequency to be configured for the I2S peripheral.

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **704** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, **BSP_AUDIO_IN_ClockConfig()**, **BSP_AUDIO_IN_MspInit()**, **hAudioInI2s**, **I2S2**, **I2S2_Init()**, and **PDMDecoder_Init()**.

void BSP_AUDIO_IN_MspDeInit (I2S_HandleTypeDef * hi2s,
void * Params
)

DeInitializes BSP_AUDIO_IN MSP.

Parameters:

hi2s,: I2S handle

Params : pointer on additional configuration parameters, can be NULL.

Definition at line **966** of file **stm32f4_discovery_audio.c**.

References **I2S2**, **I2S2_CLK_DISABLE**, **I2S2_DMAx_IRQ**, **I2S2_MOSI_GPIO_PORT**, **I2S2_MOSI_PIN**, **I2S2_SCK_GPIO_PORT**, and **I2S2_SCK_PIN**.

void BSP_AUDIO_IN_MspInit (I2S_HandleTypeDef * hi2s,
void * Params
)

BSP AUDIO IN MSP Init.

Parameters:

hi2s,: I2S handle

Params : pointer on additional configuration parameters, can be NULL.

Definition at line **900** of file **stm32f4_discovery_audio.c**.

References **AUDIO_IN_IRQ_PREPRIO**, **I2S2**, **I2S2_CLK_ENABLE**, **I2S2_DMAx_CHANNEL**, **I2S2_DMAx_CLK_ENABLE**, **I2S2_DMAx_IRQ**, **I2S2_DMAx_MEM_DATA_SIZE**, **I2S2_DMAx_PERIPH_DATA_SIZE**, **I2S2_DMAx_STREAM**, **I2S2_MOSI_AF**, **I2S2_MOSI_GPIO_CLK_ENABLE**, **I2S2_MOSI_GPIO_PORT**, **I2S2_MOSI_PIN**, **I2S2_SCK_AF**, **I2S2_SCK_GPIO_CLK_ENABLE**, **I2S2_SCK_GPIO_PORT**, and **I2S2_SCK_PIN**.

Referenced by **BSP_AUDIO_IN_Init()**.

uint8_t BSP_AUDIO_IN_Pause (void)

Pauses the audio file stream.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **768** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, and **hAudioInI2s**.

uint8_t BSP_AUDIO_IN_PDMToPCM (uint16_t * PDMBuf,
uint16_t * PCMBuf
)

Converts audio format from PDM to PCM.

Parameters:

PDMBuf,: Pointer to data PDM buffer

PCMBuf,: Pointer to data PCM buffer

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **811** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, **AudioInVolume**, **DEFAULT_AUDIO_IN_CHANNEL_NBR**, **Filter**, **INTERNAL_BUFF_SIZE**, and **PCM_OUT_SIZE**.

uint8_t BSP_AUDIO_IN_Record (uint16_t * pbuf,

uint32_t size
)

Starts audio recording.

Parameters:

pbuf,: Main buffer pointer for the recorded data storing

size,: Current size of the recorded buffer

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **734** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, and **hAudioInI2s**.

uint8_t BSP_AUDIO_IN_Resume (void)

Resumes the audio file stream.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **781** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, and **hAudioInI2s**.

uint8_t BSP_AUDIO_IN_SetVolume (uint8_t Volume)

Controls the audio in volume level.

Parameters:

Volume,: Volume level to be set in percentage from 0% to

100% (0 for Mute and 100 for Max volume level).

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **796** of file **stm32f4_discovery_audio.c**.

References **AUDIO_OK**, and **AudioInVolume**.

uint8_t BSP_AUDIO_IN_Stop (void)

Stops audio recording.

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **751** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, and **hAudioInI2s**.

void BSP_AUDIO_IN_TransferComplete_CallBack (void)

User callback when record buffer is filled.

Definition at line **998** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_RxCpltCallback()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
AUDIO OUT Private Functions

STM32F4 DISCOVERY AUDIO

Functions

uint8_t	**BSP_AUDIO_OUT_Init** (uint16_t OutputDevice, uint8_t Volume, uint32_t AudioFreq) Configures the audio peripherals.
uint8_t	**BSP_AUDIO_OUT_Play** (uint16_t *pBuffer, uint32_t Size) Starts playing audio stream from a data buffer for a determined size.
void	**BSP_AUDIO_OUT_ChangeBuffer** (uint16_t *pData, uint16_t Size) Sends n-Bytes on the I2S interface.
uint8_t	**BSP_AUDIO_OUT_Pause** (void) Pauses the audio file stream.
uint8_t	**BSP_AUDIO_OUT_Resume** (void) Resumes the audio file streaming.
uint8_t	**BSP_AUDIO_OUT_Stop** (uint32_t Option) Stops audio playing and Power down the Audio Codec.
uint8_t	**BSP_AUDIO_OUT_SetVolume** (uint8_t Volume) Controls the current audio volume level.
uint8_t	**BSP_AUDIO_OUT_SetMute** (uint32_t Cmd) Enables or disables the MUTE mode by software.
uint8_t	**BSP_AUDIO_OUT_SetOutputMode** (uint8_t Output) Switch dynamically (while audio file is played) the output target (speaker or headphone).
void	**BSP_AUDIO_OUT_SetFrequency** (uint32_t AudioFreq) Update the audio frequency.
void	**HAL_I2S_TxCpltCallback** (I2S_HandleTypeDef *hi2s) Tx Transfer completed callbacks.
void	**HAL_I2S_TxHalfCpltCallback** (I2S_HandleTypeDef *hi2s) Tx Half Transfer completed callbacks.

__weak void	**BSP_AUDIO_OUT_ClockConfig** (I2S_HandleTypeDef *hi2s, uint32_t AudioFreq, void *Params) Clock Config.
__weak void	**BSP_AUDIO_OUT_MspInit** (I2S_HandleTypeDef *hi2s, void *Params) AUDIO OUT I2S MSP Init.
__weak void	**BSP_AUDIO_OUT_MspDeInit** (I2S_HandleTypeDef *hi2s, void *Params) De-Initializes BSP_AUDIO_OUT MSP.
__weak void	**BSP_AUDIO_OUT_TransferComplete_CallBack** (void) Manages the DMA full Transfer complete event.
__weak void	**BSP_AUDIO_OUT_HalfTransfer_CallBack** (void) Manages the DMA Half Transfer complete event.
__weak void	**BSP_AUDIO_OUT_Error_CallBack** (void) Manages the DMA FIFO error event.
static uint8_t	**I2S3_Init** (uint32_t AudioFreq) Initializes the Audio Codec audio interface (I2S).

Function Documentation

void BSP_AUDIO_OUT_ChangeBuffer (uint16_t * pData,
uint16_t Size
)

Sends n-Bytes on the I2S interface.

Parameters:

pData,: Pointer to data address

Size,: Number of data to be written

Definition at line **287** of file **stm32f4_discovery_audio.c**.

References **hAudioOutI2s**.

**__weak void BSP_AUDIO_OUT_ClockConfig (I2S_HandleTypeDef *
uint32_t
void *
)**

Clock Config.

Parameters:

hi2s,: might be required to set audio peripheral predivider if any.

AudioFreq,: Audio frequency used to play the audio stream.

Note:
This API is called by **BSP_AUDIO_OUT_Init()** and **BSP_AUDIO_OUT_SetFrequency()** Being __weak it can be overwritten by the application

Parameters:

Params : pointer on additional configuration parameters, can be NULL.

Definition at line **486** of file **stm32f4_discovery_audio.c**.

References **I2SFreq**, **I2SPLLN**, and **I2SPLLR**.

Referenced by **BSP_AUDIO_OUT_Init()**, and **BSP_AUDIO_OUT_SetFrequency()**.

__weak void BSP_AUDIO_OUT_Error_CallBack (void)

Manages the DMA FIFO error event.

Definition at line **650** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_ErrorCallback()**.

__weak void BSP_AUDIO_OUT_HalfTransfer_CallBack (void)

Manages the DMA Half Transfer complete event.

Definition at line **643** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_TxHalfCpltCallback()**.

uint8_t BSP_AUDIO_OUT_Init (uint16_t OutputDevice,
uint8_t Volume,
uint32_t AudioFreq
)

Configures the audio peripherals.

Parameters:

OutputDevice,: OUTPUT_DEVICE_SPEAKER, OUTPUT_DEVICE_HEADPHONE, OUTPUT_DEVICE_BOTH or OUTPUT_DEVICE_AUTO .

Volume,: Initial volume level (from 0 (Mute) to 100 (Max))

AudioFreq,: Audio frequency used to play the audio stream.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **213** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **BSP_AUDIO_OUT_ClockConfig()**, **BSP_AUDIO_OUT_MspInit()**, **hAudioOutI2s**, **I2S3**, **I2S3_Init()**, and **pAudioDrv**.

__weak void BSP_AUDIO_OUT_MspDeInit (I2S_HandleTypeDef * h
void * P
)

De-Initializes BSP_AUDIO_OUT MSP.

Parameters:

hi2s,: might be required to set audio peripheral predivider if any.

Params : pointer on additional configuration parameters, can be NULL.

Definition at line **598** of file **stm32f4_discovery_audio.c**.

References **I2S3**, **I2S3_CLK_DISABLE**, **I2S3_DMAx_IRQ**, **I2S3_MCK_GPIO_PORT**, **I2S3_MCK_PIN**, **I2S3_SCK_PIN**,

I2S3_SCK_SD_GPIO_PORT, I2S3_SD_PIN, I2S3_WS_GPIO_PORT, and I2S3_WS_PIN.

__weak void BSP_AUDIO_OUT_MspInit (I2S_HandleTypeDef * hi2s
void * Para
)

AUDIO OUT I2S MSP Init.

Parameters:

hi2s,: might be required to set audio peripheral predivider if any.

Params : pointer on additional configuration parameters, can be NULL.

Definition at line 528 of file stm32f4_discovery_audio.c.

References AUDIO_OUT_IRQ_PREPRIO, I2S3, I2S3_CLK_ENABLE, I2S3_DMAx_CHANNEL, I2S3_DMAx_CLK_ENABLE, I2S3_DMAx_IRQ, I2S3_DMAx_MEM_DATA_SIZE, I2S3_DMAx_PERIPH_DATA_SIZE, I2S3_DMAx_STREAM, I2S3_MCK_CLK_ENABLE, I2S3_MCK_GPIO_PORT, I2S3_MCK_PIN, I2S3_SCK_PIN, I2S3_SCK_SD_CLK_ENABLE, I2S3_SCK_SD_GPIO_PORT, I2S3_SCK_SD_WS_AF, I2S3_SD_PIN, I2S3_WS_CLK_ENABLE, I2S3_WS_GPIO_PORT, and I2S3_WS_PIN.

Referenced by BSP_AUDIO_OUT_Init().

uint8_t BSP_AUDIO_OUT_Pause (void)

Pauses the audio file stream.

In case of using DMA, the DMA Pause feature is used. WARNING: When calling BSP_AUDIO_OUT_Pause() function for pause, only the

BSP_AUDIO_OUT_Resume() function should be called for resume (use of **BSP_AUDIO_OUT_Play()** function for resume could lead to unexpected behavior).

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **300** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **hAudioOutI2s**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_Play (uint16_t * pBuffer,
uint32_t Size
)

Starts playing audio stream from a data buffer for a determined size.

Parameters:

pBuffer,: Pointer to the buffer

Size,: Number of audio data BYTES.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **265** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIODATA_SIZE**, **DMA_MAX**, **hAudioOutI2s**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_Resume (void)

Resumes the audio file streaming.

WARNING: When calling **BSP_AUDIO_OUT_Pause()** function for pause, only **BSP_AUDIO_OUT_Resume()** function should be called for resume (use of **BSP_AUDIO_OUT_Play()** function for resume could lead to unexpected behavior).

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **324** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **hAudioOutI2s**, and **pAudioDrv**.

void BSP_AUDIO_OUT_SetFrequency (uint32_t AudioFreq)

Update the audio frequency.

Parameters:

AudioFreq,: Audio frequency used to play the audio stream.

Note:

This API should be called after the **BSP_AUDIO_OUT_Init()** to adjust the audio frequency.

Definition at line **442** of file **stm32f4_discovery_audio.c**.

References **BSP_AUDIO_OUT_ClockConfig()**, **hAudioOutI2s**, and **I2S3_Init()**.

uint8_t BSP_AUDIO_OUT_SetMute (uint32_t Cmd)

Enables or disables the MUTE mode by software.

Parameters:

Cmd,: could be AUDIO_MUTE_ON to mute sound or

AUDIO_MUTE_OFF to unmute the codec and restore previous volume level.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **400** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_SetOutputMode (uint8_t Output)

Switch dynamically (while audio file is played) the output target (speaker or headphone).

Note:

This function modifies a global variable of the audio codec driver: OutputDev.

Parameters:

Output,: specifies the audio output target: OUTPUT_DEVICE_SPEAKER, OUTPUT_DEVICE_HEADPHONE, OUTPUT_DEVICE_BOTH or OUTPUT_DEVICE_AUTO

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **422** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_SetVolume (uint8_t Volume)

Controls the current audio volume level.

Parameters:

Volume,: Volume level to be set in percentage from 0% to 100% (0 for Mute and 100 for Max volume level).

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **380** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_Stop (uint32_t Option)

Stops audio playing and Power down the Audio Codec.

Parameters:

Option,: could be one of the following parameters

- CODEC_PDWN_HW: completely shut down the codec (physically). Then need to reconfigure the Codec after power on.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **348** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIO_RESET_GPIO**, **AUDIO_RESET_PIN**, **hAudioOutI2s**, and

pAudioDrv.

__weak void BSP_AUDIO_OUT_TransferComplete_CallBack (void

Manages the DMA full Transfer complete event.

Definition at line **636** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_TxCpltCallback()**.

void HAL_I2S_TxCpltCallback (I2S_HandleTypeDef * hi2s)

Tx Transfer completed callbacks.

Parameters:

hi2s,: I2S handle

Definition at line **455** of file **stm32f4_discovery_audio.c**.

References **BSP_AUDIO_OUT_TransferComplete_CallBack()**, and **I2S3**.

void HAL_I2S_TxHalfCpltCallback (I2S_HandleTypeDef * hi2s)

Tx Half Transfer completed callbacks.

Parameters:

hi2s,: I2S handle

Definition at line **468** of file **stm32f4_discovery_audio.c**.

References **BSP_AUDIO_OUT_HalfTransfer_CallBack()**, and **I2S3**.

static uint8_t I2S3_Init (uint32_t AudioFreq) [static]

Initializes the Audio Codec audio interface (I2S).

Parameters:

AudioFreq,: Audio frequency to be configured for the I2S peripheral.

Definition at line **662** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **hAudioOutI2s**, and **I2S3**.

Referenced by **BSP_AUDIO_OUT_Init()**, and **BSP_AUDIO_OUT_SetFrequency()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
AUDIO OUT Exported Functions

STM32F4 DISCOVERY AUDIO

Functions

uint8_t	**BSP_AUDIO_OUT_Init** (uint16_t OutputDevice, uint8_t Volume, uint32_t AudioFreq) Configures the audio peripherals.
uint8_t	**BSP_AUDIO_OUT_Play** (uint16_t *pBuffer, uint32_t Size) Starts playing audio stream from a data buffer for a determined size.
void	**BSP_AUDIO_OUT_ChangeBuffer** (uint16_t *pData, uint16_t Size) Sends n-Bytes on the I2S interface.
uint8_t	**BSP_AUDIO_OUT_Pause** (void) Pauses the audio file stream.
uint8_t	**BSP_AUDIO_OUT_Resume** (void) Resumes the audio file streaming.
uint8_t	**BSP_AUDIO_OUT_Stop** (uint32_t Option) Stops audio playing and Power down the Audio Codec.
uint8_t	**BSP_AUDIO_OUT_SetVolume** (uint8_t Volume) Controls the current audio volume level.
void	**BSP_AUDIO_OUT_SetFrequency** (uint32_t AudioFreq) Update the audio frequency.
uint8_t	**BSP_AUDIO_OUT_SetMute** (uint32_t Cmd) Enables or disables the MUTE mode by software.
uint8_t	**BSP_AUDIO_OUT_SetOutputMode** (uint8_t Output) Switch dynamically (while audio file is played) the output target (speaker or headphone).
void	**BSP_AUDIO_OUT_TransferComplete_CallBack** (void) Manages the DMA full Transfer complete event.
void	**BSP_AUDIO_OUT_HalfTransfer_CallBack** (void) Manages the DMA Half Transfer complete event.
void	**BSP_AUDIO_OUT_Error_CallBack** (void) Manages the DMA FIFO error event.
	BSP_AUDIO_OUT_ClockConfig (I2S_HandleTypeDef *hi2s,

void	uint32_t AudioFreq, void *Params) Clock Config.
void	**BSP_AUDIO_OUT_MspInit** (I2S_HandleTypeDef *hi2s, void *Params) AUDIO OUT I2S MSP Init.
void	**BSP_AUDIO_OUT_MspDeInit** (I2S_HandleTypeDef *hi2s, void *Params) De-Initializes BSP_AUDIO_OUT MSP.

Function Documentation

void BSP_AUDIO_OUT_ChangeBuffer (uint16_t * pData,
uint16_t Size
)

Sends n-Bytes on the I2S interface.

Parameters:

pData,: Pointer to data address

Size,: Number of data to be written

Definition at line **287** of file **stm32f4_discovery_audio.c**.

References **hAudioOutI2s**.

void BSP_AUDIO_OUT_ClockConfig (I2S_HandleTypeDef * hi2s,
uint32_t AudioF
void * Param
)

Clock Config.

Parameters:

hi2s,: might be required to set audio peripheral predivider if any.

AudioFreq,: Audio frequency used to play the audio stream.

Note:
This API is called by **BSP_AUDIO_OUT_Init()** and **BSP_AUDIO_OUT_SetFrequency()** Being __weak it can be overwritten by the application

Parameters:

Params : pointer on additional configuration parameters, can be NULL.

Definition at line **486** of file **stm32f4_discovery_audio.c**.

References **I2SFreq**, **I2SPLLN**, and **I2SPLLR**.

Referenced by **BSP_AUDIO_OUT_Init()**, and **BSP_AUDIO_OUT_SetFrequency()**.

void BSP_AUDIO_OUT_Error_CallBack (void)

Manages the DMA FIFO error event.

Definition at line **650** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_ErrorCallback()**.

void BSP_AUDIO_OUT_HalfTransfer_CallBack (void)

Manages the DMA Half Transfer complete event.

Definition at line **643** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_TxHalfCpltCallback()**.

uint8_t BSP_AUDIO_OUT_Init (uint16_t OutputDevice, uint8_t Volume, uint32_t AudioFreq)

Configures the audio peripherals.

Parameters:

OutputDevice,: OUTPUT_DEVICE_SPEAKER, OUTPUT_DEVICE_HEADPHONE, OUTPUT_DEVICE_BOTH or OUTPUT_DEVICE_AUTO .

Volume,: Initial volume level (from 0 (Mute) to 100 (Max))

AudioFreq,: Audio frequency used to play the audio stream.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **213** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **BSP_AUDIO_OUT_ClockConfig()**, **BSP_AUDIO_OUT_MspInit()**, **hAudioOutI2s**, **I2S3**, **I2S3_Init()**, and **pAudioDrv**.

void BSP_AUDIO_OUT_MspDeInit (I2S_HandleTypeDef * hi2s, void * Params)

De-Initializes BSP_AUDIO_OUT MSP.

Parameters:

hi2s,: might be required to set audio peripheral predivider if any.

Params : pointer on additional configuration parameters, can be NULL.

Definition at line **598** of file **stm32f4_discovery_audio.c**.

References **I2S3**, **I2S3_CLK_DISABLE**, **I2S3_DMAx_IRQ**, **I2S3_MCK_GPIO_PORT**, **I2S3_MCK_PIN**, **I2S3_SCK_PIN**,

I2S3_SCK_SD_GPIO_PORT, I2S3_SD_PIN, I2S3_WS_GPIO_PORT, and I2S3_WS_PIN.

void BSP_AUDIO_OUT_MspInit (I2S_HandleTypeDef * hi2s, void * Params)

AUDIO OUT I2S MSP Init.

Parameters:

- **hi2s,:** might be required to set audio peripheral predivider if any.
- **Params** : pointer on additional configuration parameters, can be NULL.

Definition at line 528 of file stm32f4_discovery_audio.c.

References AUDIO_OUT_IRQ_PREPRIO, I2S3, I2S3_CLK_ENABLE, I2S3_DMAx_CHANNEL, I2S3_DMAx_CLK_ENABLE, I2S3_DMAx_IRQ, I2S3_DMAx_MEM_DATA_SIZE, I2S3_DMAx_PERIPH_DATA_SIZE, I2S3_DMAx_STREAM, I2S3_MCK_CLK_ENABLE, I2S3_MCK_GPIO_PORT, I2S3_MCK_PIN, I2S3_SCK_PIN, I2S3_SCK_SD_CLK_ENABLE, I2S3_SCK_SD_GPIO_PORT, I2S3_SCK_SD_WS_AF, I2S3_SD_PIN, I2S3_WS_CLK_ENABLE, I2S3_WS_GPIO_PORT, and I2S3_WS_PIN.

