STM32L4R9I_EVAL BSP User Manual

Private Types Definitions

STM32L4R9I_EVAL Common

This file provides firmware functions to manage Leds, push-buttons, Joystick, and COM ports available on STM32L4R9I-EVAL evaluation board from STMicroelectronics. More...

This file provides firmware functions to manage Leds, push-buttons, Joystick, and COM ports available on STM32L4R9I-EVAL evaluation board from STMicroelectronics.

STM32L4R9I_EVAL BSP User Manual

Private Functions

STM32L4R9I_EVAL Common

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP User Manual by doxygen 1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Data Fields

AUDIO_OUT_TypeDef Struct Reference

Private Types

Data Fields

AUDIO_DrvTypeDef *	**AudioDrv**
uint32_t	**OutputDevice**
uint32_t	**Frequency**
Audio_CallbackTypeDef	**CbError**
Audio_CallbackTypeDef	**CbHalfTransfer**
Audio_CallbackTypeDef	**CbTransferComplete**

Detailed Description

Definition at line **143** of file **stm32l4r9i_eval_audio.c**.

Field Documentation

AUDIO_DrvTypeDef* AUDIO_OUT_TypeDef::AudioDrv

Definition at line 145 of file stm32l4r9i_eval_audio.c.

Referenced by AUDIO_CODEC_Reset(), BSP_AUDIO_OUT_DeInit(), 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().

Audio_CallbackTypeDef AUDIO_OUT_TypeDef::CbError

Definition at line 148 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_RegisterCallbacks(), and HAL_SAI_ErrorCallback().

Audio_CallbackTypeDef AUDIO_OUT_TypeDef::CbHalfTransfer

Definition at line 149 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_RegisterCallbacks(), and HAL_SAI_TxHalfCpltCallback().

Audio_CallbackTypeDef AUDIO_OUT_TypeDef::CbTransferComple

Definition at line 150 of file stm32l4r9i_eval_audio.c.

Referenced by **BSP_AUDIO_OUT_Init()**, **BSP_AUDIO_OUT_RegisterCallbacks()**, and **HAL_SAI_TxCpltCallback()**.

uint32_t AUDIO_OUT_TypeDef::Frequency

Definition at line **147** of file **stm32l4r9i_eval_audio.c**.

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

uint32_t AUDIO_OUT_TypeDef::OutputDevice

Definition at line **146** of file **stm32l4r9i_eval_audio.c**.

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

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

- **stm32l4r9i_eval_audio.c**

STM32L4R9I_EVAL BSP User Manual

AUDIO_IN_TypeDef Struct Reference

Private Types

Data Fields

AUDIO_DrvTypeDef *	**AudioDrv**
DFSDM_Channel_HandleTypeDef	**hDfsdmLeftChannel**
DFSDM_Channel_HandleTypeDef	**hDfsdmRightChannel**
DMA_HandleTypeDef	**hDmaDfsdmLeft**
DMA_HandleTypeDef	**hDmaDfsdmRight**
int32_t *	**LeftRecBuff**
int32_t *	**RightRecBuff**
uint32_t	**InputDevice**
uint32_t	**Frequency**
uint32_t	**BitResolution**
uint32_t	**ChannelNbr**
uint16_t *	**pRecBuf**
uint32_t	**RecSize**
Audio_CallbackTypeDef	**CbError**
Audio_CallbackTypeDef	**CbHalfTransfer**
Audio_CallbackTypeDef	**CbTransferComplete**

Detailed Description

Definition at line **153** of file **stm32l4r9i_eval_audio.c**.

Field Documentation

AUDIO_DrvTypeDef* AUDIO_IN_TypeDef::AudioDrv

Definition at line 155 of file stm32l4r9i_eval_audio.c.

uint32_t AUDIO_IN_TypeDef::BitResolution

Definition at line 164 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Init().

Audio_CallbackTypeDef AUDIO_IN_TypeDef::CbError

Definition at line 168 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Init(), BSP_AUDIO_IN_RegisterCallbacks(), and HAL_DFSDM_FilterErrorCallback().

Audio_CallbackTypeDef AUDIO_IN_TypeDef::CbHalfTransfer

Definition at line 169 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Init(), BSP_AUDIO_IN_RegisterCallbacks(), and HAL_DFSDM_FilterRegConvHalfCpltCallback().

Audio_CallbackTypeDef AUDIO_IN_TypeDef::CbTransferComplete

Definition at line 170 of file stm32l4r9i_eval_audio.c.

Referenced by **BSP_AUDIO_IN_Init()**, **BSP_AUDIO_IN_RegisterCallbacks()**, and **HAL_DFSDM_FilterRegConvCpltCallback()**.

uint32_t AUDIO_IN_TypeDef::ChannelNbr

Definition at line **165** of file **stm32l4r9i_eval_audio.c**.

Referenced by **BSP_AUDIO_IN_Init()**, **BSP_AUDIO_IN_Record()**, **BSP_AUDIO_IN_Resume()**, **HAL_DFSDM_FilterRegConvCpltCallback()**, and **HAL_DFSDM_FilterRegConvHalfCpltCallback()**.

uint32_t AUDIO_IN_TypeDef::Frequency

Definition at line **163** of file **stm32l4r9i_eval_audio.c**.

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

DFSDM_Channel_HandleTypeDef AUDIO_IN_TypeDef::hDfsdmLeft

Definition at line **156** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_DeInit()**, and **AUDIO_DFSDMx_Init()**.

DFSDM_Channel_HandleTypeDef AUDIO_IN_TypeDef::hDfsdmRigh

Definition at line **157** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_DeInit()**, and **AUDIO_DFSDMx_Init()**.

DMA_HandleTypeDef AUDIO_IN_TypeDef::hDmaDfsdmLeft

Definition at line 158 of file stm32l4r9i_eval_audio.c.

Referenced by HAL_DFSDM_FilterMspDeInit(), and HAL_DFSDM_FilterMspInit().

DMA_HandleTypeDef AUDIO_IN_TypeDef::hDmaDfsdmRight

Definition at line 159 of file stm32l4r9i_eval_audio.c.

Referenced by HAL_DFSDM_FilterMspDeInit(), and HAL_DFSDM_FilterMspInit().

uint32_t AUDIO_IN_TypeDef::InputDevice

Definition at line 162 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Init(), and BSP_AUDIO_OUT_Init().

int32_t* AUDIO_IN_TypeDef::LeftRecBuff

Definition at line 160 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Record(), BSP_AUDIO_IN_Resume(), BSP_AUDIO_IN_Stop(), HAL_DFSDM_FilterRegConvCpltCallback(), and HAL_DFSDM_FilterRegConvHalfCpltCallback().

uint16_t* AUDIO_IN_TypeDef::pRecBuf

Definition at line 166 of file stm32l4r9i_eval_audio.c.

Referenced by **BSP_AUDIO_IN_Record()**, **HAL_DFSDM_FilterRegConvCpltCallback()**, and **HAL_DFSDM_FilterRegConvHalfCpltCallback()**.

uint32_t AUDIO_IN_TypeDef::RecSize

Definition at line **167** of file **stm32l4r9i_eval_audio.c**.

Referenced by **BSP_AUDIO_IN_Record()**, **BSP_AUDIO_IN_Resume()**, **HAL_DFSDM_FilterRegConvCpltCallback()**, and **HAL_DFSDM_FilterRegConvHalfCpltCallback()**.

int32_t* AUDIO_IN_TypeDef::RightRecBuff

Definition at line **161** of file **stm32l4r9i_eval_audio.c**.

Referenced by **BSP_AUDIO_IN_Record()**, **BSP_AUDIO_IN_Resume()**, **BSP_AUDIO_IN_Stop()**, **HAL_DFSDM_FilterRegConvCpltCallback()**, and **HAL_DFSDM_FilterRegConvHalfCpltCallback()**.

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

- **stm32l4r9i_eval_audio.c**

STM32L4R9I_EVAL BSP User Manual

Private FunctionPrototypes

STM32L4R9I_EVAL DSI LCD

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP User Manual by doxygen 1.7.6.1

STM32L4R9I_EVAL BSP User Manual

LCD_DrawPropTypeDef Struct Reference

Exported Types

`#include <stm32l4r9i_eval_lcd.h>`

Data Fields

uint32_t	**TextColor**
uint32_t	**BackColor**
sFONT *	**pFont**

Detailed Description

Definition at line **151** of file **stm32l4r9i_eval_lcd.h**.

Field Documentation

uint32_t LCD_DrawPropTypeDef::BackColor

Specifies the background color below the text

Definition at line 154 of file stm32l4r9i_eval_lcd.h.

Referenced by BSP_DSI_LCD_ClearStringLine(), BSP_DSI_LCD_GetBackColor(), and BSP_DSI_LCD_SetBackColor().

sFONT* LCD_DrawPropTypeDef::pFont

Specifies the font used for the text

Definition at line 155 of file stm32l4r9i_eval_lcd.h.

Referenced by BSP_DSI_LCD_ClearStringLine(), BSP_DSI_LCD_DisplayChar(), BSP_DSI_LCD_DisplayStringAt(), BSP_DSI_LCD_GetFont(), BSP_DSI_LCD_SetFont(), and DrawChar().

uint32_t LCD_DrawPropTypeDef::TextColor

Specifies the color of text

Definition at line 153 of file stm32l4r9i_eval_lcd.h.

Referenced by BSP_DSI_LCD_ClearStringLine(), BSP_DSI_LCD_GetTextColor(), and BSP_DSI_LCD_SetTextColor().

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

- **stm32l4r9i_eval_lcd.h**

STM32L4R9I_EVAL BSP User Manual

Point Struct Reference

Exported Types

```
#include <stm32l4r9i_eval_lcd.h>
```

Data Fields

int16_t	**X**
int16_t	**Y**

Detailed Description

Definition at line **159** of file **stm32l4r9i_eval_lcd.h**.

Field Documentation

int16_t Point::X

geometric X position of drawing

Definition at line **161** of file **stm32l4r9i_eval_lcd.h**.

Referenced by **BSP_DSI_LCD_DrawPolygon()**, and **BSP_DSI_LCD_FillPolygon()**.

int16_t Point::Y

geometric Y position of drawing

Definition at line **162** of file **stm32l4r9i_eval_lcd.h**.

Referenced by **BSP_DSI_LCD_DrawPolygon()**, and **BSP_DSI_LCD_FillPolygon()**.

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

- **stm32l4r9i_eval_lcd.h**

STM32L4R9I_EVAL BSP User Manual

TS_StateTypeDef Struct Reference

Exported Types

TS_StateTypeDef Define TS State structure. More...

```
#include <stm32l4r9i_eval_ts.h>
```

Data Fields

uint8_t	**touchDetected**
uint16_t	**touchX** [TS_MAX_NB_TOUCH]
uint16_t	**touchY** [TS_MAX_NB_TOUCH]
uint8_t	**touchWeight** [TS_MAX_NB_TOUCH]
uint8_t	**touchEventId** [TS_MAX_NB_TOUCH]
uint8_t	**touchArea** [TS_MAX_NB_TOUCH]
uint32_t	**gestureId**

Detailed Description

TS_StateTypeDef Define TS State structure.

Definition at line **110** of file **stm32l4r9i_eval_ts.h**.

Field Documentation

uint32_t TS_StateTypeDef::gestureId

type of gesture detected : take value of type **TS_GestureIdTypeDef**

Definition at line **118** of file **stm32l4r9i_eval_ts.h**.

uint8_t TS_StateTypeDef::touchArea[TS_MAX_NB_TOUCH]

Touch_Area[0], Touch_Area[1] : touch area of each touch

Definition at line **117** of file **stm32l4r9i_eval_ts.h**.

uint8_t TS_StateTypeDef::touchDetected

Total number of active touches detected at last scan

Definition at line **112** of file **stm32l4r9i_eval_ts.h**.

uint8_t TS_StateTypeDef::touchEventId[TS_MAX_NB_TOUCH]

Touch_EventId[0], Touch_EventId[1] : take value of type **TS_TouchEventTypeDef**

Definition at line **116** of file **stm32l4r9i_eval_ts.h**.

uint8_t TS_StateTypeDef::touchWeight[TS_MAX_NB_TOUCH]

Touch_Weight[0], Touch_Weight[1] : weight property of touches

Definition at line **115** of file **stm32l4r9i_eval_ts.h**.

uint16_t TS_StateTypeDef::touchX[TS_MAX_NB_TOUCH]

Touch X[0], X[1] coordinates on 12 bits

Definition at line **113** of file **stm32l4r9i_eval_ts.h**.

uint16_t TS_StateTypeDef::touchY[TS_MAX_NB_TOUCH]

Touch Y[0], Y[1] coordinates on 12 bits

Definition at line **114** of file **stm32l4r9i_eval_ts.h**.

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

- **stm32l4r9i_eval_ts.h**

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL TS

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP User Manual by doxygen 1.7.6.1

STM32L4R9I_EVAL BSP User Manual

OSPI_NOR_Info Struct Reference

Exported Types

```
#include <stm32l4r9i_eval_ospi_nor.h>
```

Data Fields

uint32_t	**FlashSize**
uint32_t	**EraseSectorSize**
uint32_t	**EraseSectorsNumber**
uint32_t	**ProgPageSize**
uint32_t	**ProgPagesNumber**

Detailed Description

Definition at line **81** of file **stm32l4r9i_eval_ospi_nor.h**.

Field Documentation

uint32_t OSPI_NOR_Info::EraseSectorSize

Size of sectors for the erase operation

Definition at line **83** of file **stm32l4r9i_eval_ospi_nor.h**.

Referenced by **BSP_OSPI_NOR_GetInfo()**.

uint32_t OSPI_NOR_Info::EraseSectorsNumber

Number of sectors for the erase operation

Definition at line **84** of file **stm32l4r9i_eval_ospi_nor.h**.

Referenced by **BSP_OSPI_NOR_GetInfo()**.

uint32_t OSPI_NOR_Info::FlashSize

Size of the flash

Definition at line **82** of file **stm32l4r9i_eval_ospi_nor.h**.

Referenced by **BSP_OSPI_NOR_GetInfo()**.

uint32_t OSPI_NOR_Info::ProgPageSize

Size of pages for the program operation

Definition at line **85** of file **stm32l4r9i_eval_ospi_nor.h**.

Referenced by **BSP_OSPI_NOR_GetInfo()**.

uint32_t OSPI_NOR_Info::ProgPagesNumber

Number of pages for the program operation

Definition at line **86** of file **stm32l4r9i_eval_ospi_nor.h**.

Referenced by **BSP_OSPI_NOR_GetInfo()**.

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

- **stm32l4r9i_eval_ospi_nor.h**

STM32L4R9I_EVAL BSP User Manual

STM32L476G_EVAL_SRAM_Private_Functions

Exported Functions

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP User Manual by doxygen 1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Here is a list of all struct and union fields with links to the structures/unions they belong to:

- a -

- AudioDrv : **AUDIO_OUT_TypeDef** , **AUDIO_IN_TypeDef**

- b -

- BackColor : **LCD_DrawPropTypeDef**
- BitResolution : **AUDIO_IN_TypeDef**

- c -

- CbError : **AUDIO_OUT_TypeDef** , **AUDIO_IN_TypeDef**
- CbHalfTransfer : **AUDIO_IN_TypeDef** , **AUDIO_OUT_TypeDef**
- CbTransferComplete : **AUDIO_IN_TypeDef** , **AUDIO_OUT_TypeDef**
- ChannelNbr : **AUDIO_IN_TypeDef**

- e -

- EraseSectorSize : **OSPI_NOR_Info**
- EraseSectorsNumber : **OSPI_NOR_Info**

- f -

- FlashSize : **OSPI_NOR_Info**
- Frequency : **AUDIO_IN_TypeDef** , **AUDIO_OUT_TypeDef**

- g -

- gestureId : **TS_StateTypeDef**

- h -

- hDfsdmLeftChannel : **AUDIO_IN_TypeDef**
- hDfsdmRightChannel : **AUDIO_IN_TypeDef**
- hDmaDfsdmLeft : **AUDIO_IN_TypeDef**
- hDmaDfsdmRight : **AUDIO_IN_TypeDef**

- i -

- InputDevice : **AUDIO_IN_TypeDef**

- l -

- LeftRecBuff : **AUDIO_IN_TypeDef**

- o -

- OutputDevice : **AUDIO_OUT_TypeDef**

- p -

- pFont : **LCD_DrawPropTypeDef**
- pRecBuf : **AUDIO_IN_TypeDef**
- ProgPageSize : **OSPI_NOR_Info**
- ProgPagesNumber : **OSPI_NOR_Info**

- r -

- RecSize : **AUDIO_IN_TypeDef**
- RightRecBuff : **AUDIO_IN_TypeDef**

- t -

- TextColor : **LCD_DrawPropTypeDef**
- touchArea : **TS_StateTypeDef**
- touchDetected : **TS_StateTypeDef**
- touchEventId : **TS_StateTypeDef**
- touchWeight : **TS_StateTypeDef**
- touchX : **TS_StateTypeDef**
- touchY : **TS_StateTypeDef**

- x -

- X : **Point**

- y -

- Y : **Point**

STM32L4R9I_EVAL BSP User Manual

- a -

- AudioDrv : **AUDIO_OUT_TypeDef** , **AUDIO_IN_TypeDef**

- b -

- BackColor : **LCD_DrawPropTypeDef**
- BitResolution : **AUDIO_IN_TypeDef**

- c -

- CbError : **AUDIO_OUT_TypeDef** , **AUDIO_IN_TypeDef**
- CbHalfTransfer : **AUDIO_IN_TypeDef** , **AUDIO_OUT_TypeDef**
- CbTransferComplete : **AUDIO_IN_TypeDef** , **AUDIO_OUT_TypeDef**
- ChannelNbr : **AUDIO_IN_TypeDef**

- e -

- EraseSectorSize : **OSPI_NOR_Info**
- EraseSectorsNumber : **OSPI_NOR_Info**

- f -

- FlashSize : **OSPI_NOR_Info**
- Frequency : **AUDIO_IN_TypeDef** , **AUDIO_OUT_TypeDef**

- g -

- gestureId : **TS_StateTypeDef**

- h -

- hDfsdmLeftChannel : **AUDIO_IN_TypeDef**
- hDfsdmRightChannel : **AUDIO_IN_TypeDef**
- hDmaDfsdmLeft : **AUDIO_IN_TypeDef**
- hDmaDfsdmRight : **AUDIO_IN_TypeDef**

- i -

- InputDevice : **AUDIO_IN_TypeDef**

- l -

- LeftRecBuff : **AUDIO_IN_TypeDef**

- o -

- OutputDevice : **AUDIO_OUT_TypeDef**

- p -

- pFont : **LCD_DrawPropTypeDef**
- pRecBuf : **AUDIO_IN_TypeDef**
- ProgPageSize : **OSPI_NOR_Info**
- ProgPagesNumber : **OSPI_NOR_Info**

- r -

- RecSize : **AUDIO_IN_TypeDef**
- RightRecBuff : **AUDIO_IN_TypeDef**

- t -

- TextColor : **LCD_DrawPropTypeDef**
- touchArea : **TS_StateTypeDef**
- touchDetected : **TS_StateTypeDef**
- touchEventId : **TS_StateTypeDef**
- touchWeight : **TS_StateTypeDef**
- touchX : **TS_StateTypeDef**
- touchY : **TS_StateTypeDef**

- x -

- X : **Point**

- y -

- Y : **Point**

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_dsi_ts.c File Reference

This file provides a set of functions needed to manage the Touch Screen on DSI LCD of STM32L4R9I-EVAL evaluation board. More...

```
#include "stm32l4r9i_eval_ts.h" #include "math.h"
```

Go to the source code of this file.

Detailed Description

This file provides a set of functions needed to manage the Touch Screen on DSI LCD of STM32L4R9I-EVAL evaluation board.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_dsi_ts.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_rgb_lcd.c File Reference

This file includes the driver for Liquid Crystal Display (LCD) module mounted on STM32L4R9I_EVAL board. More...

```
#include "stm32l4r9i_eval_lcd.h" #include
"../../../Utilities/Fonts/fonts.h"
#include "../../../Utilities/Fonts/font24.c"
#include "../../../Utilities/Fonts/font20.c"
#include "../../../Utilities/Fonts/font16.c"
#include "../../../Utilities/Fonts/font12.c"
#include "../../../Utilities/Fonts/font8.c"
```

Go to the source code of this file.

Detailed Description

This file includes the driver for Liquid Crystal Display (LCD) module mounted on STM32L4R9I_EVAL board.

Author:

MCD Application Team

```
  1. How To use this driver:
  --------------------------
     - This driver is used to drive directly a
n LCD TFT using the LTDC controller.
     - This driver uses timing and setting for
 RK043FN48H LCD.

  2. Driver description:
  ----------------------
    + Initialization steps:
       o Initialize the LCD using the BSP_RGB_
LCD_Init() function.
       o Apply the Layer configuration using t
he BSP_RGB_LCD_LayerDefaultInit() function.
       o Select the LCD layer to be used using
 the BSP_RGB_LCD_SelectLayer() function.
       o Enable the LCD display using the BSP_
RGB_LCD_DisplayOn() function.

    + Options
       o Configure and enable the color keying
 functionality using the
         BSP_RGB_LCD_SetColorKeying() function
.
       o Modify in the fly the transparency an
d/or the frame buffer address
         using the following functions:
         - BSP_RGB_LCD_SetTransparency()
         - BSP_RGB_LCD_SetLayerAddress()
```

```
    + Display on LCD
       o Clear the hole LCD using BSP_RGB_LCD_
Clear() function or only one specified string
         line using the BSP_RGB_LCD_ClearStrin
gLine() function.
       o Display a character on the specified
line and column using the BSP_RGB_LCD_DisplayC
har()
         function or a complete string line us
ing the BSP_RGB_LCD_DisplayStringAtLine() func
tion.
       o Display a string line on the specifie
d position (x,y in pixel) and align mode
         using the BSP_RGB_LCD_DisplayStringAt
Line() function.
       o Draw and fill a basic shapes (dot, li
ne, rectangle, circle, ellipse, .. bitmap)
         on LCD using the available set of fun
ctions.
```

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_rgb_lcd.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_rgb_ts.c File Reference

This file provides a set of functions needed to manage the Touch Screen on STM32746G-Discovery board. More...

```
#include "stm32l4r9i_eval_ts.h"
```

Go to the source code of this file.

Detailed Description

This file provides a set of functions needed to manage the Touch Screen on STM32746G-Discovery board.

Author:

MCD Application Team

```
   1. How To use this driver:
   --------------------------
      - This driver is used to drive the touch
 screen module of the STM32L4R9I_EVAL
        board on the RK043FN48H-CT672B 480x272
 LCD screen with capacitive touch screen.
      - The FT5336 component driver must be in
cluded in project files according to
        the touch screen driver present on thi
s board.

   2. Driver description:
   ----------------------
     + Initialization steps:
        o Initialize the TS module using the B
SP_RGB_TS_Init() function. This
          function includes the MSP layer hard
ware resources initialization and the
          communication layer configuration to
 start the TS use. The LCD size properties
          (x and y) are passed as parameters.
        o If TS interrupt mode is desired, you
 must configure the TS interrupt mode
          by calling the function BSP_RGB_TS_I
TConfig(). The TS interrupt mode is generated
          as an external interrupt whenever a
touch is detected.
          The interrupt mode internally uses t
he IO functionalities driver driven by
```

```
          the IO expander, to configure the IT
 line.

     + Touch screen use
        o The touch screen state is captured w
henever the function BSP_RGB_TS_GetState() is
          used. This function returns informat
ion about the last LCD touch occurred
          in the TS_StateTypeDef structure.
        o If TS interrupt mode is used, the fu
nction BSP_RGB_TS_ITGetStatus() is needed to g
et
          the interrupt status. To clear the I
T pending bits, you should call the
          function BSP_RGB_TS_ITClear().
        o The IT is handled using the correspo
nding external interrupt IRQ handler,
          the user IT callback treatment is im
plemented on the same external interrupt
          callback.
```

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_rgb_ts.c**.

STM32L4R9I_EVAL BSP User Manual

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

- _ -

- __STM32L4R9I_EVAL_BSP_VERSION : **stm32l4r9i_eval.c**
- __STM32L4R9I_EVAL_BSP_VERSION_MAIN : **stm32l4r9i_eval.c**
- __STM32L4R9I_EVAL_BSP_VERSION_RC : **stm32l4r9i_eval.c**
- __STM32L4R9I_EVAL_BSP_VERSION_SUB1 : **stm32l4r9i_eval.c**
- __STM32L4R9I_EVAL_BSP_VERSION_SUB2 : **stm32l4r9i_eval.c**

STM32L4R9I_EVAL BSP User Manual

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

- a -

- ABS : **stm32l4r9i_eval_dsi_lcd.c**
- ActiveLayer : **stm32l4r9i_eval_dsi_lcd.c**
- ADCx : **stm32l4r9i_eval.h**
- ADCx_CHANNEL : **stm32l4r9i_eval.h**
- ADCx_CHANNEL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- ADCx_CHANNEL_GPIO_PORT : **stm32l4r9i_eval.h**
- ADCx_CHANNEL_PIN : **stm32l4r9i_eval.h**
- ADCx_CLK_ENABLE : **stm32l4r9i_eval.h**
- ADCx_FORCE_RESET : **stm32l4r9i_eval.h**
- ADCx_POLL_TIMEOUT : **stm32l4r9i_eval.h**
- ADCx_RELEASE_RESET : **stm32l4r9i_eval.h**
- AGPIO_PIN_0 : **stm32l4r9i_eval_io.h**
- AGPIO_PIN_1 : **stm32l4r9i_eval_io.h**
- Audio_CallbackTypeDef : **stm32l4r9i_eval_audio.h**
- AUDIO_CODEC_Reset() : **stm32l4r9i_eval_audio.c**
- AUDIO_DFSDM_DMAx_LEFT_IRQHandler : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDM_DMAx_RIGHT_IRQHandler : **stm32l4r9i_eval_audio.h**

- AUDIO_DFSDMx_CKOUT_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CKOUT_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CKOUT_DATIN_AF : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CKOUT_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CKOUT_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DATIN_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DATIN_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DATIN_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DATIN_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DeInit() : **stm32l4r9i_eval_audio.c**
- AUDIO_DFSDMx_DMAx_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_LEFT_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_LEFT_IRQ : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_LEFT_REQUEST : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_RIGHT_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_RIGHT_IRQ : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_RIGHT_REQUEST : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_Init() : **stm32l4r9i_eval_audio.c**

- AUDIO_DFSDMx_LEFT_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_LEFT_FILTER : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_RIGHT_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_RIGHT_FILTER : **stm32l4r9i_eval_audio.h**
- AUDIO_ERROR : **stm32l4r9i_eval_audio.h**
- AUDIO_I2C_ADDRESS : **stm32l4r9i_eval.h**
- AUDIO_IN_IRQ_PREPRIO : **stm32l4r9i_eval_audio.h**
- AUDIO_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- AUDIO_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- AUDIO_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- AUDIO_INT_PIN : **stm32l4r9i_eval.h**
- AUDIO_IO_DeInit() : **stm32l4r9i_eval.c**
- AUDIO_IO_Delay() : **stm32l4r9i_eval.c**
- AUDIO_IO_Init() : **stm32l4r9i_eval.c**
- AUDIO_IO_Read() : **stm32l4r9i_eval.c**
- AUDIO_IO_Write() : **stm32l4r9i_eval.c**
- AUDIO_OK : **stm32l4r9i_eval_audio.h**
- AUDIO_OUT_IRQ_PREPRIO : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIPLLConfig() : **stm32l4r9i_eval_audio.c**
- AUDIO_SAIx : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DeInit() : **stm32l4r9i_eval_audio.c**
- AUDIO_SAIx_DMAx_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_IRQ : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_IRQHandler : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_MEM_DATA_SIZE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_PERIPH_DATA_SIZE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_REQUEST : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_FS_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_FS_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_FS_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_FS_PIN : **stm32l4r9i_eval_audio.h**

- AUDIO_SAIx_Init() : **stm32l4r9i_eval_audio.c**
- AUDIO_SAIx_MCK_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_SCK_SD_FS_AF : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_PLL_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SCK_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SCK_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SCK_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SCK_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SD_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SD_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SD_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SD_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_TIMEOUT : **stm32l4r9i_eval_audio.h**
- AUDIODATA_SIZE : **stm32l4r9i_eval_audio.h**

STM32L4R9I_EVAL BSP User Manual

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

- b -

- BLOCKERASE_TIMEOUT : **stm32l4r9i_eval_nor.h**
- BSP_AUDIO_FREQUENCY_11K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_16K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_22K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_32K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_44K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_48K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_8K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_96K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_hDfsdmLeftFilter : **stm32l4r9i_eval_audio.c** , **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_hDfsdmRightFilter : **stm32l4r9i_eval_audio.c** , **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_hSai : **stm32l4r9i_eval_audio.c** , **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_IN_DeInit() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Init() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Pause() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Record() : **stm32l4r9i_eval_audio.c**

- BSP_AUDIO_IN_RegisterCallbacks() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Resume() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_SetFrequency() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Stop() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_ChangeAudioConfig() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_ChangeBuffer() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_CIRCULARMODE : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_OUT_DeInit() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_Init() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_MONOMODE : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_OUT_NORMALMODE : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_OUT_Pause() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_Play() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_RegisterCallbacks() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_Resume() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_SetFrequency() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_SetMute() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_SetOutputMode() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_SetVolume() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_STEREOMODE : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_OUT_Stop() : **stm32l4r9i_eval_audio.c**
- BSP_COM_DeInit() : **stm32l4r9i_eval.c**
- BSP_COM_Init() : **stm32l4r9i_eval.c**
- BSP_DSI_LCD_Clear() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_ClearStringLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DeInit() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayChar() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayOff() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayOn() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayStringAt() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayStringAtLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DMA2D_IRQHandler() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawBitmap() : **stm32l4r9i_eval_dsi_lcd.c**

- BSP_DSI_LCD_DrawCircle() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawEllipse() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawHLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawPixel() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawPolygon() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawRect() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawVLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DSI_IRQHandler() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_FillCircle() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_FillEllipse() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_FillPolygon() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_FillRect() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetBackColor() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetFont() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetTextColor() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetXSize() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetYSize() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_Init() : **stm32l4r9i_eval_dsi_lcd.h**
- BSP_DSI_LCD_IsFrameBufferAvailable() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_LTDC_ER_IRQHandler() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_LTDC_IRQHandler() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_MspDeInit() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_MspInit() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_ReadPixel() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_Refresh() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_ResetColorKeying() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SelectLayer() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetBackColor() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetBrightness() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetColorKeying() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetFont() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetLayerVisible() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetTextColor() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetTransparency() : **stm32l4r9i_eval_dsi_lcd.c**

- BSP_DSI_TS_DeInit() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_GestureConfig() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_Get_GestureId() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_GetState() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_Init() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_ITClear() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_ITConfig() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_ITGetStatus() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_ResetTouchData() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_EEPROM_DeInit() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_Init() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_ReadBuffer() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_TIMEOUT_UserCallback() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_WaitEepromStandbyState() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_WriteBuffer() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_WritePage() : **stm32l4r9i_eval_eeprom.c**
- BSP_ErrorHandler() : **stm32l4r9i_eval.c**
- BSP_GetVersion() : **stm32l4r9i_eval.c**
- BSP_I2C_ALL_USERS : **stm32l4r9i_eval.c**
- BSP_I2C_AUDIO_USER : **stm32l4r9i_eval.c**
- BSP_I2C_EEPROM_USER : **stm32l4r9i_eval.c**
- BSP_I2C_MFX_USER : **stm32l4r9i_eval.c**
- BSP_I2C_NO_USER : **stm32l4r9i_eval.c**
- BSP_I2C_TS_USER : **stm32l4r9i_eval.c**
- BSP_IDD_ClearIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_Config() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_DeInit() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_DisableIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_EnableIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorClearIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorDisableIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorEnableIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorGetCode() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorGetITStatus() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_GetITStatus() : **stm32l4r9i_eval_idd.c**

- BSP_IDD_GetValue() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_Init() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_LowPower() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_Reset() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_StartMeasure() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_WakeUp() : **stm32l4r9i_eval_idd.c**
- BSP_IO_ConfigPin() : **stm32l4r9i_eval_io.c**
- BSP_IO_DeInit() : **stm32l4r9i_eval_io.c**
- BSP_IO_Init() : **stm32l4r9i_eval_io.c**
- BSP_IO_ITClear() : **stm32l4r9i_eval_io.c**
- BSP_IO_ITGetStatus() : **stm32l4r9i_eval_io.c**
- BSP_IO_ReadPin() : **stm32l4r9i_eval_io.c**
- BSP_IO_TogglePin() : **stm32l4r9i_eval_io.c**
- BSP_IO_WritePin() : **stm32l4r9i_eval_io.c**
- BSP_JOY_DeInit() : **stm32l4r9i_eval.c**
- BSP_JOY_GetState() : **stm32l4r9i_eval.c**
- BSP_JOY_Init() : **stm32l4r9i_eval.c**
- bsp_lcd_hse_to_disable : **stm32l4r9i_eval_dsi_lcd.c**
- bsp_lcd_initialized : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_LED_DeInit() : **stm32l4r9i_eval.c**
- BSP_LED_Init() : **stm32l4r9i_eval.c**
- BSP_LED_Off() : **stm32l4r9i_eval.c**
- BSP_LED_On() : **stm32l4r9i_eval.c**
- BSP_LED_Toggle() : **stm32l4r9i_eval.c**
- BSP_NOR_Erase_Block() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_Erase_Chip() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_Init() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_ProgramData() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_Read_ID() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_ReadData() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_ReturnToReadMode() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_WriteData() : **stm32l4r9i_eval_nor.c**
- BSP_OSPI_NOR_DeInit() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_EnableMemoryMappedMode() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_EnterDeepPowerDown() : **stm32l4r9i_eval_ospi_nor.c**