Referenced by BSP_AUDIO_OUT_Init().

uint8_t BSP_AUDIO_OUT_Pause (void)

Pauses the audio file stream.

In case of using DMA, the DMA Pause feature is used. WARNING: When calling BSP_AUDIO_OUT_Pause() function for pause, only the

BSP_AUDIO_OUT_Resume() function should be called for resume (use of **BSP_AUDIO_OUT_Play()** function for resume could lead to unexpected behavior).

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **300** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **hAudioOutI2s**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_Play (uint16_t * pBuffer,
uint32_t Size
)

Starts playing audio stream from a data buffer for a determined size.

Parameters:

pBuffer,: Pointer to the buffer

Size,: Number of audio data BYTES.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **265** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIODATA_SIZE**, **DMA_MAX**, **hAudioOutI2s**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_Resume (void)

Resumes the audio file streaming.

WARNING: When calling **BSP_AUDIO_OUT_Pause()** function for pause, only **BSP_AUDIO_OUT_Resume()** function should be called for resume (use of **BSP_AUDIO_OUT_Play()** function for resume could lead to unexpected behavior).

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **324** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **hAudioOutI2s**, and **pAudioDrv**.

void BSP_AUDIO_OUT_SetFrequency (uint32_t AudioFreq)

Update the audio frequency.

Parameters:

AudioFreq,: Audio frequency used to play the audio stream.

Note:

This API should be called after the **BSP_AUDIO_OUT_Init()** to adjust the audio frequency.

Definition at line **442** of file **stm32f4_discovery_audio.c**.

References **BSP_AUDIO_OUT_ClockConfig()**, **hAudioOutI2s**, and **I2S3_Init()**.

uint8_t BSP_AUDIO_OUT_SetMute (uint32_t Cmd)

Enables or disables the MUTE mode by software.

Parameters:

Cmd,: could be AUDIO_MUTE_ON to mute sound or

AUDIO_MUTE_OFF to unmute the codec and restore previous volume level.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **400** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_SetOutputMode (uint8_t Output)

Switch dynamically (while audio file is played) the output target (speaker or headphone).

Note:

This function modifies a global variable of the audio codec driver: OutputDev.

Parameters:

Output,: specifies the audio output target: OUTPUT_DEVICE_SPEAKER, OUTPUT_DEVICE_HEADPHONE, OUTPUT_DEVICE_BOTH or OUTPUT_DEVICE_AUTO

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **422** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_SetVolume (uint8_t Volume)

Controls the current audio volume level.

Parameters:

Volume,: Volume level to be set in percentage from 0% to 100% (0 for Mute and 100 for Max volume level).

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **380** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **pAudioDrv**.

uint8_t BSP_AUDIO_OUT_Stop (uint32_t Option)

Stops audio playing and Power down the Audio Codec.

Parameters:

Option,: could be one of the following parameters

- CODEC_PDWN_HW: completely shut down the codec (physically). Then need to reconfigure the Codec after power on.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **348** of file **stm32f4_discovery_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIO_RESET_GPIO**, **AUDIO_RESET_PIN**, **hAudioOutI2s**, and

pAudioDrv.

void BSP_AUDIO_OUT_TransferComplete_CallBack (void)

Manages the DMA full Transfer complete event.

Definition at line **636** of file **stm32f4_discovery_audio.c**.

Referenced by **HAL_I2S_TxCpltCallback()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
LOW LEVEL Exported Functions

STM32F4 DISCOVERY LOW LEVEL

Functions

uint32_t	**BSP_GetVersion** (void) This method returns the STM32F4 DISCO 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** Mode) Configures Button GPIO and EXTI Line.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Returns the selected Button state.

Function Documentation

uint32_t BSP_GetVersion (void)

This method returns the STM32F4 DISCO BSP Driver revision.

Return values:

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

Definition at line **158** of file **stm32f4_discovery.c**.

References **__STM32F4_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:

- LED4
- LED3
- LED5
- LED6

Definition at line **172** of file **stm32f4_discovery.c**.

References **GPIO_PIN**, **GPIO_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:

- LED4
- LED3
- LED5
- LED6

Definition at line **213** of file **stm32f4_discovery.c**.

References **GPIO_PIN**, and **GPIO_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:

- LED4
- LED3
- LED5
- LED6

Definition at line **199** of file **stm32f4_discovery.c**.

References **GPIO_PIN**, and **GPIO_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:

- LED4
- LED3
- LED5
- LED6

Definition at line **227** of file **stm32f4_discovery.c**.

References **GPIO_PIN**, and **GPIO_PORT**.

uint32_t BSP_PB_GetState (Button_TypeDef Button)

Returns the selected Button state.

Parameters:

Button,: Specifies the Button to be checked. This parameter should be: BUTTON_KEY

Return values:

The Button GPIO pin value.

Definition at line **289** of file **stm32f4_discovery.c**.

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

void BSP_PB_Init (Button_TypeDef Button,
ButtonMode_TypeDef Mode
)

Configures Button GPIO and EXTI Line.

Parameters:

Button,: Specifies the Button to be configured. This parameter should be: BUTTON_KEY

Mode,: 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

Definition at line **250** of file **stm32f4_discovery.c**.

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

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
LOW LEVEL LED Functions

STM32F4 DISCOVERY LOW LEVEL

Functions

uint32_t	**BSP_GetVersion** (void) This method returns the STM32F4 DISCO 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.

Function Documentation

uint32_t BSP_GetVersion (void)

This method returns the STM32F4 DISCO BSP Driver revision.

Return values:

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

Definition at line **158** of file **stm32f4_discovery.c**.

References **__STM32F4_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:

- LED4
- LED3
- LED5
- LED6

Definition at line **172** of file **stm32f4_discovery.c**.

References **GPIO_PIN**, **GPIO_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:

- LED4
- LED3
- LED5
- LED6

Definition at line **213** of file **stm32f4_discovery.c**.

References **GPIO_PIN**, and **GPIO_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:

- LED4
- LED3
- LED5
- LED6

Definition at line **199** of file **stm32f4_discovery.c**.

References **GPIO_PIN**, and **GPIO_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:

- LED4
- LED3
- LED5
- LED6

Definition at line **227** of file **stm32f4_discovery.c**.

References **GPIO_PIN**, and **GPIO_PORT**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
LOW LEVEL BUTTON Functions

STM32F4 DISCOVERY LOW LEVEL

Functions

void	**BSP_PB_Init** (**Button_TypeDef** Button, **ButtonMode_TypeDef** Mode) Configures Button GPIO and EXTI Line.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Returns the selected Button state.

Function Documentation

uint32_t BSP_PB_GetState (Button_TypeDef Button)

Returns the selected Button state.

Parameters:

Button,: Specifies the Button to be checked. This parameter should be: BUTTON_KEY

Return values:

The Button GPIO pin value.

Definition at line **289** of file **stm32f4_discovery.c**.

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

void BSP_PB_Init (Button_TypeDef Button, ButtonMode_TypeDef Mode)

Configures Button GPIO and EXTI Line.

Parameters:

Button,: Specifies the Button to be configured. This parameter should be: BUTTON_KEY

Mode,: 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

Definition at line **250** of file **stm32f4_discovery.c**.

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

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
LOW LEVEL Private Variables

STM32F4 DISCOVERY LOW LEVEL

Variables

GPIO_TypeDef *	**GPIO_PORT** [**LEDn**]
const uint16_t	**GPIO_PIN** [**LEDn**]
GPIO_TypeDef *	**BUTTON_PORT** [**BUTTONn**] = {**KEY_BUTTON_GPIO_PORT**}
const uint16_t	**BUTTON_PIN** [**BUTTONn**] = {**KEY_BUTTON_PIN**}
const uint8_t	**BUTTON_IRQn** [**BUTTONn**] = {**KEY_BUTTON_EXTI_IRQn**}
uint32_t	**I2cxTimeout** = **I2Cx_TIMEOUT_MAX**
uint32_t	**SpixTimeout** = **SPIx_TIMEOUT_MAX**
static SPI_HandleTypeDef	**SpiHandle**
static I2C_HandleTypeDef	**I2cHandle**

Variable Documentation

const uint8_t BUTTON_IRQn[BUTTONn] = {KEY_BUTTON_EXTI_IR

Definition at line 103 of file stm32f4_discovery.c.

Referenced by BSP_PB_Init().

const uint16_t BUTTON_PIN[BUTTONn] = {KEY_BUTTON_PIN}

Definition at line 102 of file stm32f4_discovery.c.

Referenced by BSP_PB_GetState(), and BSP_PB_Init().

GPIO_TypeDef* BUTTON_PORT[BUTTONn] = {KEY_BUTTON_GPIC

Definition at line 101 of file stm32f4_discovery.c.

Referenced by BSP_PB_GetState(), and BSP_PB_Init().

const uint16_t GPIO_PIN[LEDn]

Initial value:

```
{LED4_PIN,
                                      LED3_PIN,
                                      LED5_PIN,
                                      LED6_PIN}
```

Definition at line 96 of file stm32f4_discovery.c.

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

GPIO_TypeDef* GPIO_PORT[LEDn]

Initial value:

```
{LED4_GPIO_PORT,
                                    LED3_GPIO_PORT,
                                    LED5_GPIO_PORT,
                                    LED6_GPIO_PORT}
```

Definition at line **92** of file **stm32f4_discovery.c**.

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

I2C_HandleTypeDef I2cHandle [static]

Definition at line **109** of file **stm32f4_discovery.c**.

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

uint32_t I2cxTimeout = I2Cx_TIMEOUT_MAX

Definition at line **105** of file **stm32f4_discovery.c**.

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

SPI_HandleTypeDef SpiHandle [static]

Definition at line **108** of file **stm32f4_discovery.c**.

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

uint32_t SpixTimeout = SPIx_TIMEOUT_MAX

Definition at line **106** of file **stm32f4_discovery.c**.

Referenced by **SPIx_WriteRead()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL_Exported_Types

STM32F4 DISCOVERY LOW LEVEL

Enumerations

enum	**Led_TypeDef** { **LED4** = 0, **LED3** = 1, **LED5** = 2, **LED6** = 3 }
enum	**Button_TypeDef** { **BUTTON_KEY** = 0 }
enum	**ButtonMode_TypeDef** { **BUTTON_MODE_GPIO** = 0, **BUTTON_MODE_EXTI** = 1 }

Enumeration Type Documentation

enum Button_TypeDef

Enumerator:

BUTTON_KEY

Definition at line **73** of file **stm32f4_discovery.h**.

enum ButtonMode_TypeDef

Enumerator:

BUTTON_MODE_GPIO

BUTTON_MODE_EXTI

Definition at line **78** of file **stm32f4_discovery.h**.

enum Led_TypeDef

Enumerator:

LED4

LED3

LED5

LED6

Definition at line **65** of file **stm32f4_discovery.h**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL BUTTON

STM32F4 DISCOVERY LOW LEVEL Exported Constants

Defines

#define	**BUTTONn** 1
#define	**KEY_BUTTON_PIN** GPIO_PIN_0 Wakeup push-button.
#define	**KEY_BUTTON_GPIO_PORT** GPIOA
#define	**KEY_BUTTON_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOA
#define	**KEY_BUTTON_GPIO_CLK_DISABLE**() __HAL_RCC_GPIO
#define	**KEY_BUTTON_EXTI_IRQn** EXTI0_IRQn
#define	**BUTTONx_GPIO_CLK_ENABLE**(__INDEX__)
#define	**BUTTONx_GPIO_CLK_DISABLE**(__INDEX__)

Define Documentation

#define BUTTONn 1

Definition at line 141 of file stm32f4_discovery.h.

#define BUTTONx_GPIO_CLK_DISABLE (__INDEX__)

Value:

```
do{if((__INDEX__) == 0) KEY_BUTTON_GPIO_CLK_DISAB
LE(); \
                                                  }
while(0)
```

Definition at line 155 of file stm32f4_discovery.h.

#define BUTTONx_GPIO_CLK_ENABLE (__INDEX__)

Value:

```
do{if((__INDEX__) == 0) KEY_BUTTON_GPIO_CLK_ENABLE
(); \
                                                    }
while(0)
```

Definition at line 152 of file stm32f4_discovery.h.

Referenced by BSP_PB_Init().

#define KEY_BUTTON_EXTI_IRQn EXTI0_IRQn

Definition at line 150 of file stm32f4_discovery.h.

#define KEY_BUTTON_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPI

Definition at line 149 of file stm32f4_discovery.h.

#define KEY_BUTTON_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPI(

Definition at line 148 of file stm32f4_discovery.h.

#define KEY_BUTTON_GPIO_PORT GPIOA

Definition at line 147 of file stm32f4_discovery.h.

#define KEY_BUTTON_PIN GPIO_PIN_0

Wakeup push-button.

Definition at line 146 of file stm32f4_discovery.h.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO Exported Macros

STM32F4 DISCOVERY AUDIO

Defines

#define	**DMA_MAX**(_X_) (((_X_) <= **DMA_MAX_SZE**)? (_X_):**DMA_MAX_SZE**)

Define Documentation

#define DMA_MAX (_X_) (((_X_) <= DMA_MAX_SZE)? (_X_):DMA

Definition at line **186** of file **stm32f4_discovery_audio.h**.

Referenced by **BSP_AUDIO_OUT_Play()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY
AUDIO IN OUT Private Functions

STM32F4 DISCOVERY AUDIO

Functions

void	**HAL_I2S_ErrorCallback** (I2S_HandleTypeDef *hi2s) I2S error callbacks.

Function Documentation

void HAL_I2S_ErrorCallback (I2S_HandleTypeDef * hi2s)

I2S error callbacks.

Parameters:

hi2s,: I2S handle

Definition at line **1101** of file **stm32f4_discovery_audio.c**.

References **BSP_AUDIO_IN_Error_Callback()**, **BSP_AUDIO_OUT_Error_CallBack()**, **I2S2**, and **I2S3**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY AUDIO Private Defines

STM32F4 DISCOVERY AUDIO

Variables

const uint32_t	**I2SFreq** [8] = {8000, 11025, 16000, 22050, 32000, 44100, 48000, 96000}
const uint32_t	**I2SPLLN** [8] = {256, 429, 213, 429, 426, 271, 258, 344}
const uint32_t	**I2SPLLR** [8] = {5, 4, 4, 4, 4, 6, 3, 1}

Variable Documentation

const uint32_t I2SFreq[8] = {8000, 11025, 16000, 22050, 32000, 4410

Definition at line **161** of file **stm32f4_discovery_audio.c**.

Referenced by **BSP_AUDIO_OUT_ClockConfig()**.

const uint32_t I2SPLLN[8] = {256, 429, 213, 429, 426, 271, 258, 344}

Definition at line **162** of file **stm32f4_discovery_audio.c**.

Referenced by **BSP_AUDIO_OUT_ClockConfig()**.

const uint32_t I2SPLLR[8] = {5, 4, 4, 4, 4, 6, 3, 1}

Definition at line **163** of file **stm32f4_discovery_audio.c**.

Referenced by **BSP_AUDIO_OUT_ClockConfig()**.

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY LOW LEVEL LED

STM32F4 DISCOVERY LOW LEVEL Exported Constants

Define for STM32F4_DISCOVERY board. More...

Defines

#define	**LEDn** 4
#define	**LED4_PIN** GPIO_PIN_12
#define	**LED4_GPIO_PORT** GPIOD
#define	**LED4_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOD_CLK_E
#define	**LED4_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOD_CLK_D
#define	**LED3_PIN** GPIO_PIN_13
#define	**LED3_GPIO_PORT** GPIOD
#define	**LED3_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOD_CLK_E
#define	**LED3_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOD_CLK_D
#define	**LED5_PIN** GPIO_PIN_14
#define	**LED5_GPIO_PORT** GPIOD
#define	**LED5_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOD_CLK_E
#define	**LED5_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOD_CLK_D
#define	**LED6_PIN** GPIO_PIN_15
#define	**LED6_GPIO_PORT** GPIOD
#define	**LED6_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOD_CLK_E
#define	**LED6_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOD_CLK_D
#define	**LEDx_GPIO_CLK_ENABLE**(__INDEX__)
#define	**LEDx_GPIO_CLK_DISABLE**(__INDEX__)

Detailed Description

Define for STM32F4_DISCOVERY board.

Define Documentation

#define LED3_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOD_CLK

Definition at line 111 of file stm32f4_discovery.h.

#define LED3_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOD_CLK_

Definition at line 110 of file stm32f4_discovery.h.

#define LED3_GPIO_PORT GPIOD

Definition at line 109 of file stm32f4_discovery.h.

#define LED3_PIN GPIO_PIN_13

Definition at line 108 of file stm32f4_discovery.h.

#define LED4_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOD_CLK

Definition at line 106 of file stm32f4_discovery.h.

#define LED4_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOD_CLK_

Definition at line 105 of file stm32f4_discovery.h.

#define LED4_GPIO_PORT GPIOD

Definition at line 104 of file stm32f4_discovery.h.

#define LED4_PIN GPIO_PIN_12

Definition at line 103 of file stm32f4_discovery.h.

#define LED5_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOD_CLK

Definition at line 116 of file stm32f4_discovery.h.

#define LED5_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOD_CLK_

Definition at line 115 of file stm32f4_discovery.h.

#define LED5_GPIO_PORT GPIOD

Definition at line 114 of file stm32f4_discovery.h.

#define LED5_PIN GPIO_PIN_14

Definition at line 113 of file stm32f4_discovery.h.

#define LED6_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOD_CLK

Definition at line 121 of file stm32f4_discovery.h.

#define LED6_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOD_CLK_

Definition at line 120 of file stm32f4_discovery.h.

#define LED6_GPIO_PORT GPIOD

Definition at line 119 of file stm32f4_discovery.h.

#define LED6_PIN GPIO_PIN_15

Definition at line 118 of file stm32f4_discovery.h.

#define LEDn 4

Definition at line 101 of file stm32f4_discovery.h.

#define LEDx_GPIO_CLK_DISABLE (__INDEX__)

Value:

```
do{if((__INDEX__) == 0) LED4_GPIO_CLK_DISABLE();
else \
                                                 if((_
_INDEX__) == 1) LED3_GPIO_CLK_DISABLE(); else \
                                                 if((_
_INDEX__) == 2) LED5_GPIO_CLK_DISABLE(); else \
                                                 if((_
_INDEX__) == 3) LED6_GPIO_CLK_DISABLE(); \
                                                 }whil
e(0)
```

Definition at line 129 of file stm32f4_discovery.h.

#define LEDx_GPIO_CLK_ENABLE (__INDEX__)

Value:

```
do{if((__INDEX__) == 0) LED4_GPIO_CLK_ENABLE(); e
lse \
```

```
                                                  if((__
INDEX__) == 1) LED3_GPIO_CLK_ENABLE(); else \
                                                  if((__
INDEX__) == 2) LED5_GPIO_CLK_ENABLE(); else \
                                                  if((__
INDEX__) == 3) LED6_GPIO_CLK_ENABLE(); \
                                                  }while
(0)
```

Definition at line **123** of file **stm32f4_discovery.h**.

Referenced by **BSP_LED_Init()**.

STM32F4-Discovery BSP User Manual

Drivers Directory Reference

Directories

directory	**BSP**

STM32F4-Discovery BSP User Manual

BSP Directory Reference

Directories

directory	**STM32F4-Discovery**

STM32F4-Discovery BSP User Manual

STM32F4-Discovery Directory Reference

Files

file **stm32f4_discovery.c** [code]

This file provides set of firmware functions to manage Leds and push-button available on STM32F4-Discovery Kit from STMicroelectronics.

file **stm32f4_discovery.h** [code]

This file contains definitions for STM32F4-Discovery Kit's Leds and push-button hardware resources.

file **stm32f4_discovery_accelerometer.c** [code]

This file provides a set of functions needed to manage the MEMS accelerometers available on STM32F4-Discovery Kit.

file **stm32f4_discovery_accelerometer.h** [code]

This file contains all the functions prototypes for the **stm32f4_discovery_accelerometer.c** firmware driver.

file **stm32f4_discovery_audio.c** [code]

This file provides the Audio driver for the STM32F4-Discovery board.

file **stm32f4_discovery_audio.h** [code]

This file contains the common defines and functions prototypes for **stm32f4_discovery_audio.c** driver.

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP
User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

stm32f4_discovery.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32f4_discovery.h
00004   * @author  MCD Application Team
00005   * @version V2.1.2
00006   * @date    31-January-2017
00007   * @brief   This file contains definitions for STM32F4-Discovery Kit's Leds and
00008   *          push-button hardware resources.
00009   ******************************************************************************
00010   * @attention
00011   *
00012   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00013   *
00014   * Redistribution and use in source and binary forms, with or without modification,
00015   * are permitted provided that the following conditions are met:
00016   *   1. Redistributions of source code must retain the above copyright notice,
00017   *      this list of conditions and the fol
```

```
lowing disclaimer.
00018   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00019   *      this list of conditions and the fol
lowing disclaimer in the documentation
00020   *      and/or other materials provided wit
h the distribution.
00021   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00022   *      may be used to endorse or promote p
roducts derived from this software
00023   *      without specific prior written perm
ission.
00024   *
00025   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00026   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00027   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00028   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00029   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00030   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00031   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00032   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00033   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00034   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00035   *
00036   ****************************************
**********************************
```

```
00037   */
00038
00039 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00040 #ifndef __STM32F4_DISCOVERY_H
00041 #define __STM32F4_DISCOVERY_H
00042
00043 #ifdef __cplusplus
00044  extern "C" {
00045 #endif
00046

00047 /* Includes ------------------------------
-------------------------------------*/
00048 #include "stm32f4xx_hal.h"
00049
00050 /** @addtogroup BSP
00051   * @{
00052   */
00053
00054 /** @addtogroup STM32F4_DISCOVERY
00055   * @{
00056   */
00057
00058 /** @addtogroup STM32F4_DISCOVERY_LOW_LEVEL
00059   * @{
00060   */
00061
00062 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Ex
ported_Types STM32F4 DISCOVERY LOW LEVEL_Exported_
Types
00063   * @{
00064   */
00065 typedef enum
00066 {
00067   LED4 = 0,
00068   LED3 = 1,
```

```
00069   LED5 = 2,
00070   LED6 = 3
00071 } Led_TypeDef;
00072
00073 typedef enum
00074 {
00075   BUTTON_KEY = 0,
00076 } Button_TypeDef;
00077
00078 typedef enum
00079 {
00080   BUTTON_MODE_GPIO = 0,
00081   BUTTON_MODE_EXTI = 1
00082 } ButtonMode_TypeDef;
00083 /**
00084   * @}
00085   */
00086
00087 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Ex
ported_Constants STM32F4 DISCOVERY LOW LEVEL Expor
ted Constants
00088   * @{
00089   */
00090
00091 /**
00092 * @brief  Define for STM32F4_DISCOVERY board

00093 */
00094 #if !defined (USE_STM32F4_DISCO)
00095  #define USE_STM32F4_DISCO
00096 #endif
00097
00098 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_LE
D STM32F4 DISCOVERY LOW LEVEL LED
00099   * @{
00100   */
00101 #define LEDn                             4
```

```
00102
00103 #define LED4_PIN                                GPI
O_PIN_12
00104 #define LED4_GPIO_PORT                          GPI
OD
00105 #define LED4_GPIO_CLK_ENABLE()                  __H
AL_RCC_GPIOD_CLK_ENABLE()
00106 #define LED4_GPIO_CLK_DISABLE()                 __H
AL_RCC_GPIOD_CLK_DISABLE()
00107
00108 #define LED3_PIN                                GPI
O_PIN_13
00109 #define LED3_GPIO_PORT                          GPI
OD
00110 #define LED3_GPIO_CLK_ENABLE()                  __H
AL_RCC_GPIOD_CLK_ENABLE()
00111 #define LED3_GPIO_CLK_DISABLE()                 __H
AL_RCC_GPIOD_CLK_DISABLE()
00112
00113 #define LED5_PIN                                GPI
O_PIN_14
00114 #define LED5_GPIO_PORT                          GPI
OD
00115 #define LED5_GPIO_CLK_ENABLE()                  __H
AL_RCC_GPIOD_CLK_ENABLE()
00116 #define LED5_GPIO_CLK_DISABLE()                 __H
AL_RCC_GPIOD_CLK_DISABLE()
00117
00118 #define LED6_PIN                                GPI
O_PIN_15
00119 #define LED6_GPIO_PORT                          GPI
OD
00120 #define LED6_GPIO_CLK_ENABLE()                  __H
AL_RCC_GPIOD_CLK_ENABLE()
00121 #define LED6_GPIO_CLK_DISABLE()                 __H
AL_RCC_GPIOD_CLK_DISABLE()
00122
```

```
00123 #define LEDx_GPIO_CLK_ENABLE(__INDEX__) do{i
f((__INDEX__) == 0) LED4_GPIO_CLK_ENABLE(); else \
00124                                             i
f((__INDEX__) == 1) LED3_GPIO_CLK_ENABLE(); else \
00125                                             i
f((__INDEX__) == 2) LED5_GPIO_CLK_ENABLE(); else \
00126                                             i
f((__INDEX__) == 3) LED6_GPIO_CLK_ENABLE(); \
00127                                             }
while(0)
00128
00129 #define LEDx_GPIO_CLK_DISABLE(__INDEX__) do{
if((__INDEX__) == 0) LED4_GPIO_CLK_DISABLE(); else
 \
00130
if((__INDEX__) == 1) LED3_GPIO_CLK_DISABLE(); else
 \
00131
if((__INDEX__) == 2) LED5_GPIO_CLK_DISABLE(); else
 \
00132
if((__INDEX__) == 3) LED6_GPIO_CLK_DISABLE(); \
00133
}while(0)
00134 /**
00135   * @}
00136   */
00137
00138 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_BU
TTON STM32F4 DISCOVERY LOW LEVEL BUTTON
00139   * @{
00140   */
00141 #define BUTTONn                                 1
00142
00143 /**
00144  * @brief Wakeup push-button
00145  */
```

```
00146 #define KEY_BUTTON_PIN                 GPIO_P
IN_0
00147 #define KEY_BUTTON_GPIO_PORT           GPIOA
00148 #define KEY_BUTTON_GPIO_CLK_ENABLE()  __HAL_
RCC_GPIOA_CLK_ENABLE()
00149 #define KEY_BUTTON_GPIO_CLK_DISABLE() __HAL_
RCC_GPIOA_CLK_DISABLE()
00150 #define KEY_BUTTON_EXTI_IRQn          EXTI0_
IRQn
00151
00152 #define BUTTONx_GPIO_CLK_ENABLE(__INDEX__)
  do{if((__INDEX__) == 0) KEY_BUTTON_GPIO_CLK_ENAB
LE(); \
00153
    }while(0)
00154
00155 #define BUTTONx_GPIO_CLK_DISABLE(__INDEX__)
   do{if((__INDEX__) == 0) KEY_BUTTON_GPIO_CLK_DIS
ABLE(); \
00156
     }while(0)
00157 /**
00158   * @}
00159   */
00160
00161 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_BU
S STM32F4 DISCOVERY LOW LEVEL BUS
00162   * @{
00163   */
00164
00165 /*################################## SPI1 #####
####################################*/
00166 #define DISCOVERY_SPIx
        SPI1
00167 #define DISCOVERY_SPIx_CLK_ENABLE()
        __HAL_RCC_SPI1_CLK_ENABLE()
00168 #define DISCOVERY_SPIx_GPIO_PORT
```

```
        GPIOA                           /* GPIOA */
00169 #define DISCOVERY_SPIx_AF
        GPIO_AF5_SPI1
00170 #define DISCOVERY_SPIx_GPIO_CLK_ENABLE()
        __HAL_RCC_GPIOA_CLK_ENABLE()
00171 #define DISCOVERY_SPIx_GPIO_CLK_DISABLE()
        __HAL_RCC_GPIOA_CLK_DISABLE()
00172 #define DISCOVERY_SPIx_SCK_PIN
        GPIO_PIN_5                      /* PA.05 */
00173 #define DISCOVERY_SPIx_MISO_PIN
        GPIO_PIN_6                      /* PA.06 */
00174 #define DISCOVERY_SPIx_MOSI_PIN
        GPIO_PIN_7                      /* PA.07 */
00175
00176 /* Maximum Timeout values for flags waiting
loops. These timeouts are not based
00177    on accurate values, they just guarantee t
hat the application will not remain
00178    stuck if the SPI communication is corrupt
ed.
00179    You may modify these timeout values depen
ding on CPU frequency and application
00180    conditions (interrupts routines ...). */

00181 #define SPIx_TIMEOUT_MAX
        0x1000 /*<! The value of the maximal timeo
ut for BUS waiting loops */
00182
00183
00184 /*################################ I2C1 #######
####################################*/
00185 /* I2C clock speed configuration (in Hz) */
00186 #ifndef BSP_I2C_SPEED
00187  #define BSP_I2C_SPEED
      100000
00188 #endif /* BSP_I2C_SPEED */
00189
```

```
00190 /* I2C peripheral configuration defines (con
trol interface of the audio codec) */
00191 #define DISCOVERY_I2Cx
      I2C1
00192 #define DISCOVERY_I2Cx_CLK_ENABLE()
      __HAL_RCC_I2C1_CLK_ENABLE()
00193 #define DISCOVERY_I2Cx_SCL_SDA_GPIO_CLK_ENAB
LE()  __HAL_RCC_GPIOB_CLK_ENABLE()
00194 #define DISCOVERY_I2Cx_SCL_SDA_AF
      GPIO_AF4_I2C1
00195 #define DISCOVERY_I2Cx_SCL_SDA_GPIO_PORT
      GPIOB
00196 #define DISCOVERY_I2Cx_SCL_PIN
      GPIO_PIN_6
00197 #define DISCOVERY_I2Cx_SDA_PIN
      GPIO_PIN_9
00198
00199 #define DISCOVERY_I2Cx_FORCE_RESET()
      __HAL_RCC_I2C1_FORCE_RESET()
00200 #define DISCOVERY_I2Cx_RELEASE_RESET()
      __HAL_RCC_I2C1_RELEASE_RESET()
00201
00202 /* I2C interrupt requests */

00203 #define DISCOVERY_I2Cx_EV_IRQn
      I2C1_EV_IRQn
00204 #define DISCOVERY_I2Cx_ER_IRQn
      I2C1_ER_IRQn
00205
00206 /* Maximum Timeout values for flags waiting
loops. These timeouts are not based
00207    on accurate values, they just guarantee t
hat the application will not remain
00208    stuck if the SPI communication is corrupt
ed.
00209    You may modify these timeout values depen
ding on CPU frequency and application
```

```
00210    conditions (interrupts routines ...). */

00211 #define I2Cx_TIMEOUT_MAX    0x1000 /*<! The
value of the maximal timeout for BUS waiting loops
 */
00212
00213
00214 /*################################ ACCELEROMETE
R ###################################*/
00215 /* Read/Write command */
00216 #define READWRITE_CMD                     ((
uint8_t)0x80)
00217 /* Multiple byte read/write command */
00218 #define MULTIPLEBYTE_CMD                  ((
uint8_t)0x40)
00219 /* Dummy Byte Send by the SPI Master device
in order to generate the Clock to the Slave device
 */
00220 #define DUMMY_BYTE                        ((
uint8_t)0x00)
00221
00222 /* Chip Select macro definition */
00223 #define ACCELERO_CS_LOW()       HAL_GPIO_Wri
tePin(ACCELERO_CS_GPIO_PORT, ACCELERO_CS_PIN, GPIO
_PIN_RESET)
00224 #define ACCELERO_CS_HIGH()      HAL_GPIO_Wri
tePin(ACCELERO_CS_GPIO_PORT, ACCELERO_CS_PIN, GPIO
_PIN_SET)
00225
00226 /**
00227   * @brief  ACCELEROMETER Interface pins
00228   */
00229 #define ACCELERO_CS_PIN
   GPIO_PIN_3                 /* PE.03 */
00230 #define ACCELERO_CS_GPIO_PORT
   GPIOE                      /* GPIOE */
00231 #define ACCELERO_CS_GPIO_CLK_ENABLE()
```

```
   __HAL_RCC_GPIOE_CLK_ENABLE()
00232 #define ACCELERO_CS_GPIO_CLK_DISABLE()
   __HAL_RCC_GPIOE_CLK_DISABLE()
00233 #define ACCELERO_INT_GPIO_PORT
   GPIOE                        /* GPIOE */
00234 #define ACCELERO_INT_GPIO_CLK_ENABLE()
   __HAL_RCC_GPIOE_CLK_ENABLE()
00235 #define ACCELERO_INT_GPIO_CLK_DISABLE()
   __HAL_RCC_GPIOE_CLK_DISABLE()
00236 #define ACCELERO_INT1_PIN
   GPIO_PIN_0                   /* PE.00 */
00237 #define ACCELERO_INT1_EXTI_IRQn
   EXTI0_IRQn
00238 #define ACCELERO_INT2_PIN
   GPIO_PIN_1                   /* PE.01 */
00239 #define ACCELERO_INT2_EXTI_IRQn
   EXTI1_IRQn
00240 /**
00241   * @}
00242   */
00243
00244
00245 /*################################## AUDIO ####
####################################*/
00246 /**
00247   * @brief  AUDIO I2C Interface pins
00248   */
00249 #define AUDIO_I2C_ADDRESS
  0x94
00250
00251   /* Audio Reset Pin definition */
00252 #define AUDIO_RESET_GPIO_CLK_ENABLE()
  __HAL_RCC_GPIOD_CLK_ENABLE()
00253 #define AUDIO_RESET_PIN
  GPIO_PIN_4
00254 #define AUDIO_RESET_GPIO
  GPIOD
```

```
00255 /**
00256   * @}
00257   */
00258
00259 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Ex
ported_Macros STM32F4 DISCOVERY LOW LEVEL Exported
 Macros
00260   * @{
00261   */
00262 /**
00263   * @}
00264   */
00265
00266 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Ex
ported_Functions STM32F4 DISCOVERY LOW LEVEL Expor
ted Functions
00267   * @{
00268   */
00269 uint32_t BSP_GetVersion(void);
00270 void     BSP_LED_Init(Led_TypeDef Led);
00271 void     BSP_LED_On(Led_TypeDef Led);
00272 void     BSP_LED_Off(Led_TypeDef Led);
00273 void     BSP_LED_Toggle(Led_TypeDef Led);
00274 void     BSP_PB_Init(Button_TypeDef Button,
ButtonMode_TypeDef Mode);
00275 uint32_t BSP_PB_GetState(Button_TypeDef Butt
on);
00276
00277 /**
00278   * @}
00279   */
00280
00281 /**
00282   * @}
00283   */
00284
00285 /**
```

```
00286   * @}
00287   */
00288
00289 /**
00290   * @}
00291   */
00292
00293 /**
00294   * @}
00295   */
00296
00297 #ifdef __cplusplus
00298 }
00299 #endif
00300
00301 #endif /* __STM32F4_DISCOVERY_H */
00302
00303 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F4-Discovery BSP User Manual

stm32f4_discovery.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32f4_discovery.c
00004   * @author  MCD Application Team
00005   * @version V2.1.2
00006   * @date    31-January-2017
00007   * @brief   This file provides set of firmware functions to manage Leds and
00008   *          push-button available on STM32F4-Discovery Kit from STMicroelectronics.
00009   ******************************************************************************
00010   * @attention
00011   *
00012   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00013   *
00014   * Redistribution and use in source and binary forms, with or without modification,
00015   * are permitted provided that the following conditions are met:
00016   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00017   *      this list of conditions and the fol
lowing disclaimer.
00018   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00019   *      this list of conditions and the fol
lowing disclaimer in the documentation
00020   *      and/or other materials provided wit
h the distribution.
00021   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00022   *      may be used to endorse or promote p
roducts derived from this software
00023   *      without specific prior written perm
ission.
00024   *
00025   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00026   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00027   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00028   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00029   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00030   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00031   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00032   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00033   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00034   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00035   *
00036   ****************************************
```

```
**********************************
00037   */
00038 /* Includes ------------------------------
-----------------------------------*/
00039 #include "stm32f4_discovery.h"
00040
00041 /** @defgroup BSP BSP
00042   * @{
00043   */
00044
00045 /** @defgroup STM32F4_DISCOVERY STM32F4 DISC
OVERY
00046   * @{
00047   */
00048
00049 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL ST
M32F4 DISCOVERY LOW LEVEL
00050   * @brief This file provides set of firmwar
e functions to manage Leds and push-button
00051   *        available on STM32F4-Discovery Ki
t from STMicroelectronics.
00052   * @{
00053   */
00054
00055 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Pr
ivate_TypesDefinitions STM32F4 DISCOVERY LOW LEVEL
 Private TypesDefinitions
00056   * @{
00057   */
00058 /**
00059   * @}
00060   */
00061
00062 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Pr
ivate_Defines STM32F4 DISCOVERY LOW LEVEL Private
Defines
00063   * @{
```

```
00064   */
00065
00066   /**
00067   * @brief STM32F4 DISCO BSP Driver version
number V2.1.2
00068   */
00069 #define __STM32F4_DISCO_BSP_VERSION_MAIN   (
0x02) /*!< [31:24] main version */
00070 #define __STM32F4_DISCO_BSP_VERSION_SUB1   (
0x01) /*!< [23:16] sub1 version */
00071 #define __STM32F4_DISCO_BSP_VERSION_SUB2   (
0x02) /*!< [15:8]  sub2 version */
00072 #define __STM32F4_DISCO_BSP_VERSION_RC     (
0x00) /*!< [7:0]  release candidate */
00073 #define __STM32F4_DISCO_BSP_VERSION
((__STM32F4_DISCO_BSP_VERSION_MAIN << 24)\
00074
 |(__STM32F4_DISCO_BSP_VERSION_SUB1 << 16)\
00075
 |(__STM32F4_DISCO_BSP_VERSION_SUB2 << 8 )\
00076
 |(__STM32F4_DISCO_BSP_VERSION_RC))
00077 /**
00078   * @}
00079   */
00080
00081
00082 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Pr
ivate_Macros STM32F4 DISCOVERY LOW LEVEL Private M
acros
00083   * @{
00084   */
00085 /**
00086   * @}
00087   */
00088
00089 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Pr
```

```
ivate_Variables STM32F4 DISCOVERY LOW LEVEL Privat
e Variables
00090   * @{
00091   */
00092 GPIO_TypeDef* GPIO_PORT[LEDn] = {LED4_GPIO_P
ORT,
00093                                  LED3_GPIO_P
ORT,
00094                                  LED5_GPIO_P
ORT,
00095                                  LED6_GPIO_P
ORT};
00096 const uint16_t GPIO_PIN[LEDn] = {LED4_PIN,
00097                                  LED3_PIN,
00098                                  LED5_PIN,
00099                                  LED6_PIN};
00100
00101 GPIO_TypeDef* BUTTON_PORT[BUTTONn] = {KEY_BU
TTON_GPIO_PORT};
00102 const uint16_t BUTTON_PIN[BUTTONn] = {KEY_BU
TTON_PIN};
00103 const uint8_t BUTTON_IRQn[BUTTONn] = {KEY_BU
TTON_EXTI_IRQn};
00104
00105 uint32_t I2cxTimeout = I2Cx_TIMEOUT_MAX;
/*<! Value of Timeout when I2C communication fails
 */
00106 uint32_t SpixTimeout = SPIx_TIMEOUT_MAX;
/*<! Value of Timeout when SPI communication fails
 */
00107
00108 static SPI_HandleTypeDef    SpiHandle;
00109 static I2C_HandleTypeDef    I2cHandle;
00110 /**
00111   * @}
00112   */
00113
```

```
00114 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Pr
ivate_FunctionPrototypes STM32F4 DISCOVERY LOW LEV
EL Private FunctionPrototypes
00115   * @{
00116   */
00117 /**
00118   * @}
00119   */
00120
00121 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_Pr
ivate_Functions STM32F4 DISCOVERY LOW LEVEL Privat
e Functions
00122   * @{
00123   */
00124 static void     I2Cx_Init(void);
00125 static void     I2Cx_WriteData(uint8_t Addr,
 uint8_t Reg, uint8_t Value);
00126 static uint8_t  I2Cx_ReadData(uint8_t Addr,
uint8_t Reg);
00127 static void     I2Cx_MspInit(void);
00128 static void     I2Cx_Error(uint8_t Addr);
00129
00130 static void     SPIx_Init(void);
00131 static void     SPIx_MspInit(void);
00132 static uint8_t  SPIx_WriteRead(uint8_t Byte)
;
00133 static  void    SPIx_Error(void);
00134
00135 /* Link functions for Accelerometer peripher
al */
00136 void            ACCELERO_IO_Init(void);
00137 void            ACCELERO_IO_ITConfig(void);
00138 void            ACCELERO_IO_Write(uint8_t *p
Buffer, uint8_t WriteAddr, uint16_t NumByteToWrite
);
00139 void            ACCELERO_IO_Read(uint8_t *pB
uffer, uint8_t ReadAddr, uint16_t NumByteToRead);
```

```
00140
00141 /* Link functions for Audio peripheral */
00142 void            AUDIO_IO_Init(void);
00143 void            AUDIO_IO_DeInit(void);
00144 void            AUDIO_IO_Write(uint8_t Addr,
 uint8_t Reg, uint8_t Value);
00145 uint8_t         AUDIO_IO_Read(uint8_t Addr,
uint8_t Reg);
00146 /**
00147   * @}
00148   */
00149
00150 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_LE
D_Functions STM32F4 DISCOVERY LOW LEVEL LED Functi
ons
00151   * @{
00152   */
00153
00154 /**
00155   * @brief  This method returns the STM32F4
DISCO BSP Driver revision
00156   * @retval version : 0xXYZR (8bits for each
 decimal, R for RC)
00157   */
00158 uint32_t BSP_GetVersion(void)
00159 {
00160   return __STM32F4_DISCO_BSP_VERSION;
00161 }
00162
00163 /**
00164   * @brief  Configures LED GPIO.
00165   * @param  Led: Specifies the Led to be con
figured.
00166   *   This parameter can be one of following
 parameters:
00167   *     @arg LED4
00168   *     @arg LED3
```