- BSP_OSPI_NOR_Erase_Block() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Erase_Chip() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Erase_Sector() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_GetInfo() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_GetStatus() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Init() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_LeaveDeepPowerDown() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Read() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_ResumeErase() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_SuspendErase() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Write() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_RAM_DeInit() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_DMA_IRQHandler() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_EnableMemoryMappedMode() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_EnterDeepPowerDown() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Init() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_IRQHandler() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_LeaveDeepPowerDown() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Read() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Read_DMA() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Write() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Write_DMA() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_PB_DeInit() : **stm32l4r9i_eval.c**
- BSP_PB_GetState() : **stm32l4r9i_eval.c**
- BSP_PB_Init() : **stm32l4r9i_eval.c**
- BSP_POTENTIOMETER_DeInit() : **stm32l4r9i_eval.c**
- BSP_POTENTIOMETER_ERROR : **stm32l4r9i_eval.h**
- BSP_POTENTIOMETER_GetLevel() : **stm32l4r9i_eval.c**
- BSP_POTENTIOMETER_Init() : **stm32l4r9i_eval.c**
- BSP_POTENTIOMETER_OK : **stm32l4r9i_eval.h**
- BSP_RGB_LCD_Clear() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_ClearStringLine() : **stm32l4r9i_eval_rgb_lcd.h**

- BSP_RGB_LCD_ClockConfig() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DeInit() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayChar() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayOff() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayOn() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayStringAt() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayStringAtLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DMA2D_IRQHandler() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawBitmap() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawCircle() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawEllipse() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawHLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawPixel() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawPolygon() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawRect() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawVLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_FillCircle() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_FillEllipse() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_FillPolygon() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_FillRect() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetBackColor() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetFont() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetTextColor() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetXSize() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetYSize() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_Init() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_LayerDefaultInit() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_LayerRgb565Init() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_LTDC_ER_IRQHandler() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_LTDC_IRQHandler() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_MspDeInit() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_MspInit() : **stm32l4r9i_eval_rgb_lcd.h**

- BSP_RGB_LCD_ReadPixel() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_Reload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_ResetColorKeying() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_ResetColorKeying_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SelectLayer() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetBackColor() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetColorKeying() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetColorKeying_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetFont() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerAddress() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerAddress_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerVisible() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerVisible_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerWindow() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerWindow_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetTextColor() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetTransparency() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetTransparency_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetXSize() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetYSize() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_TS_DeInit() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_Get_GestureId() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_GetState() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_Init() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_ITClear() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_ITConfig() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_ITGetStatus() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_ResetTouchData() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_SD_AbortCallback() : **stm32l4r9i_eval_sd.c**
- BSP_SD_CardInfo : **stm32l4r9i_eval_sd.h**

- BSP_SD_DeInit() : **stm32l4r9i_eval_sd.c**
- BSP_SD_Erase() : **stm32l4r9i_eval_sd.c**
- BSP_SD_GetCardInfo() : **stm32l4r9i_eval_sd.c**
- BSP_SD_GetCardState() : **stm32l4r9i_eval_sd.c**
- BSP_SD_Init() : **stm32l4r9i_eval_sd.c**
- BSP_SD_IsDetected() : **stm32l4r9i_eval_sd.c**
- BSP_SD_ITConfig() : **stm32l4r9i_eval_sd.c**
- BSP_SD_MspDeInit() : **stm32l4r9i_eval_sd.c**
- BSP_SD_MspInit() : **stm32l4r9i_eval_sd.c**
- BSP_SD_ReadBlocks() : **stm32l4r9i_eval_sd.c**
- BSP_SD_ReadBlocks_DMA() : **stm32l4r9i_eval_sd.c**
- BSP_SD_ReadCpltCallback() : **stm32l4r9i_eval_sd.c**
- BSP_SD_WriteBlocks() : **stm32l4r9i_eval_sd.c**
- BSP_SD_WriteBlocks_DMA() : **stm32l4r9i_eval_sd.c**
- BSP_SD_WriteCpltCallback() : **stm32l4r9i_eval_sd.c**
- BSP_SRAM_DMA_IRQHandler() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_Init() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_ReadData() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_ReadData_DMA() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_WriteData() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_WriteData_DMA() : **stm32l4r9i_eval_sram.c**
- BUTTON_DOWN : **stm32l4r9i_eval.h**
- BUTTON_IRQn : **stm32l4r9i_eval.c**
- BUTTON_LEFT : **stm32l4r9i_eval.h**
- BUTTON_MODE_EXTI : **stm32l4r9i_eval.h**
- BUTTON_MODE_GPIO : **stm32l4r9i_eval.h**
- BUTTON_PIN : **stm32l4r9i_eval.c**
- BUTTON_PORT : **stm32l4r9i_eval.c**
- BUTTON_RIGHT : **stm32l4r9i_eval.h**
- BUTTON_SEL : **stm32l4r9i_eval.h**
- Button_TypeDef : **stm32l4r9i_eval.h**
- BUTTON_UP : **stm32l4r9i_eval.h**
- BUTTON_WAKEUP : **stm32l4r9i_eval.h**
- ButtonMode_TypeDef : **stm32l4r9i_eval.h**
- BUTTONn : **stm32l4r9i_eval.h**
- BUTTONx_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- BUTTONx_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by doxygen 1.7.6.1

STM32L4R9I_EVAL BSP User Manual

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

- c -

- CENTER_MODE : **stm32l4r9i_eval_lcd.h**
- CHIPERASE_TIMEOUT : **stm32l4r9i_eval_nor.h**
- CODEC_RESET_DELAY : **stm32l4r9i_eval_audio.h**
- COM1 : **stm32l4r9i_eval.h**
- COM_RX_AF : **stm32l4r9i_eval.c**
- COM_RX_PIN : **stm32l4r9i_eval.c**
- COM_RX_PORT : **stm32l4r9i_eval.c**
- COM_TX_AF : **stm32l4r9i_eval.c**
- COM_TX_PIN : **stm32l4r9i_eval.c**
- COM_TX_PORT : **stm32l4r9i_eval.c**
- COM_TypeDef : **stm32l4r9i_eval.h**
- COM_USART : **stm32l4r9i_eval.c**
- COMn : **stm32l4r9i_eval.h**
- COMx_CLK_DISABLE : **stm32l4r9i_eval.h**
- COMx_CLK_ENABLE : **stm32l4r9i_eval.h**
- COMx_RX_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- COMx_RX_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- COMx_TX_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- COMx_TX_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by doxygen 1.7.6.1

STM32L4R9I_EVAL 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 : **stm32l4r9i_eval_audio.h**
- DEFAULT_AUDIO_IN_CHANNEL_NBR : **stm32l4r9i_eval_audio.h**
- DEFAULT_AUDIO_IN_FREQ : **stm32l4r9i_eval_audio.h**
- DEFAULT_AUDIO_IN_VOLUME : **stm32l4r9i_eval_audio.h**
- DFSDMClockDivider : **stm32l4r9i_eval_audio.c**
- DFSDMFilterOrder : **stm32l4r9i_eval_audio.c**
- DFSDMOverSampling : **stm32l4r9i_eval_audio.c**
- DFSDMRightBitShift : **stm32l4r9i_eval_audio.c**
- DMA_MAX : **stm32l4r9i_eval_audio.h**
- DMA_MAX_SZE : **stm32l4r9i_eval_audio.h**
- DmaLeftRecBuffCplt : **stm32l4r9i_eval_audio.c**
- DmaLeftRecHalfBuffCplt : **stm32l4r9i_eval_audio.c**
- DmaOspiRamHandle : **stm32l4r9i_eval_ospi_ram.c**
- DmaRightRecBuffCplt : **stm32l4r9i_eval_audio.c**
- DmaRightRecHalfBuffCplt : **stm32l4r9i_eval_audio.c**
- DOWN_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- DOWN_JOY_PIN : **stm32l4r9i_eval.h**

- DrawChar() : **stm32l4r9i_eval_dsi_lcd.c**
- DrawProp : **stm32l4r9i_eval_dsi_lcd.c**
- DSI_RST_PIN : **stm32l4r9i_eval.h**
- DUMMY_CYCLES_CR_CFG : **stm32l4r9i_eval_ospi_nor.c**
- DUMMY_CYCLES_READ_OCTAL : **stm32l4r9i_eval_ospi_nor.c**

STM32L4R9I_EVAL BSP User Manual

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

- e -

- EEPROM_FAIL : **stm32l4r9i_eval_eeprom.h**
- EEPROM_I2C_ADDRESS : **stm32l4r9i_eval.h**
- EEPROM_IO_Init() : **stm32l4r9i_eval_eeprom.h** , **stm32l4r9i_eval.c**
- EEPROM_IO_IsDeviceReady() : **stm32l4r9i_eval.c** , **stm32l4r9i_eval_eeprom.h**
- EEPROM_IO_ReadData() : **stm32l4r9i_eval_eeprom.h** , **stm32l4r9i_eval.c**
- EEPROM_IO_WriteData() : **stm32l4r9i_eval.c** , **stm32l4r9i_eval_eeprom.h**
- EEPROM_MAX_SIZE : **stm32l4r9i_eval_eeprom.h**
- EEPROM_MAX_TRIALS : **stm32l4r9i_eval_eeprom.h**
- EEPROM_OK : **stm32l4r9i_eval_eeprom.h**
- EEPROM_PAGESIZE : **stm32l4r9i_eval_eeprom.h**
- EEPROM_TIMEOUT : **stm32l4r9i_eval_eeprom.h**
- EEPROMAddress : **stm32l4r9i_eval_eeprom.c**
- EEPROMDataRead : **stm32l4r9i_eval_eeprom.c**
- EEPROMDataWrite : **stm32l4r9i_eval_eeprom.c**
- EVAL_COM1 : **stm32l4r9i_eval.h**

- EVAL_COM1_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_IRQn : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_AF : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_PIN : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_AF : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_PIN : **stm32l4r9i_eval.h**
- EVAL_I2Cx : **stm32l4r9i_eval.h**
- EVAL_I2Cx_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_ER_IRQHandler : **stm32l4r9i_eval.h**
- EVAL_I2Cx_ER_IRQn : **stm32l4r9i_eval.h**
- EVAL_I2Cx_EV_IRQHandler : **stm32l4r9i_eval.h**
- EVAL_I2Cx_EV_IRQn : **stm32l4r9i_eval.h**
- EVAL_I2Cx_FORCE_RESET : **stm32l4r9i_eval.h**
- EVAL_I2Cx_RELEASE_RESET : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_PIN : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_SDA_AF : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SDA_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SDA_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SDA_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SDA_PIN : **stm32l4r9i_eval.h**
- EVAL_I2Cx_TIMEOUT_MAX : **stm32l4r9i_eval.h**
- EVAL_IDD_AMPLI_GAIN : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT0_STABDELAY : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT0_VALUE : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT1_STABDELAY : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT1_VALUE : **stm32l4r9i_eval_idd.h**

- EVAL_IDD_SHUNT2_STABDELAY : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT2_VALUE : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT4_STABDELAY : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT4_VALUE : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_VDD_MIN : **stm32l4r9i_eval_idd.h**
- EVAL_RCC_CLKSOURCE_I2C : **stm32l4r9i_eval.h**
- EVAL_RCC_PERIPHCLK_I2C : **stm32l4r9i_eval.h**
- EVAL_TS_DMAx_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_ER_IRQn : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_EV_IRQn : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_FORCE_RESET : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_RELEASE_RESET : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_PIN : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_SDA_AF : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SDA_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SDA_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SDA_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SDA_PIN : **stm32l4r9i_eval.h**
- EVAL_TS_RCC_CLKSOURCE_I2C : **stm32l4r9i_eval.h**
- EVAL_TS_RCC_PERIPHCLK_I2C : **stm32l4r9i_eval.h**
- EXT_RST_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

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

- f -

- FillTriangle() : **stm32l4r9i_eval_dsi_lcd.c**

STM32L4R9I_EVAL BSP User Manual

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

- g -

- GEST_ID_DOUBLE_CLICK : **stm32l4r9i_eval_ts.h**
- GEST_ID_MOVE_DOWN : **stm32l4r9i_eval_ts.h**
- GEST_ID_MOVE_LEFT : **stm32l4r9i_eval_ts.h**
- GEST_ID_MOVE_RIGHT : **stm32l4r9i_eval_ts.h**
- GEST_ID_MOVE_UP : **stm32l4r9i_eval_ts.h**
- GEST_ID_NB_MAX : **stm32l4r9i_eval_ts.h**
- GEST_ID_NO_GESTURE : **stm32l4r9i_eval_ts.h**
- GEST_ID_SINGLE_CLICK : **stm32l4r9i_eval_ts.h**
- GEST_ID_ZOOM_IN : **stm32l4r9i_eval_ts.h**
- GEST_ID_ZOOM_OUT : **stm32l4r9i_eval_ts.h**
- gfxmmu_lut_config_argb8888 : **stm32l4r9i_eval_gfxmmu_lut.h**

STM32L4R9I_EVAL 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_DFSDM_ChannelMspDeInit() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_ChannelMspInit() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterErrorCallback() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterMspDeInit() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterMspInit() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterRegConvCpltCallback() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterRegConvHalfCpltCallback() : **stm32l4r9i_eval_audio.c**
- HAL_DSI_EndOfRefreshCallback() : **stm32l4r9i_eval_dsi_lcd.c**
- HAL_NOR_MspWait() : **stm32l4r9i_eval_nor.c**
- HAL_SAI_ErrorCallback() : **stm32l4r9i_eval_audio.c**
- HAL_SAI_MspDeInit() : **stm32l4r9i_eval_audio.c**
- HAL_SAI_MspInit() : **stm32l4r9i_eval_audio.c**
- HAL_SAI_TxCpltCallback() : **stm32l4r9i_eval_audio.c**
- HAL_SAI_TxHalfCpltCallback() : **stm32l4r9i_eval_audio.c**
- HAL_SD_AbortCallback() : **stm32l4r9i_eval_sd.c**
- HAL_SD_RxCpltCallback() : **stm32l4r9i_eval_sd.c**
- HAL_SD_TxCpltCallback() : **stm32l4r9i_eval_sd.c**

- hAudioIn : **stm32l4r9i_eval_audio.c**
- hAudioOut : **stm32l4r9i_eval_audio.c**
- hdma2d_eval : **stm32l4r9i_eval_dsi_lcd.h** , **stm32l4r9i_eval_dsi_lcd.c**
- hDmaSai : **stm32l4r9i_eval_audio.c**
- hEvalADC : **stm32l4r9i_eval.c**
- hi2c_eval : **stm32l4r9i_eval.c**
- hi2c_evalTimeout : **stm32l4r9i_eval.c**
- hltdc_eval : **stm32l4r9i_eval_rgb_lcd.h**
- hsd_eval : **stm32l4r9i_eval_sd.h** , **stm32l4r9i_eval_sd.c**

STM32L4R9I_EVAL BSP User Manual

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

- i -

- I2Cx_DeInit() : **stm32l4r9i_eval.c**
- I2Cx_Error() : **stm32l4r9i_eval.c**
- I2Cx_Init() : **stm32l4r9i_eval.c**
- I2Cx_IsDeviceReady() : **stm32l4r9i_eval.c**
- I2Cx_MspDeInit() : **stm32l4r9i_eval.c**
- I2Cx_MspInit() : **stm32l4r9i_eval.c**
- I2Cx_ReadBuffer() : **stm32l4r9i_eval.c**
- I2Cx_ReadData() : **stm32l4r9i_eval.c**
- I2Cx_ReadMultiple() : **stm32l4r9i_eval.c**
- I2Cx_WriteBuffer() : **stm32l4r9i_eval.c**
- I2Cx_WriteData() : **stm32l4r9i_eval.c**
- I2Cx_WriteMultiple() : **stm32l4r9i_eval.c**
- IDD_AMP_CONTROL_PIN : **stm32l4r9i_eval.h**
- IDD_ERROR : **stm32l4r9i_eval_idd.h**
- IDD_I2C_ADDRESS : **stm32l4r9i_eval.h**
- IDD_INT_EXTI_IRQn : **stm32l4r9i_eval.h**
- IDD_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- IDD_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- IDD_INT_GPIO_PORT : **stm32l4r9i_eval.h**

- IDD_INT_PIN : **stm32l4r9i_eval.h**
- IDD_OK : **stm32l4r9i_eval_idd.h**
- IDD_StatusTypeDef : **stm32l4r9i_eval_idd.h**
- IDD_TIMEOUT : **stm32l4r9i_eval_idd.h**
- IDD_WAKEUP_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- IDD_WAKEUP_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- IDD_WAKEUP_GPIO_PORT : **stm32l4r9i_eval.h**
- IDD_WAKEUP_PIN : **stm32l4r9i_eval.h**
- IDD_ZERO_VALUE : **stm32l4r9i_eval_idd.h**
- IddDrv : **stm32l4r9i_eval_idd.c**
- INTERNAL_BUFF_SIZE : **stm32l4r9i_eval_audio.h**
- io_driver : **stm32l4r9i_eval_io.c**
- IO_ERROR : **stm32l4r9i_eval_io.h**
- IO_I2C_ADDRESS : **stm32l4r9i_eval.h**
- IO_OK : **stm32l4r9i_eval_io.h**
- IO_PIN_0 : **stm32l4r9i_eval_io.h**
- IO_PIN_1 : **stm32l4r9i_eval_io.h**
- IO_PIN_10 : **stm32l4r9i_eval_io.h**
- IO_PIN_11 : **stm32l4r9i_eval_io.h**
- IO_PIN_12 : **stm32l4r9i_eval_io.h**
- IO_PIN_13 : **stm32l4r9i_eval_io.h**
- IO_PIN_14 : **stm32l4r9i_eval_io.h**
- IO_PIN_15 : **stm32l4r9i_eval_io.h**
- IO_PIN_2 : **stm32l4r9i_eval_io.h**
- IO_PIN_3 : **stm32l4r9i_eval_io.h**
- IO_PIN_4 : **stm32l4r9i_eval_io.h**
- IO_PIN_5 : **stm32l4r9i_eval_io.h**
- IO_PIN_6 : **stm32l4r9i_eval_io.h**
- IO_PIN_7 : **stm32l4r9i_eval_io.h**
- IO_PIN_8 : **stm32l4r9i_eval_io.h**
- IO_PIN_9 : **stm32l4r9i_eval_io.h**
- IO_PIN_ALL : **stm32l4r9i_eval_io.h**
- IO_StatusTypeDef : **stm32l4r9i_eval_io.h**
- IO_TIMEOUT : **stm32l4r9i_eval_io.h**

User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

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

- j -

- JOY_ALL_PINS : **stm32l4r9i_eval.h**
- JOY_DOWN : **stm32l4r9i_eval.h**
- JOY_IRQn : **stm32l4r9i_eval.c**
- JOY_LEFT : **stm32l4r9i_eval.h**
- JOY_MODE_EXTI : **stm32l4r9i_eval.h**
- JOY_MODE_GPIO : **stm32l4r9i_eval.h**
- JOY_NONE : **stm32l4r9i_eval.h**
- JOY_NONE_PIN : **stm32l4r9i_eval.h**
- JOY_PIN : **stm32l4r9i_eval.c**
- JOY_RIGHT : **stm32l4r9i_eval.h**
- JOY_SEL : **stm32l4r9i_eval.h**
- JOY_UP : **stm32l4r9i_eval.h**
- JOYMode_TypeDef : **stm32l4r9i_eval.h**
- JOYn : **stm32l4r9i_eval.h**
- JOYState_TypeDef : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

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

- l -

- LCD_COLOR_BLACK : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_BLUE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_BROWN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_CYAN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKBLUE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKCYAN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKGRAY : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKGREEN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKMAGENTA : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKRED : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKYELLOW : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_GRAY : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_GREEN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTBLUE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTCYAN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTGRAY : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTGREEN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTMAGENTA : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTRED : **stm32l4r9i_eval_lcd.h**

- LCD_COLOR_LIGHTYELLOW : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_MAGENTA : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_ORANGE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_RED : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_TRANSPARENT : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_WHITE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_YELLOW : **stm32l4r9i_eval_lcd.h**
- LCD_DEFAULT_FONT : **stm32l4r9i_eval_lcd.h**
- LCD_DISP_PIN : **stm32l4r9i_eval.h**
- LCD_DSI_BL_CTRL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LCD_DSI_BL_CTRL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LCD_DSI_BL_CTRL_GPIO_PORT : **stm32l4r9i_eval.h**
- LCD_DSI_BL_CTRL_PIN : **stm32l4r9i_eval.h**
- LCD_DSI_TE_CTRL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LCD_DSI_TE_CTRL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LCD_DSI_TE_CTRL_GPIO_PORT : **stm32l4r9i_eval.h**
- LCD_DSI_TE_PIN : **stm32l4r9i_eval.h**
- LCD_ERROR : **stm32l4r9i_eval_lcd.h**
- LCD_FB_START_ADDRESS : **stm32l4r9i_eval_rgb_lcd.h**
- LCD_LayerCfgTypeDef : **stm32l4r9i_eval_lcd.h**
- LCD_OK : **stm32l4r9i_eval_lcd.h**
- LCD_ORIENTATION_INVALID : **stm32l4r9i_eval_lcd.h**
- LCD_ORIENTATION_LANDSCAPE : **stm32l4r9i_eval_lcd.h**
- LCD_ORIENTATION_PORTRAIT : **stm32l4r9i_eval_lcd.h**
- LCD_OrientationTypeDef : **stm32l4r9i_eval_lcd.h**
- LCD_RELOAD_IMMEDIATE : **stm32l4r9i_eval_rgb_lcd.h**
- LCD_RELOAD_VERTICAL_BLANKING : **stm32l4r9i_eval_rgb_lcd.h**
- LCD_Reset() : **stm32l4r9i_eval_dsi_lcd.c**
- LCD_RGB_BL_CTRL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LCD_RGB_BL_CTRL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LCD_RGB_BL_CTRL_GPIO_PORT : **stm32l4r9i_eval.h**
- LCD_RGB_BL_CTRL_PIN : **stm32l4r9i_eval.h**
- LCD_RGB_DISP_PIN : **stm32l4r9i_eval.h**
- LCD_RGB_I2C_ADDRESS : **stm32l4r9i_eval.h**
- LCD_RGB_RST_PIN : **stm32l4r9i_eval.h**
- LCD_RST_PIN : **stm32l4r9i_eval.h**

- LCD_TIMEOUT : **stm32l4r9i_eval_lcd.h**
- LED1 : **stm32l4r9i_eval.h**
- LED1_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LED1_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LED1_GPIO_PORT : **stm32l4r9i_eval.h**
- LED1_PIN : **stm32l4r9i_eval.h**
- LED2 : **stm32l4r9i_eval.h**
- LED2_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LED2_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LED2_GPIO_PORT : **stm32l4r9i_eval.h**
- LED2_PIN : **stm32l4r9i_eval.h**
- LED3 : **stm32l4r9i_eval.h**
- LED3_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LED3_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LED3_GPIO_PORT : **stm32l4r9i_eval.h**
- LED3_PIN : **stm32l4r9i_eval.h**
- LED4 : **stm32l4r9i_eval.h**
- LED4_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LED4_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LED4_GPIO_PORT : **stm32l4r9i_eval.h**
- LED4_PIN : **stm32l4r9i_eval.h**
- LED_BLUE : **stm32l4r9i_eval.h**
- LED_GREEN : **stm32l4r9i_eval.h**
- LED_ORANGE : **stm32l4r9i_eval.h**
- LED_PIN : **stm32l4r9i_eval.c**
- LED_PORT : **stm32l4r9i_eval.c**
- LED_RED : **stm32l4r9i_eval.h**
- Led_TypeDef : **stm32l4r9i_eval.h**
- LEDn : **stm32l4r9i_eval.h**
- LEDx_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LEDx_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LEFT_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- LEFT_JOY_PIN : **stm32l4r9i_eval.h**
- LEFT_MODE : **stm32l4r9i_eval_lcd.h**
- LL_ConvertLineToARGB8888() : **stm32l4r9i_eval_dsi_lcd.c**
- LL_FillBuffer() : **stm32l4r9i_eval_dsi_lcd.c**
- LTDC_ACTIVE_LAYER_BACKGROUND : **stm32l4r9i_eval_lcd.h**

- LTDC_ACTIVE_LAYER_FOREGROUND : **stm32l4r9i_eval_lcd.h**
- LTDC_DEFAULT_ACTIVE_LAYER : **stm32l4r9i_eval_lcd.h**
- LTDC_MAX_LAYER_NUMBER : **stm32l4r9i_eval_lcd.h**
- LTDC_NB_OF_LAYERS : **stm32l4r9i_eval_lcd.h**

STM32L4R9I_EVAL BSP User Manual

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

- m -

- MFX_INT_EXTI_IRQn : **stm32l4r9i_eval.h**
- MFX_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- MFX_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- MFX_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- MFX_INT_PIN : **stm32l4r9i_eval.h**
- MFX_IO_DeInit() : **stm32l4r9i_eval.c**
- MFX_IO_Delay() : **stm32l4r9i_eval.c**
- MFX_IO_EnableWakeupPin() : **stm32l4r9i_eval.c**
- MFX_IO_Init() : **stm32l4r9i_eval.c**
- MFX_IO_ITConfig() : **stm32l4r9i_eval.c**
- MFX_IO_Read() : **stm32l4r9i_eval.c**
- MFX_IO_ReadMultiple() : **stm32l4r9i_eval.c**
- MFX_IO_Wakeup() : **stm32l4r9i_eval.c**
- MFX_IO_Write() : **stm32l4r9i_eval.c**
- MFX_WAKEUP_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- MFX_WAKEUP_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- MFX_WAKEUP_GPIO_PORT : **stm32l4r9i_eval.h**
- MFX_WAKEUP_PIN : **stm32l4r9i_eval.h**
- MSD_ERROR : **stm32l4r9i_eval_sd.h**

- MSD_ERROR_SD_NOT_PRESENT : **stm32l4r9i_eval_sd.h**
- MSD_OK : **stm32l4r9i_eval_sd.h**

STM32L4R9I_EVAL BSP User Manual

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

- n -

- NOR_ADDR_SETUP_TIME : **stm32l4r9i_eval_nor.c**
- NOR_BURSTACCESS : **stm32l4r9i_eval_nor.h**
- NOR_BUSY_STATE : **stm32l4r9i_eval_nor.h**
- NOR_DATA_HOLD_TIME : **stm32l4r9i_eval_nor.c**
- NOR_DATA_SETUP_TIME : **stm32l4r9i_eval_nor.c**
- NOR_DEVICE_ADDR : **stm32l4r9i_eval_nor.h**
- NOR_MAX_BUFFER_SIZE : **stm32l4r9i_eval_nor.c**
- NOR_MEMORY_WIDTH : **stm32l4r9i_eval_nor.h**
- NOR_MspInit() : **stm32l4r9i_eval_nor.c**
- NOR_READY_BUSY_GPIO : **stm32l4r9i_eval_nor.h**
- NOR_READY_BUSY_PIN : **stm32l4r9i_eval_nor.h**
- NOR_READY_STATE : **stm32l4r9i_eval_nor.h**
- NOR_STATUS_ERROR : **stm32l4r9i_eval_nor.h**
- NOR_STATUS_OK : **stm32l4r9i_eval_nor.h**
- NOR_TURN_AROUND_TIME : **stm32l4r9i_eval_nor.c**
- NOR_WRITEBURST : **stm32l4r9i_eval_nor.h**
- norHandle : **stm32l4r9i_eval_nor.c**

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

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

- o -

- OPAMPx : **stm32l4r9i_eval.h**
- OPAMPx_CLK_DISABLE : **stm32l4r9i_eval.h**
- OPAMPx_CLK_ENABLE : **stm32l4r9i_eval.h**
- OPAMPx_FORCE_RESET : **stm32l4r9i_eval.h**
- OPAMPx_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- OPAMPx_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- OPAMPx_INM_GPIO_PORT : **stm32l4r9i_eval.h**
- OPAMPx_INM_PIN : **stm32l4r9i_eval.h**
- OPAMPx_INP_GPIO_PORT : **stm32l4r9i_eval.h**
- OPAMPx_INP_PIN : **stm32l4r9i_eval.h**
- OPAMPx_RELEASE_RESET : **stm32l4r9i_eval.h**
- OPAMPx_VOUT_GPIO_PORT : **stm32l4r9i_eval.h**
- OPAMPx_VOUT_PIN : **stm32l4r9i_eval.h**
- OSPI_NOR_AutoPollingMemReady() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_BUSY : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_ERROR : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_MspDeInit() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_MspInit() : **stm32l4r9i_eval_ospi_nor.c**

- OSPI_NOR_NOT_SUPPORTED : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_OCTAL_DISABLE : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_OCTAL_ENABLE : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_OctalMode() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_OK : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_ResetMemory() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_SUSPENDED : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_WriteEnable() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_RAM_ERROR : **stm32l4r9i_eval_ospi_ram.h**
- OSPI_RAM_HIGH_PERF_DISABLE : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_HIGH_PERF_ENABLE : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_MspDeInit() : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_MspInit() : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_OK : **stm32l4r9i_eval_ospi_ram.h**
- OSPI_RAM_QUAD_DISABLE : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_QUAD_ENABLE : **stm32l4r9i_eval_ospi_ram.c**
- OSPINORHandle : **stm32l4r9i_eval_ospi_nor.c**
- OspiRamHandle : **stm32l4r9i_eval_ospi_ram.c**
- OTG_FS_OVER_CURRENT_PIN : **stm32l4r9i_eval.h**
- OTG_FS_POWER_SWITCH_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

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

- p -

- PhysFrameBuffer : **stm32l4r9i_eval_rgb_lcd.h**
- POLY_X : **stm32l4r9i_eval_dsi_lcd.c**
- POLY_Y : **stm32l4r9i_eval_dsi_lcd.c**
- pPoint : **stm32l4r9i_eval_lcd.h**
- PROGRAM_TIMEOUT : **stm32l4r9i_eval_nor.h**

STM32L4R9I_EVAL BSP User Manual

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

- r -

- RIGHT_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- RIGHT_JOY_PIN : **stm32l4r9i_eval.h**
- RIGHT_MODE : **stm32l4r9i_eval_lcd.h**

STM32L4R9I_EVAL BSP User Manual

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

- s -

- SAIClockDivider : **stm32l4r9i_eval_audio.c**
- SAMPLINGTIME : **stm32l4r9i_eval.h**
- SaturaLH : **stm32l4r9i_eval_audio.c**
- SD_DATATIMEOUT : **stm32l4r9i_eval_sd.h**
- SD_DETECT_PIN : **stm32l4r9i_eval.h**
- SD_NOT_PRESENT : **stm32l4r9i_eval_sd.h**
- SD_PRESENT : **stm32l4r9i_eval_sd.h**
- SD_TRANSFER_BUSY : **stm32l4r9i_eval_sd.h**
- SD_TRANSFER_ERROR : **stm32l4r9i_eval_sd.h**
- SD_TRANSFER_OK : **stm32l4r9i_eval_sd.h**
- SDMMCx_IRQHandler : **stm32l4r9i_eval_sd.h**
- SDMMCx_IRQn : **stm32l4r9i_eval_sd.h**
- SEL_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- SEL_JOY_PIN : **stm32l4r9i_eval.h**
- SRAM_ADDR_SETUP_TIME : **stm32l4r9i_eval_sram.c**
- SRAM_BURSTACCESS : **stm32l4r9i_eval_sram.h**
- SRAM_DATA_HOLD_TIME : **stm32l4r9i_eval_sram.c**
- SRAM_DATA_SETUP_TIME : **stm32l4r9i_eval_sram.c**
- SRAM_DEVICE_ADDR : **stm32l4r9i_eval_sram.h**

- SRAM_DEVICE_SIZE : **stm32l4r9i_eval_sram.h**
- SRAM_DMAx_CHANNEL : **stm32l4r9i_eval_sram.h**
- SRAM_DMAx_CLK_ENABLE : **stm32l4r9i_eval_sram.h**
- SRAM_DMAx_IRQHandler : **stm32l4r9i_eval_sram.h**
- SRAM_DMAx_IRQn : **stm32l4r9i_eval_sram.h**
- SRAM_ERROR : **stm32l4r9i_eval_sram.h**
- SRAM_MEMORY_WIDTH : **stm32l4r9i_eval_sram.h**
- SRAM_MspInit() : **stm32l4r9i_eval_sram.c**
- SRAM_OK : **stm32l4r9i_eval_sram.h**
- SRAM_TURN_AROUND_TIME : **stm32l4r9i_eval_sram.c**
- SRAM_WRITEBURST : **stm32l4r9i_eval_sram.h**
- sramHandle : **stm32l4r9i_eval_sram.c**
- SWIRE_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

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

- t -

- Text_AlignModeTypdef : **stm32l4r9i_eval_lcd.h**
- Timing : **stm32l4r9i_eval_nor.c** , **stm32l4r9i_eval_sram.c**
- TOUCH_EVENT_CONTACT : **stm32l4r9i_eval_ts.h**
- TOUCH_EVENT_LIFT_UP : **stm32l4r9i_eval_ts.h**
- TOUCH_EVENT_NB_MAX : **stm32l4r9i_eval_ts.h**
- TOUCH_EVENT_NO_EVT : **stm32l4r9i_eval_ts.h**
- TOUCH_EVENT_PRESS_DOWN : **stm32l4r9i_eval_ts.h**
- TS_DEVICE_NOT_FOUND : **stm32l4r9i_eval_ts.h**
- TS_DSI_I2C_ADDRESS : **stm32l4r9i_eval.h**
- TS_DSI_INT_EXTI_IRQn : **stm32l4r9i_eval.h**
- TS_DSI_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- TS_DSI_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- TS_DSI_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- TS_DSI_INT_PIN : **stm32l4r9i_eval.h**
- TS_ERROR : **stm32l4r9i_eval_ts.h**
- ts_event_string_tab : **stm32l4r9i_eval_rgb_ts.h**
- ts_gesture_id_string_tab : **stm32l4r9i_eval_rgb_ts.h**
- TS_GestureIdTypeDef : **stm32l4r9i_eval_ts.h**
- TS_IO_Delay() : **stm32l4r9i_eval.c**