```
00169   *     @arg LED5
00170   *     @arg LED6
00171   */
00172 void BSP_LED_Init(Led_TypeDef Led)
00173 {
00174   GPIO_InitTypeDef  GPIO_InitStruct;
00175
00176   /* Enable the GPIO_LED Clock */
00177   LEDx_GPIO_CLK_ENABLE(Led);
00178
00179   /* Configure the GPIO_LED pin */
00180   GPIO_InitStruct.Pin = GPIO_PIN[Led];
00181   GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP
;
00182   GPIO_InitStruct.Pull = GPIO_PULLUP;
00183   GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
00184
00185   HAL_GPIO_Init(GPIO_PORT[Led], &GPIO_InitSt
ruct);
00186
00187   HAL_GPIO_WritePin(GPIO_PORT[Led], GPIO_PIN
[Led], GPIO_PIN_RESET);
00188 }
00189
00190 /**
00191   * @brief  Turns selected LED On.
00192   * @param  Led: Specifies the Led to be set
 on.
00193   *   This parameter can be one of following
 parameters:
00194   *     @arg LED4
00195   *     @arg LED3
00196   *     @arg LED5
00197   *     @arg LED6
00198   */
00199 void BSP_LED_On(Led_TypeDef Led)
00200 {
```

```
00201   HAL_GPIO_WritePin(GPIO_PORT[Led], GPIO_PIN
[Led], GPIO_PIN_SET);
00202 }
00203
00204 /**
00205   * @brief  Turns selected LED Off.
00206   * @param  Led: Specifies the Led to be set
 off.
00207   *   This parameter can be one of following
 parameters:
00208   *     @arg LED4
00209   *     @arg LED3
00210   *     @arg LED5
00211   *     @arg LED6
00212   */
00213 void BSP_LED_Off(Led_TypeDef Led)
00214 {
00215   HAL_GPIO_WritePin(GPIO_PORT[Led], GPIO_PIN
[Led], GPIO_PIN_RESET);
00216 }
00217
00218 /**
00219   * @brief  Toggles the selected LED.
00220   * @param  Led: Specifies the Led to be tog
gled.
00221   *   This parameter can be one of following
 parameters:
00222   *     @arg LED4
00223   *     @arg LED3
00224   *     @arg LED5
00225   *     @arg LED6
00226   */
00227 void BSP_LED_Toggle(Led_TypeDef Led)
00228 {
00229   HAL_GPIO_TogglePin(GPIO_PORT[Led], GPIO_PIN
[Led]);
00230 }
```

```
00231
00232 /**
00233   * @}
00234   */
00235
00236 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_BU
TTON_Functions STM32F4 DISCOVERY LOW LEVEL BUTTON
Functions
00237   * @{
00238   */
00239
00240 /**
00241   * @brief  Configures Button GPIO and EXTI
Line.
00242   * @param  Button: Specifies the Button to
be configured.
00243   *   This parameter should be: BUTTON_KEY
00244   * @param  Mode: Specifies Button mode.
00245   *   This parameter can be one of following
 parameters:
00246   *     @arg BUTTON_MODE_GPIO: Button will b
e used as simple IO
00247   *     @arg BUTTON_MODE_EXTI: Button will b
e connected to EXTI line with interrupt
00248   *                                generation ca
pability
00249   */
00250 void BSP_PB_Init(Button_TypeDef Button, Butt
onMode_TypeDef Mode)
00251 {
00252   GPIO_InitTypeDef GPIO_InitStruct;
00253
00254   /* Enable the BUTTON Clock */
00255   BUTTONx_GPIO_CLK_ENABLE(Button);
00256
00257   if (Mode == BUTTON_MODE_GPIO)
00258   {
```

```
00259     /* Configure Button pin as input */
00260     GPIO_InitStruct.Pin = BUTTON_PIN[Button]
;
00261     GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
00262     GPIO_InitStruct.Pull = GPIO_NOPULL;
00263     GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
00264
00265     HAL_GPIO_Init(BUTTON_PORT[Button], &GPIO
_InitStruct);
00266   }
00267
00268   if (Mode == BUTTON_MODE_EXTI)
00269   {
00270     /* Configure Button pin as input with Ex
ternal interrupt */
00271     GPIO_InitStruct.Pin = BUTTON_PIN[Button]
;
00272     GPIO_InitStruct.Pull = GPIO_NOPULL;
00273     GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
00274     GPIO_InitStruct.Mode = GPIO_MODE_IT_RISI
NG;
00275     HAL_GPIO_Init(BUTTON_PORT[Button], &GPIO
_InitStruct);
00276
00277     /* Enable and set Button EXTI Interrupt
to the lowest priority */
00278     HAL_NVIC_SetPriority((IRQn_Type)(BUTTON_
IRQn[Button]), 0x0F, 0);
00279     HAL_NVIC_EnableIRQ((IRQn_Type)(BUTTON_IR
Qn[Button]));
00280   }
00281 }
00282
00283 /**
00284   * @brief  Returns the selected Button stat
e.
00285   * @param  Button: Specifies the Button to
```

```
be checked.
00286   *   This parameter should be: BUTTON_KEY
00287   * @retval The Button GPIO pin value.
00288   */
00289 uint32_t BSP_PB_GetState(Button_TypeDef Butt
on)
00290 {
00291   return HAL_GPIO_ReadPin(BUTTON_PORT[Button
], BUTTON_PIN[Button]);
00292 }
00293
00294 /**
00295   * @}
00296   */
00297
00298 /** @defgroup STM32F4_DISCOVERY_LOW_LEVEL_BU
S_Functions STM32F4 DISCOVERY LOW LEVEL BUS Functi
ons
00299   * @{
00300   */
00301
00302 /*******************************************
**********************************
00303                                     BUS OPERATIONS
00304 *******************************************
**********************************/
00305
00306 /****************************** SPI Routine
s ******************************/
00307
00308 /**
00309   * @brief  SPIx Bus initialization
00310   */
00311 static void SPIx_Init(void)
00312 {
00313   if(HAL_SPI_GetState(&SpiHandle) == HAL_SPI
_STATE_RESET)
```

```
00314   {
00315     /* SPI configuration ------------------
-----------------------------------*/
00316     SpiHandle.Instance = DISCOVERY_SPIx;
00317     SpiHandle.Init.BaudRatePrescaler = SPI_B
AUDRATEPRESCALER_16;
00318     SpiHandle.Init.Direction = SPI_DIRECTION
_2LINES;
00319     SpiHandle.Init.CLKPhase = SPI_PHASE_1EDG
E;
00320     SpiHandle.Init.CLKPolarity = SPI_POLARIT
Y_LOW;
00321     SpiHandle.Init.CRCCalculation = SPI_CRCC
ALCULATION_DISABLED;
00322     SpiHandle.Init.CRCPolynomial = 7;
00323     SpiHandle.Init.DataSize = SPI_DATASIZE_8
BIT;
00324     SpiHandle.Init.FirstBit = SPI_FIRSTBIT_M
SB;
00325     SpiHandle.Init.NSS = SPI_NSS_SOFT;
00326     SpiHandle.Init.TIMode = SPI_TIMODE_DISAB
LED;
00327     SpiHandle.Init.Mode = SPI_MODE_MASTER;
00328
00329     SPIx_MspInit();
00330     HAL_SPI_Init(&SpiHandle);
00331   }
00332 }
00333
00334 /**
00335   * @brief  Sends a Byte through the SPI int
erface and return the Byte received
00336   *         from the SPI bus.
00337   * @param  Byte: Byte send.
00338   * @retval The received byte value
00339   */
00340 static uint8_t SPIx_WriteRead(uint8_t Byte)
```

```
00341 {
00342   uint8_t receivedbyte = 0;
00343 
00344   /* Send a Byte through the SPI peripheral
*/
00345   /* Read byte from the SPI bus */
00346   if(HAL_SPI_TransmitReceive(&SpiHandle, (ui
nt8_t*) &Byte, (uint8_t*) &receivedbyte, 1, SpixTi
meout) != HAL_OK)
00347   {
00348     SPIx_Error();
00349   }
00350 
00351   return receivedbyte;
00352 }
00353 
00354 /**
00355   * @brief  SPIx error treatment function.
00356   */
00357 static void SPIx_Error(void)
00358 {
00359   /* De-initialize the SPI communication bus
 */
00360   HAL_SPI_DeInit(&SpiHandle);
00361 
00362   /* Re-Initialize the SPI communication bus
 */
00363   SPIx_Init();
00364 }
00365 
00366 /**
00367   * @brief  SPI MSP Init.
00368   */
00369 static void SPIx_MspInit(void)
00370 {
00371   GPIO_InitTypeDef   GPIO_InitStructure;
00372 
```

```
00373   /* Enable the SPI peripheral */
00374   DISCOVERY_SPIx_CLK_ENABLE();
00375
00376   /* Enable SCK, MOSI and MISO GPIO clocks */

00377   DISCOVERY_SPIx_GPIO_CLK_ENABLE();
00378
00379   /* SPI SCK, MOSI, MISO pin configuration */

00380   GPIO_InitStructure.Pin = (DISCOVERY_SPIx_S
CK_PIN | DISCOVERY_SPIx_MISO_PIN | DISCOVERY_SPIx_
MOSI_PIN);
00381   GPIO_InitStructure.Mode = GPIO_MODE_AF_PP;
00382   GPIO_InitStructure.Pull  = GPIO_PULLDOWN;
00383   GPIO_InitStructure.Speed = GPIO_SPEED_MEDI
UM;
00384   GPIO_InitStructure.Alternate = DISCOVERY_S
PIx_AF;
00385   HAL_GPIO_Init(DISCOVERY_SPIx_GPIO_PORT, &G
PIO_InitStructure);
00386 }
00387
00388 /****************************** I2C Routine
s******************************/
00389 /**
00390   * @brief  Configures I2C interface.
00391   */
00392 static void I2Cx_Init(void)
00393 {
00394   if(HAL_I2C_GetState(&I2cHandle) == HAL_I2C
_STATE_RESET)
00395   {
00396     /* DISCOVERY_I2Cx peripheral configurati
on */
00397     I2cHandle.Init.ClockSpeed = BSP_I2C_SPEE
D;
00398     I2cHandle.Init.DutyCycle = I2C_DUTYCYCLE
```

```
_2;
00399     I2cHandle.Init.OwnAddress1 = 0x33;
00400     I2cHandle.Init.AddressingMode = I2C_ADDR
ESSINGMODE_7BIT;
00401     I2cHandle.Instance = DISCOVERY_I2Cx;
00402
00403     /* Init the I2C */
00404     I2Cx_MspInit();
00405     HAL_I2C_Init(&I2cHandle);
00406   }
00407 }
00408
00409 /**
00410   * @brief  Write a value in a register of t
he device through BUS.
00411   * @param  Addr: Device address on BUS Bus.

00412   * @param  Reg: The target register address
 to write
00413   * @param  Value: The target register value
 to be written
00414   * @retval HAL status
00415   */
00416 static void I2Cx_WriteData(uint8_t Addr, uin
t8_t Reg, uint8_t Value)
00417 {
00418   HAL_StatusTypeDef status = HAL_OK;
00419
00420   status = HAL_I2C_Mem_Write(&I2cHandle, Add
r, (uint16_t)Reg, I2C_MEMADD_SIZE_8BIT, &Value, 1,
I2cxTimeout);
00421
00422   /* Check the communication status */
00423   if(status != HAL_OK)
00424   {
00425     /* Execute user timeout callback */
00426     I2Cx_Error(Addr);
```

```
00427   }
00428 }
00429
00430 /**
00431   * @brief  Read a register of the device th
rough BUS
00432   * @param  Addr: Device address on BUS
00433   * @param  Reg: The target register address
 to read
00434   * @retval HAL status
00435   */
00436 static uint8_t  I2Cx_ReadData(uint8_t Addr,
uint8_t Reg)
00437 {
00438   HAL_StatusTypeDef status = HAL_OK;
00439   uint8_t value = 0;
00440
00441   status = HAL_I2C_Mem_Read(&I2cHandle, Addr
, (uint16_t)Reg, I2C_MEMADD_SIZE_8BIT, &value, 1,I
2cxTimeout);
00442
00443   /* Check the communication status */
00444   if(status != HAL_OK)
00445   {
00446     /* Execute user timeout callback */
00447     I2Cx_Error(Addr);
00448   }
00449   return value;
00450 }
00451
00452 /**
00453   * @brief  Manages error callback by re-ini
tializing I2C.
00454   * @param  Addr: I2C Address
00455   */
00456 static void I2Cx_Error(uint8_t Addr)
00457 {
```

```
00458   /* De-initialize the I2C communication bus
 */
00459   HAL_I2C_DeInit(&I2cHandle);
00460
00461   /* Re-Initialize the I2C communication bus
 */
00462   I2Cx_Init();
00463 }
00464
00465 /**
00466   * @brief I2C MSP Initialization
00467   */
00468 static void I2Cx_MspInit(void)
00469 {
00470   GPIO_InitTypeDef  GPIO_InitStruct;
00471
00472   /* Enable I2C GPIO clocks */
00473   DISCOVERY_I2Cx_SCL_SDA_GPIO_CLK_ENABLE();
00474
00475   /* DISCOVERY_I2Cx SCL and SDA pins configu
ration -------------------------*/
00476   GPIO_InitStruct.Pin = DISCOVERY_I2Cx_SCL_P
IN | DISCOVERY_I2Cx_SDA_PIN;
00477   GPIO_InitStruct.Mode = GPIO_MODE_AF_OD;
00478   GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
00479   GPIO_InitStruct.Pull  = GPIO_NOPULL;
00480   GPIO_InitStruct.Alternate  = DISCOVERY_I2C
x_SCL_SDA_AF;
00481   HAL_GPIO_Init(DISCOVERY_I2Cx_SCL_SDA_GPIO_
PORT, &GPIO_InitStruct);
00482
00483   /* Enable the DISCOVERY_I2Cx peripheral cl
ock */
00484   DISCOVERY_I2Cx_CLK_ENABLE();
00485
00486   /* Force the I2C peripheral clock reset */
00487   DISCOVERY_I2Cx_FORCE_RESET();
```

```
00488
00489   /* Release the I2C peripheral clock reset
*/
00490   DISCOVERY_I2Cx_RELEASE_RESET();
00491
00492   /* Enable and set I2Cx Interrupt to the hi
ghest priority */
00493   HAL_NVIC_SetPriority(DISCOVERY_I2Cx_EV_IRQn
, 0, 0);
00494   HAL_NVIC_EnableIRQ(DISCOVERY_I2Cx_EV_IRQn)
;
00495
00496   /* Enable and set I2Cx Interrupt to the hi
ghest priority */
00497   HAL_NVIC_SetPriority(DISCOVERY_I2Cx_ER_IRQn
, 0, 0);
00498   HAL_NVIC_EnableIRQ(DISCOVERY_I2Cx_ER_IRQn)
;
00499 }
00500
00501 /**********************************************
**********************************
00502                                   LINK OPERATIONS
00503 **********************************************
**********************************/
00504
00505 /****************************** LINK ACCELERO
METER ******************************/
00506
00507 /**
00508   * @brief  Configures the Accelerometer SPI
 interface.
00509   */
00510 void ACCELERO_IO_Init(void)
00511 {
00512   GPIO_InitTypeDef GPIO_InitStructure;
00513
```

```
00514   /* Configure the Accelerometer Control pin
s ------------------------------*/
00515   /* Enable CS GPIO clock and configure GPIO
 pin for Accelerometer Chip select */
00516   ACCELERO_CS_GPIO_CLK_ENABLE();
00517
00518   /* Configure GPIO PIN for LIS Chip select
*/
00519   GPIO_InitStructure.Pin = ACCELERO_CS_PIN;
00520   GPIO_InitStructure.Mode = GPIO_MODE_OUTPUT
_PP;
00521   GPIO_InitStructure.Pull  = GPIO_NOPULL;
00522   GPIO_InitStructure.Speed = GPIO_SPEED_MEDI
UM;
00523   HAL_GPIO_Init(ACCELERO_CS_GPIO_PORT, &GPIO
_InitStructure);
00524
00525   /* Deselect: Chip Select high */
00526   ACCELERO_CS_HIGH();
00527
00528   SPIx_Init();
00529 }
00530
00531 /**
00532   * @brief  Configures the Accelerometer INT
2.
00533   *         EXTI0 is already used by user bu
tton so INT1 is not configured here.
00534   */
00535 void ACCELERO_IO_ITConfig(void)
00536 {
00537   GPIO_InitTypeDef GPIO_InitStructure;
00538
00539   /* Enable INT2 GPIO clock and configure GP
IO PINs to detect Interrupts */
00540   ACCELERO_INT_GPIO_CLK_ENABLE();
00541
```

```
00542   /* Configure GPIO PINs to detect Interrupt
s */
00543   GPIO_InitStructure.Pin = ACCELERO_INT2_PIN
;
00544   GPIO_InitStructure.Mode = GPIO_MODE_IT_RIS
ING;
00545   GPIO_InitStructure.Speed = GPIO_SPEED_FAST
;
00546   GPIO_InitStructure.Pull = GPIO_NOPULL;
00547   HAL_GPIO_Init(ACCELERO_INT_GPIO_PORT, &GPI
O_InitStructure);
00548
00549   /* Enable and set Accelerometer INT2 to th
e lowest priority */
00550   HAL_NVIC_SetPriority((IRQn_Type)ACCELERO_I
NT2_EXTI_IRQn, 0x0F, 0);
00551   HAL_NVIC_EnableIRQ((IRQn_Type)ACCELERO_INT
2_EXTI_IRQn);
00552 }
00553
00554 /**
00555   * @brief  Writes one byte to the Accelerom
eter.
00556   * @param  pBuffer: pointer to the buffer c
ontaining the data to be written to the Accelerome
ter.
00557   * @param  WriteAddr: Accelerometer's inter
nal address to write to.
00558   * @param  NumByteToWrite: Number of bytes
to write.
00559   */
00560 void ACCELERO_IO_Write(uint8_t *pBuffer, uin
t8_t WriteAddr, uint16_t NumByteToWrite)
00561 {
00562   /* Configure the MS bit:
00563      - When 0, the address will remain uncha
nged in multiple read/write commands.
```

```
00564       - When 1, the address will be auto incr
emented in multiple read/write commands.
00565   */
00566   if(NumByteToWrite > 0x01)
00567   {
00568     WriteAddr |= (uint8_t)MULTIPLEBYTE_CMD;
00569   }
00570   /* Set chip select Low at the start of the
 transmission */
00571   ACCELERO_CS_LOW();
00572
00573   /* Send the Address of the indexed registe
r */
00574   SPIx_WriteRead(WriteAddr);
00575
00576   /* Send the data that will be written into
 the device (MSB First) */
00577   while(NumByteToWrite >= 0x01)
00578   {
00579     SPIx_WriteRead(*pBuffer);
00580     NumByteToWrite--;
00581     pBuffer++;
00582   }
00583
00584   /* Set chip select High at the end of the
transmission */
00585   ACCELERO_CS_HIGH();
00586 }
00587
00588 /**
00589   * @brief  Reads a block of data from the A
ccelerometer.
00590   * @param  pBuffer: pointer to the buffer t
hat receives the data read from the Accelerometer.
00591   * @param  ReadAddr: Accelerometer's intern
al address to read from.
00592   * @param  NumByteToRead: number of bytes t
```

```
o read from the Accelerometer.
00593   */
00594 void ACCELERO_IO_Read(uint8_t *pBuffer, uint
8_t ReadAddr, uint16_t NumByteToRead)
00595 {
00596   if(NumByteToRead > 0x01)
00597   {
00598     ReadAddr |= (uint8_t)(READWRITE_CMD | MU
LTIPLEBYTE_CMD);
00599   }
00600   else
00601   {
00602     ReadAddr |= (uint8_t)READWRITE_CMD;
00603   }
00604   /* Set chip select Low at the start of the
 transmission */
00605   ACCELERO_CS_LOW();
00606
00607   /* Send the Address of the indexed registe
r */
00608   SPIx_WriteRead(ReadAddr);
00609
00610   /* Receive the data that will be read from
 the device (MSB First) */
00611   while(NumByteToRead > 0x00)
00612   {
00613     /* Send dummy byte (0x00) to generate th
e SPI clock to ACCELEROMETER (Slave device) */
00614     *pBuffer = SPIx_WriteRead(DUMMY_BYTE);
00615     NumByteToRead--;
00616     pBuffer++;
00617   }
00618
00619   /* Set chip select High at the end of the
transmission */
00620   ACCELERO_CS_HIGH();
00621 }
```

```
00622
00623 /********************************** LINK AUDI
O ***********************************/
00624
00625 /**
00626   * @brief  Initializes Audio low level.
00627   */
00628 void AUDIO_IO_Init(void)
00629 {
00630   GPIO_InitTypeDef  GPIO_InitStruct;
00631
00632   /* Enable Reset GPIO Clock */
00633   AUDIO_RESET_GPIO_CLK_ENABLE();
00634
00635   /* Audio reset pin configuration */
00636   GPIO_InitStruct.Pin = AUDIO_RESET_PIN;
00637   GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP
;
00638   GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
00639   GPIO_InitStruct.Pull  = GPIO_NOPULL;
00640   HAL_GPIO_Init(AUDIO_RESET_GPIO, &GPIO_Init
Struct);
00641
00642   I2Cx_Init();
00643
00644   /* Power Down the codec */
00645   HAL_GPIO_WritePin(AUDIO_RESET_GPIO, AUDIO_
RESET_PIN, GPIO_PIN_RESET);
00646
00647   /* Wait for a delay to insure registers er
asing */
00648   HAL_Delay(5);
00649
00650   /* Power on the codec */
00651   HAL_GPIO_WritePin(AUDIO_RESET_GPIO, AUDIO_
RESET_PIN, GPIO_PIN_SET);
00652
```