- TS_IO_Init() : **stm32l4r9i_eval.c**
- TS_IO_Read() : **stm32l4r9i_eval.c**
- TS_IO_ReadMultiple() : **stm32l4r9i_eval.c**
- TS_IO_Write() : **stm32l4r9i_eval.c**
- TS_IRQ_PENDING : **stm32l4r9i_eval_ts.h**
- TS_NO_IRQ_PENDING : **stm32l4r9i_eval_ts.h**
- TS_OK : **stm32l4r9i_eval_ts.h**
- TS_ORIENTATION_LANDSCAPE : **stm32l4r9i_eval_ts.h**
- TS_ORIENTATION_PORTRAIT : **stm32l4r9i_eval_ts.h**
- TS_RGB_I2C_ADDRESS : **stm32l4r9i_eval.h**
- TS_RGB_INT_EXTI_IRQn : **stm32l4r9i_eval.h**
- TS_RGB_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- TS_RGB_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- TS_RGB_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- TS_RGB_INT_PIN : **stm32l4r9i_eval.h**
- TS_StatusTypeDef : **stm32l4r9i_eval_ts.h**
- TS_SWAP_NONE : **stm32l4r9i_eval_ts.h**
- TS_SWAP_X : **stm32l4r9i_eval_ts.h**
- TS_SWAP_XY : **stm32l4r9i_eval_ts.h**
- TS_SWAP_Y : **stm32l4r9i_eval_ts.h**
- TS_TIMEOUT : **stm32l4r9i_eval_ts.h**
- TS_TouchEventTypeDef : **stm32l4r9i_eval_ts.h**

STM32L4R9I_EVAL BSP User Manual

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

- u -

- UP_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- UP_JOY_PIN : **stm32l4r9i_eval.h**
- UseExtiModeDetection : **stm32l4r9i_eval_sd.c**

STM32L4R9I_EVAL BSP User Manual

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

- v -

- v_bspI2cUsers : **stm32l4r9i_eval.c**

STM32L4R9I_EVAL BSP User Manual

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

- w -

- WAKEUP_BUTTON_EXTI_IRQn : **stm32l4r9i_eval.h**
- WAKEUP_BUTTON_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- WAKEUP_BUTTON_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- WAKEUP_BUTTON_GPIO_PORT : **stm32l4r9i_eval.h**
- WAKEUP_BUTTON_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- a -

- AUDIO_CODEC_Reset() : **stm32l4r9i_eval_audio.c**
- AUDIO_DFSDMx_DeInit() : **stm32l4r9i_eval_audio.c**
- AUDIO_DFSDMx_Init() : **stm32l4r9i_eval_audio.c**
- AUDIO_IO_DeInit() : **stm32l4r9i_eval.c**
- AUDIO_IO_Delay() : **stm32l4r9i_eval.c**
- AUDIO_IO_Init() : **stm32l4r9i_eval.c**
- AUDIO_IO_Read() : **stm32l4r9i_eval.c**
- AUDIO_IO_Write() : **stm32l4r9i_eval.c**
- AUDIO_SAIPLLConfig() : **stm32l4r9i_eval_audio.c**
- AUDIO_SAIx_DeInit() : **stm32l4r9i_eval_audio.c**
- AUDIO_SAIx_Init() : **stm32l4r9i_eval_audio.c**

STM32L4R9I_EVAL BSP User Manual

- b -

- BSP_AUDIO_IN_DeInit() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Init() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Pause() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Record() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_RegisterCallbacks() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Resume() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_SetFrequency() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_IN_Stop() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_ChangeAudioConfig() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_ChangeBuffer() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_DeInit() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_Init() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_Pause() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_Play() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_RegisterCallbacks() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_Resume() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_SetFrequency() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_SetMute() : **stm32l4r9i_eval_audio.c**

- BSP_AUDIO_OUT_SetOutputMode() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_SetVolume() : **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_OUT_Stop() : **stm32l4r9i_eval_audio.c**
- BSP_COM_DeInit() : **stm32l4r9i_eval.c**
- BSP_COM_Init() : **stm32l4r9i_eval.c**
- BSP_DSI_LCD_Clear() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_ClearStringLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DeInit() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayChar() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayOff() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayOn() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayStringAt() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DisplayStringAtLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DMA2D_IRQHandler() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawBitmap() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawCircle() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawEllipse() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawHLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawPixel() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawPolygon() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawRect() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DrawVLine() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_DSI_IRQHandler() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_FillCircle() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_FillEllipse() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_FillPolygon() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_FillRect() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetBackColor() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetFont() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetTextColor() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetXSize() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_GetYSize() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_Init() : **stm32l4r9i_eval_dsi_lcd.h**
- BSP_DSI_LCD_IsFrameBufferAvailable() :

stm32l4r9i_eval_dsi_lcd.c
- BSP_DSI_LCD_LTDC_ER_IRQHandler() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_LTDC_IRQHandler() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_MspDeInit() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_MspInit() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_ReadPixel() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_Refresh() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_ResetColorKeying() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SelectLayer() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetBackColor() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetBrightness() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetColorKeying() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetFont() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetLayerVisible() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetTextColor() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_LCD_SetTransparency() : **stm32l4r9i_eval_dsi_lcd.c**
- BSP_DSI_TS_DeInit() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_GestureConfig() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_Get_GestureId() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_GetState() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_Init() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_ITClear() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_ITConfig() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_ITGetStatus() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_DSI_TS_ResetTouchData() : **stm32l4r9i_eval_dsi_ts.h**
- BSP_EEPROM_DeInit() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_Init() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_ReadBuffer() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_TIMEOUT_UserCallback() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_WaitEepromStandbyState() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_WriteBuffer() : **stm32l4r9i_eval_eeprom.c**
- BSP_EEPROM_WritePage() : **stm32l4r9i_eval_eeprom.c**
- BSP_ErrorHandler() : **stm32l4r9i_eval.c**
- BSP_GetVersion() : **stm32l4r9i_eval.c**

- BSP_IDD_ClearIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_Config() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_DeInit() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_DisableIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_EnableIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorClearIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorDisableIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorEnableIT() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorGetCode() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_ErrorGetITStatus() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_GetITStatus() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_GetValue() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_Init() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_LowPower() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_Reset() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_StartMeasure() : **stm32l4r9i_eval_idd.c**
- BSP_IDD_WakeUp() : **stm32l4r9i_eval_idd.c**
- BSP_IO_ConfigPin() : **stm32l4r9i_eval_io.c**
- BSP_IO_DeInit() : **stm32l4r9i_eval_io.c**
- BSP_IO_Init() : **stm32l4r9i_eval_io.c**
- BSP_IO_ITClear() : **stm32l4r9i_eval_io.c**
- BSP_IO_ITGetStatus() : **stm32l4r9i_eval_io.c**
- BSP_IO_ReadPin() : **stm32l4r9i_eval_io.c**
- BSP_IO_TogglePin() : **stm32l4r9i_eval_io.c**
- BSP_IO_WritePin() : **stm32l4r9i_eval_io.c**
- BSP_JOY_DeInit() : **stm32l4r9i_eval.c**
- BSP_JOY_GetState() : **stm32l4r9i_eval.c**
- BSP_JOY_Init() : **stm32l4r9i_eval.c**
- BSP_LED_DeInit() : **stm32l4r9i_eval.c**
- BSP_LED_Init() : **stm32l4r9i_eval.c**
- BSP_LED_Off() : **stm32l4r9i_eval.c**
- BSP_LED_On() : **stm32l4r9i_eval.c**
- BSP_LED_Toggle() : **stm32l4r9i_eval.c**
- BSP_NOR_Erase_Block() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_Erase_Chip() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_Init() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_ProgramData() : **stm32l4r9i_eval_nor.c**

- BSP_NOR_Read_ID() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_ReadData() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_ReturnToReadMode() : **stm32l4r9i_eval_nor.c**
- BSP_NOR_WriteData() : **stm32l4r9i_eval_nor.c**
- BSP_OSPI_NOR_DeInit() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_EnableMemoryMappedMode() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_EnterDeepPowerDown() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Erase_Block() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Erase_Chip() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Erase_Sector() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_GetInfo() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_GetStatus() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Init() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_LeaveDeepPowerDown() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Read() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_ResumeErase() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_SuspendErase() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_NOR_Write() : **stm32l4r9i_eval_ospi_nor.c**
- BSP_OSPI_RAM_DeInit() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_DMA_IRQHandler() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_EnableMemoryMappedMode() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_EnterDeepPowerDown() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Init() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_IRQHandler() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_LeaveDeepPowerDown() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Read() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Read_DMA() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Write() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_OSPI_RAM_Write_DMA() : **stm32l4r9i_eval_ospi_ram.c**
- BSP_PB_DeInit() : **stm32l4r9i_eval.c**

- BSP_PB_GetState() : **stm32l4r9i_eval.c**
- BSP_PB_Init() : **stm32l4r9i_eval.c**
- BSP_POTENTIOMETER_DeInit() : **stm32l4r9i_eval.c**
- BSP_POTENTIOMETER_GetLevel() : **stm32l4r9i_eval.c**
- BSP_POTENTIOMETER_Init() : **stm32l4r9i_eval.c**
- BSP_RGB_LCD_Clear() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_ClearStringLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_ClockConfig() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DeInit() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayChar() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayOff() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayOn() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayStringAt() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DisplayStringAtLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DMA2D_IRQHandler() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawBitmap() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawCircle() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawEllipse() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawHLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawPixel() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawPolygon() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawRect() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_DrawVLine() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_FillCircle() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_FillEllipse() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_FillPolygon() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_FillRect() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetBackColor() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetFont() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetTextColor() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetXSize() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_GetYSize() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_Init() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_LayerDefaultInit() : **stm32l4r9i_eval_rgb_lcd.h**

- BSP_RGB_LCD_LayerRgb565Init() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_LTDC_ER_IRQHandler() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_LTDC_IRQHandler() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_MspDeInit() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_MspInit() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_ReadPixel() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_Reload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_ResetColorKeying() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_ResetColorKeying_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SelectLayer() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetBackColor() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetColorKeying() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetColorKeying_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetFont() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerAddress() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerAddress_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerVisible() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerVisible_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerWindow() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetLayerWindow_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetTextColor() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetTransparency() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetTransparency_NoReload() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetXSize() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_LCD_SetYSize() : **stm32l4r9i_eval_rgb_lcd.h**
- BSP_RGB_TS_DeInit() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_Get_GestureId() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_GetState() : **stm32l4r9i_eval_rgb_ts.h**

- BSP_RGB_TS_Init() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_ITClear() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_ITConfig() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_ITGetStatus() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_RGB_TS_ResetTouchData() : **stm32l4r9i_eval_rgb_ts.h**
- BSP_SD_AbortCallback() : **stm32l4r9i_eval_sd.c**
- BSP_SD_DeInit() : **stm32l4r9i_eval_sd.c**
- BSP_SD_Erase() : **stm32l4r9i_eval_sd.c**
- BSP_SD_GetCardInfo() : **stm32l4r9i_eval_sd.c**
- BSP_SD_GetCardState() : **stm32l4r9i_eval_sd.c**
- BSP_SD_Init() : **stm32l4r9i_eval_sd.c**
- BSP_SD_IsDetected() : **stm32l4r9i_eval_sd.c**
- BSP_SD_ITConfig() : **stm32l4r9i_eval_sd.c**
- BSP_SD_MspDeInit() : **stm32l4r9i_eval_sd.c**
- BSP_SD_MspInit() : **stm32l4r9i_eval_sd.c**
- BSP_SD_ReadBlocks() : **stm32l4r9i_eval_sd.c**
- BSP_SD_ReadBlocks_DMA() : **stm32l4r9i_eval_sd.c**
- BSP_SD_ReadCpltCallback() : **stm32l4r9i_eval_sd.c**
- BSP_SD_WriteBlocks() : **stm32l4r9i_eval_sd.c**
- BSP_SD_WriteBlocks_DMA() : **stm32l4r9i_eval_sd.c**
- BSP_SD_WriteCpltCallback() : **stm32l4r9i_eval_sd.c**
- BSP_SRAM_DMA_IRQHandler() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_Init() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_ReadData() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_ReadData_DMA() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_WriteData() : **stm32l4r9i_eval_sram.c**
- BSP_SRAM_WriteData_DMA() : **stm32l4r9i_eval_sram.c**

STM32L4R9I_EVAL BSP User Manual

- d -

- DrawChar() : **stm32l4r9i_eval_dsi_lcd.c**

STM32L4R9I_EVAL BSP User Manual

- e -

- EEPROM_IO_Init() : **stm32l4r9i_eval.c** , **stm32l4r9i_eval_eeprom.h**
- EEPROM_IO_IsDeviceReady() : **stm32l4r9i_eval_eeprom.h** , **stm32l4r9i_eval.c**
- EEPROM_IO_ReadData() : **stm32l4r9i_eval.c** , **stm32l4r9i_eval_eeprom.h**
- EEPROM_IO_WriteData() : **stm32l4r9i_eval_eeprom.h** , **stm32l4r9i_eval.c**

STM32L4R9I_EVAL BSP User Manual

- f -

- FillTriangle() : **stm32l4r9i_eval_dsi_lcd.c**

STM32L4R9I_EVAL BSP User Manual

- h -

- HAL_DFSDM_ChannelMspDeInit() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_ChannelMspInit() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterErrorCallback() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterMspDeInit() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterMspInit() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterRegConvCpltCallback() : **stm32l4r9i_eval_audio.c**
- HAL_DFSDM_FilterRegConvHalfCpltCallback() : **stm32l4r9i_eval_audio.c**
- HAL_DSI_EndOfRefreshCallback() : **stm32l4r9i_eval_dsi_lcd.c**
- HAL_NOR_MspWait() : **stm32l4r9i_eval_nor.c**
- HAL_SAI_ErrorCallback() : **stm32l4r9i_eval_audio.c**
- HAL_SAI_MspDeInit() : **stm32l4r9i_eval_audio.c**
- HAL_SAI_MspInit() : **stm32l4r9i_eval_audio.c**
- HAL_SAI_TxCpltCallback() : **stm32l4r9i_eval_audio.c**
- HAL_SAI_TxHalfCpltCallback() : **stm32l4r9i_eval_audio.c**
- HAL_SD_AbortCallback() : **stm32l4r9i_eval_sd.c**
- HAL_SD_RxCpltCallback() : **stm32l4r9i_eval_sd.c**
- HAL_SD_TxCpltCallback() : **stm32l4r9i_eval_sd.c**

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

- i -

- I2Cx_DeInit() : **stm32l4r9i_eval.c**
- I2Cx_Error() : **stm32l4r9i_eval.c**
- I2Cx_Init() : **stm32l4r9i_eval.c**
- I2Cx_IsDeviceReady() : **stm32l4r9i_eval.c**
- I2Cx_MspDeInit() : **stm32l4r9i_eval.c**
- I2Cx_MspInit() : **stm32l4r9i_eval.c**
- I2Cx_ReadBuffer() : **stm32l4r9i_eval.c**
- I2Cx_ReadData() : **stm32l4r9i_eval.c**
- I2Cx_ReadMultiple() : **stm32l4r9i_eval.c**
- I2Cx_WriteBuffer() : **stm32l4r9i_eval.c**
- I2Cx_WriteData() : **stm32l4r9i_eval.c**
- I2Cx_WriteMultiple() : **stm32l4r9i_eval.c**

STM32L4R9I_EVAL BSP User Manual

- l -

- LCD_Reset() : **stm32l4r9i_eval_dsi_lcd.c**
- LL_ConvertLineToARGB8888() : **stm32l4r9i_eval_dsi_lcd.c**
- LL_FillBuffer() : **stm32l4r9i_eval_dsi_lcd.c**

STM32L4R9I_EVAL BSP User Manual

- m -

- MFX_IO_DeInit() : **stm32l4r9i_eval.c**
- MFX_IO_Delay() : **stm32l4r9i_eval.c**
- MFX_IO_EnableWakeupPin() : **stm32l4r9i_eval.c**
- MFX_IO_Init() : **stm32l4r9i_eval.c**
- MFX_IO_ITConfig() : **stm32l4r9i_eval.c**
- MFX_IO_Read() : **stm32l4r9i_eval.c**
- MFX_IO_ReadMultiple() : **stm32l4r9i_eval.c**
- MFX_IO_Wakeup() : **stm32l4r9i_eval.c**
- MFX_IO_Write() : **stm32l4r9i_eval.c**

STM32L4R9I_EVAL BSP User Manual

- n -

- NOR_MspInit() : **stm32l4r9i_eval_nor.c**

STM32L4R9I_EVAL BSP User Manual

- o -

- OSPI_NOR_AutoPollingMemReady() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_MspDeInit() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_MspInit() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_OctalMode() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_ResetMemory() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_WriteEnable() : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_RAM_MspDeInit() : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_MspInit() : **stm32l4r9i_eval_ospi_ram.c**

STM32L4R9I_EVAL BSP User Manual

- s -

- SRAM_MspInit() : **stm32l4r9i_eval_sram.c**

STM32L4R9I_EVAL BSP User Manual

- t -

- TS_IO_Delay() : **stm32l4r9i_eval.c**
- TS_IO_Init() : **stm32l4r9i_eval.c**
- TS_IO_Read() : **stm32l4r9i_eval.c**
- TS_IO_ReadMultiple() : **stm32l4r9i_eval.c**
- TS_IO_Write() : **stm32l4r9i_eval.c**

STM32L4R9I_EVAL BSP User Manual

- a -

- ActiveLayer : **stm32l4r9i_eval_dsi_lcd.c**

- b -

- BSP_AUDIO_hDfsdmLeftFilter : **stm32l4r9i_eval_audio.c** , **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_hDfsdmRightFilter : **stm32l4r9i_eval_audio.h** , **stm32l4r9i_eval_audio.c**
- BSP_AUDIO_hSai : **stm32l4r9i_eval_audio.c** , **stm32l4r9i_eval_audio.h**
- bsp_lcd_hse_to_disable : **stm32l4r9i_eval_dsi_lcd.c**
- bsp_lcd_initialized : **stm32l4r9i_eval_dsi_lcd.c**
- BUTTON_IRQn : **stm32l4r9i_eval.c**
- BUTTON_PIN : **stm32l4r9i_eval.c**
- BUTTON_PORT : **stm32l4r9i_eval.c**

- c -

- COM_RX_AF : **stm32l4r9i_eval.c**
- COM_RX_PIN : **stm32l4r9i_eval.c**

- COM_RX_PORT : **stm32l4r9i_eval.c**
- COM_TX_AF : **stm32l4r9i_eval.c**
- COM_TX_PIN : **stm32l4r9i_eval.c**
- COM_TX_PORT : **stm32l4r9i_eval.c**
- COM_USART : **stm32l4r9i_eval.c**

- d -

- DmaLeftRecBuffCplt : **stm32l4r9i_eval_audio.c**
- DmaLeftRecHalfBuffCplt : **stm32l4r9i_eval_audio.c**
- DmaOspiRamHandle : **stm32l4r9i_eval_ospi_ram.c**
- DmaRightRecBuffCplt : **stm32l4r9i_eval_audio.c**
- DmaRightRecHalfBuffCplt : **stm32l4r9i_eval_audio.c**
- DrawProp : **stm32l4r9i_eval_dsi_lcd.c**

- e -

- EEPROMAddress : **stm32l4r9i_eval_eeprom.c**
- EEPROMDataRead : **stm32l4r9i_eval_eeprom.c**
- EEPROMDataWrite : **stm32l4r9i_eval_eeprom.c**

- g -

- gfxmmu_lut_config_argb8888 : **stm32l4r9i_eval_gfxmmu_lut.h**

- h -

- hAudioIn : **stm32l4r9i_eval_audio.c**
- hAudioOut : **stm32l4r9i_eval_audio.c**
- hdma2d_eval : **stm32l4r9i_eval_dsi_lcd.h** , **stm32l4r9i_eval_dsi_lcd.c**
- hDmaSai : **stm32l4r9i_eval_audio.c**
- hEvalADC : **stm32l4r9i_eval.c**
- hi2c_eval : **stm32l4r9i_eval.c**
- hi2c_evalTimeout : **stm32l4r9i_eval.c**
- hltdc_eval : **stm32l4r9i_eval_rgb_lcd.h**
- hsd_eval : **stm32l4r9i_eval_sd.h** , **stm32l4r9i_eval_sd.c**

- i -

- IddDrv : **stm32l4r9i_eval_idd.c**
- io_driver : **stm32l4r9i_eval_io.c**

- j -

- JOY_IRQn : **stm32l4r9i_eval.c**
- JOY_PIN : **stm32l4r9i_eval.c**

- l -

- LED_PIN : **stm32l4r9i_eval.c**
- LED_PORT : **stm32l4r9i_eval.c**

- n -

- norHandle : **stm32l4r9i_eval_nor.c**

- o -

- OSPINORHandle : **stm32l4r9i_eval_ospi_nor.c**
- OspiRamHandle : **stm32l4r9i_eval_ospi_ram.c**

- p -

- PhysFrameBuffer : **stm32l4r9i_eval_rgb_lcd.h**

- s -

- sramHandle : **stm32l4r9i_eval_sram.c**

- t -

- Timing : **stm32l4r9i_eval_nor.c** , **stm32l4r9i_eval_sram.c**
- ts_event_string_tab : **stm32l4r9i_eval_rgb_ts.h**
- ts_gesture_id_string_tab : **stm32l4r9i_eval_rgb_ts.h**

- u -

- UseExtiModeDetection : **stm32l4r9i_eval_sd.c**

- v -

- v_bspI2cUsers : **stm32l4r9i_eval.c**

STM32L4R9I_EVAL BSP User Manual

- Audio_CallbackTypeDef : **stm32l4r9i_eval_audio.h**
- pPoint : **stm32l4r9i_eval_lcd.h**

STM32L4R9I_EVAL BSP User Manual

- Button_TypeDef : **stm32l4r9i_eval.h**
- ButtonMode_TypeDef : **stm32l4r9i_eval.h**
- COM_TypeDef : **stm32l4r9i_eval.h**
- IDD_StatusTypeDef : **stm32l4r9i_eval_idd.h**
- IO_StatusTypeDef : **stm32l4r9i_eval_io.h**
- JOYMode_TypeDef : **stm32l4r9i_eval.h**
- JOYState_TypeDef : **stm32l4r9i_eval.h**
- LCD_OrientationTypeDef : **stm32l4r9i_eval_lcd.h**
- Led_TypeDef : **stm32l4r9i_eval.h**
- Text_AlignModeTypdef : **stm32l4r9i_eval_lcd.h**
- TS_GestureIdTypeDef : **stm32l4r9i_eval_ts.h**
- TS_StatusTypeDef : **stm32l4r9i_eval_ts.h**
- TS_TouchEventTypeDef : **stm32l4r9i_eval_ts.h**

STM32L4R9I_EVAL BSP User Manual

- b -

- BUTTON_DOWN : **stm32l4r9i_eval.h**
- BUTTON_LEFT : **stm32l4r9i_eval.h**
- BUTTON_MODE_EXTI : **stm32l4r9i_eval.h**
- BUTTON_MODE_GPIO : **stm32l4r9i_eval.h**
- BUTTON_RIGHT : **stm32l4r9i_eval.h**
- BUTTON_SEL : **stm32l4r9i_eval.h**
- BUTTON_UP : **stm32l4r9i_eval.h**
- BUTTON_WAKEUP : **stm32l4r9i_eval.h**

- c -

- CENTER_MODE : **stm32l4r9i_eval_lcd.h**
- COM1 : **stm32l4r9i_eval.h**

- g -

- GEST_ID_DOUBLE_CLICK : **stm32l4r9i_eval_ts.h**
- GEST_ID_MOVE_DOWN : **stm32l4r9i_eval_ts.h**
- GEST_ID_MOVE_LEFT : **stm32l4r9i_eval_ts.h**
- GEST_ID_MOVE_RIGHT : **stm32l4r9i_eval_ts.h**

- GEST_ID_MOVE_UP : **stm32l4r9i_eval_ts.h**
- GEST_ID_NB_MAX : **stm32l4r9i_eval_ts.h**
- GEST_ID_NO_GESTURE : **stm32l4r9i_eval_ts.h**
- GEST_ID_SINGLE_CLICK : **stm32l4r9i_eval_ts.h**
- GEST_ID_ZOOM_IN : **stm32l4r9i_eval_ts.h**
- GEST_ID_ZOOM_OUT : **stm32l4r9i_eval_ts.h**

- i -

- IDD_ERROR : **stm32l4r9i_eval_idd.h**
- IDD_OK : **stm32l4r9i_eval_idd.h**
- IDD_TIMEOUT : **stm32l4r9i_eval_idd.h**
- IDD_ZERO_VALUE : **stm32l4r9i_eval_idd.h**
- IO_ERROR : **stm32l4r9i_eval_io.h**
- IO_OK : **stm32l4r9i_eval_io.h**
- IO_TIMEOUT : **stm32l4r9i_eval_io.h**

- j -

- JOY_DOWN : **stm32l4r9i_eval.h**
- JOY_LEFT : **stm32l4r9i_eval.h**
- JOY_MODE_EXTI : **stm32l4r9i_eval.h**
- JOY_MODE_GPIO : **stm32l4r9i_eval.h**
- JOY_NONE : **stm32l4r9i_eval.h**
- JOY_RIGHT : **stm32l4r9i_eval.h**
- JOY_SEL : **stm32l4r9i_eval.h**
- JOY_UP : **stm32l4r9i_eval.h**

- l -

- LCD_ORIENTATION_INVALID : **stm32l4r9i_eval_lcd.h**
- LCD_ORIENTATION_LANDSCAPE : **stm32l4r9i_eval_lcd.h**
- LCD_ORIENTATION_PORTRAIT : **stm32l4r9i_eval_lcd.h**
- LED1 : **stm32l4r9i_eval.h**
- LED2 : **stm32l4r9i_eval.h**
- LED3 : **stm32l4r9i_eval.h**
- LED4 : **stm32l4r9i_eval.h**
- LED_BLUE : **stm32l4r9i_eval.h**

- LED_GREEN : **stm32l4r9i_eval.h**
- LED_ORANGE : **stm32l4r9i_eval.h**
- LED_RED : **stm32l4r9i_eval.h**
- LEFT_MODE : **stm32l4r9i_eval_lcd.h**

- r -

- RIGHT_MODE : **stm32l4r9i_eval_lcd.h**

- t -

- TOUCH_EVENT_CONTACT : **stm32l4r9i_eval_ts.h**
- TOUCH_EVENT_LIFT_UP : **stm32l4r9i_eval_ts.h**
- TOUCH_EVENT_NB_MAX : **stm32l4r9i_eval_ts.h**
- TOUCH_EVENT_NO_EVT : **stm32l4r9i_eval_ts.h**
- TOUCH_EVENT_PRESS_DOWN : **stm32l4r9i_eval_ts.h**
- TS_DEVICE_NOT_FOUND : **stm32l4r9i_eval_ts.h**
- TS_ERROR : **stm32l4r9i_eval_ts.h**
- TS_OK : **stm32l4r9i_eval_ts.h**
- TS_TIMEOUT : **stm32l4r9i_eval_ts.h**

STM32L4R9I_EVAL BSP User Manual

- _ -

- __STM32L4R9I_EVAL_BSP_VERSION : **stm32l4r9i_eval.c**
- __STM32L4R9I_EVAL_BSP_VERSION_MAIN : **stm32l4r9i_eval.c**
- __STM32L4R9I_EVAL_BSP_VERSION_RC : **stm32l4r9i_eval.c**
- __STM32L4R9I_EVAL_BSP_VERSION_SUB1 : **stm32l4r9i_eval.c**
- __STM32L4R9I_EVAL_BSP_VERSION_SUB2 : **stm32l4r9i_eval.c**

STM32L4R9I_EVAL BSP User Manual

- a -

- ABS : **stm32l4r9i_eval_dsi_lcd.c**
- ADCx : **stm32l4r9i_eval.h**
- ADCx_CHANNEL : **stm32l4r9i_eval.h**
- ADCx_CHANNEL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- ADCx_CHANNEL_GPIO_PORT : **stm32l4r9i_eval.h**
- ADCx_CHANNEL_PIN : **stm32l4r9i_eval.h**
- ADCx_CLK_ENABLE : **stm32l4r9i_eval.h**
- ADCx_FORCE_RESET : **stm32l4r9i_eval.h**
- ADCx_POLL_TIMEOUT : **stm32l4r9i_eval.h**
- ADCx_RELEASE_RESET : **stm32l4r9i_eval.h**
- AGPIO_PIN_0 : **stm32l4r9i_eval_io.h**
- AGPIO_PIN_1 : **stm32l4r9i_eval_io.h**
- AUDIO_DFSDM_DMAx_LEFT_IRQHandler : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDM_DMAx_RIGHT_IRQHandler : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CKOUT_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CKOUT_CLK_ENABLE : **stm32l4r9i_eval_audio.h**

- AUDIO_DFSDMx_CKOUT_DATIN_AF : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CKOUT_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CKOUT_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DATIN_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DATIN_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DATIN_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DATIN_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_LEFT_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_LEFT_IRQ : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_LEFT_REQUEST : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_RIGHT_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_RIGHT_IRQ : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_DMAx_RIGHT_REQUEST : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_LEFT_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_LEFT_FILTER : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_RIGHT_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_DFSDMx_RIGHT_FILTER : **stm32l4r9i_eval_audio.h**
- AUDIO_ERROR : **stm32l4r9i_eval_audio.h**
- AUDIO_I2C_ADDRESS : **stm32l4r9i_eval.h**

- AUDIO_IN_IRQ_PREPRIO : **stm32l4r9i_eval_audio.h**
- AUDIO_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- AUDIO_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- AUDIO_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- AUDIO_INT_PIN : **stm32l4r9i_eval.h**
- AUDIO_OK : **stm32l4r9i_eval_audio.h**
- AUDIO_OUT_IRQ_PREPRIO : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_CHANNEL : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_IRQ : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_IRQHandler : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_MEM_DATA_SIZE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_PERIPH_DATA_SIZE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_DMAx_REQUEST : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_FS_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_FS_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_FS_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_FS_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_MCK_SCK_SD_FS_AF : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_PLL_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SCK_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SCK_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SCK_GPIO_PORT : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SCK_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SD_CLK_DISABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SD_CLK_ENABLE : **stm32l4r9i_eval_audio.h**
- AUDIO_SAIx_SD_GPIO_PORT : **stm32l4r9i_eval_audio.h**

- AUDIO_SAIx_SD_PIN : **stm32l4r9i_eval_audio.h**
- AUDIO_TIMEOUT : **stm32l4r9i_eval_audio.h**
- AUDIODATA_SIZE : **stm32l4r9i_eval_audio.h**

STM32L4R9I_EVAL BSP User Manual

- b -

- BLOCKERASE_TIMEOUT : **stm32l4r9i_eval_nor.h**
- BSP_AUDIO_FREQUENCY_11K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_16K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_22K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_32K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_44K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_48K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_8K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_FREQUENCY_96K : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_OUT_CIRCULARMODE : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_OUT_MONOMODE : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_OUT_NORMALMODE : **stm32l4r9i_eval_audio.h**
- BSP_AUDIO_OUT_STEREOMODE : **stm32l4r9i_eval_audio.h**
- BSP_I2C_ALL_USERS : **stm32l4r9i_eval.c**
- BSP_I2C_AUDIO_USER : **stm32l4r9i_eval.c**
- BSP_I2C_EEPROM_USER : **stm32l4r9i_eval.c**
- BSP_I2C_MFX_USER : **stm32l4r9i_eval.c**
- BSP_I2C_NO_USER : **stm32l4r9i_eval.c**
- BSP_I2C_TS_USER : **stm32l4r9i_eval.c**
- BSP_POTENTIOMETER_ERROR : **stm32l4r9i_eval.h**

- BSP_POTENTIOMETER_OK : **stm32l4r9i_eval.h**
- BSP_SD_CardInfo : **stm32l4r9i_eval_sd.h**
- BUTTONn : **stm32l4r9i_eval.h**
- BUTTONx_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- BUTTONx_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- c -

- CHIPERASE_TIMEOUT : **stm32l4r9i_eval_nor.h**
- CODEC_RESET_DELAY : **stm32l4r9i_eval_audio.h**
- COMn : **stm32l4r9i_eval.h**
- COMx_CLK_DISABLE : **stm32l4r9i_eval.h**
- COMx_CLK_ENABLE : **stm32l4r9i_eval.h**
- COMx_RX_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- COMx_RX_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- COMx_TX_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- COMx_TX_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- d -

- DEFAULT_AUDIO_IN_BIT_RESOLUTION : **stm32l4r9i_eval_audio.h**
- DEFAULT_AUDIO_IN_CHANNEL_NBR : **stm32l4r9i_eval_audio.h**
- DEFAULT_AUDIO_IN_FREQ : **stm32l4r9i_eval_audio.h**
- DEFAULT_AUDIO_IN_VOLUME : **stm32l4r9i_eval_audio.h**
- DFSDMClockDivider : **stm32l4r9i_eval_audio.c**
- DFSDMFilterOrder : **stm32l4r9i_eval_audio.c**
- DFSDMOverSampling : **stm32l4r9i_eval_audio.c**
- DFSDMRightBitShift : **stm32l4r9i_eval_audio.c**
- DMA_MAX : **stm32l4r9i_eval_audio.h**
- DMA_MAX_SZE : **stm32l4r9i_eval_audio.h**
- DOWN_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- DOWN_JOY_PIN : **stm32l4r9i_eval.h**
- DSI_RST_PIN : **stm32l4r9i_eval.h**
- DUMMY_CYCLES_CR_CFG : **stm32l4r9i_eval_ospi_nor.c**
- DUMMY_CYCLES_READ_OCTAL : **stm32l4r9i_eval_ospi_nor.c**

User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

- e -

- EEPROM_FAIL : **stm32l4r9i_eval_eeprom.h**
- EEPROM_I2C_ADDRESS : **stm32l4r9i_eval.h**
- EEPROM_MAX_SIZE : **stm32l4r9i_eval_eeprom.h**
- EEPROM_MAX_TRIALS : **stm32l4r9i_eval_eeprom.h**
- EEPROM_OK : **stm32l4r9i_eval_eeprom.h**
- EEPROM_PAGESIZE : **stm32l4r9i_eval_eeprom.h**
- EEPROM_TIMEOUT : **stm32l4r9i_eval_eeprom.h**
- EVAL_COM1 : **stm32l4r9i_eval.h**
- EVAL_COM1_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_IRQn : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_AF : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_COM1_RX_PIN : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_AF : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_COM1_TX_GPIO_PORT : **stm32l4r9i_eval.h**