```
00653   /* Wait for a delay to insure registers er
asing */
00654   HAL_Delay(5);
00655 }
00656
00657 /**
00658   * @brief  DeInitializes Audio low level.
00659   */
00660 void AUDIO_IO_DeInit(void)
00661 {
00662
00663 }
00664
00665 /**
00666   * @brief  Writes a single data.
00667   * @param  Addr: I2C address
00668   * @param  Reg: Reg address
00669   * @param  Value: Data to be written
00670   */
00671 void AUDIO_IO_Write (uint8_t Addr, uint8_t R
eg, uint8_t Value)
00672 {
00673   I2Cx_WriteData(Addr, Reg, Value);
00674 }
00675
00676 /**
00677   * @brief  Reads a single data.
00678   * @param  Addr: I2C address
00679   * @param  Reg: Reg address
00680   * @retval Data to be read
00681   */
00682 uint8_t AUDIO_IO_Read(uint8_t Addr, uint8_t
Reg)
00683 {
00684   return I2Cx_ReadData(Addr, Reg);
00685 }
00686
```

```
00687 /**
00688   * @}
00689   */
00690
00691 /**
00692   * @}
00693   */
00694
00695 /**
00696   * @}
00697   */
00698
00699 /**
00700   * @}
00701   */
00702
00703 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32F4-Discovery BSP User Manual

stm32f4_discovery_accelerometer.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32f4_discovery_accelerometer.h
00004   * @author  MCD Application Team
00005   * @version V2.1.2
00006   * @date    31-January-2017
00007   * @brief   This file contains all the functions prototypes for the
00008   *          stm32f4_discovery_accelerometer.c firmware driver.
00009   ******************************************************************************
00010   * @attention
00011   *
00012   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00013   *
00014   * Redistribution and use in source and binary forms, with or without modification,
00015   * are permitted provided that the following conditions are met:
00016   *   1. Redistributions of source code must
```

```
 retain the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer.
00018   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00019   *      this list of conditions and the fol
lowing disclaimer in the documentation
00020   *      and/or other materials provided wit
h the distribution.
00021   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00022   *      may be used to endorse or promote p
roducts derived from this software
00023   *      without specific prior written perm
ission.
00024   *
00025   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00026   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00027   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00028   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00029   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00030   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 PROCUREMENT OF SUBSTITUTE GOODS OR
00031   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00032   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00033   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00034   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00035   *
```

```
00036   ******************************************************************************
00037   */
00038
00039 /* Define to prevent recursive inclusion -------------------------------------*/
00040 #ifndef __STM32F4_DISCOVERY_ACCELEROMETER_H
00041 #define __STM32F4_DISCOVERY_ACCELEROMETER_H
00042
00043 #ifdef __cplusplus
00044  extern "C" {
00045 #endif
00046
00047 /* Includes ------------------------------------------------------------------*/
00048 #include "stm32f4_discovery.h"
00049
00050 /* Include Accelerometer component drivers */
00051 #include "../Components/lis302dl/lis302dl.h"
00052 #include "../Components/lis3dsh/lis3dsh.h"
00053
00054 /** @addtogroup BSP
00055   * @{
00056   */
00057
00058 /** @addtogroup STM32F4_DISCOVERY
00059   * @{
00060   */
00061
00062 /** @addtogroup STM32F4_DISCOVERY_ACCELEROMETER
00063   * @{
00064   */
00065
```

```
00066 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETE
R_Exported_Types STM32F4 DISCOVERY ACCELEROMETER E
xported Types
00067   * @{
00068   */
00069 typedef enum
00070 {
00071   ACCELERO_OK = 0,
00072   ACCELERO_ERROR = 1,
00073   ACCELERO_TIMEOUT = 2
00074 }ACCELERO_StatusTypeDef;
00075 /**
00076   * @}
00077   */
00078
00079 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETE
R_Exported_Functions STM32F4 DISCOVERY ACCELEROMET
ER Exported Functions
00080   * @{
00081   */
00082 /* Accelerometer functions */
00083 uint8_t BSP_ACCELERO_Init(void);
00084 uint8_t BSP_ACCELERO_ReadID(void);
00085 void    BSP_ACCELERO_Reset(void);
00086 void    BSP_ACCELERO_Click_ITConfig(void);
00087 void    BSP_ACCELERO_Click_ITClear(void);
00088 void    BSP_ACCELERO_GetXYZ(int16_t *pDataXY
Z);
00089
00090 /**
00091   * @}
00092   */
00093
00094 /**
00095   * @}
00096   */
00097
```

```
00098 /**
00099   * @}
00100   */
00101
00102 /**
00103   * @}
00104   */
00105
00106 #ifdef __cplusplus
00107 }
00108 #endif
00109
00110 #endif /* __STM32F4_DISCOVERY_ACCELEROMETER_
H */
00111
00112 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32F4-Discovery BSP User Manual

stm32f4_discovery_accelerometer.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32f4_discovery_accelerometer.c
00004   * @author  MCD Application Team
00005   * @version V2.1.2
00006   * @date    31-January-2017
00007   * @brief   This file provides a set of functions needed to manage the
00008   *          MEMS accelerometers available on STM32F4-Discovery Kit.
00009   ******************************************************************************
00010   * @attention
00011   *
00012   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00013   *
00014   * Redistribution and use in source and binary forms, with or without modification,
00015   * are permitted provided that the following conditions are met:
00016   *   1. Redistributions of source code must
```

```
 retain the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer.
00018   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00019   *      this list of conditions and the fol
lowing disclaimer in the documentation
00020   *      and/or other materials provided wit
h the distribution.
00021   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00022   *      may be used to endorse or promote p
roducts derived from this software
00023   *      without specific prior written perm
ission.
00024   *
00025   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00026   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00027   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00028   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00029   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00030   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 PROCUREMENT OF SUBSTITUTE GOODS OR
00031   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00032   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00033   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00034   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00035   *
```

```
00036   ******************************************************************************
00037   */
00038
00039 /* Includes ------------------------------------------------------------------*/
00040 #include "stm32f4_discovery_accelerometer.h"
00041
00042 /** @addtogroup BSP
00043   * @{
00044   */
00045
00046 /** @addtogroup STM32F4_DISCOVERY
00047   * @{
00048   */
00049
00050 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETER STM32F4 DISCOVERY ACCELEROMETER
00051   * @brief  This file includes the motion sensor driver for ACCELEROMETER motion sensor
00052   *         devices.
00053   * @{
00054   */
00055
00056 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETER_Private_TypesDefinitions STM32F4 DISCOVERY ACCELEROMETER Private TypesDefinitions
00057   * @{
00058   */
00059 /**
00060   * @}
00061   */
00062
00063 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETER_Private_Defines STM32F4 DISCOVERY ACCELEROMETER Private Defines
00064   * @{
```

```
00065   */
00066 /**
00067   * @}
00068   */
00069
00070 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETE
R_Private_Macros STM32F4 DISCOVERY ACCELEROMETER P
rivate Macros
00071   * @{
00072   */
00073 /**
00074   * @}
00075   */
00076
00077 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETE
R_Private_Variables STM32F4 DISCOVERY ACCELEROMETE
R Private Variables
00078   * @{
00079   */
00080 static ACCELERO_DrvTypeDef *AcceleroDrv;
00081 /**
00082   * @}
00083   */
00084
00085 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETE
R_Private_FunctionPrototypes STM32F4 DISCOVERY ACC
ELEROMETER Private FunctionPrototypes
00086   * @{
00087   */
00088 /**
00089   * @}
00090   */
00091
00092 /** @defgroup STM32F4_DISCOVERY_ACCELEROMETE
R_Private_Functions STM32F4 DISCOVERY ACCELEROMETE
R Private Functions
00093   * @{
```

```
00094   */
00095
00096 /**
00097   * @brief  Setx Accelerometer Initializatio
n.
00098   * @retval ACCELERO_OK if no problem during
 initialization
00099   */
00100 uint8_t BSP_ACCELERO_Init(void)
00101 {
00102   uint8_t ret = ACCELERO_ERROR;
00103   uint16_t ctrl = 0x0000;
00104   LIS302DL_InitTypeDef         lis302dl_init
struct;
00105   LIS302DL_FilterConfigTypeDef lis302dl_filt
er = {0,0,0};
00106   LIS3DSH_InitTypeDef          l1s3dsh_InitS
truct;
00107
00108   if(Lis302dlDrv.ReadID() == I_AM_LIS302DL)
00109   {
00110     /* Initialize the accelerometer driver s
tructure */
00111     AcceleroDrv = &Lis302dlDrv;
00112
00113     /* Set configuration of LIS302DL MEMS Ac
celerometer **********************/
00114     lis302dl_initstruct.Power_Mode = LIS302D
L_LOWPOWERMODE_ACTIVE;
00115     lis302dl_initstruct.Output_DataRate = LI
S302DL_DATARATE_100;
00116     lis302dl_initstruct.Axes_Enable = LIS302
DL_XYZ_ENABLE;
00117     lis302dl_initstruct.Full_Scale = LIS302D
L_FULLSCALE_2_3;
00118     lis302dl_initstruct.Self_Test = LIS302DL
_SELFTEST_NORMAL;
```

```
00119
00120     /* Configure MEMS: data rate, power mode
, full scale, self test and axes */
00121     ctrl = (uint16_t) (lis302dl_initstruct.O
utput_DataRate | lis302dl_initstruct.Power_Mode |
\
00122                        lis302dl_initstruct.F
ull_Scale | lis302dl_initstruct.Self_Test | \
00123                        lis302dl_initstruct.A
xes_Enable);
00124
00125     /* Configure the accelerometer main para
meters */
00126     AcceleroDrv->Init(ctrl);
00127
00128     /* MEMS High Pass Filter configuration */

00129     lis302dl_filter.HighPassFilter_Data_Sele
ction = LIS302DL_FILTEREDDATASELECTION_OUTPUTREGIS
TER;
00130     lis302dl_filter.HighPassFilter_CutOff_Fr
equency = LIS302DL_HIGHPASSFILTER_LEVEL_1;
00131     lis302dl_filter.HighPassFilter_Interrupt
 = LIS302DL_HIGHPASSFILTERINTERRUPT_1_2;
00132
00133     /* Configure MEMS high pass filter cut-o
ff level, interrupt and data selection bits */

00134     ctrl = (uint8_t)(lis302dl_filter.HighPas
sFilter_Data_Selection | \
00135                      lis302dl_filter.HighPas
sFilter_CutOff_Frequency | \
00136                      lis302dl_filter.HighPas
sFilter_Interrupt);
00137
00138     /* Configure the accelerometer LPF main
parameters */
```

```
00139     AcceleroDrv->FilterConfig(ctrl);
00140
00141     ret = ACCELERO_OK;
00142   }
00143   else if(Lis3dshDrv.ReadID() == I_AM_LIS3DS
H)
00144   {
00145     /* Initialize the accelerometer driver s
tructure */
00146     AcceleroDrv = &Lis3dshDrv;
00147
00148     /* Set configuration of LIS3DSH MEMS Acc
elerometer **********************/
00149     l1s3dsh_InitStruct.Output_DataRate = LIS
3DSH_DATARATE_100;
00150     l1s3dsh_InitStruct.Axes_Enable = LIS3DSH
_XYZ_ENABLE;
00151     l1s3dsh_InitStruct.SPI_Wire = LIS3DSH_SE
RIALINTERFACE_4WIRE;
00152     l1s3dsh_InitStruct.Self_Test = LIS3DSH_S
ELFTEST_NORMAL;
00153     l1s3dsh_InitStruct.Full_Scale = LIS3DSH_
FULLSCALE_2;
00154     l1s3dsh_InitStruct.Filter_BW = LIS3DSH_F
ILTER_BW_800;
00155
00156     /* Configure MEMS: power mode(ODR) and a
xes enable */
00157     ctrl = (uint16_t) (l1s3dsh_InitStruct.Ou
tput_DataRate | \
00158                        l1s3dsh_InitStruct.Ax
es_Enable);
00159
00160     /* Configure MEMS: full scale and self t
est */
00161     ctrl |= (uint16_t) ((l1s3dsh_InitStruct.
SPI_Wire    | \
```

```
00162                                 l1s3dsh_InitStruct.
Self_Test   | \
00163                                 l1s3dsh_InitStruct.
Full_Scale  | \
00164                                 l1s3dsh_InitStruct.
Filter_BW) << 8);
00165
00166     /* Configure the accelerometer main para
meters */
00167     AcceleroDrv->Init(ctrl);
00168
00169     ret = ACCELERO_OK;
00170   }
00171
00172   else
00173   {
00174     ret = ACCELERO_ERROR;
00175   }
00176   return ret;
00177 }
00178
00179 /**
00180   * @brief  Read ID of Accelerometer compone
nt.
00181   * @retval ID
00182   */
00183 uint8_t BSP_ACCELERO_ReadID(void)
00184 {
00185   uint8_t id = 0x00;
00186
00187   if(AcceleroDrv->ReadID != NULL)
00188   {
00189     id = AcceleroDrv->ReadID();
00190   }
00191   return id;
00192 }
00193
```

```
00194 /**
00195   * @brief  Reboot memory content of Acceler
ometer.
00196   */
00197 void BSP_ACCELERO_Reset(void)
00198 {
00199   if(AcceleroDrv->Reset != NULL)
00200   {
00201     AcceleroDrv->Reset();
00202   }
00203 }
00204
00205 /**
00206   * @brief  Configure Accelerometer click IT
.
00207   */
00208 void BSP_ACCELERO_Click_ITConfig(void)
00209 {
00210   if(AcceleroDrv->ConfigIT != NULL)
00211   {
00212     AcceleroDrv->ConfigIT();
00213   }
00214 }
00215
00216 /**
00217   * @brief  Clear Accelerometer click IT.
00218   */
00219 void BSP_ACCELERO_Click_ITClear(void)
00220 {
00221   if(AcceleroDrv->ClearIT != NULL)
00222   {
00223     AcceleroDrv->ClearIT();
00224   }
00225 }
00226
00227 /**
00228   * @brief  Get XYZ axes acceleration.
```

```
00229   * @param  pDataXYZ: Pointer to 3 angular a
cceleration axes.
00230   *                   pDataXYZ[0] = X axis,
pDataXYZ[1] = Y axis, pDataXYZ[2] = Z axis
00231   */
00232 void BSP_ACCELERO_GetXYZ(int16_t *pDataXYZ)
00233 {
00234   int16_t SwitchXY = 0;
00235
00236   if(AcceleroDrv->GetXYZ != NULL)
00237   {
00238     AcceleroDrv->GetXYZ(pDataXYZ);
00239
00240     /* Switch X and Y Axes in case of LIS302
DL MEMS */
00241     if(AcceleroDrv == &Lis302dlDrv)
00242     {
00243       SwitchXY  = pDataXYZ[0];
00244       pDataXYZ[0] = pDataXYZ[1];
00245       /* Invert Y Axis to be compliant with
LIS3DSH MEMS */
00246       pDataXYZ[1] = -SwitchXY;
00247     }
00248   }
00249 }
00250
00251 /**
00252   * @}
00253   */
00254
00255 /**
00256   * @}
00257   */
00258
00259 /**
00260   * @}
00261   */
```

```
00262
00263 /**
00264   * @}
00265   */
00266
00267 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F4-Discovery BSP User Manual

stm32f4_discovery_audio.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32f4_discovery_audio.h
00004   * @author  MCD Application Team
00005   * @version V2.1.2
00006   * @date    31-January-2017
00007   * @brief   This file contains the common defines and functions prototypes for
00008   *          stm32f4_discovery_audio.c driver.
00009   ******************************************************************************
00010   * @attention
00011   *
00012   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00013   *
00014   * Redistribution and use in source and binary forms, with or without modification,
00015   * are permitted provided that the following conditions are met:
00016   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00017   *      this list of conditions and the fol
lowing disclaimer.
00018   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00019   *      this list of conditions and the fol
lowing disclaimer in the documentation
00020   *      and/or other materials provided wit
h the distribution.
00021   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00022   *      may be used to endorse or promote p
roducts derived from this software
00023   *      without specific prior written perm
ission.
00024   *
00025   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00026   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00027   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00028   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00029   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00030   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00031   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00032   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00033   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00034   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00035   *
00036   ****************************************
```

```
**********************************
00037   */
00038
00039 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00040 #ifndef __STM32F4_DISCOVERY_AUDIO_H
00041 #define __STM32F4_DISCOVERY_AUDIO_H
00042
00043 #ifdef __cplusplus
00044  extern "C" {
00045 #endif
00046
00047 /* Includes ------------------------------
-------------------------------------*/
00048 /* Include audio component Driver */
00049 #include "../Components/cs43l22/cs43l22.h"
00050
00051 #include "stm32f4_discovery.h"
00052 #include "../../../Middlewares/ST/STM32_Audi
o/Addons/PDM/pdm_filter.h"
00053
00054 /** @addtogroup BSP
00055   * @{
00056   */
00057
00058 /** @addtogroup STM32F4_DISCOVERY
00059   * @{
00060   */
00061
00062 /** @addtogroup STM32F4_DISCOVERY_AUDIO
00063   * @{
00064   */
00065
00066
00067 /** @defgroup STM32F4_DISCOVERY_AUDIO_Export
ed_Types STM32F4 DISCOVERY AUDIO Exported Types
00068   * @{
```

```
00069   */
00070 /**
00071   * @}
00072   */
00073
00074 /** @defgroup STM32F4_DISCOVERY_AUDIO_OUT_Ex
ported_Constants STM32F4 DISCOVERY AUDIO OUT Expor
ted Constants
00075   * @{
00076   */
00077
00078 /*-----------------------------------------
-----------------------------------
00079                              AUDIO OUT CONFIGUR
ATION
00080 -----------------------------------------
-----------------------------------*/
00081
00082 /* I2S peripheral configuration defines */
00083 #define I2S                              SPI3
00084 #define I2S3_CLK_ENABLE()                __HA
L_RCC_SPI3_CLK_ENABLE()
00085 #define I2S3_CLK_DISABLE()               __HA
L_RCC_SPI3_CLK_DISABLE()
00086 #define I2S3_SCK_SD_WS_AF                GPIO
_AF6_SPI3
00087 #define I2S3_SCK_SD_CLK_ENABLE()         __HA
L_RCC_GPIOC_CLK_ENABLE()
00088 #define I2S3_MCK_CLK_ENABLE()            __HA
L_RCC_GPIOC_CLK_ENABLE()
00089 #define I2S3_WS_CLK_ENABLE()             __HA
L_RCC_GPIOA_CLK_ENABLE()
00090 #define I2S3_WS_PIN                      GPIO
_PIN_4
00091 #define I2S3_SCK_PIN                     GPIO
_PIN_10
00092 #define I2S3_SD_PIN                      GPIO
```

```
_PIN_12
00093 #define I2S3_MCK_PIN                       GPIO
_PIN_7
00094 #define I2S3_SCK_SD_GPIO_PORT              GPIOC

00095 #define I2S3_WS_GPIO_PORT                  GPIOA

00096 #define I2S3_MCK_GPIO_PORT                 GPIOC

00097
00098 /* I2S DMA Stream definitions */
00099 #define I2S3_DMAx_CLK_ENABLE()             __HA
L_RCC_DMA1_CLK_ENABLE()
00100 #define I2S3_DMAx_CLK_DISABLE()            __HA
L_RCC_DMA1_CLK_DISABLE()
00101 #define I2S3_DMAx_STREAM                   DMA1
_Stream7
00102 #define I2S3_DMAx_CHANNEL                  DMA_
CHANNEL_0
00103 #define I2S3_DMAx_IRQ                      DMA1
_Stream7_IRQn
00104 #define I2S3_DMAx_PERIPH_DATA_SIZE         DMA_
PDATAALIGN_HALFWORD
00105 #define I2S3_DMAx_MEM_DATA_SIZE            DMA_
MDATAALIGN_HALFWORD
00106 #define DMA_MAX_SZE                        0xFF
FF
00107
00108 #define I2S3_IRQHandler                    DMA1
_Stream7_IRQHandler
00109
00110 /* Select the interrupt preemption priority
and subpriority for the DMA interrupt */
00111 #define AUDIO_OUT_IRQ_PREPRIO              0x0E
   /* Select the preemption priority level(0 is th
e highest) */
00112
```

```
00113 /*-----------------------------------------
-------------------------------------
00114                               AUDIO IN CONFIGURA
TION
00115 -----------------------------------------
-----------------------------------*/
00116 /* SPI Configuration defines */
00117 #define I2S2                                SPI2
00118 #define I2S2_CLK_ENABLE()                   __HA
L_RCC_SPI2_CLK_ENABLE()
00119 #define I2S2_CLK_DISABLE()                  __HA
L_RCC_SPI2_CLK_DISABLE()
00120 #define I2S2_SCK_PIN                        GPIO
_PIN_10
00121 #define I2S2_SCK_GPIO_PORT                  GPIOB