- EVAL_COM1_TX_PIN : **stm32l4r9i_eval.h**
- EVAL_I2Cx : **stm32l4r9i_eval.h**
- EVAL_I2Cx_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_ER_IRQHandler : **stm32l4r9i_eval.h**
- EVAL_I2Cx_ER_IRQn : **stm32l4r9i_eval.h**
- EVAL_I2Cx_EV_IRQHandler : **stm32l4r9i_eval.h**
- EVAL_I2Cx_EV_IRQn : **stm32l4r9i_eval.h**
- EVAL_I2Cx_FORCE_RESET : **stm32l4r9i_eval.h**
- EVAL_I2Cx_RELEASE_RESET : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_PIN : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SCL_SDA_AF : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SDA_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SDA_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SDA_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_I2Cx_SDA_PIN : **stm32l4r9i_eval.h**
- EVAL_I2Cx_TIMEOUT_MAX : **stm32l4r9i_eval.h**
- EVAL_IDD_AMPLI_GAIN : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT0_STABDELAY : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT0_VALUE : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT1_STABDELAY : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT1_VALUE : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT2_STABDELAY : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT2_VALUE : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT4_STABDELAY : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_SHUNT4_VALUE : **stm32l4r9i_eval_idd.h**
- EVAL_IDD_VDD_MIN : **stm32l4r9i_eval_idd.h**
- EVAL_RCC_CLKSOURCE_I2C : **stm32l4r9i_eval.h**
- EVAL_RCC_PERIPHCLK_I2C : **stm32l4r9i_eval.h**
- EVAL_TS_DMAx_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_ER_IRQn : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_EV_IRQn : **stm32l4r9i_eval.h**

- EVAL_TS_I2Cx_FORCE_RESET : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_RELEASE_RESET : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_PIN : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SCL_SDA_AF : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SDA_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SDA_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SDA_GPIO_PORT : **stm32l4r9i_eval.h**
- EVAL_TS_I2Cx_SDA_PIN : **stm32l4r9i_eval.h**
- EVAL_TS_RCC_CLKSOURCE_I2C : **stm32l4r9i_eval.h**
- EVAL_TS_RCC_PERIPHCLK_I2C : **stm32l4r9i_eval.h**
- EXT_RST_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- i -

- IDD_AMP_CONTROL_PIN : **stm32l4r9i_eval.h**
- IDD_I2C_ADDRESS : **stm32l4r9i_eval.h**
- IDD_INT_EXTI_IRQn : **stm32l4r9i_eval.h**
- IDD_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- IDD_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- IDD_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- IDD_INT_PIN : **stm32l4r9i_eval.h**
- IDD_WAKEUP_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- IDD_WAKEUP_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- IDD_WAKEUP_GPIO_PORT : **stm32l4r9i_eval.h**
- IDD_WAKEUP_PIN : **stm32l4r9i_eval.h**
- INTERNAL_BUFF_SIZE : **stm32l4r9i_eval_audio.h**
- IO_I2C_ADDRESS : **stm32l4r9i_eval.h**
- IO_PIN_0 : **stm32l4r9i_eval_io.h**
- IO_PIN_1 : **stm32l4r9i_eval_io.h**
- IO_PIN_10 : **stm32l4r9i_eval_io.h**
- IO_PIN_11 : **stm32l4r9i_eval_io.h**
- IO_PIN_12 : **stm32l4r9i_eval_io.h**
- IO_PIN_13 : **stm32l4r9i_eval_io.h**
- IO_PIN_14 : **stm32l4r9i_eval_io.h**

- IO_PIN_15 : **stm32l4r9i_eval_io.h**
- IO_PIN_2 : **stm32l4r9i_eval_io.h**
- IO_PIN_3 : **stm32l4r9i_eval_io.h**
- IO_PIN_4 : **stm32l4r9i_eval_io.h**
- IO_PIN_5 : **stm32l4r9i_eval_io.h**
- IO_PIN_6 : **stm32l4r9i_eval_io.h**
- IO_PIN_7 : **stm32l4r9i_eval_io.h**
- IO_PIN_8 : **stm32l4r9i_eval_io.h**
- IO_PIN_9 : **stm32l4r9i_eval_io.h**
- IO_PIN_ALL : **stm32l4r9i_eval_io.h**

STM32L4R9I_EVAL BSP User Manual

- j -

- JOY_ALL_PINS : **stm32l4r9i_eval.h**
- JOY_NONE_PIN : **stm32l4r9i_eval.h**
- JOYn : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- l -

- LCD_COLOR_BLACK : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_BLUE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_BROWN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_CYAN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKBLUE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKCYAN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKGRAY : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKGREEN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKMAGENTA : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKRED : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_DARKYELLOW : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_GRAY : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_GREEN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTBLUE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTCYAN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTGRAY : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTGREEN : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTMAGENTA : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTRED : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_LIGHTYELLOW : **stm32l4r9i_eval_lcd.h**

- LCD_COLOR_MAGENTA : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_ORANGE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_RED : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_TRANSPARENT : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_WHITE : **stm32l4r9i_eval_lcd.h**
- LCD_COLOR_YELLOW : **stm32l4r9i_eval_lcd.h**
- LCD_DEFAULT_FONT : **stm32l4r9i_eval_lcd.h**
- LCD_DISP_PIN : **stm32l4r9i_eval.h**
- LCD_DSI_BL_CTRL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LCD_DSI_BL_CTRL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LCD_DSI_BL_CTRL_GPIO_PORT : **stm32l4r9i_eval.h**
- LCD_DSI_BL_CTRL_PIN : **stm32l4r9i_eval.h**
- LCD_DSI_TE_CTRL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LCD_DSI_TE_CTRL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LCD_DSI_TE_CTRL_GPIO_PORT : **stm32l4r9i_eval.h**
- LCD_DSI_TE_PIN : **stm32l4r9i_eval.h**
- LCD_ERROR : **stm32l4r9i_eval_lcd.h**
- LCD_FB_START_ADDRESS : **stm32l4r9i_eval_rgb_lcd.h**
- LCD_LayerCfgTypeDef : **stm32l4r9i_eval_lcd.h**
- LCD_OK : **stm32l4r9i_eval_lcd.h**
- LCD_RELOAD_IMMEDIATE : **stm32l4r9i_eval_rgb_lcd.h**
- LCD_RELOAD_VERTICAL_BLANKING : **stm32l4r9i_eval_rgb_lcd.h**
- LCD_RGB_BL_CTRL_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LCD_RGB_BL_CTRL_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LCD_RGB_BL_CTRL_GPIO_PORT : **stm32l4r9i_eval.h**
- LCD_RGB_BL_CTRL_PIN : **stm32l4r9i_eval.h**
- LCD_RGB_DISP_PIN : **stm32l4r9i_eval.h**
- LCD_RGB_I2C_ADDRESS : **stm32l4r9i_eval.h**
- LCD_RGB_RST_PIN : **stm32l4r9i_eval.h**
- LCD_RST_PIN : **stm32l4r9i_eval.h**
- LCD_TIMEOUT : **stm32l4r9i_eval_lcd.h**
- LED1_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LED1_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LED1_GPIO_PORT : **stm32l4r9i_eval.h**
- LED1_PIN : **stm32l4r9i_eval.h**
- LED2_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**

- LED2_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LED2_GPIO_PORT : **stm32l4r9i_eval.h**
- LED2_PIN : **stm32l4r9i_eval.h**
- LED3_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LED3_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LED3_GPIO_PORT : **stm32l4r9i_eval.h**
- LED3_PIN : **stm32l4r9i_eval.h**
- LED4_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LED4_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LED4_GPIO_PORT : **stm32l4r9i_eval.h**
- LED4_PIN : **stm32l4r9i_eval.h**
- LEDn : **stm32l4r9i_eval.h**
- LEDx_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- LEDx_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- LEFT_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- LEFT_JOY_PIN : **stm32l4r9i_eval.h**
- LTDC_ACTIVE_LAYER_BACKGROUND : **stm32l4r9i_eval_lcd.h**
- LTDC_ACTIVE_LAYER_FOREGROUND : **stm32l4r9i_eval_lcd.h**
- LTDC_DEFAULT_ACTIVE_LAYER : **stm32l4r9i_eval_lcd.h**
- LTDC_MAX_LAYER_NUMBER : **stm32l4r9i_eval_lcd.h**
- LTDC_NB_OF_LAYERS : **stm32l4r9i_eval_lcd.h**

STM32L4R9I_EVAL BSP User Manual

- m -

- MFX_INT_EXTI_IRQn : **stm32l4r9i_eval.h**
- MFX_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- MFX_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- MFX_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- MFX_INT_PIN : **stm32l4r9i_eval.h**
- MFX_WAKEUP_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- MFX_WAKEUP_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- MFX_WAKEUP_GPIO_PORT : **stm32l4r9i_eval.h**
- MFX_WAKEUP_PIN : **stm32l4r9i_eval.h**
- MSD_ERROR : **stm32l4r9i_eval_sd.h**
- MSD_ERROR_SD_NOT_PRESENT : **stm32l4r9i_eval_sd.h**
- MSD_OK : **stm32l4r9i_eval_sd.h**

STM32L4R9I_EVAL BSP User Manual

- n -

- NOR_ADDR_SETUP_TIME : **stm32l4r9i_eval_nor.c**
- NOR_BURSTACCESS : **stm32l4r9i_eval_nor.h**
- NOR_BUSY_STATE : **stm32l4r9i_eval_nor.h**
- NOR_DATA_HOLD_TIME : **stm32l4r9i_eval_nor.c**
- NOR_DATA_SETUP_TIME : **stm32l4r9i_eval_nor.c**
- NOR_DEVICE_ADDR : **stm32l4r9i_eval_nor.h**
- NOR_MAX_BUFFER_SIZE : **stm32l4r9i_eval_nor.c**
- NOR_MEMORY_WIDTH : **stm32l4r9i_eval_nor.h**
- NOR_READY_BUSY_GPIO : **stm32l4r9i_eval_nor.h**
- NOR_READY_BUSY_PIN : **stm32l4r9i_eval_nor.h**
- NOR_READY_STATE : **stm32l4r9i_eval_nor.h**
- NOR_STATUS_ERROR : **stm32l4r9i_eval_nor.h**
- NOR_STATUS_OK : **stm32l4r9i_eval_nor.h**
- NOR_TURN_AROUND_TIME : **stm32l4r9i_eval_nor.c**
- NOR_WRITEBURST : **stm32l4r9i_eval_nor.h**

STM32L4R9I_EVAL BSP User Manual

- o -

- OPAMPx : **stm32l4r9i_eval.h**
- OPAMPx_CLK_DISABLE : **stm32l4r9i_eval.h**
- OPAMPx_CLK_ENABLE : **stm32l4r9i_eval.h**
- OPAMPx_FORCE_RESET : **stm32l4r9i_eval.h**
- OPAMPx_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- OPAMPx_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- OPAMPx_INM_GPIO_PORT : **stm32l4r9i_eval.h**
- OPAMPx_INM_PIN : **stm32l4r9i_eval.h**
- OPAMPx_INP_GPIO_PORT : **stm32l4r9i_eval.h**
- OPAMPx_INP_PIN : **stm32l4r9i_eval.h**
- OPAMPx_RELEASE_RESET : **stm32l4r9i_eval.h**
- OPAMPx_VOUT_GPIO_PORT : **stm32l4r9i_eval.h**
- OPAMPx_VOUT_PIN : **stm32l4r9i_eval.h**
- OSPI_NOR_BUSY : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_ERROR : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_NOT_SUPPORTED : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_OCTAL_DISABLE : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_OCTAL_ENABLE : **stm32l4r9i_eval_ospi_nor.c**
- OSPI_NOR_OK : **stm32l4r9i_eval_ospi_nor.h**
- OSPI_NOR_SUSPENDED : **stm32l4r9i_eval_ospi_nor.h**

- OSPI_RAM_ERROR : **stm32l4r9i_eval_ospi_ram.h**
- OSPI_RAM_HIGH_PERF_DISABLE : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_HIGH_PERF_ENABLE : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_OK : **stm32l4r9i_eval_ospi_ram.h**
- OSPI_RAM_QUAD_DISABLE : **stm32l4r9i_eval_ospi_ram.c**
- OSPI_RAM_QUAD_ENABLE : **stm32l4r9i_eval_ospi_ram.c**
- OTG_FS_OVER_CURRENT_PIN : **stm32l4r9i_eval.h**
- OTG_FS_POWER_SWITCH_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- p -

- POLY_X : **stm32l4r9i_eval_dsi_lcd.c**
- POLY_Y : **stm32l4r9i_eval_dsi_lcd.c**
- PROGRAM_TIMEOUT : **stm32l4r9i_eval_nor.h**

STM32L4R9I_EVAL BSP User Manual

- r -

- RIGHT_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- RIGHT_JOY_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- s -

- SAIClockDivider : **stm32l4r9i_eval_audio.c**
- SAMPLINGTIME : **stm32l4r9i_eval.h**
- SaturaLH : **stm32l4r9i_eval_audio.c**
- SD_DATATIMEOUT : **stm32l4r9i_eval_sd.h**
- SD_DETECT_PIN : **stm32l4r9i_eval.h**
- SD_NOT_PRESENT : **stm32l4r9i_eval_sd.h**
- SD_PRESENT : **stm32l4r9i_eval_sd.h**
- SD_TRANSFER_BUSY : **stm32l4r9i_eval_sd.h**
- SD_TRANSFER_ERROR : **stm32l4r9i_eval_sd.h**
- SD_TRANSFER_OK : **stm32l4r9i_eval_sd.h**
- SDMMCx_IRQHandler : **stm32l4r9i_eval_sd.h**
- SDMMCx_IRQn : **stm32l4r9i_eval_sd.h**
- SEL_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- SEL_JOY_PIN : **stm32l4r9i_eval.h**
- SRAM_ADDR_SETUP_TIME : **stm32l4r9i_eval_sram.c**
- SRAM_BURSTACCESS : **stm32l4r9i_eval_sram.h**
- SRAM_DATA_HOLD_TIME : **stm32l4r9i_eval_sram.c**
- SRAM_DATA_SETUP_TIME : **stm32l4r9i_eval_sram.c**
- SRAM_DEVICE_ADDR : **stm32l4r9i_eval_sram.h**
- SRAM_DEVICE_SIZE : **stm32l4r9i_eval_sram.h**

- SRAM_DMAx_CHANNEL : **stm32l4r9i_eval_sram.h**
- SRAM_DMAx_CLK_ENABLE : **stm32l4r9i_eval_sram.h**
- SRAM_DMAx_IRQHandler : **stm32l4r9i_eval_sram.h**
- SRAM_DMAx_IRQn : **stm32l4r9i_eval_sram.h**
- SRAM_ERROR : **stm32l4r9i_eval_sram.h**
- SRAM_MEMORY_WIDTH : **stm32l4r9i_eval_sram.h**
- SRAM_OK : **stm32l4r9i_eval_sram.h**
- SRAM_TURN_AROUND_TIME : **stm32l4r9i_eval_sram.c**
- SRAM_WRITEBURST : **stm32l4r9i_eval_sram.h**
- SWIRE_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- t -

- TS_DSI_I2C_ADDRESS : **stm32l4r9i_eval.h**
- TS_DSI_INT_EXTI_IRQn : **stm32l4r9i_eval.h**
- TS_DSI_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- TS_DSI_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- TS_DSI_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- TS_DSI_INT_PIN : **stm32l4r9i_eval.h**
- TS_IRQ_PENDING : **stm32l4r9i_eval_ts.h**
- TS_NO_IRQ_PENDING : **stm32l4r9i_eval_ts.h**
- TS_ORIENTATION_LANDSCAPE : **stm32l4r9i_eval_ts.h**
- TS_ORIENTATION_PORTRAIT : **stm32l4r9i_eval_ts.h**
- TS_RGB_I2C_ADDRESS : **stm32l4r9i_eval.h**
- TS_RGB_INT_EXTI_IRQn : **stm32l4r9i_eval.h**
- TS_RGB_INT_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- TS_RGB_INT_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- TS_RGB_INT_GPIO_PORT : **stm32l4r9i_eval.h**
- TS_RGB_INT_PIN : **stm32l4r9i_eval.h**
- TS_SWAP_NONE : **stm32l4r9i_eval_ts.h**
- TS_SWAP_X : **stm32l4r9i_eval_ts.h**
- TS_SWAP_XY : **stm32l4r9i_eval_ts.h**
- TS_SWAP_Y : **stm32l4r9i_eval_ts.h**

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by doxygen 1.7.6.1

STM32L4R9I_EVAL BSP User Manual

- u -

- UP_JOY_EXTI_IRQn : **stm32l4r9i_eval.h**
- UP_JOY_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

- w -

- WAKEUP_BUTTON_EXTI_IRQn : **stm32l4r9i_eval.h**
- WAKEUP_BUTTON_GPIO_CLK_DISABLE : **stm32l4r9i_eval.h**
- WAKEUP_BUTTON_GPIO_CLK_ENABLE : **stm32l4r9i_eval.h**
- WAKEUP_BUTTON_GPIO_PORT : **stm32l4r9i_eval.h**
- WAKEUP_BUTTON_PIN : **stm32l4r9i_eval.h**

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval.c File Reference

This file provides a set of firmware functions to manage Leds, push-button and COM ports for STM32L4R9I_EVAL. More...

```
#include "stm32l4r9i_eval.h" #include "stm32l4r9i_eval_io.h"
```

Go to the source code of this file.

Defines

#define	**__STM32L4R9I_EVAL_BSP_VERSION_MAIN** (0x01) STM32L4R9I EVAL BSP Driver version number.
#define	**__STM32L4R9I_EVAL_BSP_VERSION_SUB1** (0x00)
#define	**__STM32L4R9I_EVAL_BSP_VERSION_SUB2** (0x00)
#define	**__STM32L4R9I_EVAL_BSP_VERSION_RC** (0x00)
#define	**__STM32L4R9I_EVAL_BSP_VERSION**
#define	**BSP_I2C_NO_USER** 0x00000000U BSP I2C users.
#define	**BSP_I2C_AUDIO_USER** 0x00000001U
#define	**BSP_I2C_TS_USER** 0x00000002U
#define	**BSP_I2C_MFX_USER** 0x00000004U
#define	**BSP_I2C_EEPROM_USER** 0x00000008U
#define	**BSP_I2C_ALL_USERS** 0x0000000FU

Functions

static void	**I2Cx_Init** (uint32_t user) Eval I2Cx Bus initialization.
static void	**I2Cx_MspInit** (I2C_HandleTypeDef *hi2c) Eval I2Cx MSP Initialization.
static void	**I2Cx_DeInit** (uint32_t user) Eval I2Cx Bus deinitialization.
static void	**I2Cx_MspDeInit** (I2C_HandleTypeDef *hi2c) Eval I2Cx MSP Deinitialization.
static HAL_StatusTypeDef	**I2Cx_WriteBuffer** (uint16_t Addr, uint16_t Reg, uint16_t RegSize, uint8_t *pBuffer, uint16_t Length) Write a value in a register of the device through BUS.
static HAL_StatusTypeDef	**I2Cx_ReadBuffer** (uint16_t Addr, uint16_t Reg, uint16_t RegSize, uint8_t *pBuffer, uint16_t Length) Read multiple data on the BUS.
static HAL_StatusTypeDef	**I2Cx_IsDeviceReady** (uint16_t DevAddress, uint32_t Trials) Checks if target device is ready for communication.
static void	**I2Cx_Error** (void) Eval I2Cx error treatment function.
static void	**I2Cx_WriteData** (uint16_t Addr, uint16_t Reg, uint16_t RegSize, uint8_t Value) Write a value in a register of the device through BUS.
static void	**I2Cx_WriteMultiple** (uint16_t Addr, uint16_t Reg, uint16_t MemAddress, uint8_t *Buffer, uint16_t Length) Write multiple data value in a register of

	the device through BUS.
static uint8_t	**I2Cx_ReadData** (uint16_t Addr, uint16_t Reg, uint16_t RegSize) Read a register of the device through BUS.
static uint8_t	**I2Cx_ReadMultiple** (uint16_t Addr, uint16_t Reg, uint16_t MemAddress, uint8_t *Buffer, uint16_t Length) Read multiple data from a register of the device through BUS.
void	**MFX_IO_Init** (void) Initialize MFX_IO low level.
void	**MFX_IO_DeInit** (void) DeInitializes MFX_IO low level.
void	**MFX_IO_Wakeup** (void) MFX_IO wakeup.
void	**MFX_IO_EnableWakeupPin** (void) MFX_IO enable wakeup pin.
void	**MFX_IO_ITConfig** (void) Configure MFX_IO low level Interrupt.
void	**MFX_IO_Delay** (uint32_t Delay) MFX_IO delay.
void	**MFX_IO_Write** (uint16_t Addr, uint8_t Reg, uint8_t Value) MFX_IO writes single data.
uint8_t	**MFX_IO_Read** (uint16_t Addr, uint8_t Reg) MFX_IO reads single data.
uint16_t	**MFX_IO_ReadMultiple** (uint16_t Addr, uint8_t Reg, uint8_t *Buffer, uint16_t Length) MFX_IO reads multiple data.
void	**TS_IO_Init** (void) Initializes Touchscreen low level.

void	**TS_IO_Write** (uint8_t Addr, uint8_t Reg, uint8_t Value) Writes a single data.
uint8_t	**TS_IO_Read** (uint8_t Addr, uint8_t Reg) Reads a single data.
uint16_t	**TS_IO_ReadMultiple** (uint8_t Addr, uint8_t Reg, uint8_t *Buffer, uint16_t Length) Reads multiple data with I2C communication channel from TouchScreen.
void	**TS_IO_Delay** (uint32_t Delay) TS delay.
void	**EEPROM_IO_Init** (void) Initialize peripherals used by the I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_WriteData** (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize) Write data to I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_ReadData** (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize) Read data from I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_IsDeviceReady** (uint16_t DevAddress, uint32_t Trials) Checks if target device is ready for communication.
void	**AUDIO_IO_Init** (void) Initialize Audio low level.
void	**AUDIO_IO_DeInit** (void) Deinitialize Audio low level.
void	**AUDIO_IO_Write** (uint8_t Addr, uint16_t Reg, uint16_t Value)

	Writes a single data.
uint16_t	**AUDIO_IO_Read** (uint8_t Addr, uint16_t Reg) Reads a single data.
void	**AUDIO_IO_Delay** (uint32_t Delay) AUDIO Codec delay.
__weak void	**BSP_ErrorHandler** (void) Error Handler.
uint32_t	**BSP_GetVersion** (void) This method returns the STM32L4R9I EVAL BSP Driver revision.
void	**BSP_LED_Init** (**Led_TypeDef** Led) Initialize LED GPIO.
void	**BSP_LED_DeInit** (**Led_TypeDef** Led) DeInitialize LED GPIO.
void	**BSP_LED_On** (**Led_TypeDef** Led) Turn selected LED On.
void	**BSP_LED_Off** (**Led_TypeDef** Led) Turn selected LED Off.
void	**BSP_LED_Toggle** (**Led_TypeDef** Led) Toggle the selected LED.
void	**BSP_PB_Init** (**Button_TypeDef** Button, **ButtonMode_TypeDef** ButtonMode) Initialize push button GPIO and EXTI Line.
void	**BSP_PB_DeInit** (**Button_TypeDef** Button) DeInitialize push button.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Return the selected button state.
uint8_t	**BSP_JOY_Init** (**JOYMode_TypeDef** JoyMode) Initialize all buttons of the joystick in

	GPIO or EXTI modes.
void	**BSP_JOY_DeInit** (void) DeInitialize all buttons of the joystick.
JOYState_TypeDef	**BSP_JOY_GetState** (void) Return the current joystick status.
void	**BSP_COM_Init** (**COM_TypeDef** COM, UART_HandleTypeDef *huart) Initialize COM port.
void	**BSP_COM_DeInit** (**COM_TypeDef** COM, UART_HandleTypeDef *huart) DeInitialize COM port.
uint32_t	**BSP_POTENTIOMETER_Init** (void) Init the potentioneter RV2 measurement thanks to ADC.
uint32_t	**BSP_POTENTIOMETER_DeInit** (void) DeInitialize the potentioneter RV2 measurement thanks to ADC.
uint32_t	**BSP_POTENTIOMETER_GetLevel** (void) Get the potentioneter RV2 measurement thanks to ADC.

Variables

GPIO_TypeDef *	**LED_PORT** [**LEDn**] LED variables.
const uint16_t	**LED_PIN** [**LEDn**]
GPIO_TypeDef *	**BUTTON_PORT** [**BUTTONn**] BUTTON variables.
const uint16_t	**BUTTON_PIN** [**BUTTONn**]
const uint8_t	**BUTTON_IRQn** [**BUTTONn**]
const uint16_t	**JOY_PIN** [**JOYn**] JOYSTICK variables.
const uint8_t	**JOY_IRQn** [**JOYn**]
USART_TypeDef *	**COM_USART** [**COMn**] = {**EVAL_COM1**} COM variables.
GPIO_TypeDef *	**COM_TX_PORT** [**COMn**] = {**EVAL_COM1_TX_GPIO_PORT**}
GPIO_TypeDef *	**COM_RX_PORT** [**COMn**] = {**EVAL_COM1_RX_GPIO_PORT**}
const uint16_t	**COM_TX_PIN** [**COMn**] = {**EVAL_COM1_TX_PIN**}
const uint16_t	**COM_RX_PIN** [**COMn**] = {**EVAL_COM1_RX_PIN**}
const uint16_t	**COM_TX_AF** [**COMn**] = {**EVAL_COM1_TX_AF**}
const uint16_t	**COM_RX_AF** [**COMn**] = {**EVAL_COM1_RX_AF**}
uint32_t	**hi2c_evalTimeout** = **EVAL_I2Cx_TIMEOUT_MAX** BUS variables.
I2C_HandleTypeDef	**hi2c_eval**
static uint32_t	**v_bspI2cUsers** = **BSP_I2C_NO_USER**
static ADC_HandleTypeDef	**hEvalADC** = {0}

Detailed Description

This file provides a set of firmware functions to manage Leds, push-button and COM ports for STM32L4R9I_EVAL.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval.h File Reference

This file contains definitions for STM32L4R9I_EVAL's LEDs, push-buttons and COM ports hardware resources. More...

```
#include "stm32l4xx_hal.h" #include "stm32l4r9i_eval_io.h"
```

Go to the source code of this file.

Defines

#define	**LEDn** 4
#define	**LED1_PIN** GPIO_PIN_15
#define	**LED1_GPIO_PORT** GPIOB
#define	**LED1_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOB_CLK_E
#define	**LED1_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOB_CLK_D
#define	**LED2_PIN** GPIO_PIN_13
#define	**LED2_GPIO_PORT** GPIOH
#define	**LED2_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOH_CLK_E
#define	**LED2_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOH_CLK_D
#define	**LED3_PIN** GPIO_PIN_14
#define	**LED3_GPIO_PORT** GPIOH
#define	**LED3_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOH_CLK_E
#define	**LED3_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOH_CLK_D
#define	**LED4_PIN** GPIO_PIN_15
#define	**LED4_GPIO_PORT** GPIOH
#define	**LED4_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOH_CLK_E
#define	**LED4_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOH_CLK_D
#define	**LEDx_GPIO_CLK_ENABLE**(__LED__)
#define	**LEDx_GPIO_CLK_DISABLE**(__LED__)
#define	**JOYn** 5
#define	**BUTTONn** 1 + **JOYn**
#define	**WAKEUP_BUTTON_PIN** GPIO_PIN_13 Wakeup/Tamper push-button.
#define	**WAKEUP_BUTTON_GPIO_PORT** GPIOC
#define	**WAKEUP_BUTTON_GPIO_CLK_ENABLE**() __HAL_RCC_G
#define	**WAKEUP_BUTTON_GPIO_CLK_DISABLE**() __HAL_RCC_(
#define	**WAKEUP_BUTTON_EXTI_IRQn** EXTI15_10_IRQn
#define	**BUTTONx_GPIO_CLK_ENABLE**(__BUTTON__) do { if((__B **BUTTON_WAKEUP**) { **WAKEUP_BUTTON_GPIO_CLK_ENA**
#define	**BUTTONx_GPIO_CLK_DISABLE**(__BUTTON__) do { if((__E **BUTTON_WAKEUP**) { **WAKEUP_BUTTON_GPIO_CLK_DISA**

#define **SEL_JOY_PIN** **IO_PIN_0**
Joystick Sel push-button.

#define **SEL_JOY_EXTI_IRQn** EXTI0_IRQn

#define **DOWN_JOY_PIN** **IO_PIN_1**
Joystick Down push-button.

#define **DOWN_JOY_EXTI_IRQn** EXTI0_IRQn

#define **LEFT_JOY_PIN** **IO_PIN_2**
Joystick Left push-button.

#define **LEFT_JOY_EXTI_IRQn** EXTI0_IRQn

#define **RIGHT_JOY_PIN** **IO_PIN_3**
Joystick Right push-button.

#define **RIGHT_JOY_EXTI_IRQn** EXTI0_IRQn

#define **UP_JOY_PIN** **IO_PIN_4**
Joystick Up push-button.

#define **UP_JOY_EXTI_IRQn** EXTI0_IRQn

#define **JOY_ALL_PINS** (**RIGHT_JOY_PIN** | **LEFT_JOY_PIN** | **UP_J**
DOWN_JOY_PIN | **SEL_JOY_PIN**)

#define **JOY_NONE_PIN** **JOY_ALL_PINS**

#define **COMn** 1

#define **EVAL_COM1** LPUART1
Definition for COM port1, connected to USART1.

#define **EVAL_COM1_CLK_ENABLE**() __HAL_RCC_LPUART1_CLK

#define **EVAL_COM1_CLK_DISABLE**() __HAL_RCC_LPUART1_CL

#define **EVAL_COM1_TX_PIN** GPIO_PIN_7

#define **EVAL_COM1_TX_GPIO_PORT** GPIOG

#define **EVAL_COM1_TX_GPIO_CLK_ENABLE**() __HAL_RCC_GPI

#define **EVAL_COM1_TX_GPIO_CLK_DISABLE**() __HAL_RCC_GP

#define **EVAL_COM1_TX_AF** GPIO_AF8_LPUART1

#define **EVAL_COM1_RX_PIN** GPIO_PIN_8

#define **EVAL_COM1_RX_GPIO_PORT** GPIOG

#define **EVAL_COM1_RX_GPIO_CLK_ENABLE**() __HAL_RCC_GPI

#define **EVAL_COM1_RX_GPIO_CLK_DISABLE**() __HAL_RCC_GP

#define **EVAL_COM1_RX_AF** GPIO_AF8_LPUART1

#define **EVAL_COM1_IRQn** LPUART1_IRQn

#define **COMx_CLK_ENABLE**(__INDEX__) do { if((__INDEX__) == C
EVAL_COM1_CLK_ENABLE(); } } while(0)

#define **COMx_CLK_DISABLE**(__INDEX__) do { if((__INDEX__) ==
EVAL_COM1_CLK_DISABLE(); } } while(0)

#define **COMx_TX_GPIO_CLK_ENABLE**(__INDEX__) do { if((__IND
EVAL_COM1_TX_GPIO_CLK_ENABLE(); } } while(0)

#define **COMx_TX_GPIO_CLK_DISABLE**(__INDEX__) do { if((__IND
EVAL_COM1_TX_GPIO_CLK_DISABLE(); } } while(0)

#define **COMx_RX_GPIO_CLK_ENABLE**(__INDEX__) do { if((__IND
EVAL_COM1_RX_GPIO_CLK_ENABLE(); } } while(0)

#define **COMx_RX_GPIO_CLK_DISABLE**(__INDEX__) do { if((__IND
EVAL_COM1_RX_GPIO_CLK_DISABLE(); } } while(0)

#define **BSP_POTENTIOMETER_OK** 0x00
Potentiometer Status values.

#define **BSP_POTENTIOMETER_ERROR** 0xFF

#define **ADCx** ADC1
Definition for Potentiometer, connected to ADC1.

#define **ADCx_CLK_ENABLE**() __HAL_RCC_ADC_CLK_ENABLE()

#define **ADCx_CHANNEL_GPIO_CLK_ENABLE**() __HAL_RCC_GP

#define **ADCx_FORCE_RESET**() __HAL_RCC_ADC_FORCE_RESE

#define **ADCx_RELEASE_RESET**() __HAL_RCC_ADC_RELEASE_

#define **ADCx_CHANNEL_PIN** GPIO_PIN_0

#define **ADCx_CHANNEL_GPIO_PORT** GPIOA

#define **ADCx_CHANNEL** ADC_CHANNEL_5

#define **SAMPLINGTIME** ADC_SAMPLETIME_3CYCLES

#define **ADCx_POLL_TIMEOUT** 10

#define **OPAMPx_INP_PIN** GPIO_PIN_0
Definition for Potentiometer connected to OPAMP.

#define **OPAMPx_INP_GPIO_PORT** GPIOA

#define **OPAMPx_INM_PIN** GPIO_PIN_1

#define **OPAMPx_INM_GPIO_PORT** GPIOA

#define **OPAMPx_VOUT_PIN** GPIO_PIN_3

#define **OPAMPx_VOUT_GPIO_PORT** GPIOA

#define **OPAMPx** OPAMP

#define **OPAMPx_CLK_ENABLE**() __HAL_RCC_OPAMP_CLK_ENA

#define **OPAMPx_CLK_DISABLE**() __HAL_RCC_OPAMP_CLK_DIS

#define **OPAMPx_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOA_CLK

#define **OPAMPx_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOA_CLI

#define **OPAMPx_FORCE_RESET**() __HAL_RCC_OPAMP_FORCE_

#define **OPAMPx_RELEASE_RESET**() __HAL_RCC_OPAMP_RELE

#define **DSI_RST_PIN** **IO_PIN_9**

Eval Pins definition connected to MFX.

#define **EXT_RST_PIN** **IO_PIN_8**

#define **LCD_DISP_PIN** **IO_PIN_11**

#define **LCD_RST_PIN** **IO_PIN_12**

#define **OTG_FS_POWER_SWITCH_PIN** **IO_PIN_6**

#define **OTG_FS_OVER_CURRENT_PIN** **IO_PIN_7**

#define **SD_DETECT_PIN** **IO_PIN_5**

#define **SWIRE_PIN** **IO_PIN_13**

#define **EVAL_I2Cx_SCL_PIN** GPIO_PIN_4

#define **EVAL_I2Cx_SCL_GPIO_PORT** GPIOH

#define **EVAL_I2Cx_SDA_PIN** GPIO_PIN_5

#define **EVAL_I2Cx_SDA_GPIO_PORT** GPIOH

#define **EVAL_I2Cx_SCL_SDA_AF** GPIO_AF4_I2C2

#define **EVAL_I2Cx** I2C2

#define **EVAL_I2Cx_CLK_ENABLE**() __HAL_RCC_I2C2_CLK_ENAI

#define **EVAL_I2Cx_CLK_DISABLE**() __HAL_RCC_I2C2_CLK_DISA

#define **EVAL_I2Cx_SDA_GPIO_CLK_ENABLE**() __HAL_RCC_GPI

#define **EVAL_I2Cx_SDA_GPIO_CLK_DISABLE**() __HAL_RCC_GP

#define **EVAL_I2Cx_SCL_GPIO_CLK_ENABLE**() __HAL_RCC_GPI

#define **EVAL_I2Cx_SCL_GPIO_CLK_DISABLE**() __HAL_RCC_GP

#define **EVAL_I2Cx_FORCE_RESET**() __HAL_RCC_I2C2_FORCE_

#define **EVAL_I2Cx_RELEASE_RESET**() __HAL_RCC_I2C2_RELEA

#define **EVAL_I2Cx_EV_IRQn** I2C2_EV_IRQn

#define **EVAL_I2Cx_EV_IRQHandler** I2C2_EV_IRQHandler

#define **EVAL_I2Cx_ER_IRQn** I2C2_ER_IRQn

#define **EVAL_I2Cx_ER_IRQHandler** I2C2_ER_IRQHandler

#define **EVAL_RCC_PERIPHCLK_I2C** RCC_PERIPHCLK_I2C2

#define **EVAL_RCC_CLKSOURCE_I2C** RCC_I2C2CLKSOURCE_SY

#define **EVAL_TS_I2Cx** I2C2

#define **EVAL_TS_I2Cx_CLK_ENABLE**() __HAL_RCC_I2C2_CLK_E

#define **EVAL_TS_DMAx_CLK_ENABLE**() __HAL_RCC_DMA1_CLI

#define **EVAL_TS_I2Cx_SDA_GPIO_CLK_ENABLE**() __HAL_RCC_

#define **EVAL_TS_I2Cx_SDA_GPIO_CLK_DISABLE**() __HAL_RCC_

#define **EVAL_TS_I2Cx_SCL_GPIO_CLK_ENABLE**() __HAL_RCC_

#define **EVAL_TS_I2Cx_SCL_GPIO_CLK_DISABLE**() __HAL_RCC_

#define **EVAL_TS_I2Cx_FORCE_RESET**() __HAL_RCC_I2C2_FOR

#define **EVAL_TS_I2Cx_RELEASE_RESET**() __HAL_RCC_I2C2_RE

#define **EVAL_TS_I2Cx_SCL_PIN** GPIO_PIN_4

#define **EVAL_TS_I2Cx_SCL_GPIO_PORT** GPIOH

#define **EVAL_TS_I2Cx_SDA_PIN** GPIO_PIN_5

#define **EVAL_TS_I2Cx_SDA_GPIO_PORT** GPIOH

#define **EVAL_TS_I2Cx_SCL_SDA_AF** GPIO_AF4_I2C2

#define **EVAL_TS_I2Cx_EV_IRQn** I2C2_EV_IRQn

#define **EVAL_TS_I2Cx_ER_IRQn** I2C2_ER_IRQn

#define **EVAL_TS_RCC_PERIPHCLK_I2C** RCC_PERIPHCLK_I2C2

#define **EVAL_TS_RCC_CLKSOURCE_I2C** RCC_I2C2CLKSOURCE

#define **EEPROM_I2C_ADDRESS** 0xA0

#define **IO_I2C_ADDRESS** 0x84

#define **IDD_I2C_ADDRESS** **IO_I2C_ADDRESS**

#define **AUDIO_I2C_ADDRESS** 0x34

#define **LCD_RGB_I2C_ADDRESS** 0x70

#define **TS_RGB_I2C_ADDRESS** 0x70

#define **TS_DSI_I2C_ADDRESS** 0x70

#define **EVAL_I2Cx_TIMEOUT_MAX** 3000

#define **AUDIO_INT_GPIO_PORT** GPIOI /* GPIOI */

#define **AUDIO_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOI_CL

#define **AUDIO_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOI_C

#define **AUDIO_INT_PIN** GPIO_PIN_6 /* PI.06 */

#define **MFX_INT_GPIO_PORT** GPIOA /* GPIOA */

MFX interface pins.

#define **MFX_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOA_CLK

#define **MFX_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOA_CL

#define **MFX_INT_PIN** GPIO_PIN_0 /* PA.00 */

#define **MFX_INT_EXTI_IRQn** EXTI0_IRQn

#define **MFX_WAKEUP_GPIO_PORT** GPIOG /* GPIOG */

#define **MFX_WAKEUP_GPIO_CLK_ENABLE**() __HAL_RCC_GPIO

#define **MFX_WAKEUP_GPIO_CLK_DISABLE**() __HAL_RCC_GPIC

#define **MFX_WAKEUP_PIN** GPIO_PIN_9 /* PG.09 */

#define **IDD_INT_GPIO_PORT** **MFX_INT_GPIO_PORT**

#define **IDD_INT_GPIO_CLK_ENABLE**() **MFX_INT_GPIO_CLK_EN**

#define **IDD_INT_GPIO_CLK_DISABLE**() **MFX_INT_GPIO_CLK_DI**

#define **IDD_INT_PIN** **MFX_INT_PIN**

#define **IDD_INT_EXTI_IRQn** **MFX_INT_EXTI_IRQn**

#define **IDD_WAKEUP_GPIO_PORT** **MFX_WAKEUP_GPIO_PORT**

#define **IDD_WAKEUP_GPIO_CLK_ENABLE**() **MFX_WAKEUP_GPI**

#define **IDD_WAKEUP_GPIO_CLK_DISABLE**() **MFX_WAKEUP_GP**

#define **IDD_WAKEUP_PIN** **MFX_WAKEUP_PIN**

#define **IDD_AMP_CONTROL_PIN** **AGPIO_PIN_1**

Idd current measurement interface pins on MFX.

#define **LCD_DSI_TE_PIN** GPIO_PIN_6

MB1314 DSI.

#define **LCD_DSI_TE_CTRL_GPIO_PORT** GPIOG

#define **LCD_DSI_TE_CTRL_GPIO_CLK_ENABLE**() __HAL_RCC_(

#define **LCD_DSI_TE_CTRL_GPIO_CLK_DISABLE**() __HAL_RCC_

#define **LCD_DSI_BL_CTRL_PIN** GPIO_PIN_14

#define **LCD_DSI_BL_CTRL_GPIO_PORT** GPIOB

#define **LCD_DSI_BL_CTRL_GPIO_CLK_ENABLE**() __HAL_RCC_(

#define **LCD_DSI_BL_CTRL_GPIO_CLK_DISABLE**() __HAL_RCC_

#define **TS_DSI_INT_PIN** GPIO_PIN_2

#define **TS_DSI_INT_GPIO_PORT** GPIOC

#define **TS_DSI_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOC_(

#define **TS_DSI_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOC_

#define **TS_DSI_INT_EXTI_IRQn** EXTI2_IRQn

#define **LCD_RGB_DISP_PIN** **IO_PIN_11** /* MFX_GPIO11 */

MB1315 RGB.

```
#define LCD_RGB_RST_PIN   IO_PIN_12 /* MFX_GPIO12 */
#define LCD_RGB_BL_CTRL_PIN   GPIO_PIN_5
#define LCD_RGB_BL_CTRL_GPIO_PORT   GPIOA
#define LCD_RGB_BL_CTRL_GPIO_CLK_ENABLE()   __HAL_RCC_
#define LCD_RGB_BL_CTRL_GPIO_CLK_DISABLE()   __HAL_RCC
#define TS_RGB_INT_PIN   GPIO_PIN_2
#define TS_RGB_INT_GPIO_PORT   GPIOC
#define TS_RGB_INT_GPIO_CLK_ENABLE()   __HAL_RCC_GPIOC_
#define TS_RGB_INT_GPIO_CLK_DISABLE()   __HAL_RCC_GPIOC
#define TS_RGB_INT_EXTI_IRQn   EXTI2_IRQn
```

Enumerations

enum	**Led_TypeDef** {
	LED1 = 0, **LED_GREEN** = LED1, **LED2** = 1, **LED_ORANGE** = LED2,
	LED3 = 2, **LED_RED** = LED3, **LED4** = 3, **LED_BLUE** = LED4
	}
	LED Types Definition. More...

enum	**Button_TypeDef** {
	BUTTON_WAKEUP = 0, **BUTTON_SEL** = 1, **BUTTON_LEFT** = 2, **BUTTON_RIGHT** = 3,
	BUTTON_DOWN = 4, **BUTTON_UP** = 5
	}
	BUTTON Types Definition. More...

enum	**ButtonMode_TypeDef** { **BUTTON_MODE_GPIO** = 0, **BUTTON_MODE_EXTI** = 1 }

enum	**JOYState_TypeDef** {
	JOY_SEL = 0, **JOY_LEFT** = 1, **JOY_RIGHT** = 2, **JOY_DOWN** = 3,
	JOY_UP = 4, **JOY_NONE** = 5
	}
	JOYSTICK Types Definition. More...

enum	**JOYMode_TypeDef** { **JOY_MODE_GPIO** = 0, **JOY_MODE_EXTI** = 1 }
enum	**COM_TypeDef** { **COM1** = 0 } COM Types Definition. More...

Functions

uint32_t	**BSP_GetVersion** (void) This method returns the STM32L4R9I EVAL BSP Driver revision.
void	**BSP_LED_Init** (**Led_TypeDef** Led) Initialize LED GPIO.
void	**BSP_LED_DeInit** (**Led_TypeDef** Led) DeInitialize LED GPIO.
void	**BSP_LED_On** (**Led_TypeDef** Led) Turn selected LED On.
void	**BSP_LED_Off** (**Led_TypeDef** Led) Turn selected LED Off.
void	**BSP_LED_Toggle** (**Led_TypeDef** Led) Toggle the selected LED.
void	**BSP_PB_Init** (**Button_TypeDef** Button, **ButtonMode_TypeDef** ButtonMode) Initialize push button GPIO and EXTI Line.
void	**BSP_PB_DeInit** (**Button_TypeDef** Button) DeInitialize push button.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Return the selected button state.
void	**BSP_COM_Init** (**COM_TypeDef** COM, UART_HandleTypeDef *huart) Initialize COM port.
void	**BSP_COM_DeInit** (**COM_TypeDef** COM, UART_HandleTypeDef *huart) DeInitialize COM port.
uint32_t	**BSP_POTENTIOMETER_Init** (void) Init the potentioneter RV2 measurement thanks to ADC.
uint32_t	**BSP_POTENTIOMETER_DeInit** (void) DeInitialize the potentioneter RV2 measurement

	thanks to ADC.
uint32_t	**BSP_POTENTIOMETER_GetLevel** (void) Get the potentioneter RV2 measurement thanks to ADC.
uint8_t	**BSP_JOY_Init** (**JOYMode_TypeDef** JoyMode) Initialize all buttons of the joystick in GPIO or EXTI modes.
void	**BSP_JOY_DeInit** (void) DeInitialize all buttons of the joystick.
JOYState_TypeDef	**BSP_JOY_GetState** (void) Return the current joystick status.
__weak void	**BSP_ErrorHandler** (void) Error Handler.

Detailed Description

This file contains definitions for STM32L4R9I_EVAL's LEDs, push-buttons and COM ports hardware resources.

Author:
MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_audio.c File Reference

This file provides the Audio driver for the STM32L4R9I-EVAL evaluation board. More...

```
#include <string.h> #include "stm32l4r9i_eval_audio.h"
```

Go to the source code of this file.

Data Structures

struct	**AUDIO_OUT_TypeDef**
struct	**AUDIO_IN_TypeDef**

Defines

#define	**SAIClockDivider**(__FREQUENCY__)
#define	**DFSDMOverSampling**(__FREQUENCY__)
#define	**DFSDMClockDivider**(__FREQUENCY__)
#define	**DFSDMFilterOrder**(__FREQUENCY__)
#define	**DFSDMRightBitShift**(__FREQUENCY__)
#define	**SaturaLH**(N, L, H) (((N)<(L))?(L):(((N)>(H))?(H):(N)))

Functions

static void	**AUDIO_CODEC_Reset** (void) Resets the audio codec.
static uint8_t	**AUDIO_SAIx_Init** (uint32_t AudioFreq, uint16_t OutputDevice) Initializes the Audio Codec audio interface (SAI).
static uint8_t	**AUDIO_SAIx_DeInit** (void) De-initializes the Audio Codec audio interface (SAI).
static uint8_t	**AUDIO_DFSDMx_Init** (uint32_t AudioFreq) Initializes the Digital Filter for Sigma-Delta Modulators interface (DFSDM).
static uint8_t	**AUDIO_DFSDMx_DeInit** (void) De-initializes the Digital Filter for Sigma-Delta Modulators interface (DFSDM).
static uint8_t	**AUDIO_SAIPLLConfig** (uint32_t Frequency) Configures the SAI PLL clock according to the required audio frequency.
uint8_t	**BSP_AUDIO_OUT_Init** (uint16_t OutputDevice, uint8_t Volume, uint32_t AudioFreq) Configures the audio peripherals.
uint8_t	**BSP_AUDIO_OUT_DeInit** (void) De-Initializes audio codec related peripherals.
uint8_t	**BSP_AUDIO_OUT_Play** (uint16_t *pData, uint32_t Size) Starts playing audio stream from a data buffer for a determined size.
uint8_t	**BSP_AUDIO_OUT_ChangeBuffer** (uint16_t *pData, uint16_t Size) Sends n-Bytes on the SAI interface.
uint8_t	**BSP_AUDIO_OUT_Pause** (void) This function Pauses the audio file stream.
uint8_t	**BSP_AUDIO_OUT_Resume** (void)

	This function Resumes the audio file stream.
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).
uint8_t	**BSP_AUDIO_OUT_SetFrequency** (uint32_t AudioFreq) Update the audio frequency.
void	**BSP_AUDIO_OUT_ChangeAudioConfig** (uint32_t AudioOutOption) Changes the Audio Out Configuration.
void	**BSP_AUDIO_OUT_RegisterCallbacks** (**Audio_CallbackTypeDef** ErrorCallback, **Audio_CallbackTypeDef** HalfTransferCallback, **Audio_CallbackTypeDef** TransferCompleteCallback) register user callback functions
void	**HAL_SAI_TxCpltCallback** (SAI_HandleTypeDef *hsai) Tx Transfer completed callbacks.
void	**HAL_SAI_TxHalfCpltCallback** (SAI_HandleTypeDef *hsai) Tx Half Transfer completed callbacks.
void	**HAL_SAI_ErrorCallback** (SAI_HandleTypeDef *hsai) SAI error callbacks.
uint8_t	**BSP_AUDIO_IN_Init** (uint32_t AudioFreq, uint32_t BitRes, uint32_t ChnlNbr) Initializes wave recording.
uint8_t	**BSP_AUDIO_IN_DeInit** (void) De-Initializes microphone related peripherals.
uint8_t	**BSP_AUDIO_IN_Record** (uint16_t *pbuf, uint32_t size)

	Starts audio recording.
uint8_t	**BSP_AUDIO_IN_SetFrequency** (uint32_t AudioFreq) Updates the audio frequency.
void	**HAL_DFSDM_FilterRegConvCpltCallback** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter) Regular conversion complete callback.
void	**HAL_DFSDM_FilterRegConvHalfCpltCallback** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter) Half regular conversion complete callback.
void	**HAL_DFSDM_FilterErrorCallback** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter) Error callback.
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.
void	**BSP_AUDIO_IN_RegisterCallbacks** (**Audio_CallbackTypeDef** ErrorCallback, **Audio_CallbackTypeDef** HalfTransferCallback, **Audio_CallbackTypeDef** TransferCompleteCallback) register user callback functions
void	**HAL_SAI_MspInit** (SAI_HandleTypeDef *hsai) SAI MSP Init.
void	**HAL_SAI_MspDeInit** (SAI_HandleTypeDef *hsai) SAI MSP De-init.
void	**HAL_DFSDM_ChannelMspInit** (DFSDM_Channel_HandleTypeDef *hdfsdm_channel) Initializes the DFSDM channel MSP.
void	**HAL_DFSDM_ChannelMspDeInit** (DFSDM_Channel_HandleTypeDef *hdfsdm_channel) DeInitializes the DFSDM channel MSP.
void	**HAL_DFSDM_FilterMspInit**

	(DFSDM_Filter_HandleTypeDef *hdfsdm_filter) Initializes the DFSDM filter MSP.
void	**HAL_DFSDM_FilterMspDeInit** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter) DeInitializes the DFSDM filter MSP.

Variables

static **AUDIO_OUT_TypeDef**	**hAudioOut** = {0}
static **AUDIO_IN_TypeDef**	**hAudioIn** = {0}
static DMA_HandleTypeDef	**hDmaSai**
static uint32_t	**DmaLeftRecHalfBuffCplt**
static uint32_t	**DmaLeftRecBuffCplt**
static uint32_t	**DmaRightRecHalfBuffCplt**
static uint32_t	**DmaRightRecBuffCplt**
SAI_HandleTypeDef	**BSP_AUDIO_hSai**
DFSDM_Filter_HandleTypeDef	**BSP_AUDIO_hDfsdmLeftFilter**
DFSDM_Filter_HandleTypeDef	**BSP_AUDIO_hDfsdmRightFilter**

Detailed Description

This file provides the Audio driver for the STM32L4R9I-EVAL evaluation board.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_audio.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_audio.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_audio.c** driver. More...

```
#include <stdlib.h> #include "../Components/wm8994/wm8994.h"
#include "stm32l4r9i_eval.h"
```

Go to the source code of this file.

Defines

#define	**BSP_AUDIO_OUT_CIRCULARMODE** ((uint32_t)0x0000000 CIRCULAR MODE */
#define	**BSP_AUDIO_OUT_NORMALMODE** ((uint32_t)0x00000002) MODE */
#define	**BSP_AUDIO_OUT_STEREOMODE** ((uint32_t)0x00000004) /
#define	**BSP_AUDIO_OUT_MONOMODE** ((uint32_t)0x00000008) /*
#define	**BSP_AUDIO_FREQUENCY_96K** SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_48K** SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_44K** SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_32K** SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_22K** SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_16K** SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_11K** SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_8K** SAI_AUDIO_FREQUENCY_
#define	**AUDIO_SAIx** SAI1_Block_A
#define	**AUDIO_SAIx_CLK_ENABLE**() __HAL_RCC_SAI1_CLK_EN
#define	**AUDIO_SAIx_CLK_DISABLE**() __HAL_RCC_SAI1_CLK_DI
#define	**AUDIO_SAIx_MCK_SCK_SD_FS_AF** GPIO_AF13_SAI1
#define	**AUDIO_SAIx_MCK_CLK_ENABLE**() __HAL_RCC_GPIOB_
#define	**AUDIO_SAIx_MCK_CLK_DISABLE**() __HAL_RCC_GPIOB_
#define	**AUDIO_SAIx_MCK_PIN** GPIO_PIN_8
#define	**AUDIO_SAIx_MCK_GPIO_PORT** GPIOB
#define	**AUDIO_SAIx_FS_CLK_ENABLE**() __HAL_RCC_GPIOB_CL
#define	**AUDIO_SAIx_FS_CLK_DISABLE**() __HAL_RCC_GPIOB_C
#define	**AUDIO_SAIx_FS_PIN** GPIO_PIN_9
#define	**AUDIO_SAIx_FS_GPIO_PORT** GPIOB
#define	**AUDIO_SAIx_SCK_CLK_ENABLE**() __HAL_RCC_GPIOA_(
#define	**AUDIO_SAIx_SCK_CLK_DISABLE**() __HAL_RCC_GPIOA_
#define	**AUDIO_SAIx_SCK_PIN** GPIO_PIN_8
#define	**AUDIO_SAIx_SCK_GPIO_PORT** GPIOA
#define	**AUDIO_SAIx_SD_CLK_ENABLE**() __HAL_RCC_GPIOC_CL

```
#define AUDIO_SAIx_SD_CLK_DISABLE()  __HAL_RCC_GPIOC_C
#define AUDIO_SAIx_SD_PIN  GPIO_PIN_1
#define AUDIO_SAIx_SD_GPIO_PORT  GPIOC
#define AUDIO_SAIx_DMAx_CLK_ENABLE()  __HAL_RCC_DMA2_
#define AUDIO_SAIx_DMAx_CLK_DISABLE()  __HAL_RCC_DMA2_
#define AUDIO_SAIx_DMAx_REQUEST  DMA_REQUEST_SAI1_A
#define AUDIO_SAIx_DMAx_CHANNEL  DMA2_Channel2
#define AUDIO_SAIx_DMAx_IRQ  DMA2_Channel2_IRQn
#define AUDIO_SAIx_DMAx_PERIPH_DATA_SIZE  DMA_PDATAAL
#define AUDIO_SAIx_DMAx_MEM_DATA_SIZE  DMA_MDATAALIGN
#define DMA_MAX_SZE  (uint32_t)0xFFFF
#define AUDIO_SAIx_DMAx_IRQHandler  DMA2_Channel2_IRQHar
#define AUDIO_OUT_IRQ_PREPRIO  5 /* Select the preemption prior
                               highest) */
#define AUDIO_SAIx_PLL_DISABLE()  HAL_RCCEx_DisablePLLSA
#define AUDIO_DFSDMx_LEFT_CHANNEL  DFSDM1_Channel4
#define AUDIO_DFSDMx_RIGHT_CHANNEL  DFSDM1_Channel3
#define AUDIO_DFSDMx_LEFT_FILTER  DFSDM1_Filter0
#define AUDIO_DFSDMx_RIGHT_FILTER  DFSDM1_Filter1
#define AUDIO_DFSDMx_CLK_ENABLE()  __HAL_RCC_DFSDM1_
#define AUDIO_DFSDMx_CLK_DISABLE()  __HAL_RCC_DFSDM1_
#define AUDIO_DFSDMx_CKOUT_DATIN_AF  GPIO_AF6_DFSDM1
#define AUDIO_DFSDMx_CKOUT_CLK_ENABLE()  __HAL_RCC_G
#define AUDIO_DFSDMx_CKOUT_CLK_DISABLE()  __HAL_RCC_C
#define AUDIO_DFSDMx_CKOUT_PIN  GPIO_PIN_10
#define AUDIO_DFSDMx_CKOUT_GPIO_PORT  GPIOF
#define AUDIO_DFSDMx_DATIN_CLK_ENABLE()  __HAL_RCC_GP
#define AUDIO_DFSDMx_DATIN_CLK_DISABLE()  __HAL_RCC_GI
#define AUDIO_DFSDMx_DATIN_PIN  GPIO_PIN_0
#define AUDIO_DFSDMx_DATIN_GPIO_PORT  GPIOC
#define AUDIO_DFSDMx_DMAx_CLK_ENABLE()  __HAL_RCC_DM
#define AUDIO_DFSDMx_DMAx_CLK_DISABLE()  __HAL_RCC_DM
#define AUDIO_DFSDMx_DMAx_LEFT_CHANNEL  DMA1_Channel
#define AUDIO_DFSDMx_DMAx_RIGHT_CHANNEL  DMA1_Channe
```

```
#define AUDIO_DFSDMx_DMAx_LEFT_IRQ DMA1_Channel4_IRQr
#define AUDIO_DFSDMx_DMAx_RIGHT_IRQ DMA1_Channel5_IRQ
#define AUDIO_DFSDMx_DMAx_LEFT_REQUEST DMA_REQUEST
#define AUDIO_DFSDMx_DMAx_RIGHT_REQUEST DMA_REQUES
#define AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE DMA_PDAT
#define AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE DMA_MDATAA
#define AUDIO_DFSDM_DMAx_LEFT_IRQHandler DMA1_Channel
#define AUDIO_DFSDM_DMAx_RIGHT_IRQHandler DMA1_Channe
#define AUDIO_IN_IRQ_PREPRIO 6 /* Select the preemption priority
#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 BSP_AUDIO_FREQUENCY_1
#define DEFAULT_AUDIO_IN_BIT_RESOLUTION 16
#define DEFAULT_AUDIO_IN_CHANNEL_NBR 2 /* Mono = 1, Stere
#define DEFAULT_AUDIO_IN_VOLUME 64
#define INTERNAL_BUFF_SIZE 256
#define CODEC_RESET_DELAY 5
#define DMA_MAX(_X_) (((_X_) <= DMA_MAX_SZE)? (_X_):DMA_M
```

Typedefs

typedef void(* **Audio_CallbackTypeDef**)(void)

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_DeInit** (void) De-Initializes audio codec related peripherals.
uint8_t	**BSP_AUDIO_OUT_Play** (uint16_t *pData, uint32_t Size) Starts playing audio stream from a data buffer for a determined size.
uint8_t	**BSP_AUDIO_OUT_ChangeBuffer** (uint16_t *pData, uint16_t Size) Sends n-Bytes on the SAI interface.
uint8_t	**BSP_AUDIO_OUT_Pause** (void) This function Pauses the audio file stream.
uint8_t	**BSP_AUDIO_OUT_Resume** (void) This function Resumes the audio file stream.
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_SetFrequency** (uint32_t AudioFreq) Update the audio frequency.
void	**BSP_AUDIO_OUT_ChangeAudioConfig** (uint32_t AudioOutOption) Changes the Audio Out Configuration.
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_RegisterCallbacks** (**Audio_CallbackTypeDef** ErrorCallback,

	Audio_CallbackTypeDef HalfTransferCallback, **Audio_CallbackTypeDef** TransferCompleteCallback) register user callback functions
uint8_t	**BSP_AUDIO_IN_Init** (uint32_t AudioFreq, uint32_t BitRes, uint32_t ChnlNbr) Initializes wave recording.
uint8_t	**BSP_AUDIO_IN_DeInit** (void) De-Initializes microphone related peripherals.
uint8_t	**BSP_AUDIO_IN_Record** (uint16_t *pbuf, uint32_t size) Starts audio recording.
uint8_t	**BSP_AUDIO_IN_SetFrequency** (uint32_t AudioFreq) Updates the audio frequency.
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.
void	**BSP_AUDIO_IN_RegisterCallbacks** (**Audio_CallbackTypeDef** ErrorCallback, **Audio_CallbackTypeDef** HalfTransferCallback, **Audio_CallbackTypeDef** TransferCompleteCallback) register user callback functions

Variables

SAI_HandleTypeDef	**BSP_AUDIO_hSai**
DFSDM_Filter_HandleTypeDef	**BSP_AUDIO_hDfsdmLeftFilter**
DFSDM_Filter_HandleTypeDef	**BSP_AUDIO_hDfsdmRightFilter**

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_audio.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_audio.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_dsi_lcd.c File Reference

This file includes the driver for DSI Liquid Crystal Display (LCD) module mounted on STM32L4R9I_EVAL evaluation board. More...

```
#include "stm32l4r9i_eval_lcd.h" #include
"stm32l4r9i_eval_dsi_lcd.h"
#include "stm32l4r9i_eval_gfxmmu_lut.h"
#include "stm32l4r9i_eval_io.h"
#include "../../../Utilities/Fonts/fonts.h"
#include "../../../Utilities/Fonts/font24.c"
#include "../../../Utilities/Fonts/font20.c"
#include "../../../Utilities/Fonts/font16.c"
#include "../../../Utilities/Fonts/font12.c"
#include "../../../Utilities/Fonts/font8.c"
```

Go to the source code of this file.

Defines

```
#define  ABS(X)   ((X) > 0 ? (X) : -(X))
#define  POLY_X(Z)   ((int32_t)((Points + (Z))->X))
#define  POLY_Y(Z)   ((int32_t)((Points + (Z))->Y))
```

Functions

uint8_t	**BSP_DSI_LCD_DeInit** (void) De-initialize the DSI LCD.
uint32_t	**BSP_DSI_LCD_GetXSize** (void) Gets the LCD X size.
uint32_t	**BSP_DSI_LCD_GetYSize** (void) Gets the LCD Y size.
uint8_t	**BSP_DSI_LCD_SelectLayer** (uint32_t LayerIndex) Selects the LCD Layer.
uint8_t	**BSP_DSI_LCD_SetLayerVisible** (uint32_t LayerIndex, FunctionalState State) Sets an LCD Layer visible.
uint8_t	**BSP_DSI_LCD_SetTransparency** (uint32_t LayerIndex, uint8_t Transparency) Configures the transparency.
uint8_t	**BSP_DSI_LCD_SetColorKeying** (uint32_t LayerIndex, uint32_t RGBValue) Configures and sets the color keying.
uint8_t	**BSP_DSI_LCD_ResetColorKeying** (uint32_t LayerIndex) Disables the color keying.
void	**BSP_DSI_LCD_SetTextColor** (uint32_t Color) Sets the LCD text color.
uint32_t	**BSP_DSI_LCD_GetTextColor** (void) Gets the LCD text color.
void	**BSP_DSI_LCD_SetBackColor** (uint32_t Color) Sets the LCD background color.
uint32_t	**BSP_DSI_LCD_GetBackColor** (void) Gets the LCD background color.
void	**BSP_DSI_LCD_SetFont** (sFONT *fonts) Sets the LCD text font.
sFONT *	**BSP_DSI_LCD_GetFont** (void)

	Gets the LCD text font.
uint32_t	**BSP_DSI_LCD_ReadPixel** (uint16_t Xpos, uint16_t Ypos) Reads an LCD pixel.
void	**BSP_DSI_LCD_Clear** (uint32_t Color) Clears the whole currently active layer of LTDC.
void	**BSP_DSI_LCD_ClearStringLine** (uint32_t Line) Clears the selected line in currently active layer.
void	**BSP_DSI_LCD_DisplayChar** (uint16_t Xpos, uint16_t Ypos, uint8_t Ascii) Displays one character in currently active layer.
void	**BSP_DSI_LCD_DisplayStringAt** (uint16_t Xpos, uint16_t Ypos, uint8_t *Text, **Text_AlignModeTypdef** Mode) Displays characters in currently active layer.
void	**BSP_DSI_LCD_DisplayStringAtLine** (uint16_t Line, uint8_t *ptr) Displays string on specified line of the LCD.
void	**BSP_DSI_LCD_DrawHLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length) Draws an horizontal line in currently active layer.
void	**BSP_DSI_LCD_DrawVLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length) Draws a vertical line in currently active layer.
void	**BSP_DSI_LCD_DrawLine** (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2) Draws an uni-line (between two points) in currently active layer.
void	**BSP_DSI_LCD_DrawRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height) Draws a rectangle in currently active layer.
void	**BSP_DSI_LCD_DrawCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius) Draws a circle in currently active layer.

void	**BSP_DSI_LCD_DrawPolygon** (**pPoint** Points, uint16_t PointCount) Draws an poly-line (between many points) in currently active layer.
void	**BSP_DSI_LCD_DrawEllipse** (int Xpos, int Ypos, int XRadius, int YRadius) Draws an ellipse on LCD in currently active layer.
void	**BSP_DSI_LCD_DrawBitmap** (uint32_t Xpos, uint32_t Ypos, uint8_t *pbmp) Draws a bitmap picture loaded in the internal Flash (32 bpp) in currently active layer.
void	**BSP_DSI_LCD_FillRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height) Draws a full rectangle in currently active layer.
void	**BSP_DSI_LCD_FillCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius) Draws a full circle in currently active layer.
void	**BSP_DSI_LCD_FillPolygon** (**pPoint** Points, uint16_t PointCount) Draws a full poly-line (between many points) in currently active layer.
void	**BSP_DSI_LCD_FillEllipse** (int Xpos, int Ypos, int XRadius, int YRadius) Draws a full ellipse in currently active layer.
void	**BSP_DSI_LCD_DrawPixel** (uint16_t Xpos, uint16_t Ypos, uint32_t RGB_Code) Draws a pixel on LCD.
void	**BSP_DSI_LCD_DisplayOn** (void) Switch back on the display if was switched off by previous call of **BSP_DSI_LCD_DisplayOff()**.
void	**BSP_DSI_LCD_DisplayOff** (void) Switch Off the display.
void	**BSP_DSI_LCD_Refresh** (void) Refresh the display.

uint8_t	**BSP_DSI_LCD_IsFrameBufferAvailable** (void) Check if frame buffer is available.
void	**BSP_DSI_LCD_SetBrightness** (uint8_t BrightnessValue) Set the brightness value.
__weak void	**BSP_DSI_LCD_DMA2D_IRQHandler** (void) Handles DMA2D interrupt request.
__weak void	**BSP_DSI_LCD_DSI_IRQHandler** (void) Handles DSI interrupt request.
__weak void	**BSP_DSI_LCD_LTDC_IRQHandler** (void) Handles LTDC interrupt request.
__weak void	**BSP_DSI_LCD_LTDC_ER_IRQHandler** (void) This function handles LTDC Error interrupt Handler.
__weak void	**BSP_DSI_LCD_MspDeInit** (void) De-Initializes the BSP LCD Msp.
__weak void	**BSP_DSI_LCD_MspInit** (void) Initialize the BSP LCD Msp.
void	**HAL_DSI_EndOfRefreshCallback** (DSI_HandleTypeDef *hdsi) End of Refresh DSI callback.
static void	**LCD_Reset** (void) LCD Reset Hw reset the LCD DSI activating its XRES signal (active low for some time) and desactivating it later.
static void	**DrawChar** (uint16_t Xpos, uint16_t Ypos, const uint8_t *c) Draws a character on LCD.
static void	**FillTriangle** (uint16_t x1, uint16_t x2, uint16_t x3, uint16_t y1, uint16_t y2, uint16_t y3) Fills a triangle (between 3 points).
static void	**LL_FillBuffer** (uint32_t LayerIndex, void *pDst, uint32_t xSize, uint32_t ySize, uint32_t OffLine, uint32_t ColorIndex) Fills a buffer.

static void **LL_ConvertLineToARGB8888** (void *pSrc, void *pDst, uint32_t xSize, uint32_t ColorMode)

Converts a line to an ARGB8888 pixel format.