00122 #define I2S2_SCK_GPIO_CLK_ENABLE()          __HA
L_RCC_GPIOB_CLK_ENABLE()
00123 #define I2S2_SCK_AF                         GPIO
_AF5_SPI2
00124
00125 #define I2S2_MOSI_PIN                       GPIO
_PIN_3
00126 #define I2S2_MOSI_GPIO_PORT                 GPIOC

00127 #define I2S2_MOSI_GPIO_CLK_ENABLE()         __HA
L_RCC_GPIOC_CLK_ENABLE()
00128 #define I2S2_MOSI_AF                        GPIO
_AF5_SPI2
00129
00130 /* I2S DMA Stream Rx definitions */
00131 #define I2S2_DMAx_CLK_ENABLE()              __HA
L_RCC_DMA1_CLK_ENABLE()
00132 #define I2S2_DMAx_CLK_DISABLE()             __HA
L_RCC_DMA1_CLK_DISABLE()
00133 #define I2S2_DMAx_STREAM                    DMA1
_Stream3
```

```
00134 #define I2S2_DMAx_CHANNEL                DMA_
CHANNEL_0
00135 #define I2S2_DMAx_IRQ                    DMA1
_Stream3_IRQn
00136 #define I2S2_DMAx_PERIPH_DATA_SIZE       DMA_
PDATAALIGN_HALFWORD
00137 #define I2S2_DMAx_MEM_DATA_SIZE          DMA_
MDATAALIGN_HALFWORD
00138
00139 #define I2S2_IRQHandler                  DMA1
_Stream3_IRQHandler
00140
00141 /* Select the interrupt preemption priority
and subpriority for the IT/DMA interrupt */
00142 #define AUDIO_IN_IRQ_PREPRIO             0x0F
   /* Select the preemption priority level(0 is th
e highest) */
00143
00144 /*-----------------------------------------
-----------------------------------
00145              CONFIGURATION: Audio Driver Con
figuration parameters
00146 -----------------------------------------
-----------------------------------*/
00147
00148 #define AUDIODATA_SIZE                   2
/* 16-bits audio data size */
00149
00150 /* Audio status definition */
00151 #define AUDIO_OK                         0
00152 #define AUDIO_ERROR                      1
00153 #define AUDIO_TIMEOUT                    2
00154
00155 /* AudioFreq * DataSize (2 bytes) * NumChann
els (Stereo: 2) */
00156 #define DEFAULT_AUDIO_IN_FREQ
  I2S_AUDIOFREQ_16K
```

```
00157 #define DEFAULT_AUDIO_IN_BIT_RESOLUTION
  16
00158 #define DEFAULT_AUDIO_IN_CHANNEL_NBR
  1 /* Mono = 1, Stereo = 2 */
00159 #define DEFAULT_AUDIO_IN_VOLUME
  64
00160
00161 /* PDM buffer input size */
00162 #define INTERNAL_BUFF_SIZE
  128*DEFAULT_AUDIO_IN_FREQ/16000*DEFAULT_AUDIO_IN
_CHANNEL_NBR
00163 /* PCM buffer output size */
00164 #define PCM_OUT_SIZE
  DEFAULT_AUDIO_IN_FREQ/1000
00165 #define CHANNEL_DEMUX_MASK
  0x55
00166
00167 /*------------------------------------------
-----------------------------------
00168                        OPTIONAL Configuration d
efines parameters
00169 ------------------------------------------
-----------------------------------*/
00170
00171 /**
00172   * @}
00173   */
00174
00175 /** @defgroup STM32F4_DISCOVERY_AUDIO_Export
ed_Variables STM32F4 DISCOVERY AUDIO Exported Vari
ables
00176   * @{
00177   */
00178 extern __IO uint16_t AudioInVolume;
00179 /**
00180   * @}
00181   */
```

```
00182
00183 /** @defgroup STM32F4_DISCOVERY_AUDIO_Export
ed_Macros STM32F4 DISCOVERY AUDIO Exported Macros
00184   * @{
00185   */
00186 #define DMA_MAX(_X_)                (((_X_)
<= DMA_MAX_SZE)? (_X_):DMA_MAX_SZE)
00187 /**
00188   * @}
00189   */
00190
00191 /** @defgroup STM32F4_DISCOVERY_AUDIO_OUT_Ex
ported_Functions  STM32F4 DISCOVERY AUDIO OUT Expo
rted Functions
00192   * @{
00193   */
00194 uint8_t BSP_AUDIO_OUT_Init(uint16_t OutputDe
vice, uint8_t Volume, uint32_t AudioFreq);
00195 uint8_t BSP_AUDIO_OUT_Play(uint16_t* pBuffer
, uint32_t Size);
00196 void    BSP_AUDIO_OUT_ChangeBuffer(uint16_t
*pData, uint16_t Size);
00197 uint8_t BSP_AUDIO_OUT_Pause(void);
00198 uint8_t BSP_AUDIO_OUT_Resume(void);
00199 uint8_t BSP_AUDIO_OUT_Stop(uint32_t Option);
00200 uint8_t BSP_AUDIO_OUT_SetVolume(uint8_t Volu
me);
00201 void    BSP_AUDIO_OUT_SetFrequency(uint32_t
AudioFreq);
00202 uint8_t BSP_AUDIO_OUT_SetMute(uint32_t Cmd);
00203 uint8_t BSP_AUDIO_OUT_SetOutputMode(uint8_t
Output);
00204
00205 /* User Callbacks: user has to implement the
se functions in his code if they are needed. */
00206 /* This function is called when the requeste
d data has been completely transferred. */
```

```
00207 void    BSP_AUDIO_OUT_TransferComplete_CallB
ack(void);
00208
00209 /* This function is called when half of the
requested buffer has been transferred. */
00210 void    BSP_AUDIO_OUT_HalfTransfer_CallBack(
void);
00211
00212 /* This function is called when an Interrupt
 due to transfer error on or peripheral
00213    error occurs. */
00214 void    BSP_AUDIO_OUT_Error_CallBack(void);
00215
00216 /* These function can be modified in case th
e current settings (e.g. DMA stream)
00217    need to be changed for specific applicati
on needs */
00218 void  BSP_AUDIO_OUT_ClockConfig(I2S_HandleTy
peDef *hi2s, uint32_t AudioFreq, void *Params);
00219 void  BSP_AUDIO_OUT_MspInit(I2S_HandleTypeDe
f *hi2s, void *Params);
00220 void  BSP_AUDIO_OUT_MspDeInit(I2S_HandleType
Def *hi2s, void *Params);
00221
00222 /**
00223   * @}
00224   */
00225
00226 /** @defgroup STM32F4_DISCOVERY_AUDIO_IN_Exp
orted_Functions STM32F4 DISCOVERY AUDIO IN Exporte
d Functions
00227   * @{
00228   */
00229 uint8_t BSP_AUDIO_IN_Init(uint32_t AudioFreq
, uint32_t BitRes, uint32_t ChnlNbr);
00230 uint8_t BSP_AUDIO_IN_Record(uint16_t *pData,
 uint32_t Size);
```

```
00231 uint8_t BSP_AUDIO_IN_Stop(void);
00232 uint8_t BSP_AUDIO_IN_Pause(void);
00233 uint8_t BSP_AUDIO_IN_Resume(void);
00234 uint8_t BSP_AUDIO_IN_SetVolume(uint8_t Volum
e);
00235 uint8_t BSP_AUDIO_IN_PDMToPCM(uint16_t *PDMB
uf, uint16_t *PCMBuf);
00236 /* User Callbacks: user has to implement the
se functions in his code if they are needed. */
00237 /* This function should be implemented by th
e user application.
00238    It is called into this driver when the cu
rrent buffer is filled to prepare the next
00239    buffer pointer and its size. */
00240 void    BSP_AUDIO_IN_TransferComplete_CallBa
ck(void);
00241 void    BSP_AUDIO_IN_HalfTransfer_CallBack(v
oid);
00242
00243 /* This function is called when an Interrupt
 due to transfer error on or peripheral
00244    error occurs. */
00245 void    BSP_AUDIO_IN_Error_Callback(void);
00246
00247 /* These function can be modified in case th
e current settings (e.g. DMA stream)
00248    need to be changed for specific applicati
on needs */
00249 void  BSP_AUDIO_IN_ClockConfig(I2S_HandleTyp
eDef *hi2s, uint32_t AudioFreq, void *Params);
00250 void  BSP_AUDIO_IN_MspInit(I2S_HandleTypeDef
 *hi2s, void *Params);
00251 void  BSP_AUDIO_IN_MspDeInit(I2S_HandleTypeD
ef *hi2s, void *Params);
00252
00253 /**
00254   * @}
```

```
00255   */
00256
00257 /**
00258   * @}
00259   */
00260
00261 /**
00262   * @}
00263   */
00264
00265 /**
00266   * @}
00267   */
00268
00269 #ifdef __cplusplus
00270 }
00271 #endif
00272
00273 #endif /* __STM32F4_DISCOVERY_AUDIO_H */
00274
00275 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F4-Discovery BSP User Manual

stm32f4_discovery_audio.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32f4_discovery_audio.c
00004   * @author  MCD Application Team
00005   * @version V2.1.2
00006   * @date    31-January-2017
00007   * @brief   This file provides the Audio driver for the STM32F4-Discovery
00008   *          board.
00009   ******************************************************************************
00010   * @attention
00011   *
00012   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00013   *
00014   * Redistribution and use in source and binary forms, with or without modification,
00015   * are permitted provided that the following conditions are met:
00016   *   1. Redistributions of source code must retain the above copyright notice,
00017   *      this list of conditions and the fol
```

```
lowing disclaimer.
00018   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00019   *      this list of conditions and the fol
lowing disclaimer in the documentation
00020   *      and/or other materials provided wit
h the distribution.
00021   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00022   *      may be used to endorse or promote p
roducts derived from this software
00023   *      without specific prior written perm
ission.
00024   *
00025   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00026   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00027   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00028   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00029   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00030   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00031   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00032   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00033   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00034   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00035   *
00036   ****************************************
**********************************
```

```
00037   */
00038
00039 /*=========================================
=================================
00040
 User NOTES
00041 1. How To use this driver:
00042 --------------------------
00043    - This driver supports STM32F4xx devices
on STM32F4-Discovery Kit:
00044         a) to play an audio file (all functi
ons names start by BSP_AUDIO_OUT_xxx)
00045         b) to record an audio file through M
P45DT02, ST MEMS (all functions names start by AUD
IO_IN_xxx)
00046
00047 a) PLAY A FILE:
00048 ==============
00049    + Call the function BSP_AUDIO_OUT_Init(
00050                                          OutputDe
vice: physical output mode (OUTPUT_DEVICE_SPEAKER,

00051
     OUTPUT_DEVICE_HEADPHONE, OUTPUT_DEVICE_AUTO o
r
00052
     OUTPUT_DEVICE_BOTH)
00053                                          Volume:
initial volume to be set (0 is min (mute), 100 is
max (100%)
00054                                          AudioFre
q: Audio frequency in Hz (8000, 16000, 22500, 3200
0 ...)
00055                                          this par
ameter is relative to the audio file/stream type.
00056                                         )
00057       This function configures all the hardw
```

```
are required for the audio application (codec, I2C
, I2S,
00058       GPIOs, DMA and interrupt if needed). T
his function returns 0 if configuration is OK.
00059       If the returned value is different fro
m 0 or the function is stuck then the communicatio
n with
00060       the codec (try to un-plug the power or
 reset device in this case).
00061       - OUTPUT_DEVICE_SPEAKER: only speaker
will be set as output for the audio stream.
00062       - OUTPUT_DEVICE_HEADPHONE: only headph
ones will be set as output for the audio stream.
00063       - OUTPUT_DEVICE_AUTO: Selection of out
put device is made through external switch (implem
ented
00064           into the audio jack on the discover
y board). When the Headphone is connected it is us
ed
00065           as output. When the headphone is di
sconnected from the audio jack, the output is
00066           automatically switched to Speaker.
00067       - OUTPUT_DEVICE_BOTH: both Speaker and
 Headphone are used as outputs for the audio stream

00068           at the same time.
00069    + Call the function BSP_AUDIO_OUT_Play(
00070                                       pBuffer: p
ointer to the audio data file address
00071                                       Size: size
 of the buffer to be sent in Bytes
00072                                     )
00073       to start playing (for the first time)
from the audio file/stream.
00074    + Call the function BSP_AUDIO_OUT_Pause()
 to pause playing
00075    + Call the function BSP_AUDIO_OUT_Resume(
```

```
) to resume playing.
00076        Note. After calling BSP_AUDIO_OUT_Pau
se() function for pause, only BSP_AUDIO_OUT_Resume
() should be called
00077           for resume (it is not allowed to c
all BSP_AUDIO_OUT_Play() in this case).
00078        Note. This function should be called
only when the audio file is played or paused (not
stopped).
00079    + For each mode, you may need to implemen
t the relative callback functions into your code.
00080       The Callback functions are named BSP_A
UDIO_OUT_XXXCallBack() and only their prototypes a
re declared in
00081       the stm32f4_discovery_audio.h file. (r
efer to the example for more details on the callba
cks implementations)
00082    + To Stop playing, to modify the volume l
evel, the frequency or to mute, use the functions
00083        BSP_AUDIO_OUT_Stop(), BSP_AUDIO_OUT_S
etVolume(), AUDIO_OUT_SetFrequency() BSP_AUDIO_OUT
_SetOutputMode and BSP_AUDIO_OUT_SetMute().
00084    + The driver API and the callback functio
ns are at the end of the stm32f4_discovery_audio.h
 file.
00085
00086 Driver architecture:
00087 --------------------
00088    + This driver provide the High Audio Laye
r: consists of the function API exported in the st
m32f4_discovery_audio.h file
00089        (BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_
Play() ...)
00090    + This driver provide also the Media Acce
ss Layer (MAL): which consists of functions allowi
ng to access the media containing/
00091        providing the audio file/stream. Thes
```

```
e functions are also included as local functions i
nto
00092         the stm32f4_discovery_audio.c file (I
2S3_Init()...)
00093
00094 Known Limitations:
00095 -------------------
00096    1- When using the Speaker, if the audio f
ile quality is not high enough, the speaker output
00097       may produce high and uncomfortable noi
se level. To avoid this issue, to use speaker
00098       output properly, try to increase audio
 file sampling rate (typically higher than 48KHz).
00099       This operation will lead to larger fil
e size.
00100    2- Communication with the audio codec (th
rough I2C) may be corrupted if it is interrupted b
y some
00101       user interrupt routines (in this case,
 interrupts could be disabled just before the star
t of
00102       communication then re-enabled when it
is over). Note that this communication is only don
e at
00103       the configuration phase (BSP_AUDIO_OUT
_Init() or BSP_AUDIO_OUT_Stop()) and when Volume c
ontrol modification is
00104       performed (BSP_AUDIO_OUT_SetVolume() o
r BSP_AUDIO_OUT_SetMute()or BSP_AUDIO_OUT_SetOutpu
tMode()).
00105       When the audio data is played, no comm
unication is required with the audio codec.
00106    3- Parsing of audio file is not implement
ed (in order to determine audio file properties: M
ono/Stereo, Data size,
00107       File size, Audio Frequency, Audio Data
 header size ...). The configuration is fixed for
```

```
the given audio file.
00108    4- Supports only Stereo audio streaming.
To play mono audio streams, each data should be se
nt twice
00109       on the I2S or should be duplicated on
the source buffer. Or convert the stream in stereo
 before playing.
00110    5- Supports only 16-bits audio data size.
00111
00112 b) RECORD A FILE:
00113 ================
00114    + Call the function BSP_AUDIO_IN_Init(
00115                                          AudioFre
q: Audio frequency in Hz (8000, 16000, 22500, 3200
0 ...)
00116                                          )
00117       This function configures all the hardw
are required for the audio application (I2S,
00118       GPIOs, DMA and interrupt if needed). T
his function returns 0 if configuration is OK.
00119
00120    + Call the function BSP_AUDIO_IN_Record(
00121                                 pbuf Main buffer
 pointer for the recorded data storing
00122                                 size Current siz
e of the recorded buffer
00123                                 )
00124       to start recording from the microphone.

00125
00126    + User needs to implement user callbacks
to retrieve data saved in the record buffer
00127       (AUDIO_IN_RxHalfCpltCallback/BSP_AUDIO
_IN_ReceiveComplete_CallBack)
00128
00129    + Call the function AUDIO_IN_STOP() to st
op recording
```

```
00130
00131 ============================================
==================================*/
00132
00133 /* Includes ------------------------------
-----------------------------------*/
00134 #include "stm32f4_discovery_audio.h"
00135
00136 /** @addtogroup BSP
00137   * @{
00138   */
00139
00140 /** @addtogroup STM32F4_DISCOVERY
00141   * @{
00142   */
00143
00144 /** @defgroup STM32F4_DISCOVERY_AUDIO STM32F
4 DISCOVERY AUDIO
00145   * @brief This file includes the low layer
audio driver available on STM32F4-Discovery
00146   *        discovery board.
00147   * @{
00148   */
00149
00150 /** @defgroup STM32F4_DISCOVERY_AUDIO_Privat
e_Types STM32F4 DISCOVERY AUDIO Private Types
00151   * @{
00152   */
00153 /**
00154   * @}
00155   */
00156
00157 /** @defgroup STM32F4_DISCOVERY_AUDIO_Privat
e_Defines STM32F4 DISCOVERY AUDIO Private Defines
00158   * @{
00159   */
00160 /* These PLL parameters are valid when the f
```

```
(VCO clock) = 1Mhz */
00161 const uint32_t I2SFreq[8] = {8000, 11025, 16
000, 22050, 32000, 44100, 48000, 96000};
00162 const uint32_t I2SPLLN[8] = {256, 429, 213,
429, 426, 271, 258, 344};
00163 const uint32_t I2SPLLR[8] = {5, 4, 4, 4, 4,
6, 3, 1};
00164 /**
00165   * @}
00166   */
00167
00168 /** @defgroup STM32F4_DISCOVERY_AUDIO_Privat
e_Macros STM32F4 DISCOVERY AUDIO Private Macros
00169   * @{
00170   */
00171 /**
00172   * @}
00173   */
00174
00175 /** @defgroup STM32F4_DISCOVERY_AUDIO_Privat
e_Variables STM32F4 DISCOVERY AUDIO Private Variab
les
00176   * @{
00177   */
00178 /*##### PLAY #####*/
00179 static AUDIO_DrvTypeDef           *pAudioDrv
;
00180 I2S_HandleTypeDef                 hAudioOutI
2s;
00181
00182 /*### RECORDER ###*/
00183 I2S_HandleTypeDef                 hAudioInI2s
;
00184
00185 PDMFilter_InitStruct Filter[DEFAULT_AUDIO_IN
_CHANNEL_NBR];
00186 __IO uint16_t AudioInVolume = DEFAULT_AUDIO_
```

```
IN_VOLUME;
00187 /**
00188   * @}
00189   */
00190
00191 /** @defgroup STM32F4_DISCOVERY_AUDIO_Privat
e_Function_Prototypes STM32F4 DISCOVERY AUDIO Priv
ate Function Prototypes
00192   * @{
00193   */
00194 static uint8_t I2S3_Init(uint32_t AudioFreq)
;
00195 static uint8_t I2S2_Init(uint32_t AudioFreq)
;
00196 static void PDMDecoder_Init(uint32_t AudioFr
eq, uint32_t ChnlNbr);
00197 /**
00198   * @}
00199   */
00200
00201 /** @defgroup STM32F4_DISCOVERY_AUDIO_OUT_Pr
ivate_Functions STM32F4 DISCOVERY AUDIO OUT Privat
e Functions
00202   * @{
00203   */
00204
00205 /**
00206   * @brief  Configures the audio peripherals.