Variables

DMA2D_HandleTypeDef	**hdma2d_eval**
static uint32_t	**bsp_lcd_initialized** = 0
static uint32_t	**bsp_lcd_hse_to_disable** = 0
static uint32_t	**ActiveLayer** = **LTDC_ACTIVE_LAYER_BACKGROU**
static **LCD_DrawPropTypeDef**	**DrawProp** [**LTDC_MAX_LAYER_NUMBER**]

Detailed Description

This file includes the driver for DSI Liquid Crystal Display (LCD) module mounted on STM32L4R9I_EVAL evaluation board.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_dsi_lcd.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_dsi_lcd.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_dsi_lcd.c** driver. More...

```
#include "stm32l4r9i_eval.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_DSI_LCD_Init** (void)
uint8_t	**BSP_DSI_LCD_DeInit** (void) De-initialize the DSI LCD.
uint32_t	**BSP_DSI_LCD_GetXSize** (void) Gets the LCD X size.
uint32_t	**BSP_DSI_LCD_GetYSize** (void) Gets the LCD Y size.
uint8_t	**BSP_DSI_LCD_SetTransparency** (uint32_t LayerIndex, uint8_t Transparency) Configures the transparency.
uint8_t	**BSP_DSI_LCD_SetColorKeying** (uint32_t LayerIndex, uint32_t RGBValue) Configures and sets the color keying.
uint8_t	**BSP_DSI_LCD_ResetColorKeying** (uint32_t LayerIndex) Disables the color keying.
uint8_t	**BSP_DSI_LCD_SelectLayer** (uint32_t LayerIndex) Selects the LCD Layer.
uint8_t	**BSP_DSI_LCD_SetLayerVisible** (uint32_t LayerIndex, FunctionalState State) Sets an LCD Layer visible.
void	**BSP_DSI_LCD_SetTextColor** (uint32_t Color) Sets the LCD text color.
uint32_t	**BSP_DSI_LCD_GetTextColor** (void) Gets the LCD text color.
void	**BSP_DSI_LCD_SetBackColor** (uint32_t Color) Sets the LCD background color.
uint32_t	**BSP_DSI_LCD_GetBackColor** (void) Gets the LCD background color.
void	**BSP_DSI_LCD_SetFont** (sFONT *fonts) Sets the LCD text font.

sFONT *	**BSP_DSI_LCD_GetFont** (void) Gets the LCD text font.
uint32_t	**BSP_DSI_LCD_ReadPixel** (uint16_t Xpos, uint16_t Ypos) Reads an LCD pixel.
void	**BSP_DSI_LCD_DrawPixel** (uint16_t Xpos, uint16_t Ypos, uint32_t RGB_Code) Draws a pixel on LCD.
void	**BSP_DSI_LCD_Clear** (uint32_t Color) Clears the whole currently active layer of LTDC.
void	**BSP_DSI_LCD_ClearStringLine** (uint32_t Line) Clears the selected line in currently active layer.
void	**BSP_DSI_LCD_DisplayStringAtLine** (uint16_t Line, uint8_t *ptr) Displays string on specified line of the LCD.
void	**BSP_DSI_LCD_DisplayStringAt** (uint16_t Xpos, uint16_t Ypos, uint8_t *Text, **Text_AlignModeTypdef** Mode) Displays characters in currently active layer.
void	**BSP_DSI_LCD_DisplayChar** (uint16_t Xpos, uint16_t Ypos, uint8_t Ascii) Displays one character in currently active layer.
void	**BSP_DSI_LCD_DrawHLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length) Draws an horizontal line in currently active layer.
void	**BSP_DSI_LCD_DrawVLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length) Draws a vertical line in currently active layer.
void	**BSP_DSI_LCD_DrawLine** (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2) Draws an uni-line (between two points) in currently active layer.
void	**BSP_DSI_LCD_DrawRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)

	Draws a rectangle in currently active layer.
void	**BSP_DSI_LCD_DrawCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius) Draws a circle in currently active layer.
void	**BSP_DSI_LCD_DrawPolygon** (**pPoint** Points, uint16_t PointCount) Draws an poly-line (between many points) in currently active layer.
void	**BSP_DSI_LCD_DrawEllipse** (int Xpos, int Ypos, int XRadius, int YRadius) Draws an ellipse on LCD in currently active layer.
void	**BSP_DSI_LCD_DrawBitmap** (uint32_t Xpos, uint32_t Ypos, uint8_t *pbmp) Draws a bitmap picture loaded in the internal Flash (32 bpp) in currently active layer.
void	**BSP_DSI_LCD_FillRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height) Draws a full rectangle in currently active layer.
void	**BSP_DSI_LCD_FillCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius) Draws a full circle in currently active layer.
void	**BSP_DSI_LCD_FillPolygon** (**pPoint** Points, uint16_t PointCount) Draws a full poly-line (between many points) in currently active layer.
void	**BSP_DSI_LCD_FillEllipse** (int Xpos, int Ypos, int XRadius, int YRadius) Draws a full ellipse in currently active layer.
void	**BSP_DSI_LCD_DisplayOff** (void) Switch Off the display.
void	**BSP_DSI_LCD_DisplayOn** (void) Switch back on the display if was switched off by previous call of **BSP_DSI_LCD_DisplayOff()**.
void	**BSP_DSI_LCD_Refresh** (void)

	Refresh the display.
uint8_t	**BSP_DSI_LCD_IsFrameBufferAvailable** (void) Check if frame buffer is available.
void	**BSP_DSI_LCD_SetBrightness** (uint8_t BrightnessValue) Set the brightness value.
__weak void	**BSP_DSI_LCD_MspInit** (void) Initialize the BSP LCD Msp.
__weak void	**BSP_DSI_LCD_MspDeInit** (void) De-Initializes the BSP LCD Msp.
__weak void	**BSP_DSI_LCD_DMA2D_IRQHandler** (void) Handles DMA2D interrupt request.
__weak void	**BSP_DSI_LCD_DSI_IRQHandler** (void) Handles DSI interrupt request.
__weak void	**BSP_DSI_LCD_LTDC_IRQHandler** (void) Handles LTDC interrupt request.
__weak void	**BSP_DSI_LCD_LTDC_ER_IRQHandler** (void) This function handles LTDC Error interrupt Handler.

Variables

DMA2D_HandleTypeDef	**hdma2d_eval**

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_dsi_lcd.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_dsi_lcd.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_dsi_ts.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_dsi_ts.c** driver. More...

```
#include "stm32l4r9i_eval.h" #include
"../Components/ft3x67/ft3x67.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_DSI_TS_Init** (uint16_t ts_SizeX, uint16_t ts_SizeY)
uint8_t	**BSP_DSI_TS_DeInit** (void)
uint8_t	**BSP_DSI_TS_GetState** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_DSI_TS_GestureConfig** (FunctionalState State)
uint8_t	**BSP_DSI_TS_Get_GestureId** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_DSI_TS_ResetTouchData** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_DSI_TS_ITConfig** (void)
uint8_t	**BSP_DSI_TS_ITGetStatus** (void)
void	**BSP_DSI_TS_ITClear** (void)

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_dsi_ts.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_dsi_ts.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_eeprom.c File Reference

This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory. More...

```
#include "stm32l4r9i_eval_eeprom.h"
```

Go to the source code of this file.

Functions

uint32_t	**BSP_EEPROM_Init** (void) Initializes peripherals used by the I2C EEPROM driver.
uint8_t	**BSP_EEPROM_DeInit** (void) DeInitializes the EEPROM.
uint32_t	**BSP_EEPROM_ReadBuffer** (uint8_t *pBuffer, uint16_t ReadAddr, uint16_t *NumByteToRead) Reads a block of data from the EEPROM.
uint32_t	**BSP_EEPROM_WritePage** (uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite) Writes more than one byte to the EEPROM with a single WRITE cycle.
uint32_t	**BSP_EEPROM_WriteBuffer** (uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite) Writes buffer of data to the I2C EEPROM.
uint32_t	**BSP_EEPROM_WaitEepromStandbyState** (void) Wait for EEPROM Standby state.
__weak void	**BSP_EEPROM_TIMEOUT_UserCallback** (void) Basic management of the timeout situation.

Variables

__IO uint16_t	**EEPROMAddress** = 0
__IO uint16_t	**EEPROMDataRead**
__IO uint8_t	**EEPROMDataWrite**

Detailed Description

This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory.

Author:

MCD Application Team

```
            ==================================
==================================
            Notes:
             - The I2C EEPROM memory (M24LR64)
 is available on separate daughter
               board ANT7-M24LR-A, which is no
t provided with the STM32L4R9I_EVAL
               board.
               To use this driver you have to
connect the ANT7-M24LR-A to CN5
               connector of STM32L4R9I_EVAL bo
ard.
            ==================================
==================================

            It implements a high level communi
cation layer for read and write
            from/to this memory. The needed ST
M32L4xx hardware resources (I2C and
            GPIO) are defined in stm32l4r9i_ev
al.h file, and the initialization is
            performed in EEPROM_IO_Init() func
tion declared in stm32l4r9i_eval.c
            file.
            You can easily tailor this driver
to any other development board,
            by just adapting the defines for h
ardware resources and
```

```
            EEPROM_IO_Init() function.

            @note In this driver, basic read a
nd write functions (BSP_EEPROM_ReadBuffer()
                  and BSP_EEPROM_WritePage())
use DMA mode to perform the data
                  transfer to/from EEPROM memo
ry.

           @note   Regarding BSP_EEPROM_WriteP
age(), it is an optimized function to perform
                  small write (less than 1 pag
e) BUT the number of bytes (combined to write
start address) must not
                  cross the EEPROM page bounda
ry. This function can only writes into
                  the boundaries of an EEPROM
page.
                  This function doesn't check
on boundaries condition (in this driver
                  the function BSP_EEPROM_Writ
eBuffer() which calls BSP_EEPROM_WritePage() i
s
                  responsible of checking on P
age boundaries).


       +-------------------------------------
---------------------------+
       |                Pin assignment for M24L
R64 EEPROM                    |
       +-------------------------------------
-+-----------+-------------+
       |  STM32L4xx I2C Pins
 |   EEPROM  |   Pin       |
       +-------------------------------------
-+-----------+-------------+
```

```
        | .
 |   E0(GND) |    1  (0V)  |
        | .
 |   AC0     |    2        |
        | .
 |   AC1     |    3        |
        | .
 |   VSS     |    4  (0V)  |
        | SDA
 |   SDA     |    5        |
        | SCL
 |   SCL     |    6        |
        | .
 |   E1(GND) |    7  (0V)  |
        | .
 |   VDD     |    8 (3.3V) |
       +-----------------------------------------
-+-----------+-------------+
```

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_eeprom.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_eeprom.h File Reference

This file contains all the functions prototypes for the **stm32l4r9i_eval_eeprom.c** firmware driver. More...

```
#include "stm32l4r9i_eval.h"
```

Go to the source code of this file.

Defines

```
#define EEPROM_PAGESIZE   ((uint8_t)4)
#define EEPROM_MAX_SIZE   ((uint16_t)0x2000) /* 64Kbit */
#define EEPROM_MAX_TRIALS   ((uint32_t)3000)
#define EEPROM_OK   ((uint32_t)0)
#define EEPROM_FAIL   ((uint32_t)1)
#define EEPROM_TIMEOUT   ((uint32_t)2)
```

Functions

uint32_t	**BSP_EEPROM_Init** (void) Initializes peripherals used by the I2C EEPROM driver.
uint8_t	**BSP_EEPROM_DeInit** (void) DeInitializes the EEPROM.
uint32_t	**BSP_EEPROM_ReadBuffer** (uint8_t *pBuffer, uint16_t ReadAddr, uint16_t *NumByteToRead) Reads a block of data from the EEPROM.
uint32_t	**BSP_EEPROM_WritePage** (uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite) Writes more than one byte to the EEPROM with a single WRITE cycle.
uint32_t	**BSP_EEPROM_WriteBuffer** (uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite) Writes buffer of data to the I2C EEPROM.
uint32_t	**BSP_EEPROM_WaitEepromStandbyState** (void) Wait for EEPROM Standby state.
__weak void	**BSP_EEPROM_TIMEOUT_UserCallback** (void) Basic management of the timeout situation.
void	**EEPROM_IO_Init** (void) Initialize peripherals used by the I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_WriteData** (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize) Write data to I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_ReadData** (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize) Read data from I2C EEPROM driver.

HAL_StatusTypeDef	**EEPROM_IO_IsDeviceReady** (uint16_t DevAddress, uint32_t Trials) Checks if target device is ready for communication.

Detailed Description

This file contains all the functions prototypes for the **stm32l4r9i_eval_eeprom.c** firmware driver.

Author:
MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_eeprom.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_gfxmmu_lut.h File Reference

This file contains GFXMMU look-up-table for DSI round LCD. More...

Go to the source code of this file.

Variables

const uint32_t **gfxmmu_lut_config_argb8888** [780]

Detailed Description

This file contains GFXMMU look-up-table for DSI round LCD.

Author:
MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_gfxmmu_lut.h**.

Variable Documentation

const uint32_t gfxmmu_lut_config_argb8888[780]

Definition at line **45** of file **stm32l4r9i_eval_gfxmmu_lut.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_idd.c File Reference

This file provides a set of firmware functions to manage the Idd measurement driver for STM32L4R9I_EVAL board. More...

```
#include "stm32l4r9i_eval_idd.h" #include "stm32l4r9i_eval_io.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_IDD_Init** (void) Configures IDD measurement component.
void	**BSP_IDD_DeInit** (void) Unconfigures IDD measurement component.
void	**BSP_IDD_Reset** (void) Reset Idd measurement component.
void	**BSP_IDD_LowPower** (void) Turn Idd measurement component in low power (standby/sleep) mode.
void	**BSP_IDD_StartMeasure** (void) Start Measurement campaign.
void	**BSP_IDD_Config** (IDD_ConfigTypeDef IddConfig) Configure Idd component.
void	**BSP_IDD_GetValue** (uint32_t *IddValue) Get Idd current value.
void	**BSP_IDD_EnableIT** (void) Enable Idd interrupt that warn end of measurement.
void	**BSP_IDD_ClearIT** (void) Clear Idd interrupt that warn end of measurement.
uint8_t	**BSP_IDD_GetITStatus** (void) Get Idd interrupt status.
void	**BSP_IDD_DisableIT** (void) Disable Idd interrupt that warn end of measurement.
uint8_t	**BSP_IDD_ErrorGetCode** (void) Get Error Code .
void	**BSP_IDD_ErrorEnableIT** (void) Enable error interrupt that warn end of measurement.
void	**BSP_IDD_ErrorClearIT** (void) Clear Error interrupt that warn end of measurement.
uint8_t	**BSP_IDD_ErrorGetITStatus** (void)

	Get Error interrupt status.
void	**BSP_IDD_ErrorDisableIT** (void) Disable Error interrupt.
void	**BSP_IDD_WakeUp** (void) Wake up Idd measurement component.

Variables

static IDD_DrvTypeDef * **IddDrv** = NULL

Detailed Description

This file provides a set of firmware functions to manage the Idd measurement driver for STM32L4R9I_EVAL board.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_idd.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_idd.h File Reference

Header file for **stm32l4r9i_eval_idd.c** module. More...

```
#include "stm32l4r9i_eval.h" #include
"../Components/mfxstm32l152/mfxstm32l152.h"
```

Go to the source code of this file.

Defines

#define	**EVAL_IDD_SHUNT0_VALUE** ((uint16_t) 1000) Shunt values on evaluation board in milli ohms.
#define	**EVAL_IDD_SHUNT1_VALUE** ((uint16_t) 24)
#define	**EVAL_IDD_SHUNT2_VALUE** ((uint16_t) 620)
#define	**EVAL_IDD_SHUNT4_VALUE** ((uint16_t) 10000)
#define	**EVAL_IDD_SHUNT0_STABDELAY** ((uint8_t) 149) Shunt stabilization delay on evaluation board in milli ohms.
#define	**EVAL_IDD_SHUNT1_STABDELAY** ((uint8_t) 149)
#define	**EVAL_IDD_SHUNT2_STABDELAY** ((uint8_t) 149)
#define	**EVAL_IDD_SHUNT4_STABDELAY** ((uint8_t) 255)
#define	**EVAL_IDD_AMPLI_GAIN** ((uint16_t) 4967) IDD Ampli Gain on evaluation board.
#define	**EVAL_IDD_VDD_MIN** ((uint16_t) 1710) IDD Vdd Min on evaluation board.

Enumerations

enum	**IDD_StatusTypeDef** { **IDD_OK** = 0, **IDD_TIMEOUT** = 1, **IDD_ZERO_VALUE** = 2, **IDD_ERROR** = 0xFF } IDD Status values. More...

Functions

uint8_t	**BSP_IDD_Init** (void) Configures IDD measurement component.
void	**BSP_IDD_DeInit** (void) Unconfigures IDD measurement component.
void	**BSP_IDD_Reset** (void) Reset Idd measurement component.
void	**BSP_IDD_LowPower** (void) Turn Idd measurement component in low power (standby/sleep) mode.
void	**BSP_IDD_WakeUp** (void) Wake up Idd measurement component.
void	**BSP_IDD_StartMeasure** (void) Start Measurement campaign.
void	**BSP_IDD_Config** (IDD_ConfigTypeDef IddConfig) Configure Idd component.
void	**BSP_IDD_GetValue** (uint32_t *IddValue) Get Idd current value.
void	**BSP_IDD_EnableIT** (void) Enable Idd interrupt that warn end of measurement.
void	**BSP_IDD_ClearIT** (void) Clear Idd interrupt that warn end of measurement.
uint8_t	**BSP_IDD_GetITStatus** (void) Get Idd interrupt status.
void	**BSP_IDD_DisableIT** (void) Disable Idd interrupt that warn end of measurement.
uint8_t	**BSP_IDD_ErrorGetCode** (void) Get Error Code .
void	**BSP_IDD_ErrorEnableIT** (void) Enable error interrupt that warn end of measurement.
void	**BSP_IDD_ErrorClearIT** (void)

	Clear Error interrupt that warn end of measurement.
uint8_t	**BSP_IDD_ErrorGetITStatus** (void) Get Error interrupt status.
void	**BSP_IDD_ErrorDisableIT** (void) Disable Error interrupt.

Detailed Description

Header file for **stm32l4r9i_eval_idd.c** module.

Author:
MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_idd.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_io.c File Reference

This file provides a set of functions needed to manage the IO pins on STM32L4R9I_EVAL evaluation board. More...

```
#include "stm32l4r9i_eval_io.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_IO_Init** (void) Initialize and configure the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).
uint8_t	**BSP_IO_DeInit** (void) DeInitialize the IO to allow Mfx Initialization to be executed again.
uint32_t	**BSP_IO_ITGetStatus** (uint32_t IO_Pin) Get the selected pins IT status.
void	**BSP_IO_ITClear** (uint32_t IO_Pin) Clear the selected IO IT pending bit.
uint8_t	**BSP_IO_ConfigPin** (uint32_t IO_Pin, IO_ModeTypedef IO_Mode) Configure the IO pin(s) according to IO mode structure value.
void	**BSP_IO_WritePin** (uint32_t IO_Pin, uint8_t PinState) Set the selected IO pin(s) state.
uint32_t	**BSP_IO_ReadPin** (uint32_t IO_Pin) Get the selected IO pin(s) current state.
void	**BSP_IO_TogglePin** (uint32_t IO_Pin) Toggle the selected IO pin(s) state.

Variables

static IO_DrvTypeDef *	**io_driver**

Detailed Description

This file provides a set of functions needed to manage the IO pins on STM32L4R9I_EVAL evaluation board.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_io.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_io.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_io.c** driver. More...

```
#include "stm32l4r9i_eval.h" #include
"../Components/mfxstm32l152/mfxstm32l152.h"
```

Go to the source code of this file.

Defines

```
#define IO_PIN_0    ((uint32_t)0x00000001)
#define IO_PIN_1    ((uint32_t)0x00000002)
#define IO_PIN_2    ((uint32_t)0x00000004)
#define IO_PIN_3    ((uint32_t)0x00000008)
#define IO_PIN_4    ((uint32_t)0x00000010)
#define IO_PIN_5    ((uint32_t)0x00000020)
#define IO_PIN_6    ((uint32_t)0x00000040)
#define IO_PIN_7    ((uint32_t)0x00000080)
#define IO_PIN_8    ((uint32_t)0x00000100)
#define IO_PIN_9    ((uint32_t)0x00000200)
#define IO_PIN_10   ((uint32_t)0x00000400)
#define IO_PIN_11   ((uint32_t)0x00000800)
#define IO_PIN_12   ((uint32_t)0x00001000)
#define IO_PIN_13   ((uint32_t)0x00002000)
#define IO_PIN_14   ((uint32_t)0x00004000)
#define IO_PIN_15   ((uint32_t)0x00008000)
#define AGPIO_PIN_0   ((uint32_t)0x00010000)
#define AGPIO_PIN_1   ((uint32_t)0x00020000)
#define IO_PIN_ALL   ((uint32_t)0x0003FFFF)
```

Enumerations

enum	**IO_StatusTypeDef** { **IO_OK** = 0x00, **IO_ERROR** = 0x01, **IO_TIMEOUT** = 0x02 }

Functions

uint8_t	**BSP_IO_Init** (void) Initialize and configure the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).
uint8_t	**BSP_IO_DeInit** (void) DeInitialize the IO to allow Mfx Initialization to be executed again.
uint32_t	**BSP_IO_ITGetStatus** (uint32_t IO_Pin) Get the selected pins IT status.
void	**BSP_IO_ITClear** (uint32_t IO_Pin) Clear the selected IO IT pending bit.
uint8_t	**BSP_IO_ConfigPin** (uint32_t IO_Pin, IO_ModeTypedef IO_Mode) Configure the IO pin(s) according to IO mode structure value.
void	**BSP_IO_WritePin** (uint32_t IO_Pin, uint8_t PinState) Set the selected IO pin(s) state.
uint32_t	**BSP_IO_ReadPin** (uint32_t IO_Pin) Get the selected IO pin(s) current state.
void	**BSP_IO_TogglePin** (uint32_t IO_Pin) Toggle the selected IO pin(s) state.

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_io.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_io.h**.

STM32L4R9I_EVAL BSP User Manual

Data Structures | Defines | Typedefs | Enumerations

stm32l4r9i_eval_lcd.h File Reference

This file contains alias for compatibility with DSI and TFT LCD. More...

```
#include "../../../Utilities/Fonts/fonts.h"
```

Go to the source code of this file.

Data Structures

struct	**LCD_DrawPropTypeDef**
struct	**Point**

Defines

#define	**LCD_LayerCfgTypeDef** LTDC_LayerCfgTypeDef
#define	**LTDC_MAX_LAYER_NUMBER** ((uint32_t) 2)
#define	**LTDC_ACTIVE_LAYER_BACKGROUND** LTDC_LAYER_1
#define	**LTDC_ACTIVE_LAYER_FOREGROUND** LTDC_LAYER_2
#define	**LTDC_NB_OF_LAYERS** ((uint32_t) 2)
#define	**LTDC_DEFAULT_ACTIVE_LAYER** **LTDC_ACTIVE_LAYER_**
#define	**LCD_OK** 0x00
#define	**LCD_ERROR** 0x01
#define	**LCD_TIMEOUT** 0x02
#define	**LCD_COLOR_BLUE** ((uint32_t) 0xFF0000FF)
#define	**LCD_COLOR_GREEN** ((uint32_t) 0xFF00FF00)
#define	**LCD_COLOR_RED** ((uint32_t) 0xFFFF0000)
#define	**LCD_COLOR_CYAN** ((uint32_t) 0xFF00FFFF)
#define	**LCD_COLOR_MAGENTA** ((uint32_t) 0xFFFF00FF)
#define	**LCD_COLOR_YELLOW** ((uint32_t) 0xFFFFFF00)
#define	**LCD_COLOR_LIGHTBLUE** ((uint32_t) 0xFF8080FF)
#define	**LCD_COLOR_LIGHTGREEN** ((uint32_t) 0xFF80FF80)
#define	**LCD_COLOR_LIGHTRED** ((uint32_t) 0xFFFF8080)
#define	**LCD_COLOR_LIGHTCYAN** ((uint32_t) 0xFF80FFFF)
#define	**LCD_COLOR_LIGHTMAGENTA** ((uint32_t) 0xFFFF80FF)
#define	**LCD_COLOR_LIGHTYELLOW** ((uint32_t) 0xFFFFFF80)
#define	**LCD_COLOR_DARKBLUE** ((uint32_t) 0xFF000080)
#define	**LCD_COLOR_DARKGREEN** ((uint32_t) 0xFF008000)
#define	**LCD_COLOR_DARKRED** ((uint32_t) 0xFF800000)
#define	**LCD_COLOR_DARKCYAN** ((uint32_t) 0xFF008080)
#define	**LCD_COLOR_DARKMAGENTA** ((uint32_t) 0xFF800080)
#define	**LCD_COLOR_DARKYELLOW** ((uint32_t) 0xFF808000)
#define	**LCD_COLOR_WHITE** ((uint32_t) 0xFFFFFFFF)
#define	**LCD_COLOR_LIGHTGRAY** ((uint32_t) 0xFFD3D3D3)
#define	**LCD_COLOR_GRAY** ((uint32_t) 0xFF808080)
#define	**LCD_COLOR_DARKGRAY** ((uint32_t) 0xFF404040)

```
#define LCD_COLOR_BLACK ((uint32_t) 0xFF000000)
#define LCD_COLOR_BROWN ((uint32_t) 0xFFA52A2A)
#define LCD_COLOR_ORANGE ((uint32_t) 0xFFFFA500)
#define LCD_COLOR_TRANSPARENT ((uint32_t) 0xFF000000)
#define LCD_DEFAULT_FONT Font24
```

Typedefs

typedef **Point** * **pPoint**

Enumerations

enum	**Text_AlignModeTypdef** { **CENTER_MODE** = 0x01, **RIGHT_MODE** = 0x02, **LEFT_MODE** = 0x03 }
enum	**LCD_OrientationTypeDef** { **LCD_ORIENTATION_PORTRAIT** = 0x00, **LCD_ORIENTATION_LANDSCAPE** = 0x01, **LCD_ORIENTATION_INVALID** = 0x02 }

Detailed Description

This file contains alias for compatibility with DSI and TFT LCD.

Author:
MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_lcd.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_nor.c File Reference

This file includes a standard driver for the M29W128GL70ZA6E NOR memories mounted on STM32L4R9I-EVAL board. More...

```
#include "stm32l4r9i_eval_nor.h"
```

Go to the source code of this file.

Defines

#define	**NOR_ADDR_SETUP_TIME** 9 /* 70ns with a clock at 120 MHz (period of 8.33 ns) */
#define	**NOR_DATA_SETUP_TIME** 5 /* 45ns with a clock at 120 MHz (period of 8.33 ns) */
#define	**NOR_DATA_HOLD_TIME** 0
#define	**NOR_TURN_AROUND_TIME** 4 /* 30ns with a clock at 120 MHz (perido of 8.33 ns) */
#define	**NOR_MAX_BUFFER_SIZE** 32 /* 16-bits bus so 32 words */

Functions

static void	**NOR_MspInit** (void) Initializes the NOR MSP.
uint8_t	**BSP_NOR_Init** (void) Initializes the NOR device.
uint8_t	**BSP_NOR_ReadData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the NOR device.
void	**BSP_NOR_ReturnToReadMode** (void) Returns the NOR memory to read mode.
uint8_t	**BSP_NOR_WriteData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data to the NOR device.
uint8_t	**BSP_NOR_ProgramData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Programs an amount of data to the NOR device.
uint8_t	**BSP_NOR_Erase_Block** (uint32_t BlockAddress) Erases the specified block of the NOR device.
uint8_t	**BSP_NOR_Erase_Chip** (void) Erases the entire NOR chip.
uint8_t	**BSP_NOR_Read_ID** (NOR_IDTypeDef *pNOR_ID) Reads NOR flash IDs.
void	**HAL_NOR_MspWait** (NOR_HandleTypeDef *hnor, uint32_t Timeout) NOR BSP Wait for Ready/Busy signal.

Variables

static NOR_HandleTypeDef	**norHandle**
static FMC_NORSRAM_TimingTypeDef	**Timing**

Detailed Description

This file includes a standard driver for the M29W128GL70ZA6E NOR memories mounted on STM32L4R9I-EVAL board.

Author:

MCD Application Team

```
  ==========================================
==================================
                     ##### How to use this dri
ver #####
  ==========================================
==================================
  [..]
   (#) This driver is used to drive the M29W12
8GL70ZA6E NOR flash external
       memory mounted on STM32L4R9I-EVAL evalu
ation board.

   (#) This driver does not need a specific co
mponent driver for the NOR device
       to be included with.

   (#) Initialization steps:
       (++) Initialize the NOR external memory
 using the BSP_NOR_Init() function. This
            function includes the MSP layer ha
rdware resources initialization and the
            FMC controller configuration to in
terface with the external NOR memory.

   (#) NOR flash operations
       (++) NOR external memory can be accesse
d with read/write operations once it is
            initialized.
            Read/write operation can be perfor
```

```
med with AHB access using the functions
            BSP_NOR_ReadData()/BSP_NOR_WriteDa
ta(). The BSP_NOR_WriteData() performs write o
peration
            of an amount of data by unit (half
word). You can also perform a program data
            operation of an amount of data usi
ng the function BSP_NOR_ProgramData().
       (++) The function BSP_NOR_Read_ID() ret
urns the chip IDs stored in the structure
            "NOR_IDTypeDef". (see the NOR IDs
in the memory data sheet)
       (++) Perform erase block operation usin
g the function BSP_NOR_Erase_Block() and by
            specifying the block address. You
can perform an erase operation of the whole
            chip by calling the function BSP_N
OR_Erase_Chip().
       (++) After other operations, the functi
on BSP_NOR_ReturnToReadMode() allows the NOR
            flash to return to read mode to pe
rform read operations on it.
```

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_nor.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_nor.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_nor.c** driver. More...

```
#include "stm32l4xx_hal.h"
```

Go to the source code of this file.

Defines

#define	**NOR_STATUS_OK** 0x00 NOR status structure definition.
#define	**NOR_STATUS_ERROR** 0x01
#define	**NOR_DEVICE_ADDR** ((uint32_t)0x68000000)
#define	**NOR_MEMORY_WIDTH** FMC_NORSRAM_MEM_BUS_WID
#define	**NOR_BURSTACCESS** FMC_BURST_ACCESS_MODE_DIS
#define	**NOR_WRITEBURST** FMC_WRITE_BURST_DISABLE
#define	**BLOCKERASE_TIMEOUT** 2000U /* NOR block erase timeou */
#define	**CHIPERASE_TIMEOUT** 400000U /* NOR chip erase timeout s */
#define	**PROGRAM_TIMEOUT** 13U /* NOR program timeout : 64 byte 200 us = 12.8 ms */
#define	**NOR_READY_BUSY_PIN** GPIO_PIN_6
#define	**NOR_READY_BUSY_GPIO** GPIOD
#define	**NOR_READY_STATE** GPIO_PIN_SET
#define	**NOR_BUSY_STATE** GPIO_PIN_RESET

Functions

uint8_t	**BSP_NOR_Init** (void) Initializes the NOR device.
uint8_t	**BSP_NOR_ReadData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the NOR device.
uint8_t	**BSP_NOR_WriteData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data to the NOR device.
uint8_t	**BSP_NOR_ProgramData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Programs an amount of data to the NOR device.
uint8_t	**BSP_NOR_Erase_Block** (uint32_t BlockAddress) Erases the specified block of the NOR device.
uint8_t	**BSP_NOR_Erase_Chip** (void) Erases the entire NOR chip.
uint8_t	**BSP_NOR_Read_ID** (NOR_IDTypeDef *pNOR_ID) Reads NOR flash IDs.
void	**BSP_NOR_ReturnToReadMode** (void) Returns the NOR memory to read mode.

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_nor.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_nor.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ospi_nor.c File Reference

This file includes a standard driver for the MX25UM51245G OSPI memory mounted on STM32L4R9I-EVAL board. More...

```
#include "stm32l4r9i_eval_ospi_nor.h"
```

Go to the source code of this file.

Defines

#define	**OSPI_NOR_OCTAL_DISABLE** 0x0
#define	**OSPI_NOR_OCTAL_ENABLE** 0x1
#define	**DUMMY_CYCLES_READ_OCTAL** MX25LM51245G_DUMM
#define	**DUMMY_CYCLES_CR_CFG** MX25LM51245G_CR2_DC_66

Functions

static void	**OSPI_NOR_MspInit** (void) Initializes the OSPI MSP.
static void	**OSPI_NOR_MspDeInit** (void) De-Initializes the OSPI MSP.
static uint8_t	**OSPI_NOR_ResetMemory** (OSPI_HandleTypeDef *hospi) This function reset the OSPI memory.
static uint8_t	**OSPI_NOR_WriteEnable** (OSPI_HandleTypeDef *hospi) This function send a Write Enable and wait it is effective.
static uint8_t	**OSPI_NOR_AutoPollingMemReady** (OSPI_HandleTypeDef *hospi, uint32_t Timeout) This function read the SR of the memory and wait the EOP.
static uint8_t	**OSPI_NOR_OctalMode** (OSPI_HandleTypeDef *hospi, uint8_t Operation) This function enables/disables the octal mode of the memory.
uint8_t	**BSP_OSPI_NOR_Init** (void) Initializes the OSPI interface.
uint8_t	**BSP_OSPI_NOR_DeInit** (void) De-Initializes the OSPI interface.
uint8_t	**BSP_OSPI_NOR_Read** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Write** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Block** (uint32_t BlockAddress)

	Erases the specified block of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Sector** (uint32_t Sector) Erases the specified sector of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Chip** (void) Erases the entire OSPI memory.
uint8_t	**BSP_OSPI_NOR_GetStatus** (void) Reads current status of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_GetInfo** (**OSPI_NOR_Info** *pInfo) Return the configuration of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_EnableMemoryMappedMode** (void) Configure the OSPI in memory-mapped mode.
uint8_t	**BSP_OSPI_NOR_SuspendErase** (void) This function suspends an ongoing erase command.
uint8_t	**BSP_OSPI_NOR_ResumeErase** (void) This function resumes a paused erase command.
uint8_t	**BSP_OSPI_NOR_EnterDeepPowerDown** (void) This function enter the OSPI memory in deep power down mode.
uint8_t	**BSP_OSPI_NOR_LeaveDeepPowerDown** (void) This function leave the OSPI memory from deep power down mode.

Variables

OSPI_HandleTypeDef	**OSPINORHandle**

Detailed Description

This file includes a standard driver for the MX25UM51245G OSPI memory mounted on STM32L4R9I-EVAL board.

Author:

MCD Application Team

```
  ==========================================
==================================
                     ##### How to use this dri
ver #####
  ==========================================
==================================
  [..]
   (#) This driver is used to drive the MX25UM
51245G OSPI external
       memory mounted on STM32L4R9I-EVAL evalu
ation board.

   (#) This driver need a specific component d
river (MX25UM51245G) to be included with.

   (#) Initialization steps:
       (++) Initialize the OPSI external memor
y using the BSP_OSPI_NOR_Init() function. This
            function includes the MSP layer ha
rdware resources initialization and the
            OSPI interface with the external m
emory.

   (#) OSPI memory operations
       (++) OSPI memory can be accessed with r
ead/write operations once it is
            initialized.
            Read/write operation can be perfor
med with AHB access using the functions
```

```
            BSP_OSPI_NOR_Read()/BSP_OSPI_NOR_W
rite().
       (++) The function BSP_OSPI_NOR_GetInfo(
) returns the configuration of the OSPI memory
.
            (see the OSPI memory data sheet)
       (++) Perform erase block operation usin
g the function BSP_OSPI_NOR_Erase_Block() and
by
            specifying the block address. You
can perform an erase operation of the whole
            chip by calling the function BSP_O
SPI_NOR_Erase_Chip().
       (++) The function BSP_OSPI_NOR_GetStatu
s() returns the current status of the OSPI mem
ory.
            (see the OSPI memory data sheet)
```

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_ospi_nor.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ospi_nor.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_ospi_nor.c** driver. More...

```
#include "stm32l4xx_hal.h" #include
"../Components/mx25lm51245g/mx25lm51245g.h"
```

Go to the source code of this file.

Data Structures

struct **OSPI_NOR_Info**

Defines

#define	**OSPI_NOR_OK** ((uint8_t)0x00)
#define	**OSPI_NOR_ERROR** ((uint8_t)0x01)
#define	**OSPI_NOR_BUSY** ((uint8_t)0x02)
#define	**OSPI_NOR_NOT_SUPPORTED** ((uint8_t)0x04)
#define	**OSPI_NOR_SUSPENDED** ((uint8_t)0x08)

Functions

uint8_t	**BSP_OSPI_NOR_Init** (void) Initializes the OSPI interface.
uint8_t	**BSP_OSPI_NOR_DeInit** (void) De-Initializes the OSPI interface.
uint8_t	**BSP_OSPI_NOR_Read** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Write** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Block** (uint32_t BlockAddress) Erases the specified block of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Sector** (uint32_t Sector) Erases the specified sector of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Chip** (void) Erases the entire OSPI memory.
uint8_t	**BSP_OSPI_NOR_GetStatus** (void) Reads current status of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_GetInfo** (**OSPI_NOR_Info** *pInfo) Return the configuration of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_EnableMemoryMappedMode** (void) Configure the OSPI in memory-mapped mode.
uint8_t	**BSP_OSPI_NOR_SuspendErase** (void) This function suspends an ongoing erase command.
uint8_t	**BSP_OSPI_NOR_ResumeErase** (void) This function resumes a paused erase command.
uint8_t	**BSP_OSPI_NOR_EnterDeepPowerDown** (void) This function enter the OSPI memory in deep power down mode.
uint8_t	**BSP_OSPI_NOR_LeaveDeepPowerDown** (void) This function leave the OSPI memory from deep power down

mode.

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_ospi_nor.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_ospi_nor.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ospi_ram.c File Reference

This file includes a standard driver for the ISS66WVH8M8 HyperRAM memory mounted on STM32L4R9I-EVAL board. More...

```
#include "stm32l4r9i_eval_ospi_ram.h"
```

Go to the source code of this file.

Defines

#define	**OSPI_RAM_QUAD_DISABLE** 0x0
#define	**OSPI_RAM_QUAD_ENABLE** 0x1
#define	**OSPI_RAM_HIGH_PERF_DISABLE** 0x0
#define	**OSPI_RAM_HIGH_PERF_ENABLE** 0x1

Functions

static void	**OSPI_RAM_MspInit** (void) Initializes the OSPI MSP.
static void	**OSPI_RAM_MspDeInit** (void) De-Initializes the OSPI MSP.
uint8_t	**BSP_OSPI_RAM_Init** (void) Initializes the OSPI interface.
uint8_t	**BSP_OSPI_RAM_DeInit** (void) De-Initializes the OSPI interface.
uint8_t	**BSP_OSPI_RAM_Read** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the HyperRAM memory.
uint8_t	**BSP_OSPI_RAM_Read_DMA** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the HyperRAM memory in DMA mode.
uint8_t	**BSP_OSPI_RAM_Write** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the HyperRAM memory.
uint8_t	**BSP_OSPI_RAM_Write_DMA** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the HyperRAM memory in DMA mode.
uint8_t	**BSP_OSPI_RAM_EnableMemoryMappedMode** (void) Configure the OSPI in memory-mapped mode.
uint8_t	**BSP_OSPI_RAM_EnterDeepPowerDown** (void) This function enter the HyperRAM memory in deep power down mode.
uint8_t	**BSP_OSPI_RAM_LeaveDeepPowerDown** (void) This function leave the HyperRAM memory from deep power down mode.
void	**BSP_OSPI_RAM_DMA_IRQHandler** (void)

	Handles OctoSPI HyperRAM DMA transfer interrupt request.
void	**BSP_OSPI_RAM_IRQHandler** (void) Handles OctoSPI HyperRAM interrupt request.

Variables

OSPI_HandleTypeDef	**OspiRamHandle**
DMA_HandleTypeDef	**DmaOspiRamHandle**

Detailed Description

This file includes a standard driver for the ISS66WVH8M8 HyperRAM memory mounted on STM32L4R9I-EVAL board.

Author:

MCD Application Team

```
  ==========================================
==================================
                     ##### How to use this dri
ver #####
  ==========================================
==================================
  [..]
   (#) This driver is used to drive the ISS66W
VH8M8 HyperRAM external
       memory mounted on STM32L4R9I-EVAL evalu
ation board.

   (#) This driver need a specific component d
river (ISS66WVH8M8) to be included with.

   (#) Initialization steps:
       (++) Initialize the HyperRAM external m
emory using the BSP_OSPI_RAM_Init() function.
This
            function includes the MSP layer ha
rdware resources initialization and the
            OSPI interface with the external m
emory.

   (#) HyperRAM memory operations
       (++) HyperRAM memory can be accessed wi
th read/write operations once it is
            initialized.
            Read/write operation can be perfor
```

```
med with AHB access using the functions
            BSP_OSPI_RAM_Read()/BSP_OSPI_RAM_W
rite().
       (++) The function BSP_OSPI_RAM_GetInfo(
) returns the configuration of the HyperRAM me
mory.
            (see the HyperRAM memory data shee
t)
       (++) Perform erase block operation usin
g the function BSP_OSPI_RAM_Erase_Block() and
by
            specifying the block address. You
can perform an erase operation of the whole
            chip by calling the function BSP_O
SPI_RAM_Erase_Chip().
       (++) The function BSP_OSPI_RAM_GetStatu
s() returns the current status of the HyperRAM
 memory.
            (see the HyperRAM memory data shee
t)
```

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_ospi_ram.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ospi_ram.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_ospi_ram.c** driver. More...

```
#include "stm32l4xx_hal.h" #include
"../Components/iss66wvh8m8/iss66wvh8m8.h"
```

Go to the source code of this file.

Defines

```
#define OSPI_RAM_OK    ((uint8_t)0x00)
#define OSPI_RAM_ERROR    ((uint8_t)0x01)
```

Functions

uint8_t	**BSP_OSPI_RAM_Init** (void) Initializes the OSPI interface.
uint8_t	**BSP_OSPI_RAM_DeInit** (void) De-Initializes the OSPI interface.
uint8_t	**BSP_OSPI_RAM_Read** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the HyperRAM memory.
uint8_t	**BSP_OSPI_RAM_Read_DMA** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the HyperRAM memory in DMA mode.
uint8_t	**BSP_OSPI_RAM_Write** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the HyperRAM memory.
uint8_t	**BSP_OSPI_RAM_Write_DMA** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the HyperRAM memory in DMA mode.
uint8_t	**BSP_OSPI_RAM_EnableMemoryMappedMode** (void) Configure the OSPI in memory-mapped mode.
uint8_t	**BSP_OSPI_RAM_EnterDeepPowerDown** (void) This function enter the HyperRAM memory in deep power down mode.
uint8_t	**BSP_OSPI_RAM_LeaveDeepPowerDown** (void) This function leave the HyperRAM memory from deep power down mode.
void	**BSP_OSPI_RAM_DMA_IRQHandler** (void) Handles OctoSPI HyperRAM DMA transfer interrupt request.
void	**BSP_OSPI_RAM_IRQHandler** (void) Handles OctoSPI HyperRAM interrupt request.

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_ospi_ram.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_ospi_ram.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_rgb_lcd.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_rgb_lcd.c** driver. More...

```
#include "stm32l4r9i_eval.h" #include
"../Components/rk043fn48h/rk043fn48h.h"
```

Go to the source code of this file.

Defines

#define	**LCD_FB_START_ADDRESS** (uint32_t)**PhysFrameBuffer** /* SRAM used for LCD Frame buffer */ LCD FB_StartAddress.
#define	**LCD_RELOAD_IMMEDIATE** ((uint32_t)LTDC_SRCR_IMR) LCD Reload Types.
#define	**LCD_RELOAD_VERTICAL_BLANKING** ((uint32_t)LTDC_SF

Functions

uint8_t	**BSP_RGB_LCD_Init** (void)
uint8_t	**BSP_RGB_LCD_DeInit** (void)
uint32_t	**BSP_RGB_LCD_GetXSize** (void)
uint32_t	**BSP_RGB_LCD_GetYSize** (void)
void	**BSP_RGB_LCD_SetXSize** (uint32_t imageWidthPixels)
void	**BSP_RGB_LCD_SetYSize** (uint32_t imageHeightPixels)
void	**BSP_RGB_LCD_LayerDefaultInit** (uint16_t LayerIndex, uint32_t FrameBuffer)
void	**BSP_RGB_LCD_LayerRgb565Init** (uint16_t LayerIndex, uint32_t FB_Address)
void	**BSP_RGB_LCD_SetTransparency** (uint32_t LayerIndex, uint8_t Transparency)
void	**BSP_RGB_LCD_SetTransparency_NoReload** (uint32_t LayerIndex, uint8_t Transparency)
void	**BSP_RGB_LCD_SetLayerAddress** (uint32_t LayerIndex, uint32_t Address)
void	**BSP_RGB_LCD_SetLayerAddress_NoReload** (uint32_t LayerIndex, uint32_t Address)
void	**BSP_RGB_LCD_SetColorKeying** (uint32_t LayerIndex, uint32_t RGBValue)
void	**BSP_RGB_LCD_SetColorKeying_NoReload** (uint32_t LayerIndex, uint32_t RGBValue)
void	**BSP_RGB_LCD_ResetColorKeying** (uint32_t LayerIndex)
void	**BSP_RGB_LCD_ResetColorKeying_NoReload** (uint32_t LayerIndex)
void	**BSP_RGB_LCD_SetLayerWindow** (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
void	**BSP_RGB_LCD_SetLayerWindow_NoReload** (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
void	**BSP_RGB_LCD_SelectLayer** (uint32_t LayerIndex)

void	**BSP_RGB_LCD_SetLayerVisible** (uint32_t LayerIndex, FunctionalState State)
void	**BSP_RGB_LCD_SetLayerVisible_NoReload** (uint32_t LayerIndex, FunctionalState State)
void	**BSP_RGB_LCD_Reload** (uint32_t ReloadType)
void	**BSP_RGB_LCD_SetTextColor** (uint32_t Color)
uint32_t	**BSP_RGB_LCD_GetTextColor** (void)
void	**BSP_RGB_LCD_SetBackColor** (uint32_t Color)
uint32_t	**BSP_RGB_LCD_GetBackColor** (void)
void	**BSP_RGB_LCD_SetFont** (sFONT *fonts)
sFONT *	**BSP_RGB_LCD_GetFont** (void)
uint32_t	**BSP_RGB_LCD_ReadPixel** (uint16_t Xpos, uint16_t Ypos)
void	**BSP_RGB_LCD_DrawPixel** (uint16_t Xpos, uint16_t Ypos, uint32_t pixel)
void	**BSP_RGB_LCD_Clear** (uint32_t Color)
void	**BSP_RGB_LCD_ClearStringLine** (uint32_t Line)
void	**BSP_RGB_LCD_DisplayStringAtLine** (uint16_t Line, uint8_t *ptr)
void	**BSP_RGB_LCD_DisplayStringAt** (uint16_t Xpos, uint16_t Ypos, uint8_t *Text, **Text_AlignModeTypdef** Mode)
void	**BSP_RGB_LCD_DisplayChar** (uint16_t Xpos, uint16_t Ypos, uint8_t Ascii)
void	**BSP_RGB_LCD_DrawHLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length)
void	**BSP_RGB_LCD_DrawVLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length)
void	**BSP_RGB_LCD_DrawLine** (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2)
void	**BSP_RGB_LCD_DrawRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
void	**BSP_RGB_LCD_DrawCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)
void	**BSP_RGB_LCD_DrawPolygon** (**pPoint** Points, uint16_t PointCount)
void	**BSP_RGB_LCD_DrawEllipse** (int Xpos, int Ypos, int

	XRadius, int YRadius)
void	**BSP_RGB_LCD_DrawBitmap** (uint32_t Xpos, uint32_t Ypos, uint8_t *pbmp)
void	**BSP_RGB_LCD_FillRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
void	**BSP_RGB_LCD_FillCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)
void	**BSP_RGB_LCD_FillPolygon** (**pPoint** Points, uint16_t PointCount)
void	**BSP_RGB_LCD_FillEllipse** (int Xpos, int Ypos, int XRadius, int YRadius)
void	**BSP_RGB_LCD_DisplayOff** (void)
void	**BSP_RGB_LCD_DisplayOn** (void)
void	**BSP_RGB_LCD_MspInit** (LTDC_HandleTypeDef *hltdc, void *Params)
void	**BSP_RGB_LCD_MspDeInit** (LTDC_HandleTypeDef *hltdc, void *Params)
void	**BSP_RGB_LCD_ClockConfig** (LTDC_HandleTypeDef *hltdc, void *Params)
void	**BSP_RGB_LCD_LTDC_IRQHandler** (void)
void	**BSP_RGB_LCD_LTDC_ER_IRQHandler** (void)
void	**BSP_RGB_LCD_DMA2D_IRQHandler** (void)

Variables

LTDC_HandleTypeDef	**hltdc_eval**
uint32_t	**PhysFrameBuffer** []

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_rgb_lcd.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_rgb_lcd.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_rgb_ts.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_rgb_ts.c** driver. More...

```
#include "stm32l4r9i_eval.h" #include
"../Components/ft5336/ft5336.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_RGB_TS_Init** (uint16_t ts_SizeX, uint16_t ts_SizeY)
uint8_t	**BSP_RGB_TS_DeInit** (void)
uint8_t	**BSP_RGB_TS_GetState** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_RGB_TS_Get_GestureId** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_RGB_TS_ITConfig** (void)
uint8_t	**BSP_RGB_TS_ITGetStatus** (void)
void	**BSP_RGB_TS_ITClear** (void)
uint8_t	**BSP_RGB_TS_ResetTouchData** (**TS_StateTypeDef** *TS_State)

Variables

char *	**ts_event_string_tab** [**TOUCH_EVENT_NB_MAX**] Table for touchscreen event information display on LCD : table indexed on enum **TS_TouchEventTypeDef** information.
char *	**ts_gesture_id_string_tab** [**GEST_ID_NB_MAX**] Table for touchscreen gesture Id information display on LCD : table indexed on enum **TS_GestureIdTypeDef** information.

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_rgb_ts.c** driver.

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_rgb_ts.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_sd.c File Reference

This file includes the uSD card driver. More...

```
#include "stm32l4r9i_eval_io.h" #include "stm32l4r9i_eval_sd.h"
```

Go to the source code of this file.

Functions

uint8_t	**BSP_SD_Init** (void) Initializes the SD card device.
uint8_t	**BSP_SD_DeInit** (void) DeInitializes the SD card device.
uint8_t	**BSP_SD_ITConfig** (void) Configures Interrupt mode for SD detection pin.
uint8_t	**BSP_SD_IsDetected** (void) Detects if SD card is correctly plugged in the memory slot or not.
uint8_t	**BSP_SD_ReadBlocks** (uint32_t *pData, uint32_t ReadAddr, uint32_t NumOfBlocks, uint32_t Timeout) Reads block(s) from a specified address in an SD card, in polling mode.
uint8_t	**BSP_SD_WriteBlocks** (uint32_t *pData, uint32_t WriteAddr, uint32_t NumOfBlocks, uint32_t Timeout) Writes block(s) to a specified address in an SD card, in polling mode.
uint8_t	**BSP_SD_ReadBlocks_DMA** (uint32_t *pData, uint32_t ReadAddr, uint32_t NumOfBlocks) Reads block(s) from a specified address in an SD card, in DMA mode.
uint8_t	**BSP_SD_WriteBlocks_DMA** (uint32_t *pData, uint32_t WriteAddr, uint32_t NumOfBlocks) Writes block(s) to a specified address in an SD card, in DMA mode.
uint8_t	**BSP_SD_Erase** (uint32_t StartAddr, uint32_t EndAddr) Erases the specified memory area of the given SD card.
uint8_t	**BSP_SD_GetCardState** (void) Gets the current SD card data status.
void	**BSP_SD_GetCardInfo** (**BSP_SD_CardInfo** *CardInfo)

	Get SD information about specific SD card.
__weak void	**BSP_SD_MspInit** (SD_HandleTypeDef *hsd, void *Params) Initializes the SD MSP.
__weak void	**BSP_SD_MspDeInit** (SD_HandleTypeDef *hsd, void *Params) De-Initializes the SD MSP.
__weak void	**BSP_SD_AbortCallback** (void) BSP SD Abort callback.
__weak void	**BSP_SD_WriteCpltCallback** (void) BSP Tx Transfer completed callback.
__weak void	**BSP_SD_ReadCpltCallback** (void) BSP Rx Transfer completed callback.
void	**HAL_SD_AbortCallback** (SD_HandleTypeDef *hsd) SD Abort callback.
void	**HAL_SD_TxCpltCallback** (SD_HandleTypeDef *hsd) Tx Transfer completed callback.
void	**HAL_SD_RxCpltCallback** (SD_HandleTypeDef *hsd) Rx Transfer completed callback.

Variables

SD_HandleTypeDef	**hsd_eval**
static uint8_t	**UseExtiModeDetection** = 0

Detailed Description

This file includes the uSD card driver.

Author:

MCD Application Team

```
  ==========================================
==================================
                     ##### How to use this dri
ver #####
  ==========================================
==================================
  (#) This driver is used to drive the micro S
D external card mounted on STM32L4R9I_EVAL
     evaluation board.

  (#) This driver does not need a specific com
ponent driver for the micro SD device
     to be included with.

  (#) Initialization steps:
       (++) Initialize the micro SD card using
 the BSP_SD_Init() function. This
            function includes the MSP layer ha
rdware resources initialization (BSP_SD_MspIni
t())
            and the SDMMC1 interface configura
tion to interface with the external micro SD.
It
            also includes the micro SD initial
ization sequence.
       (++) To check the SD card presence you
can use the function BSP_SD_IsDetected() which
            returns the detection status.
       (++) If SD presence detection interrupt
 mode is desired, you must configure the
```

```
            SD detection interrupt mode by cal
ling the functions BSP_SD_ITConfig().
            The interrupt is generated as an e
xternal interrupt whenever the
            micro SD card is plugged/unplugged
 in/from the evaluation board. The SD detectio
n
            is managed by MFX, so the SD detec
tion interrupt has to be treated by MFX_IRQOUT
            gpio pin IRQ handler.
       (++) The function BSP_SD_GetCardInfo()
is used to get the micro SD card information
            which is stored in the structure "
HAL_SD_CardInfoTypedef".

  (#) Micro SD card operations
       (++) The micro SD card can be accessed
with read/write block(s) operations once
            it is reay for access. The access
can be performed whether using the polling
            mode by calling the functions BSP_
SD_ReadBlocks()/BSP_SD_WriteBlocks(),
            or by DMA transfer using the funct
ions BSP_SD_ReadBlocks_DMA()/BSP_SD_WriteBlock
s_DMA().
       (++) The DMA transfer complete is used
with interrupt mode. Once the SD transfer
            is complete, the DMA Tx/Rx transfe
r complete are handled using the
            BSP_SD_WriteCpltCallback()/BSP_SD_
ReadCpltCallback() user callback functions imp
lemented
            by the user at application level.
       (++) The SD erase block(s) is performed
 using the function BSP_SD_Erase() with specif
ying
            the number of blocks to erase.
```

```
       (++) The SD runtime status is returned
when calling the function BSP_SD_GetStatus().
   [..]
```

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_sd.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_sd.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_sd.c** driver. More...

```
#include "stm32l4xx_hal.h"
```

Go to the source code of this file.

Defines

#define	**BSP_SD_CardInfo** HAL_SD_CardInfoTypeDef SD Card information structure.
#define	**MSD_OK** ((uint8_t)0x00) SD status structure definition.
#define	**MSD_ERROR** ((uint8_t)0x01)
#define	**MSD_ERROR_SD_NOT_PRESENT** ((uint8_t)0x02)
#define	**SD_TRANSFER_OK** ((uint8_t)0x00) SD transfer state definition.
#define	**SD_TRANSFER_BUSY** ((uint8_t)0x01)
#define	**SD_TRANSFER_ERROR** ((uint8_t)0x02)
#define	**SD_DATATIMEOUT** ((uint32_t)100000000)
#define	**SD_PRESENT** ((uint8_t)0x01)
#define	**SD_NOT_PRESENT** ((uint8_t)0x00)
#define	**SDMMCx_IRQHandler** SDMMC1_IRQHandler
#define	**SDMMCx_IRQn** SDMMC1_IRQn

Functions

uint8_t	**BSP_SD_Init** (void) Initializes the SD card device.
uint8_t	**BSP_SD_DeInit** (void) DeInitializes the SD card device.
uint8_t	**BSP_SD_ITConfig** (void) Configures Interrupt mode for SD detection pin.
uint8_t	**BSP_SD_ReadBlocks** (uint32_t *pData, uint32_t ReadAddr, uint32_t NumOfBlocks, uint32_t Timeout) Reads block(s) from a specified address in an SD card, in polling mode.
uint8_t	**BSP_SD_WriteBlocks** (uint32_t *pData, uint32_t WriteAddr, uint32_t NumOfBlocks, uint32_t Timeout) Writes block(s) to a specified address in an SD card, in polling mode.
uint8_t	**BSP_SD_ReadBlocks_DMA** (uint32_t *pData, uint32_t ReadAddr, uint32_t NumOfBlocks) Reads block(s) from a specified address in an SD card, in DMA mode.
uint8_t	**BSP_SD_WriteBlocks_DMA** (uint32_t *pData, uint32_t WriteAddr, uint32_t NumOfBlocks) Writes block(s) to a specified address in an SD card, in DMA mode.
uint8_t	**BSP_SD_Erase** (uint32_t StartAddr, uint32_t EndAddr) Erases the specified memory area of the given SD card.
uint8_t	**BSP_SD_GetCardState** (void) Gets the current SD card data status.
void	**BSP_SD_GetCardInfo** (**BSP_SD_CardInfo** *CardInfo) Get SD information about specific SD card.
uint8_t	**BSP_SD_IsDetected** (void) Detects if SD card is correctly plugged in the memory

	slot or not.
__weak void	**BSP_SD_MspInit** (SD_HandleTypeDef *hsd, void *Params) Initializes the SD MSP.
__weak void	**BSP_SD_MspDeInit** (SD_HandleTypeDef *hsd, void *Params) De-Initializes the SD MSP.
__weak void	**BSP_SD_AbortCallback** (void) BSP SD Abort callback.
__weak void	**BSP_SD_WriteCpltCallback** (void) BSP Tx Transfer completed callback.
__weak void	**BSP_SD_ReadCpltCallback** (void) BSP Rx Transfer completed callback.

Variables

SD_HandleTypeDef	**hsd_eval**

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_sd.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_sd.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_sram.c File Reference

This file provides a set of functions needed to drive the IS61WV102416BLL-10MLI SRAM memory mounted on STM32L4R9I-EVAL board. More...

```
#include "stm32l4r9i_eval_sram.h"
```

Go to the source code of this file.

Defines

#define	**SRAM_ADDR_SETUP_TIME** 2 /* 10ns with a clock at 120 MHz (period of 8.33 ns) */
#define	**SRAM_DATA_SETUP_TIME** 1 /* 6ns with a clock at 120 MHz (period of 8.33 ns) */
#define	**SRAM_DATA_HOLD_TIME** 0
#define	**SRAM_TURN_AROUND_TIME** 1 /* 4ns with a clock at 120 MHz (period of 8.33 ns) */

Functions

static void	**SRAM_MspInit** (void) Initializes SRAM MSP.
uint8_t	**BSP_SRAM_Init** (void) Initializes the SRAM device.
uint8_t	**BSP_SRAM_ReadData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the SRAM device in polling mode.
uint8_t	**BSP_SRAM_ReadData_DMA** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the SRAM device in DMA mode.
uint8_t	**BSP_SRAM_WriteData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data from the SRAM device in polling mode.
uint8_t	**BSP_SRAM_WriteData_DMA** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data from the SRAM device in DMA mode.
void	**BSP_SRAM_DMA_IRQHandler** (void) Handles SRAM DMA transfer interrupt request.

Variables

static SRAM_HandleTypeDef	**sramHandle**
static FMC_NORSRAM_TimingTypeDef	**Timing**

Detailed Description

This file provides a set of functions needed to drive the IS61WV102416BLL-10MLI SRAM memory mounted on STM32L4R9I-EVAL board.

Author:

MCD Application Team

```
  ==========================================
==================================
                     ##### How to use this dri
ver #####
  ==========================================
==================================
  [..]
   (#) This driver is used to drive the IS61WV
102416BLL-10MLI SRAM external memory mounted
       on STM32L4R9I-EVAL evaluation board.

   (#) This driver does not need a specific co
mponent driver for the SRAM device
       to be included with.

   (#) Initialization steps:
       (++) Initialize the SRAM external memor
y using the BSP_SRAM_Init() function. This
            function includes the MSP layer ha
rdware resources initialization and the
            FMC controller configuration to in
terface with the external SRAM memory.

   (#) SRAM read/write operations
       (++) SRAM external memory can be access
ed with read/write operations once it is
            initialized.
            Read/write operation can be perfor
```

```
med with AHB access using the functions
            BSP_SRAM_ReadData()/BSP_SRAM_Write
Data(), or by DMA transfer using the functions
            BSP_SRAM_ReadData_DMA()/BSP_SRAM_W
riteData_DMA().
       (++) The AHB access is performed with 1
6-bit width transaction, the DMA transfer
            configuration is fixed at single (
no burst) halfword transfer
            (see the SRAM_MspInit() static fun
ction).
       (++) User can implement his own functio
ns for read/write access with his desired
            configurations.
       (++) If interrupt mode is used for DMA
transfer, the function BSP_SRAM_DMA_IRQHandler
()
            is called in IRQ handler file, to
serve the generated interrupt once the DMA
            transfer is complete.
```

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_sram.c**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_sram.h File Reference

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_sram.c** driver. More...

```
#include "stm32l4xx_hal.h"
```

Go to the source code of this file.

Defines

#define	**SRAM_OK** 0x00 SRAM status structure definition.
#define	**SRAM_ERROR** 0x01
#define	**SRAM_DEVICE_ADDR** ((uint32_t)0x60000000)
#define	**SRAM_DEVICE_SIZE** ((uint32_t)0x200000) /* SRAM device 1024 words of 16-bits */
#define	**SRAM_MEMORY_WIDTH** FMC_NORSRAM_MEM_BUS_WI
#define	**SRAM_BURSTACCESS** FMC_BURST_ACCESS_MODE_DI
#define	**SRAM_WRITEBURST** FMC_WRITE_BURST_DISABLE
#define	**SRAM_DMAx_CLK_ENABLE** __HAL_RCC_DMA1_CLK_EN
#define	**SRAM_DMAx_CHANNEL** DMA1_Channel1
#define	**SRAM_DMAx_IRQn** DMA1_Channel1_IRQn
#define	**SRAM_DMAx_IRQHandler** DMA1_Channel1_IRQHandler

Functions

uint8_t	**BSP_SRAM_Init** (void) Initializes the SRAM device.
uint8_t	**BSP_SRAM_ReadData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the SRAM device in polling mode.
uint8_t	**BSP_SRAM_ReadData_DMA** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the SRAM device in DMA mode.
uint8_t	**BSP_SRAM_WriteData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data from the SRAM device in polling mode.
uint8_t	**BSP_SRAM_WriteData_DMA** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data from the SRAM device in DMA mode.
void	**BSP_SRAM_DMA_IRQHandler** (void) Handles SRAM DMA transfer interrupt request.

Detailed Description

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_sram.c** driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_sram.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ts.h File Reference

This file contains the common defines and functions prototypes for the stm32l4r9i_eval_ts.c driver. More...