00207   * @param  OutputDevice: OUTPUT_DEVICE_SPEA
KER, OUTPUT_DEVICE_HEADPHONE,
00208   *                       OUTPUT_DEVICE_BOTH
 or OUTPUT_DEVICE_AUTO .
00209   * @param  Volume: Initial volume level (fr
om 0 (Mute) to 100 (Max))
00210   * @param  AudioFreq: Audio frequency used
to play the audio stream.
```

```
00211   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00212   */
00213 uint8_t BSP_AUDIO_OUT_Init(uint16_t OutputDe
vice, uint8_t Volume, uint32_t AudioFreq)
00214 {
00215   uint8_t ret = AUDIO_OK;
00216
00217   /* PLL clock is set depending by the Audio
Freq (44.1khz vs 48khz groups) */
00218   BSP_AUDIO_OUT_ClockConfig(&hAudioOutI2s, A
udioFreq, NULL);
00219
00220   /* I2S data transfer preparation:
00221   Prepare the Media to be used for the audio
 transfer from memory to I2S peripheral */
00222   hAudioOutI2s.Instance = I2S3;
00223   if(HAL_I2S_GetState(&hAudioOutI2s) == HAL_
I2S_STATE_RESET)
00224   {
00225     /* Init the I2S MSP: this __weak functio
n can be redefined by the application*/
00226     BSP_AUDIO_OUT_MspInit(&hAudioOutI2s, NUL
L);
00227   }
00228
00229   /* I2S data transfer preparation:
00230   Prepare the Media to be used for the audio
 transfer from memory to I2S peripheral */
00231   /* Configure the I2S peripheral */
00232   if(I2S3_Init(AudioFreq) != AUDIO_OK)
00233   {
00234     ret = AUDIO_ERROR;
00235   }
00236
00237   if(ret == AUDIO_OK)
00238   {
```

```
00239     /* Retieve audio codec identifier */
00240     if(((cs43l22_drv.ReadID(AUDIO_I2C_ADDRESS
)) & CS43L22_ID_MASK) == CS43L22_ID)
00241     {
00242       /* Initialize the audio driver structu
re */
00243       pAudioDrv = &cs43l22_drv;
00244     }
00245     else
00246     {
00247       ret = AUDIO_ERROR;
00248     }
00249   }
00250
00251   if(ret == AUDIO_OK)
00252   {
00253     pAudioDrv->Init(AUDIO_I2C_ADDRESS, Outpu
tDevice, Volume, AudioFreq);
00254   }
00255
00256   return ret;
00257 }
00258
00259 /**
00260   * @brief  Starts playing audio stream from
 a data buffer for a determined size.
00261   * @param  pBuffer: Pointer to the buffer
00262   * @param  Size: Number of audio data BYTES.

00263   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00264   */
00265 uint8_t BSP_AUDIO_OUT_Play(uint16_t* pBuffer
, uint32_t Size)
00266 {
00267   /* Call the audio Codec Play function */
00268   if(pAudioDrv->Play(AUDIO_I2C_ADDRESS, pBuf
```

```
fer, Size) != 0)
00269   {
00270     return AUDIO_ERROR;
00271   }
00272   else
00273   {
00274     /* Update the Media layer and enable it
for play */
00275     HAL_I2S_Transmit_DMA(&hAudioOutI2s, pBuf
fer, DMA_MAX(Size/AUDIODATA_SIZE));
00276
00277     /* Return AUDIO_OK when all operations a
re correctly done */
00278     return AUDIO_OK;
00279   }
00280 }
00281
00282 /**
00283   * @brief  Sends n-Bytes on the I2S interfa
ce.
00284   * @param  pData: Pointer to data address
00285   * @param  Size: Number of data to be writt
en
00286   */
00287 void BSP_AUDIO_OUT_ChangeBuffer(uint16_t *pD
ata, uint16_t Size)
00288 {
00289   HAL_I2S_Transmit_DMA(&hAudioOutI2s, pData,
 Size);
00290 }
00291
00292 /**
00293   * @brief   Pauses the audio file stream. I
n case of using DMA, the DMA Pause
00294   *          feature is used.
00295   * WARNING: When calling BSP_AUDIO_OUT_Paus
e() function for pause, only the
```

```
00296   *           BSP_AUDIO_OUT_Resume() function
 should be called for resume (use of BSP_AUDIO_OUT
_Play()
00297   *           function for resume could lead
to unexpected behavior).
00298   * @retval  AUDIO_OK if correct communicati
on, else wrong communication
00299   */
00300 uint8_t BSP_AUDIO_OUT_Pause(void)
00301 {
00302   /* Call the Audio Codec Pause/Resume funct
ion */
00303   if(pAudioDrv->Pause(AUDIO_I2C_ADDRESS) !=
0)
00304   {
00305     return AUDIO_ERROR;
00306   }
00307   else
00308   {
00309     /* Call the Media layer pause function */

00310     HAL_I2S_DMAPause(&hAudioOutI2s);
00311
00312     /* Return AUDIO_OK when all operations a
re correctly done */
00313     return AUDIO_OK;
00314   }
00315 }
00316
00317 /**
00318   * @brief   Resumes the audio file streamin
g.
00319   * WARNING: When calling BSP_AUDIO_OUT_Paus
e() function for pause, only
00320   *           BSP_AUDIO_OUT_Resume() function
 should be called for resume (use of BSP_AUDIO_OUT
_Play()
```

```
00321   *          function for resume could lead
to unexpected behavior).
00322   * @retval  AUDIO_OK if correct communicati
on, else wrong communication
00323   */
00324 uint8_t BSP_AUDIO_OUT_Resume(void)
00325 {
00326   /* Call the Audio Codec Pause/Resume funct
ion */
00327   if(pAudioDrv->Resume(AUDIO_I2C_ADDRESS) !=
 0)
00328   {
00329     return AUDIO_ERROR;
00330   }
00331   else
00332   {
00333     /* Call the Media layer resume function
*/
00334     HAL_I2S_DMAResume(&hAudioOutI2s);
00335
00336     /* Return AUDIO_OK when all operations a
re correctly done */
00337     return AUDIO_OK;
00338   }
00339 }
00340
00341 /**
00342   * @brief  Stops audio playing and Power do
wn the Audio Codec.
00343   * @param  Option: could be one of the foll
owing parameters
00344   *           - CODEC_PDWN_HW: completely sh
ut down the codec (physically).
00345   *                                Then need to
reconfigure the Codec after power on.
00346   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
```

```
00347   */
00348 uint8_t BSP_AUDIO_OUT_Stop(uint32_t Option)
00349 {
00350   /* Call DMA Stop to disable DMA stream bef
ore stopping codec */
00351   HAL_I2S_DMAStop(&hAudioOutI2s);
00352
00353   /* Call Audio Codec Stop function */
00354   if(pAudioDrv->Stop(AUDIO_I2C_ADDRESS, Opti
on) != 0)
00355   {
00356     return AUDIO_ERROR;
00357   }
00358   else
00359   {
00360     if(Option == CODEC_PDWN_HW)
00361     {
00362       /* Wait at least 1ms */
00363       HAL_Delay(1);
00364
00365       /* Reset the pin */
00366       HAL_GPIO_WritePin(AUDIO_RESET_GPIO, AU
DIO_RESET_PIN, GPIO_PIN_RESET);
00367     }
00368
00369     /* Return AUDIO_OK when all operations a
re correctly done */
00370     return AUDIO_OK;
00371   }
00372 }
00373
00374 /**
00375   * @brief  Controls the current audio volum
e level.
00376   * @param  Volume: Volume level to be set i
n percentage from 0% to 100% (0 for
00377   *         Mute and 100 for Max volume leve
```

```
l).
00378   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00379   */
00380 uint8_t BSP_AUDIO_OUT_SetVolume(uint8_t Volu
me)
00381 {
00382   /* Call the codec volume control function
with converted volume value */
00383   if(pAudioDrv->SetVolume(AUDIO_I2C_ADDRESS,
 Volume) != 0)
00384   {
00385     return AUDIO_ERROR;
00386   }
00387   else
00388   {
00389     /* Return AUDIO_OK when all operations a
re correctly done */
00390     return AUDIO_OK;
00391   }
00392 }
00393
00394 /**
00395   * @brief  Enables or disables the MUTE mod
e by software
00396   * @param  Cmd: could be AUDIO_MUTE_ON to m
ute sound or AUDIO_MUTE_OFF to
00397   *         unmute the codec and restore pre
vious volume level.
00398   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00399   */
00400 uint8_t BSP_AUDIO_OUT_SetMute(uint32_t Cmd)
00401 {
00402   /* Call the Codec Mute function */
00403   if(pAudioDrv->SetMute(AUDIO_I2C_ADDRESS, C
md) != 0)
```

```
00404   {
00405     return AUDIO_ERROR;
00406   }
00407   else
00408   {
00409     /* Return AUDIO_OK when all operations a
re correctly done */
00410     return AUDIO_OK;
00411   }
00412 }
00413
00414 /**
00415   * @brief  Switch dynamically (while audio
file is played) the output target
00416   *         (speaker or headphone).
00417   * @note   This function modifies a global
variable of the audio codec driver: OutputDev.
00418   * @param  Output: specifies the audio outp
ut target: OUTPUT_DEVICE_SPEAKER,
00419   *         OUTPUT_DEVICE_HEADPHONE, OUTPUT_
DEVICE_BOTH or OUTPUT_DEVICE_AUTO
00420   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00421   */
00422 uint8_t BSP_AUDIO_OUT_SetOutputMode(uint8_t
Output)
00423 {
00424   /* Call the Codec output Device function */

00425   if(pAudioDrv->SetOutputMode(AUDIO_I2C_ADDR
ESS, Output) != 0)
00426   {
00427     return AUDIO_ERROR;
00428   }
00429   else
00430   {
00431     /* Return AUDIO_OK when all operations a
```

```
re correctly done */
00432     return AUDIO_OK;
00433   }
00434 }
00435
00436 /**
00437   * @brief  Update the audio frequency.
00438   * @param  AudioFreq: Audio frequency used
to play the audio stream.
00439   * @note   This API should be called after
the BSP_AUDIO_OUT_Init() to adjust the
00440   *         audio frequency.
00441   */
00442 void BSP_AUDIO_OUT_SetFrequency(uint32_t Aud
ioFreq)
00443 {
00444   /* PLL clock is set depending by the Audio
Freq (44.1khz vs 48khz groups) */
00445   BSP_AUDIO_OUT_ClockConfig(&hAudioOutI2s, A
udioFreq, NULL);
00446
00447   /* Update the I2S audio frequency configur
ation */
00448   I2S3_Init(AudioFreq);
00449 }
00450
00451 /**
00452   * @brief  Tx Transfer completed callbacks.
00453   * @param  hi2s: I2S handle
00454   */
00455 void HAL_I2S_TxCpltCallback(I2S_HandleTypeDe
f *hi2s)
00456 {
00457   if(hi2s->Instance == I2S3)
00458   {
00459     /* Call the user function which will man
age directly transfer complete */
```

```
00460     BSP_AUDIO_OUT_TransferComplete_CallBack(
);
00461   }
00462 }
00463
00464 /**
00465   * @brief  Tx Half Transfer completed callb
acks.
00466   * @param  hi2s: I2S handle
00467   */
00468 void HAL_I2S_TxHalfCpltCallback(I2S_HandleTy
peDef *hi2s)
00469 {
00470   if(hi2s->Instance == I2S3)
00471   {
00472     /* Manage the remaining file size and ne
w address offset: This function should
00473        be coded by user (its prototype is al
ready declared in stm32f4_discovery_audio.h) */
00474     BSP_AUDIO_OUT_HalfTransfer_CallBack();
00475   }
00476 }
00477
00478 /**
00479   * @brief  Clock Config.
00480   * @param  hi2s: might be required to set a
udio peripheral predivider if any.
00481   * @param  AudioFreq: Audio frequency used
to play the audio stream.
00482   * @note   This API is called by BSP_AUDIO_
OUT_Init() and BSP_AUDIO_OUT_SetFrequency()
00483   *         Being __weak it can be overwritt
en by the application
00484   * @param  Params : pointer on additional c
onfiguration parameters, can be NULL.
00485   */
00486 __weak void BSP_AUDIO_OUT_ClockConfig(I2S_Ha
```

```
ndleTypeDef *hi2s, uint32_t AudioFreq, void *Param
s)
00487 {
00488   RCC_PeriphCLKInitTypeDef rccclkinit;
00489   uint8_t index = 0, freqindex = 0xFF;
00490
00491   for(index = 0; index < 8; index++)
00492   {
00493     if(I2SFreq[index] == AudioFreq)
00494     {
00495       freqindex = index;
00496     }
00497   }
00498   /* Enable PLLI2S clock */
00499   HAL_RCCEx_GetPeriphCLKConfig(&rccclkinit);
00500   /* PLLI2S_VCO Input = HSE_VALUE/PLL_M = 1
Mhz */
00501   if ((freqindex & 0x7) == 0)
00502   {
00503     /* I2S clock config
00504     PLLI2S_VCO = f(VCO clock) = f(PLLI2S clo
ck input) * (PLLI2SN/PLLM)
00505     I2SCLK = f(PLLI2S clock output) = f(VCO
clock) / PLLI2SR */
00506     rccclkinit.PeriphClockSelection = RCC_PE
RIPHCLK_I2S;
00507     rccclkinit.PLLI2S.PLLI2SN = I2SPLLN[freq
index];
00508     rccclkinit.PLLI2S.PLLI2SR = I2SPLLR[freq
index];
00509     HAL_RCCEx_PeriphCLKConfig(&rccclkinit);
00510   }
00511   else
00512   {
00513     /* I2S clock config
00514     PLLI2S_VCO = f(VCO clock) = f(PLLI2S clo
ck input) * (PLLI2SN/PLLM)
```

```
00515     I2SCLK = f(PLLI2S clock output) = f(VCO
clock) / PLLI2SR */
00516     rccclkinit.PeriphClockSelection = RCC_PE
RIPHCLK_I2S;
00517     rccclkinit.PLLI2S.PLLI2SN = 258;
00518     rccclkinit.PLLI2S.PLLI2SR = 3;
00519     HAL_RCCEx_PeriphCLKConfig(&rccclkinit);
00520   }
00521 }
00522
00523 /**
00524   * @brief  AUDIO OUT I2S MSP Init.
00525   * @param  hi2s: might be required to set a
udio peripheral predivider if any.
00526   * @param  Params : pointer on additional c
onfiguration parameters, can be NULL.
00527   */
00528 __weak void BSP_AUDIO_OUT_MspInit(I2S_Handle
TypeDef *hi2s, void *Params)
00529 {
00530   static DMA_HandleTypeDef hdma_i2sTx;
00531   GPIO_InitTypeDef  GPIO_InitStruct;
00532
00533   /* Enable I2S3 clock */
00534   I2S3_CLK_ENABLE();
00535
00536   /*** Configure the GPIOs ***/
00537   /* Enable I2S GPIO clocks */
00538   I2S3_SCK_SD_CLK_ENABLE();
00539   I2S3_WS_CLK_ENABLE();
00540
00541   /* I2S3 pins configuration: WS, SCK and SD
 pins ----------------------------*/
00542   GPIO_InitStruct.Pin         = I2S3_SCK_PIN
 | I2S3_SD_PIN;
00543   GPIO_InitStruct.Mode        = GPIO_MODE_AF
_PP;
```

```
00544   GPIO_InitStruct.Pull        = GPIO_NOPULL;
00545   GPIO_InitStruct.Speed       = GPIO_SPEED_F
AST;
00546   GPIO_InitStruct.Alternate   = I2S3_SCK_SD_
WS_AF;
00547   HAL_GPIO_Init(I2S3_SCK_SD_GPIO_PORT, &GPIO
_InitStruct);
00548
00549   GPIO_InitStruct.Pin         = I2S3_WS_PIN
;
00550   HAL_GPIO_Init(I2S3_WS_GPIO_PORT, &GPIO_Ini
tStruct);
00551
00552   /* I2S3 pins configuration: MCK pin */
00553   I2S3_MCK_CLK_ENABLE();
00554   GPIO_InitStruct.Pin         = I2S3_MCK_PIN
;
00555   HAL_GPIO_Init(I2S3_MCK_GPIO_PORT, &GPIO_In
itStruct);
00556
00557   /* Enable the I2S DMA clock */
00558   I2S3_DMAx_CLK_ENABLE();
00559
00560   if(hi2s->Instance == I2S3)
00561   {
00562     /* Configure the hdma_i2sTx handle param
eters */
00563     hdma_i2sTx.Init.Channel             = I2
S3_DMAx_CHANNEL;
00564     hdma_i2sTx.Init.Direction           = DM
A_MEMORY_TO_PERIPH;
00565     hdma_i2sTx.Init.PeriphInc           = DM
A_PINC_DISABLE;
00566     hdma_i2sTx.Init.MemInc              = DM
A_MINC_ENABLE;
00567     hdma_i2sTx.Init.PeriphDataAlignment = I2
S3_DMAx_PERIPH_DATA_SIZE;
```

```
00568     hdma_i2sTx.Init.MemDataAlignment    = I2
S3_DMAx_MEM_DATA_SIZE;
00569     hdma_i2sTx.Init.Mode                = DM
A_NORMAL;
00570     hdma_i2sTx.Init.Priority            = DM
A_PRIORITY_HIGH;
00571     hdma_i2sTx.Init.FIFOMode            = DM
A_FIFOMODE_ENABLE;
00572     hdma_i2sTx.Init.FIFOThreshold       = DM
A_FIFO_THRESHOLD_FULL;
00573     hdma_i2sTx.Init.MemBurst            = DM
A_MBURST_SINGLE;
00574     hdma_i2sTx.Init.PeriphBurst         = DM
A_PBURST_SINGLE;
00575
00576     hdma_i2sTx.Instance                 = I2
S3_DMAx_STREAM;
00577
00578     /* Associate the DMA handle */
00579     __HAL_LINKDMA(hi2s, hdmatx, hdma_i2sTx);
00580
00581     /* Deinitialize the Stream for new trans
fer */
00582     HAL_DMA_DeInit(&hdma_i2sTx);
00583
00584     /* Configure the DMA Stream */
00585     HAL_DMA_Init(&hdma_i2sTx);
00586   }
00587
00588   /* I2S DMA IRQ Channel configuration */
00589   HAL_NVIC_SetPriority(I2S3_DMAx_IRQ, AUDIO_
OUT_IRQ_PREPRIO, 0);
00590   HAL_NVIC_EnableIRQ(I2S3_DMAx_IRQ);
00591 }
00592
00593 /**
00594   * @brief  De-Initializes BSP_AUDIO_OUT MSP.
```

```
00595   * @param  hi2s: might be required to set a
udio peripheral predivider if any.
00596   * @param  Params : pointer on additional c
onfiguration parameters, can be NULL.
00597   */
00598 __weak void BSP_AUDIO_OUT_MspDeInit(I2S_Hand
leTypeDef *hi2s, void *Params)
00599 {
00600   GPIO_InitTypeDef  GPIO_InitStruct;
00601
00602   /* I2S DMA IRQ Channel deactivation */
00603   HAL_NVIC_DisableIRQ(I2S3_DMAx_IRQ);
00604
00605   if(hi2s->Instance == I2S3)
00606   {
00607     /* Deinitialize the Stream for new trans
fer */
00608     HAL_DMA_DeInit(hi2s->hdmatx);
00609   }
00610
00611  /* Disable I2S block */
00612   __HAL_I2S_DISABLE(hi2s);
00613
00614   /* CODEC_I2S pins configuration: SCK and S
D pins */
00615   GPIO_InitStruct.Pin = I2S3_SCK_PIN | I2S3_
SD_PIN;
00616   HAL_GPIO_DeInit(I2S3_SCK_SD_GPIO_PORT, GPI
O_InitStruct.Pin);
00617
00618   /* CODEC_I2S pins configuration: WS pin */
00619   GPIO_InitStruct.Pin = I2S3_WS_PIN;
00620   HAL_GPIO_DeInit(I2S3_WS_GPIO_PORT, GPIO_In
itStruct.Pin);
00621
00622   /* CODEC_I2S pins configuration: MCK pin */
```

```
00623   GPIO_InitStruct.Pin = I2S3_MCK_PIN;
00624   HAL_GPIO_DeInit(I2S3_MCK_GPIO_PORT, GPIO_I
nitStruct.Pin);
00625
00626   /* Disable I2S clock */
00627   I2S3_CLK_DISABLE();
00628
00629   /* GPIO pins clock and DMA clock can be sh
ut down in the applic
00630      by surcgarging this __weak function */

00631 }
00632
00633 /**
00634   * @brief  Manages the DMA full Transfer co
mplete event.
00635   */
00636 __weak void BSP_AUDIO_OUT_TransferComplete_C
allBack(void)
00637 {
00638 }
00639
00640 /**
00641   * @brief  Manages the DMA Half Transfer co
mplete event.
00642   */
00643 __weak void BSP_AUDIO_OUT_HalfTransfer_CallB
ack(void)
00644 {
00645 }
00646
00647 /**
00648   * @brief  Manages the DMA FIFO error event.

00649   */
00650 __weak void BSP_AUDIO_OUT_Error_CallBack(void
```

```
)
00651 {
00652 }
00653
00654 /*******************************************
*********************************
00655                                   Static Functions
00656 *******************************************
*********************************/
00657
00658 /**
00659   * @brief  Initializes the Audio Codec audi
o interface (I2S).
00660   * @param  AudioFreq: Audio frequency to be
 configured for the I2S peripheral.
00661   */
00662 static uint8_t I2S3_Init(uint32_t AudioFreq)
00663 {
00664   /* Initialize the hAudioOutI2s Instance pa
rameter */
00665   hAudioOutI2s.Instance         = I2S3;
00666
00667  /* Disable I2S block */
00668   __HAL_I2S_DISABLE(&hAudioOutI2s);
00669
00670   /* I2S3 peripheral configuration */
00671   hAudioOutI2s.Init.AudioFreq   = AudioFreq;
00672   hAudioOutI2s.Init.ClockSource = I2S_CLOCK_
PLL;
00673   hAudioOutI2s.Init.CPOL        = I2S_CPOL_L
OW;
00674   hAudioOutI2s.Init.DataFormat  = I2S_DATAFO
RMAT_16B;
00675   hAudioOutI2s.Init.MCLKOutput  = I2S_MCLKOU
TPUT_ENABLE;
00676   hAudioOutI2s.Init.Mode        = I2S_MODE_M
ASTER_TX;
```

```
00677   hAudioOutI2s.Init.Standard    = I2S_STANDA
RD;
00678   /* Initialize the I2S peripheral with the
structure above */
00679   if(HAL_I2S_Init(&hAudioOutI2s) != HAL_OK)
00680   {
00681     return AUDIO_ERROR;
00682   }
00683   else
00684   {
00685     return AUDIO_OK;
00686   }
00687 }
00688
00689 /**
00690   * @}
00691   */
00692
00693 /** @defgroup STM32F4_DISCOVERY_AUDIO_IN_Pri
vate_Functions STM32F4 DISCOVERY AUDIO IN Private
Functions
00694   * @{
00695   */
00696
00697 /**
00698   * @brief  Initializes wave recording.
00699   * @param  AudioFreq: Audio frequency to be
 configured for the I2S peripheral.
00700   * @param  BitRes: Audio frequency to be co
nfigured for the I2S peripheral.
00701   * @param  ChnlNbr: Audio frequency to be c
onfigured for the I2S peripheral.
00702   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00703   */
00704 uint8_t BSP_AUDIO_IN_Init(uint32_t AudioFreq
, uint32_t BitRes, uint32_t ChnlNbr)
```

```
00705 {
00706
00707   /* Configure PLL clock */
00708   BSP_AUDIO_IN_ClockConfig(&hAudioInI2s, Aud
ioFreq, NULL);
00709
00710   /* Configure the PDM library */
00711   PDMDecoder_Init(AudioFreq, ChnlNbr);
00712
00713   /* Configure the I2S peripheral */
00714   hAudioInI2s.Instance = I2S2;
00715   if(HAL_I2S_GetState(&hAudioInI2s) == HAL_I
2S_STATE_RESET)
00716   {
00717     /* Initialize the I2S Msp: this __weak f
unction can be rewritten by the application */
00718     BSP_AUDIO_IN_MspInit(&hAudioInI2s, NULL)
;
00719   }
00720
00721   /* Configure the I2S2 */
00722   I2S2_Init(AudioFreq);
00723
00724   /* Return AUDIO_OK when all operations are
 correctly done */
00725   return AUDIO_OK;
00726 }
00727
00728 /**
00729   * @brief  Starts audio recording.
00730   * @param  pbuf: Main buffer pointer for th
e recorded data storing
00731   * @param  size: Current size of the record
ed buffer
00732   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00733   */
```

```
00734 uint8_t BSP_AUDIO_IN_Record(uint16_t* pbuf,
uint32_t size)
00735 {
00736   uint32_t ret = AUDIO_ERROR;
00737
00738   /* Start the process receive DMA */
00739   HAL_I2S_Receive_DMA(&hAudioInI2s, pbuf, si
ze);
00740
00741   /* Return AUDIO_OK when all operations are
 correctly done */
00742   ret = AUDIO_OK;
00743
00744   return ret;
00745 }
00746
00747 /**
00748   * @brief  Stops audio recording.
00749   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00750   */
00751 uint8_t BSP_AUDIO_IN_Stop(void)
00752 {
00753   uint32_t ret = AUDIO_ERROR;
00754
00755   /* Call the Media layer pause function */
00756   HAL_I2S_DMAStop(&hAudioInI2s);
00757
00758   /* Return AUDIO_OK when all operations are
 correctly done */
00759   ret = AUDIO_OK;
00760
00761   return ret;
00762 }
00763
00764 /**
00765   * @brief  Pauses the audio file stream.
```

```
00766   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00767   */
00768 uint8_t BSP_AUDIO_IN_Pause(void)
00769 {
00770   /* Call the Media layer pause function */
00771   HAL_I2S_DMAPause(&hAudioInI2s);
00772
00773   /* Return AUDIO_OK when all operations are
 correctly done */
00774   return AUDIO_OK;
00775 }
00776
00777 /**
00778   * @brief  Resumes the audio file stream.

00779   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00780   */
00781 uint8_t BSP_AUDIO_IN_Resume(void)
00782 {
00783   /* Call the Media layer pause/resume funct
ion */
00784   HAL_I2S_DMAResume(&hAudioInI2s);
00785
00786   /* Return AUDIO_OK when all operations are
 correctly done */
00787   return AUDIO_OK;
00788 }
00789
00790 /**
00791   * @brief  Controls the audio in volume lev
el.
00792   * @param  Volume: Volume level to be set i
n percentage from 0% to 100% (0 for
00793   *         Mute and 100 for Max volume leve
l).
```

```
00794   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00795   */
00796 uint8_t BSP_AUDIO_IN_SetVolume(uint8_t Volum
e)
00797 {
00798   /* Set the Global variable AudioInVolume */

00799   AudioInVolume = Volume;
00800
00801   /* Return AUDIO_OK when all operations are
 correctly done */
00802   return AUDIO_OK;
00803 }
00804
00805 /**
00806   * @brief  Converts audio format from PDM t
o PCM.
00807   * @param  PDMBuf: Pointer to data PDM buff
er
00808   * @param  PCMBuf: Pointer to data PCM buff
er
00809   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00810   */
00811 uint8_t BSP_AUDIO_IN_PDMToPCM(uint16_t *PDMB
uf, uint16_t *PCMBuf)
00812 {
00813   uint16_t AppPDM[INTERNAL_BUFF_SIZE/2];
00814   uint32_t index = 0;
00815
00816   /* PDM Demux */
00817   for(index = 0; index<INTERNAL_BUFF_SIZE/2;
 index++)
00818   {
00819     AppPDM[index] = HTONS(PDMBuf[index]);
00820   }
```

```
00821
00822   for(index = 0; index < DEFAULT_AUDIO_IN_CH
ANNEL_NBR; index++)
00823   {
00824     /* PDM to PCM filter */
00825     PDM_Filter_64_LSB((uint8_t*)&AppPDM[inde
x], (uint16_t*)&(PCMBuf[index]), AudioInVolume , (
PDMFilter_InitStruct *)&Filter[index]);
00826   }
00827   /* Duplicate samples since a single microp
hone in mounted on STM32F4-Discovery */
00828   for(index = 0; index < PCM_OUT_SIZE; index
++)
00829   {
00830     PCMBuf[(index<<1)+1] = PCMBuf[index<<1];
00831   }
00832
00833   /* Return AUDIO_OK when all operations are
 correctly done */
00834   return AUDIO_OK;
00835 }
00836
00837 /**
00838   * @brief  Rx Transfer completed callbacks
00839   * @param  hi2s: I2S handle
00840   */
00841 void HAL_I2S_RxCpltCallback(I2S_HandleTypeDe
f *hi2s)
00842 {
00843   /* Call the record update function to get
the next buffer to fill and its size (size is igno
red) */
00844   BSP_AUDIO_IN_TransferComplete_CallBack();
00845 }
00846
00847 /**
00848   * @brief  Rx Half Transfer completed callb
```

```
acks.
00849   * @param  hi2s: I2S handle
00850   */
00851 void HAL_I2S_RxHalfCpltCallback(I2S_HandleTy
peDef *hi2s)
00852 {
00853   /* Manage the remaining file size and new
address offset: This function
00854      should be coded by user (its prototype
is already declared in stm32f4_discovery_audio.h)
*/
00855   BSP_AUDIO_IN_HalfTransfer_CallBack();
00856 }
00857
00858 /**
00859   * @brief  Audio In Clock Config.
00860   * @param  hi2s: I2S handle
00861   * @param  AudioFreq: Audio frequency used
to record the audio stream.
00862   * @param  Params : pointer on additional c
onfiguration parameters, can be NULL.
00863   * @note   This API is called by BSP_AUDIO_
IN_Init()
00864   *         Being __weak it can be overwritt
en by the application
00865   */
00866 __weak void BSP_AUDIO_IN_ClockConfig(I2S_Han
dleTypeDef *hi2s, uint32_t AudioFreq, void *Params
)
00867 {
00868   RCC_PeriphCLKInitTypeDef rccclkinit;
00869
00870   /*Enable PLLI2S clock*/
00871   HAL_RCCEx_GetPeriphCLKConfig(&rccclkinit);
00872   /* PLLI2S_VCO Input = HSE_VALUE/PLL_M = 1
Mhz */
00873   if ((AudioFreq & 0x7) == 0)
```

```
00874   {
00875     /* Audio frequency multiple of 8 (8/16/3
2/48/96/192)*/
00876     /* PLLI2S_VCO Output = PLLI2S_VCO Input
* PLLI2SN = 192 Mhz */
00877     /* I2SCLK = PLLI2S_VCO Output/PLLI2SR =
192/6 = 32 Mhz */
00878     rccclkinit.PeriphClockSelection = RCC_PE
RIPHCLK_I2S;
00879     rccclkinit.PLLI2S.PLLI2SN = 192;
00880     rccclkinit.PLLI2S.PLLI2SR = 6;
00881     HAL_RCCEx_PeriphCLKConfig(&rccclkinit);
00882   }
00883   else
00884   {
00885     /* Other Frequency (11.025/22.500/44.100
) */
00886     /* PLLI2S_VCO Output = PLLI2S_VCO Input
* PLLI2SN = 290 Mhz */
00887     /* I2SCLK = PLLI2S_VCO Output/PLLI2SR =
290/2 = 145 Mhz */
00888     rccclkinit.PeriphClockSelection = RCC_PE
RIPHCLK_I2S;
00889     rccclkinit.PLLI2S.PLLI2SN = 290;
00890     rccclkinit.PLLI2S.PLLI2SR = 2;
00891     HAL_RCCEx_PeriphCLKConfig(&rccclkinit);
00892   }
00893 }
00894
00895 /**
00896   * @brief  BSP AUDIO IN MSP Init.
00897   * @param  hi2s: I2S handle
00898   * @param  Params : pointer on additional c
onfiguration parameters, can be NULL.
00899   */
00900 __weak void BSP_AUDIO_IN_MspInit(I2S_HandleT
ypeDef *hi2s, void *Params)
```

```
00901 {
00902   static DMA_HandleTypeDef hdma_i2sRx;
00903   GPIO_InitTypeDef  GPIO_InitStruct;
00904
00905   /* Enable the I2S2 peripheral clock */
00906   I2S2_CLK_ENABLE();
00907
00908   /* Enable I2S GPIO clocks */
00909   I2S2_SCK_GPIO_CLK_ENABLE();
00910   I2S2_MOSI_GPIO_CLK_ENABLE();
00911
00912   /* I2S2 pins configuration: SCK and MOSI p
ins -----------------------------*/
00913   GPIO_InitStruct.Mode      = GPIO_MODE_AF_P
P;
00914   GPIO_InitStruct.Pull      = GPIO_NOPULL;
00915   GPIO_InitStruct.Speed     = GPIO_SPEED_FAS
T;
00916
00917   GPIO_InitStruct.Pin       = I2S2_SCK_PIN;
00918   GPIO_InitStruct.Alternate = I2S2_SCK_AF;
00919   HAL_GPIO_Init(I2S2_SCK_GPIO_PORT, &GPIO_In
itStruct);
00920
00921   GPIO_InitStruct.Pin       = I2S2_MOSI_PIN
;
00922   GPIO_InitStruct.Alternate = I2S2_MOSI_AF;
00923   HAL_GPIO_Init(I2S2_MOSI_GPIO_PORT, &GPIO_I
nitStruct);
00924
00925   /* Enable the DMA clock */
00926   I2S2_DMAx_CLK_ENABLE();
00927
00928   if(hi2s->Instance == I2S2)
00929   {
00930     /* Configure the hdma_i2sRx handle param
eters */
```

```
00931     hdma_i2sRx.Init.Channel             = I2
S2_DMAx_CHANNEL;
00932     hdma_i2sRx.Init.Direction           = DM
A_PERIPH_TO_MEMORY;
00933     hdma_i2sRx.Init.PeriphInc           = DM
A_PINC_DISABLE;
00934     hdma_i2sRx.Init.MemInc              = DM
A_MINC_ENABLE;
00935     hdma_i2sRx.Init.PeriphDataAlignment = I2
S2_DMAx_PERIPH_DATA_SIZE;
00936     hdma_i2sRx.Init.MemDataAlignment    = I2
S2_DMAx_MEM_DATA_SIZE;
00937     hdma_i2sRx.Init.Mode                = DM
A_CIRCULAR;
00938     hdma_i2sRx.Init.Priority            = DM
A_PRIORITY_HIGH;
00939     hdma_i2sRx.Init.FIFOMode            = DM
A_FIFOMODE_DISABLE;
00940     hdma_i2sRx.Init.FIFOThreshold       = DM
A_FIFO_THRESHOLD_FULL;
00941     hdma_i2sRx.Init.MemBurst            = DM
A_MBURST_SINGLE;
00942     hdma_i2sRx.Init.PeriphBurst         = DM
A_MBURST_SINGLE;
00943
00944     hdma_i2sRx.Instance = I2S2_DMAx_STREAM;
00945
00946     /* Associate the DMA handle */
00947     __HAL_LINKDMA(hi2s, hdmarx, hdma_i2sRx);
00948
00949     /* Deinitialize the Stream for new trans
fer */
00950     HAL_DMA_DeInit(&hdma_i2sRx);
00951
00952     /* Configure the DMA Stream */
00953     HAL_DMA_Init(&hdma_i2sRx);
00954   }
```

```
00955
00956   /* I2S DMA IRQ Channel configuration */
00957   HAL_NVIC_SetPriority(I2S2_DMAx_IRQ, AUDIO_
IN_IRQ_PREPRIO, 0);
00958   HAL_NVIC_EnableIRQ(I2S2_DMAx_IRQ);
00959 }
00960
00961 /**
00962   * @brief  DeInitializes BSP_AUDIO_IN MSP.
00963   * @param  hi2s: I2S handle
00964   * @param  Params : pointer on additional c
onfiguration parameters, can be NULL.
00965   */
00966 __weak void BSP_AUDIO_IN_MspDeInit(I2S_Handl
eTypeDef *hi2s, void *Params)
00967 {
00968   GPIO_InitTypeDef  gpio_init_structure;
00969
00970   /* I2S DMA IRQ Channel deactivation */
00971   HAL_NVIC_DisableIRQ(I2S2_DMAx_IRQ);
00972
00973   if(hi2s->Instance == I2S2)
00974   {
00975     /* Deinitialize the Stream for new trans
fer */
00976     HAL_DMA_DeInit(hi2s->hdmarx);
00977   }
00978
00979  /* Disable I2S block */
00980   __HAL_I2S_DISABLE(hi2s);
00981
00982   /* Disable pins: SCK and SD pins */
00983   gpio_init_structure.Pin = I2S2_SCK_PIN;
00984   HAL_GPIO_DeInit(I2S2_SCK_GPIO_PORT, gpio_i
nit_structure.Pin);
00985   gpio_init_structure.Pin = I2S2_MOSI_PIN;
00986   HAL_GPIO_DeInit(I2S2_MOSI_GPIO_PORT, gpio_
```

```
init_structure.Pin);
00987
00988   /* Disable I2S clock */
00989   I2S2_CLK_DISABLE();
00990
00991   /* GPIO pins clock and DMA clock can be sh
ut down in the applic
00992      by surcgarging this __weak function */
00993 }
00994
00995 /**
00996   * @brief  User callback when record buffer
 is filled.
00997   */
00998 __weak void BSP_AUDIO_IN_TransferComplete_Ca
llBack(void)
00999 {
01000   /* This function should be implemented by
the user application.
01001      It is called into this driver when the
current buffer is filled
01002      to prepare the next buffer pointer and
its size. */
01003 }
01004
01005 /**
01006   * @brief  Manages the DMA Half Transfer co
mplete event.
01007   */
01008 __weak void BSP_AUDIO_IN_HalfTransfer_CallBa
ck(void)
01009 {
01010   /* This function should be implemented by
the user application.
01011      It is called into this driver when the
current buffer is filled
01012      to prepare the next buffer pointer and
```

```
its size. */
01013 }
01014
01015 /**
01016   * @brief  Audio IN Error callback function.

01017   */
01018 __weak void BSP_AUDIO_IN_Error_Callback(void
)
01019 {
01020   /* This function is called when an Interru
pt due to transfer error on or peripheral
01021      error occurs. */
01022 }
01023
01024 /*******************************************
**********************************
01025                                   Static Functions
01026 *******************************************
**********************************/
01027
01028 /**
01029   * @brief  Initialize the PDM library.
01030   * @param  AudioFreq: Audio sampling freque
ncy
01031   * @param  ChnlNbr: Number of audio channel
s (1: mono; 2: stereo)
01032   */
01033 static void PDMDecoder_Init(uint32_t AudioFr
eq, uint32_t ChnlNbr)
01034 {
01035   uint32_t i = 0;
01036
01037   /* Enable CRC peripheral to unlock the PDM
 library */
01038   __CRC_CLK_ENABLE();
01039
```

```
01040   for(i = 0; i < ChnlNbr; i++)
01041   {
01042     /* Filter LP and HP Init */
01043     Filter[i].LP_HZ = AudioFreq / 2;
01044     Filter[i].HP_HZ = 10;
01045     Filter[i].Fs = AudioFreq;
01046     /* On STM32F4-Discovery a single microph
one is mounted, samples are duplicated
01047        to make stereo audio streams */
01048     Filter[i].Out_MicChannels = 2;
01049     Filter[i].In_MicChannels = ChnlNbr;
01050     PDM_Filter_Init((PDMFilter_InitStruct *)&
Filter[i]);
01051   }
01052 }
01053
01054 /**
01055   * @brief  Initializes the Audio Codec audi
o interface (I2S)
01056   * @note   This function assumes that the I
2S input clock (through PLL_R in
01057   *         Devices RevA/Z and through dedic
ated PLLI2S_R in Devices RevB/Y)
01058   *         is already configured and ready
to be used.
01059   * @param  AudioFreq: Audio frequency to be
 configured for the I2S peripheral.
01060   */
01061 static uint8_t I2S2_Init(uint32_t AudioFreq)
01062 {
01063   /* Initialize the hAudioInI2s Instance par
ameter */
01064   hAudioInI2s.Instance          = I2S2;
01065
01066   /* Disable I2S block */
01067   __HAL_I2S_DISABLE(&hAudioInI2s);
01068
```

```
01069   /* I2S2 peripheral configuration */
01070   hAudioInI2s.Init.AudioFreq    = 2 * AudioF
req;
01071   hAudioInI2s.Init.ClockSource  = I2S_CLOCK_
PLL;
01072   hAudioInI2s.Init.CPOL         = I2S_CPOL_H
IGH;
01073   hAudioInI2s.Init.DataFormat   = I2S_DATAFO
RMAT_16B;
01074   hAudioInI2s.Init.MCLKOutput   = I2S_MCLKOU
TPUT_DISABLE;
01075   hAudioInI2s.Init.Mode         = I2S_MODE_M
ASTER_RX;
01076   hAudioInI2s.Init.Standard     = I2S_STANDA
RD_LSB;
01077
01078   /* Initialize the I2S peripheral with the
structure above */
01079   if(HAL_I2S_Init(&hAudioInI2s) != HAL_OK)
01080   {
01081     return AUDIO_ERROR;
01082   }
01083   else
01084   {
01085     return AUDIO_OK;
01086   }
01087 }
01088
01089 /**
01090   * @}
01091   */
01092
01093 /** @defgroup STM32F4_DISCOVERY_AUDIO_IN_OUT
_Private_Functions STM32F4 DISCOVERY AUDIO IN OUT
Private Functions
01094   * @{
01095   */
```

```
01096
01097 /**
01098   * @brief  I2S error callbacks.
01099   * @param  hi2s: I2S handle
01100   */
01101 void HAL_I2S_ErrorCallback(I2S_HandleTypeDef
 *hi2s)
01102 {
01103   /* Manage the error generated on DMA FIFO:
 This function
01104      should be coded by user (its prototype
is already declared in stm32f4_discovery_audio.h)
*/
01105   if(hi2s->Instance == I2S3)
01106   {
01107     BSP_AUDIO_OUT_Error_CallBack();
01108   }
01109   if(hi2s->Instance == I2S2)
01110   {
01111     BSP_AUDIO_IN_Error_Callback();
01112   }
01113 }
01114
01115 /**
01116   * @}
01117   */
01118
01119 /**
01120   * @}
01121   */
01122
01123 /**
01124   * @}
01125   */
01126
01127 /**
01128   * @}
```

```
01129   */
01130
01131 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32F4-Discovery BSP User Manual

BSP

Modules

STM32F4 DISCOVERY

Generated on Thu Jan 19 2017 15:34:14 for STM32F4-Discovery BSP User Manual by doxygen 1.7.6.1

STM32F4-Discovery BSP User Manual

STM32F4 DISCOVERY

BSP

Modules

STM32F4 DISCOVERY LOW LEVEL

This file provides set of firmware functions to manage Leds and push-button available on STM32F4-Discovery Kit from STMicroelectronics.

STM32F4 DISCOVERY ACCELEROMETER

This file includes the motion sensor driver for ACCELEROMETER motion sensor devices.

STM32F4 DISCOVERY AUDIO

This file includes the low layer audio driver available on STM32F4-Discovery discovery board.