```
#include "stm32l4r9i_eval.h" #include "stm32l4r9i_eval_lcd.h"
#include "stm32l4r9i_eval_io.h"
```

Go to the source code of this file.

Data Structures

struct	**TS_StateTypeDef** **TS_StateTypeDef** Define TS State structure. More...

Defines

#define	**TS_NO_IRQ_PENDING** ((uint8_t) 0)
#define	**TS_IRQ_PENDING** ((uint8_t) 1)
#define	**TS_SWAP_NONE** ((uint8_t) 0x01)
#define	**TS_SWAP_X** ((uint8_t) 0x02)
#define	**TS_SWAP_Y** ((uint8_t) 0x04)
#define	**TS_SWAP_XY** ((uint8_t) 0x08)
#define	**TS_ORIENTATION_PORTRAIT** ((uint8_t) 0)
#define	**TS_ORIENTATION_LANDSCAPE** ((uint8_t) 1)

Enumerations

enum	**TS_StatusTypeDef** { **TS_OK** = 0x00, **TS_ERROR** = 0x01, **TS_TIMEOUT** = 0x02, **TS_DEVICE_NOT_FOUND** = 0x03 } TS_StatusTypeDef Define BSP_TS_xxx() functions possible return value, when status is returned by those functions. More...

enum	**TS_GestureIdTypeDef** {
	GEST_ID_NO_GESTURE = 0x00, **GEST_ID_MOVE_UP** = 0x01, **GEST_ID_MOVE_RIGHT** = 0x02,
	GEST_ID_MOVE_DOWN = 0x03,
	GEST_ID_MOVE_LEFT = 0x04, **GEST_ID_ZOOM_IN** = 0x05, **GEST_ID_ZOOM_OUT** = 0x06,
	GEST_ID_SINGLE_CLICK = 0x07,
	GEST_ID_DOUBLE_CLICK = 0x08, **GEST_ID_NB_MAX** = 0x09
	}
	TS_GestureIdTypeDef Define Possible managed gesture identification values returned by touch screen driver. More...

enum	**TS_TouchEventTypeDef** {
	TOUCH_EVENT_NO_EVT = 0x00,
	TOUCH_EVENT_PRESS_DOWN = 0x01,
	TOUCH_EVENT_LIFT_UP = 0x02,
	TOUCH_EVENT_CONTACT = 0x03,
	TOUCH_EVENT_NB_MAX = 0x04
	}
	TS_TouchEventTypeDef Define Possible touch events kind as returned values by touch screen IC Driver. More...

Detailed Description

This file contains the common defines and functions prototypes for the stm32l4r9i_eval_ts.c driver.

Author:

MCD Application Team

Attention:

© COPYRIGHT(c) 2017 STMicroelectronics

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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

Definition in file **stm32l4r9i_eval_ts.h**.

STM32L4R9I_EVAL BSP User Manual

Modules

Here is a list of all modules:

- BSP
  - STM32L4R9I_EVAL
    - STM32L4R9I_EVAL Common
      - Private Types Definitions
      - Exported Defines
      - Exported Variables
      - Private Functions
      - Exported Functions
      - Bus Operations Functions
      - Exported Types
      - Exported Constants
        - LED Constants
        - BUTTON Constants
        - COM Constants
        - POTENTIOMETER Constants
        - BUS Constants
    - STM32L4R9I_EVAL AUDIO
      - Private Types
      - Private Macros
      - Private Variables
      - Exported Variables
      - Private Functions
      - Exported Types
      - Exported Constants
        - BSP Audio Out Option

        - BSP Audio Sample Rate
    - Exported Macros
    - Exported Functions
- STM32L4R9I_EVAL LCD
    - STM32L4R9I_EVAL DSI LCD
        - Private Macros
        - Exported Variables
        - Private Variables
        - Private FunctionPrototypes
        - Exported Functions
        - Private Functions
    - Exported Constants
    - Exported Types
    - STM32L4R9I_EVAL RGB LCD
        - Exported Constants
        - Exported Variables
        - Exported Functions
    - Exported Functions
- STM32L4R9I_EVAL TS
    - STM32L4R9I_EVAL DSI TS
        - Exported Functions
    - STM32L4R9I_EVAL RGB TS
        - Imported Variables
        - STM32L4R9I_EVAL_RGB_TS_Exported_Functi
    - Exported Constants
    - Exported Types
    - Exported Functions
- STM32L4R9I_EVAL EEPROM
    - Private Variables
    - Exported Functions
    - Exported Constants
- STM32L4R9I_EVAL IDD
    - Private Variables
    - Exported Functions
    - Exported Defines
- STM32L4R9I_EVAL IO
    - Private Variables

    - Exported Types
    - Exported Constants
    - Exported Functions
- STM32L4R9I_EVAL NOR
    - Private Constants
    - Private Variables
    - Private Functions
    - Exported Functions
    - Exported Constants
- STM32L4R9I_EVAL OSPI NOR
    - Private Constants
    - Private Variables
    - Private Functions
    - Exported Constants
    - Exported Types
    - Exported Functions
- STM32L4R9I_EVAL OSPI RAM
    - Private Constants
    - Private Variables
    - Private Functions
    - Exported Constants
    - Exported Functions
- STM32L4R9I_EVAL SD
    - Private Variables
    - Exported Types
    - Exported Constants
    - Exported Variables
    - Exported Functions
- STM32L4R9I_EVAL SRAM
    - Private Variables
    - Private Constants
    - Private Functions
    - Exported Functions
        - STM32L476G_EVAL_SRAM_Private_Functions
    - Exported Constants

User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Data Structures

Here are the data structures with brief descriptions:

AUDIO_IN_TypeDef	
AUDIO_OUT_TypeDef	
LCD_DrawPropTypeDef	
OSPI_NOR_Info	
Point	
TS_StateTypeDef	**TS_StateTypeDef** Define TS State structure

STM32L4R9I_EVAL BSP User Manual

File List

Here is a list of all files with brief descriptions:

stm32l4r9i_eval.c [code]	This file provides a set of firmware functions to manage Leds, push-button and COM ports for STM32L4R9I_EVAL
stm32l4r9i_eval.h [code]	This file contains definitions for STM32L4R9I_EVAL's LEDs, push-buttons and COM ports hardware resources
stm32l4r9i_eval_audio.c [code]	This file provides the Audio driver for the STM32L4R9I-EVAL evaluation board
stm32l4r9i_eval_audio.h [code]	This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_audio.c** driver
stm32l4r9i_eval_dsi_lcd.c [code]	This file includes the driver for DSI Liquid Crystal Display (LCD) module mounted on STM32L4R9I_EVAL evaluation board

stm32l4r9i_eval_dsi_lcd.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_dsi_lcd.c** driver
stm32l4r9i_eval_dsi_ts.c [code]	This file provides a set of functions needed to manage the Touch Screen on DSI LCD of STM32L4R9I-EVAL evaluation board
stm32l4r9i_eval_dsi_ts.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_dsi_ts.c** driver
stm32l4r9i_eval_eeprom.c [code]	This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory
stm32l4r9i_eval_eeprom.h [code]	This file contains all the functions prototypes for the **stm32l4r9i_eval_eeprom.c** firmware driver
stm32l4r9i_eval_gfxmmu_lut.h [code]	This file contains GFXMMU look-up-table for DSI round LCD
stm32l4r9i_eval_idd.c [code]	This file provides a set of firmware functions to manage the Idd measurement driver for STM32L4R9I_EVAL boar
stm32l4r9i_eval_idd.h [code]	Header file for **stm32l4r9i_eval_idd.c** module
stm32l4r9i_eval_io.c [code]	This file provides a set of

	functions needed to manage the IO pins on STM32L4R9I_EVAL evaluation board
stm32l4r9i_eval_io.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_io.c** driver
stm32l4r9i_eval_lcd.h [code]	This file contains alias for compatibility with DSI and TFT LCD
stm32l4r9i_eval_nor.c [code]	This file includes a standard driver for the M29W128GL70ZA6E NOR memories mounted on STM32L4R9I-EVAL board
stm32l4r9i_eval_nor.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_nor.c** drive
stm32l4r9i_eval_ospi_nor.c [code]	This file includes a standard driver for the MX25UM51245G OSPI memory mounted on STM32L4R9I-EVAL board
stm32l4r9i_eval_ospi_nor.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_ospi_nor.c** driver
stm32l4r9i_eval_ospi_ram.c [code]	This file includes a standard driver for the ISS66WVH8M8 HyperRAM memory mounted on STM32L4R9I-EVAL board

stm32l4r9i_eval_ospi_ram.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_ospi_ram.** driver
stm32l4r9i_eval_rgb_lcd.c [code]	This file includes the driver fo Liquid Crystal Display (LCD) module mounted on STM32L4R9I_EVAL board
stm32l4r9i_eval_rgb_lcd.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_rgb_lcd.c** driver
stm32l4r9i_eval_rgb_ts.c [code]	This file provides a set of functions needed to manage the Touch Screen on STM32746G-Discovery boar
stm32l4r9i_eval_rgb_ts.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_rgb_ts.c** driver
stm32l4r9i_eval_sd.c [code]	This file includes the uSD card driver
stm32l4r9i_eval_sd.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_sd.c** driver
stm32l4r9i_eval_sram.c [code]	This file provides a set of functions needed to drive the IS61WV102416BLL-10MLI SRAM memory mounted on STM32L4R9I-EVAL board

stm32l4r9i_eval_sram.h [code]	This file contains the commo defines and functions prototypes for the **stm32l4r9i_eval_sram.c** driver
stm32l4r9i_eval_ts.h [code]	This file contains the commo defines and functions prototypes for the stm32l4r9i_eval_ts.c driver

STM32L4R9I_EVAL BSP User Manual

Directories

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

- Firmware
  - Drivers
    - BSP
      - STM32L4R9I_EVAL

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL Common

STM32L4R9I_EVAL

Modules

Private Types Definitions

This file provides firmware functions to manage Leds, push-buttons, Joystick, and COM ports available on STM32L4R9I-EVAL evaluation board from STMicroelectronics.

Exported Defines

Exported Variables

Private Functions

Exported Functions

Bus Operations Functions

Exported Types

Exported Constants

STM32L4R9I_EVAL BSP User Manual

Data Structure Index

A | L | O | P | T

A

AUDIO_IN_TypeDef
AUDIO_OUT_TypeDef

L

LCD_DrawPropTypeDef

O

OSPI_NOR_Info

P

Point

T

TS_StateTypeDe

A | L | O | P | T

STM32L4R9I_EVAL BSP User Manual

Private Types

STM32L4R9I_EVAL AUDIO

Data Structures

struct	**AUDIO_OUT_TypeDef**
struct	**AUDIO_IN_TypeDef**

STM32L4R9I_EVAL BSP User Manual

Exported Types

STM32L4R9I_EVAL AUDIO

Typedefs

typedef void(* **Audio_CallbackTypeDef**)(void)

Typedef Documentation

typedef void(* Audio_CallbackTypeDef)(void)

Definition at line **70** of file **stm32l4r9i_eval_audio.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_audio.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_audio.c
00004   * @author  MCD Application Team
00005   * @brief   This file provides the Audio driver for the STM32L4R9I-EVAL evaluation board.
00006   ******************************************************************************
00007   * @attention
00008   *
00009   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00010   *
00011   * Redistribution and use in source and binary forms, with or without modification,
00012   * are permitted provided that the following conditions are met:
00013   *   1. Redistributions of source code must retain the above copyright notice,
00014   *      this list of conditions and the following disclaimer.
```

```
00015   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00016   *      this list of conditions and the fol
lowing disclaimer in the documentation
00017   *      and/or other materials provided wit
h the distribution.
00018   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00019   *      may be used to endorse or promote p
roducts derived from this software
00020   *      without specific prior written perm
ission.
00021   *
00022   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00023   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00024   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00025   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00026   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00027   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00028   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00029   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00030   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00031   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00032   *
00033   ****************************************
**********************************
00034   */
```

```
/*==========================================================================
                                    User NOTES

1. How To use this driver:
--------------------------
   + This driver supports STM32L4xx devices on STM32L4R9I-EVAL (MB1313) Evaluation boards.
         a) to play an audio file (all functions names start by BSP_AUDIO_OUT_xxx)
         b) to record an audio file through MP34DT01TR, ST MEMS (all functions names start by BSP_AUDIO_IN_xxx)

a) PLAY A FILE:
==============
   + Call the function BSP_AUDIO_OUT_Init(
                                        OutputDevice: physical output mode (OUTPUT_DEVICE_SPEAKER,

      OUTPUT_DEVICE_HEADPHONE or OUTPUT_DEVICE_BOTH)
                                        Volume    : Initial volume to be set (0 is min (mute), 100 is max (100%)
                                        AudioFreq   : Audio frequency in Hz (8000, 16000, 22500, 32000...)

      this parameter is relative to the audio file/stream type.
                                       )
       This function configures all the hardware required for the audio application (codec, I2C, SAI,
       GPIOs, DMA and interrupt if needed). T
```

```
his function returns AUDIO_OK if configuration is
OK.
00056       If the returned value is different fro
m AUDIO_OK or the function is stuck then the commu
nication with
00057       the codec or the IOExpander has failed
 (try to un-plug the power or reset device in this
 case).
00058       - OUTPUT_DEVICE_SPEAKER  : only speake
r will be set as output for the audio stream.
00059       - OUTPUT_DEVICE_HEADPHONE: only headph
ones will be set as output for the audio stream.
00060       - OUTPUT_DEVICE_BOTH     : both Speake
r and Headphone are used as outputs for the audio
stream
00061                                   at the same
 time.
00062    + Call the function BSP_AUDIO_OUT_Registe
rCallbacks to register user callbacks
00063      required to manage audio data streaming
 towards the audio codec (ErrorCallback(),
00064      HalfTransfer_CallBack() and TransferCom
plete_CallBack()).
00065    + Call the function BSP_AUDIO_OUT_Play()
to start audio playback (for the first time).
00066    + Call the function BSP_AUDIO_OUT_Pause()
 to pause audio playback.
00067    + Call the function BSP_AUDIO_OUT_Resume(
) to resume audio playback.
00068        Note. After calling BSP_AUDIO_OUT_Pau
se() function for pause, only BSP_AUDIO_OUT_Resume
() should be called
00069           for resume (it is not allowed to c
all BSP_AUDIO_OUT_Play() in this case).
00070        Note. This function should be called
only when the audio file is played or paused (not
stopped).
```

```
00071    + Call the function BSP_AUDIO_OUT_Stop()
to stop audio playback.
00072    + To modify the volume level, the samplin
g frequency, the device output mode,
00073       the mute status or the audio configura
tion or the stop, use the functions: BSP_AUDIO_OUT
_SetVolume(),
00074       AUDIO_OUT_SetFrequency(), BSP_AUDIO_OU
T_SetOutputMode(), BSP_AUDIO_OUT_SetMute()and
00075       BSP_AUDIO_OUT_ChangeAudioConfig().
00076
00077 Driver architecture:
00078 --------------------
00079    + This driver provides the audio layer hi
gh level API: it consists in functions
00080      exported in the stm32l4r9i_eval_audio.h
 file (e.g. BSP_AUDIO_OUT_Init(),
00081      BSP_AUDIO_OUT_Play(), ...).
00082    + This driver also includes the Media Acc
ess Layer (MAL): it consists in
00083      functions allowing to access setup the
audio devices. These functions
00084      are  included as local functions into t
he stm32l4r9i_eval_audio.c file
00085      (e.g. AUDIO_SAIx_Init()).
00086
00087 Known Limitations:
00088 ------------------
00089    1- Communication with the audio codec (th
rough I2C) may be corrupted if it is interrupted b
y some
00090       user interrupt routines (in this case,
 interrupts could be disabled just before the star
t of
00091       communication then re-enabled when it
is over). Note that this communication is only don
e at
```

```
00092       the configuration phase (BSP_AUDIO_OUT
_Init() or BSP_AUDIO_OUT_Stop()) and when Volume c
ontrol modification is
00093       performed (BSP_AUDIO_OUT_SetVolume() o
r BSP_AUDIO_OUT_SetMute()or BSP_AUDIO_OUT_SetOutpu
tMode()).
00094       When the audio data is played, no comm
unication is required with the audio codec.
00095    2- Parsing of audio file is not implement
ed (in order to determine audio file properties: M
ono/Stereo, Data size,
00096       File size, Audio Frequency, Audio Data
 header size ...). The configuration is fixed for
the given audio file.
00097    3- Supports only 16-bits audio data size.
00098
00099 b) RECORD A FILE:
00100 ================
00101    + Call the function BSP_AUDIO_IN_Init(
00102                                          AudioFre
q: Audio frequency in Hz (8000, 16000, 22500, 3200
0 ...)
00103                                          )
00104       This function configures all the hardw
are required for the audio application (DFSDM,
00105       GPIOs, DMA and interrupt if needed). T
his function returns AUDIO_OK if the
00106       configuration completes successfully.
00107
00108    + Call the function BSP_AUDIO_IN_Register
Callbacks to register user callbacks
00109      used to stream audio data toward the re
cord buffer (ErrorCallback(),
00110      HalfTransfer_CallBack() and TransferCom
plete_CallBack()).
00111
00112    + Call the function BSP_AUDIO_IN_Record(
```

```
00113                                  pbuf Main buffer
 pointer for the recorded data storing
00114                                  size Current siz
e of the recorded buffer
00115                                  )
00116       to start recording from the microphone.

00117
00118    + Call the function BSP_AUDIO_IN_Stop() t
o stop recording.
00119 ==========================================
==================================*/
00120
00121 /* Includes ------------------------------
-----------------------------------*/
00122 #include <string.h>
00123 #include "stm32l4r9i_eval_audio.h"
00124
00125 /** @addtogroup BSP
00126   * @{
00127   */
00128
00129 /** @addtogroup STM32L4R9I_EVAL
00130   * @{
00131   */
00132
00133 /** @defgroup STM32L4R9I_EVAL_AUDIO STM32L4R
9I_EVAL AUDIO
00134   * @brief This file includes the low layer
driver for wm8994 Audio Codec
00135   *        available on STM32L4R9I-EVAL eval
uation board(MB1313).
00136   * @{
00137   */
00138
00139 /* Private typedef ------------------------
-----------------------------------*/
```

```
00140 /** @defgroup STM32L4R9I_EVAL_AUDIO_Private_
Types Private Types
00141   * @{
00142   */
00143 typedef struct
00144 {
00145   AUDIO_DrvTypeDef *    AudioDrv;
/* Audio codec driver */
00146   uint32_t              OutputDevice;
/* Output device */
00147   uint32_t              Frequency;
/* Playback frequency */
00148   Audio_CallbackTypeDef CbError;
/* pointer to the callback function invoked when .
.. */
00149   Audio_CallbackTypeDef CbHalfTransfer;
/* pointer to the callback function invoked when .
.. */
00150   Audio_CallbackTypeDef CbTransferComplete;
/* pointer to the callback function invoked when .
.. */
00151 } AUDIO_OUT_TypeDef;
00152
00153 typedef struct
00154 {
00155   AUDIO_DrvTypeDef *          AudioDrv;
      /* Audio codec driver */
00156   DFSDM_Channel_HandleTypeDef hDfsdmLeftChan
nel;  /* DFSDM channel handle used for left channe
l */
00157   DFSDM_Channel_HandleTypeDef hDfsdmRightCha
nnel; /* DFSDM channel handle used for right chann
el */
00158   DMA_HandleTypeDef           hDmaDfsdmLeft;
      /* DMA handle used for DFSDM regular convers
ions on left channel */
00159   DMA_HandleTypeDef           hDmaDfsdmRight
```

```
;     /* DMA handle used for DFSDM regular convers
ions on right channel */
00160   int32_t *                   LeftRecBuff;
      /* Buffers for left samples */
00161   int32_t *                   RightRecBuff;
      /* Buffers for right samples */
00162   uint32_t                    InputDevice;
      /* Input device */
00163   uint32_t                    Frequency;
      /* Record Frequency */
00164   uint32_t                    BitResolution;
      /* Record bit resolution */
00165   uint32_t                    ChannelNbr;
      /* Record Channel Number */
00166   uint16_t *                  pRecBuf;
      /* Pointer to record user buffer */
00167   uint32_t                    RecSize;
      /* Size to record in mono, double size to re
cord in stereo */
00168   Audio_CallbackTypeDef       CbError;
      /* pointer to the callback function invoked
when a DMA transfer fails */
00169   Audio_CallbackTypeDef       CbHalfTransfer
;     /* pointer to the callback function invoked
when half of the DMA transfer is completed */
00170   Audio_CallbackTypeDef       CbTransferComp
lete; /* pointer to the callback function invoked
when the DMA transfer is completed */
00171 } AUDIO_IN_TypeDef;
00172 /**
00173   * @}
00174   */
00175
00176 /* Private macros ------------------------
-----------------------------------*/
00177 /** @defgroup STM32L4R9I_EVAL_AUDIO_Private_
Macros Private Macros
```

```
00178   * @{
00179   */
00180 /*### PLAY ###*/
00181 /* SCK(kHz) = SAI_CK_x/(SAIClockDivider*2*25
6) */
00182 #define SAIClockDivider(__FREQUENCY__) \
00183         (__FREQUENCY__ == AUDIO_FREQUENCY_8K
)  ? 24 \
00184       : (__FREQUENCY__ == AUDIO_FREQUENCY_11
K) ? 4 \
00185       : (__FREQUENCY__ == AUDIO_FREQUENCY_16
K) ? 12 \
00186       : (__FREQUENCY__ == AUDIO_FREQUENCY_22
K) ? 2 \
00187       : (__FREQUENCY__ == AUDIO_FREQUENCY_32
K) ? 6 \
00188       : (__FREQUENCY__ == AUDIO_FREQUENCY_44
K) ? 1 \
00189       : (__FREQUENCY__ == AUDIO_FREQUENCY_48
K) ? 4 : 2  \
00190
00191 /*### RECORD ###*/
00192 #define DFSDMOverSampling(__FREQUENCY__) \
00193         (__FREQUENCY__ == AUDIO_FREQUENCY_8K
)  ? 256 \
00194       : (__FREQUENCY__ == AUDIO_FREQUENCY_11
K) ? 256 \
00195       : (__FREQUENCY__ == AUDIO_FREQUENCY_16
K) ? 128 \
00196       : (__FREQUENCY__ == AUDIO_FREQUENCY_22
K) ? 128 \
00197       : (__FREQUENCY__ == AUDIO_FREQUENCY_32
K) ? 64 \
00198       : (__FREQUENCY__ == AUDIO_FREQUENCY_44
K) ? 64  \
00199       : (__FREQUENCY__ == AUDIO_FREQUENCY_48
K) ? 32 : 16  \
```

```
00200
00201 #define DFSDMClockDivider(__FREQUENCY__) \
00202         (__FREQUENCY__ == AUDIO_FREQUENCY_8K
)  ? 24 \
00203       : (__FREQUENCY__ == AUDIO_FREQUENCY_11
K) ? 4 \
00204       : (__FREQUENCY__ == AUDIO_FREQUENCY_16
K) ? 24 \
00205       : (__FREQUENCY__ == AUDIO_FREQUENCY_22
K) ? 4 \
00206       : (__FREQUENCY__ == AUDIO_FREQUENCY_32
K) ? 24 \
00207       : (__FREQUENCY__ == AUDIO_FREQUENCY_44
K) ? 4  \
00208       : (__FREQUENCY__ == AUDIO_FREQUENCY_48
K) ? 32 : 32  \
00209
00210 #define DFSDMFilterOrder(__FREQUENCY__) \
00211         (__FREQUENCY__ == AUDIO_FREQUENCY_8K
)  ? DFSDM_FILTER_SINC3_ORDER \
00212       : (__FREQUENCY__ == AUDIO_FREQUENCY_11
K) ? DFSDM_FILTER_SINC3_ORDER \
00213       : (__FREQUENCY__ == AUDIO_FREQUENCY_16
K) ? DFSDM_FILTER_SINC3_ORDER \
00214       : (__FREQUENCY__ == AUDIO_FREQUENCY_22
K) ? DFSDM_FILTER_SINC3_ORDER \
00215       : (__FREQUENCY__ == AUDIO_FREQUENCY_32
K) ? DFSDM_FILTER_SINC4_ORDER \
00216       : (__FREQUENCY__ == AUDIO_FREQUENCY_44
K) ? DFSDM_FILTER_SINC4_ORDER  \
00217       : (__FREQUENCY__ == AUDIO_FREQUENCY_48
K) ? DFSDM_FILTER_SINC4_ORDER : DFSDM_FILTER_SINC5
_ORDER  \
00218
00219 #define DFSDMRightBitShift(__FREQUENCY__) \
00220         (__FREQUENCY__ == AUDIO_FREQUENCY_8K
)  ? 5 \
```

```
00221       : (__FREQUENCY__ == AUDIO_FREQUENCY_11
K) ? 5 \
00222       : (__FREQUENCY__ == AUDIO_FREQUENCY_16
K) ? 2 \
00223       : (__FREQUENCY__ == AUDIO_FREQUENCY_22
K) ? 2 \
00224       : (__FREQUENCY__ == AUDIO_FREQUENCY_32
K) ? 5 \
00225       : (__FREQUENCY__ == AUDIO_FREQUENCY_44
K) ? 5  \
00226       : (__FREQUENCY__ == AUDIO_FREQUENCY_48
K) ? 1 : 2  \
00227
00228 /* Saturate the record PCM sample */
00229 #define SaturaLH(N, L, H) (((N)<(L))?(L):(((
N)>(H))?(H):(N)))
00230
00231 /**
00232   * @}
00233   */
00234
00235 /* Private variables ----------------------
----------------------------------*/
00236 /** @defgroup STM32L4R9I_EVAL_AUDIO_Private_
Variables Private Variables
00237   * @{
00238   */
00239 /* Audio output context information */
00240 static AUDIO_OUT_TypeDef hAudioOut = {0};
00241
00242 /* Audio input context information */
00243 static AUDIO_IN_TypeDef hAudioIn = {0};
00244
00245 /* SAI DMA handle */
00246 static DMA_HandleTypeDef hDmaSai;
00247
00248 static uint32_t DmaLeftRecHalfBuffCplt;
```

```
00249 static uint32_t DmaLeftRecBuffCplt;
00250 static uint32_t DmaRightRecHalfBuffCplt;
00251 static uint32_t DmaRightRecBuffCplt;
00252
00253 /**
00254   * @}
00255   */
00256
00257 /* Exported variables ---------------------
---------------------------------*/
00258 /** @defgroup STM32L4R9I_EVAL_AUDIO_Exported
_Variables Exported Variables
00259   * @{
00260   */
00261 /* SAIx handle */
00262 SAI_HandleTypeDef               BSP_AUDIO_hS
ai;
00263
00264 /* DFSDM filter handle */
00265 DFSDM_Filter_HandleTypeDef      BSP_AUDIO_hD
fsdmLeftFilter;
00266 DFSDM_Filter_HandleTypeDef      BSP_AUDIO_hD
fsdmRightFilter;
00267 /**
00268   * @}
00269   */
00270
00271 /* Private function prototypes ------------
---------------------------------*/
00272 /** @defgroup STM32L4R9I_EVAL_AUDIO_Private_
Functions Private Functions
00273   * @{
00274   */
00275 static void    AUDIO_CODEC_Reset(void);
00276 static uint8_t AUDIO_SAIx_Init(uint32_t Audi
oFreq, uint16_t OutputDevice);
00277 static uint8_t AUDIO_SAIx_DeInit(void);
```

```
00278 static uint8_t AUDIO_DFSDMx_Init(uint32_t Au
dioFreq);
00279 static uint8_t AUDIO_DFSDMx_DeInit(void);
00280 static uint8_t AUDIO_SAIPLLConfig(uint32_t A
udioFreq);
00281 /**
00282   * @}
00283   */
00284
00285 /* Exported functions ---------------------
-----------------------------------*/
00286 /** @addtogroup STM32L4R9I_EVAL_AUDIO_Export
ed_Functions
00287   * @{
00288   */
00289
00290 /**
00291   * @brief  Configures the audio peripherals.

00292   * @param  OutputDevice: OUTPUT_DEVICE_SPEA
KER, OUTPUT_DEVICE_HEADPHONE,
00293   *                       or OUTPUT_DEVICE_B
OTH.
00294   * @param  Volume: Initial volume level (fr
om 0 (Mute) to 100 (Max))
00295   * @param  AudioFreq: Audio frequency used
to play the audio stream.
00296   * @retval BSP AUDIO status
00297   * @note   The SAI PLL input clock must be
configure in the user application.
00298   *         The SAI PLL configuration done w
ithin this function assumes that
00299   *         the SAI PLL input clock runs at
4 MHz.
00300   */
00301 uint8_t BSP_AUDIO_OUT_Init(uint16_t OutputDe
vice, uint8_t Volume, uint32_t AudioFreq)
```

```
00302 {
00303   /* Initialize the audio output context */
00304   hAudioOut.AudioDrv           = &wm8994_drv
;
00305   hAudioOut.OutputDevice       = OutputDevic
e;
00306   hAudioOut.Frequency          = AudioFreq;
00307   hAudioOut.CbError            = (Audio_Call
backTypeDef)NULL;
00308   hAudioOut.CbHalfTransfer     = (Audio_Call
backTypeDef)NULL;
00309   hAudioOut.CbTransferComplete = (Audio_Call
backTypeDef)NULL;
00310
00311   /* Check if input device is currently used
 */
00312   if(hAudioIn.InputDevice != 0)
00313   {
00314     /* If input device is currently used, SA
I PLL is already initialized */
00315     /* Check that AudioFreq for record and p
layback is the same */
00316     if(hAudioIn.Frequency != hAudioOut.Frequ
ency)
00317     {
00318       return AUDIO_ERROR;
00319     }
00320   }
00321   else
00322   {
00323     /* Configure the SAI PLL according to th
e requested audio frequency */
00324     if (AUDIO_SAIPLLConfig(AudioFreq) != AUD
IO_OK)
00325     {
00326       return AUDIO_ERROR;
00327     }
```

```
00328   }
00329
00330   /* SAI data transfer preparation: prepare
the Media to be used for the audio
00331      transfer from memory to SAI peripheral.
 */
00332   if (AUDIO_SAIx_Init(AudioFreq, OutputDevic
e) != AUDIO_OK)
00333   {
00334     return AUDIO_ERROR;
00335   }
00336
00337   /* Retieve audio codec identifier */
00338   if (wm8994_drv.ReadID(AUDIO_I2C_ADDRESS) !
= WM8994_ID)
00339   {
00340     return AUDIO_ERROR;
00341   }
00342
00343   /* Reset the audio codec Registers */
00344   AUDIO_CODEC_Reset();
00345
00346   /* Initialize the audio codec internal reg
isters */
00347   if (hAudioOut.AudioDrv->Init(AUDIO_I2C_ADD
RESS,
00348                                OutputDevice,
00349                                Volume,
00350                                AudioFreq) !=
 0)
00351   {
00352     return AUDIO_ERROR;
00353   }
00354
00355   /* Set the requested volume */
00356   BSP_AUDIO_OUT_SetVolume(Volume);
00357
```

```
00358   return AUDIO_OK;
00359 }
00360
00361 /**
00362   * @brief  De-Initializes audio codec relat
ed peripherals
00363   * @retval BSP AUDIO status
00364
00365   */
00366 uint8_t BSP_AUDIO_OUT_DeInit(void)
00367 {
00368   /* De-initializes the SAI interface */
00369   if (AUDIO_SAIx_DeInit() != AUDIO_OK)
00370   {
00371     return AUDIO_ERROR;
00372   }
00373
00374   /* DeInit audio codec */
00375   hAudioOut.AudioDrv->DeInit();
00376
00377   /* Reset the audio output context */
00378   memset(&hAudioOut, 0, sizeof(hAudioOut));
00379
00380   return AUDIO_OK;
00381 }
00382
00383 /**
00384   * @brief  Starts playing audio stream from
 a data buffer for a determined size.
00385   * @param  pData: pointer on PCM samples bu
ffer
00386   * @param  Size: Number of audio data HALF
WORD.
00387   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00388   */
00389 uint8_t BSP_AUDIO_OUT_Play(uint16_t* pData,
```

```
uint32_t Size)
00390 {
00391   /* Initiate a DMA transfer of PCM samples
towards the serial audio interface */
00392   if (HAL_SAI_Transmit_DMA(&BSP_AUDIO_hSai,
(uint8_t *)pData, DMA_MAX(Size))!= HAL_OK)
00393   {
00394     return AUDIO_ERROR;
00395   }
00396
00397   /* Call the audio Codec Play function */
00398   if (hAudioOut.AudioDrv->Play(AUDIO_I2C_ADD
RESS, pData, Size) != 0)
00399   {
00400     return AUDIO_ERROR;
00401   }
00402
00403   return AUDIO_OK;
00404 }
00405
00406 /**
00407   * @brief  Sends n-Bytes on the SAI interfa
ce.
00408   * @param  pData: pointer on data address
00409   * @param  Size: number of data to be writt
en
00410   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00411   */
00412 uint8_t BSP_AUDIO_OUT_ChangeBuffer(uint16_t
*pData, uint16_t Size)
00413 {
00414   /* Initiate a DMA transfer of PCM samples
towards the serial audio interface */
00415   if (HAL_SAI_Transmit_DMA(&BSP_AUDIO_hSai,
(uint8_t *)pData, Size) != HAL_OK)
00416   {
```

```
00417     return AUDIO_ERROR;
00418   }
00419
00420   return AUDIO_OK;
00421 }
00422
00423 /**
00424   * @brief  This function Pauses the audio f
ile stream. In case
00425   *         of using DMA, the DMA Pause feat
ure is used.
00426   * @note When calling BSP_AUDIO_OUT_Pause()
 function for pause, only
00427   *       BSP_AUDIO_OUT_Resume() function sh
ould be called for resume
00428   *       (use of BSP_AUDIO_OUT_Play() funct
ion for resume could lead
00429   *       to unexpected behavior).
00430   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00431   */
00432 uint8_t BSP_AUDIO_OUT_Pause(void)
00433 {
00434   /* Call the Audio Codec Pause function */
00435   if (hAudioOut.AudioDrv->Pause(AUDIO_I2C_AD
DRESS) != 0)
00436   {
00437     return AUDIO_ERROR;
00438   }
00439
00440   /* Pause DMA transfer of PCM samples towar
ds the serial audio interface */
00441   if (HAL_SAI_DMAPause(&BSP_AUDIO_hSai)!= HA
L_OK)
00442   {
00443     return AUDIO_ERROR;
00444   }
```

```
00445
00446   return AUDIO_OK;
00447 }
00448
00449 /**
00450   * @brief  This function  Resumes the audio
 file stream.
00451   * @note When calling BSP_AUDIO_OUT_Pause()
 function for pause, only
00452   *       BSP_AUDIO_OUT_Resume() function sh
ould be called for resume
00453   *       (use of BSP_AUDIO_OUT_Play() funct
ion for resume could lead to
00454   *       unexpected behavior).
00455   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00456   */
00457 uint8_t BSP_AUDIO_OUT_Resume(void)
00458 {
00459   /* Call the Audio Codec Resume function */
00460   if (hAudioOut.AudioDrv->Resume(AUDIO_I2C_A
DDRESS) != 0)
00461   {
00462     return AUDIO_ERROR;
00463   }
00464
00465   /* Resume DMA transfer of PCM samples towa
rds the serial audio interface */
00466   if (HAL_SAI_DMAResume(&BSP_AUDIO_hSai) !=
HAL_OK)
00467   {
00468     return AUDIO_ERROR;
00469   }
00470
00471   return AUDIO_OK;
00472 }
00473
```

```
00474 /**
00475   * @brief  Stops audio playing and Power do
wn the Audio Codec.
00476   * @param  Option: could be one of the foll
owing parameters
00477   *           - CODEC_PDWN_SW: for software
power off (by writing registers).
00478   *                                Then no need
to reconfigure the Codec after power on.
00479   *           - CODEC_PDWN_HW: completely sh
ut down the codec (physically).
00480   *                                Then need to
reconfigure the Codec after power on.
00481   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00482   */
00483 uint8_t BSP_AUDIO_OUT_Stop(uint32_t Option)
00484 {
00485   /* Call Audio Codec Stop function */
00486   if (hAudioOut.AudioDrv->Stop(AUDIO_I2C_ADD
RESS, Option) != 0)
00487   {
00488     return AUDIO_ERROR;
00489   }
00490
00491   if(Option == CODEC_PDWN_HW)
00492   {
00493     /* Wait at least 100us */
00494     HAL_Delay(1);
00495   }
00496
00497   /* Stop DMA transfer of PCM samples toward
s the serial audio interface */
00498   if (HAL_SAI_DMAStop(&BSP_AUDIO_hSai)!= HAL
_OK)
00499   {
00500     return AUDIO_ERROR;
```

```
00501   }
00502
00503   return AUDIO_OK;
00504 }
00505
00506 /**
00507   * @brief  Controls the current audio volum
e level.
00508   * @param  Volume: Volume level to be set i
n percentage from 0% to 100% (0 for
00509   *         Mute and 100 for Max volume leve
l).
00510   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00511   */
00512 uint8_t BSP_AUDIO_OUT_SetVolume(uint8_t Volu
me)
00513 {
00514   /* Call the codec volume control function
with converted volume value */
00515   if (hAudioOut.AudioDrv->SetVolume(AUDIO_I2
C_ADDRESS, Volume) != 0)
00516   {
00517     return AUDIO_ERROR;
00518   }
00519
00520   return AUDIO_OK;
00521 }
00522
00523 /**
00524   * @brief  Enables or disables the MUTE mod
e by software
00525   * @param  Cmd: Could be AUDIO_MUTE_ON to m
ute sound or AUDIO_MUTE_OFF to
00526   *         unmute the codec and restore pre
vious volume level.
00527   * @retval AUDIO_OK if correct communicatio
```

```
n, else wrong communication
00528   */
00529 uint8_t BSP_AUDIO_OUT_SetMute(uint32_t Cmd)
00530 {
00531   /* Call the Codec Mute function */
00532   if (hAudioOut.AudioDrv->SetMute(AUDIO_I2C_
ADDRESS, Cmd) != 0)
00533   {
00534     return AUDIO_ERROR;
00535   }
00536
00537   return AUDIO_OK;
00538 }
00539
00540 /**
00541   * @brief  Switch dynamically (while audio
file is played) the output target
00542   *         (speaker or headphone).
00543   * @param  Output: The audio output target:
 OUTPUT_DEVICE_SPEAKER,
00544   *         OUTPUT_DEVICE_HEADPHONE or OUTPU
T_DEVICE_BOTH
00545   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00546   */
00547 uint8_t BSP_AUDIO_OUT_SetOutputMode(uint8_t
Output)
00548 {
00549   /* Call the Codec output device function */

00550   if (hAudioOut.AudioDrv->SetOutputMode(AUDI
O_I2C_ADDRESS, Output) != 0)
00551   {
00552     return AUDIO_ERROR;
00553   }
00554
00555   return AUDIO_OK;
```

```
00556 }
00557
00558 /**
00559   * @brief  Update the audio frequency.
00560   * @param  AudioFreq: Audio frequency used
to play the audio stream.
00561   * @note   This API should be called after
the BSP_AUDIO_OUT_Init() to adjust the
00562   *         audio frequency.
00563   * @note   The SAI PLL configuration done w
ithin this function assumes that
00564   *         the SAI PLL input clock runs at
4 MHz.
00565   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00566   */
00567 uint8_t BSP_AUDIO_OUT_SetFrequency(uint32_t
AudioFreq)
00568 {
00569   /* Configure the SAI PLL according to the
requested audio frequency */
00570   if (AUDIO_SAIPLLConfig(AudioFreq) != AUDIO
_OK)
00571   {
00572     return AUDIO_ERROR;
00573   }
00574
00575   /* Disable SAI peripheral to allow access
to SAI internal registers */
00576   __HAL_SAI_DISABLE(&BSP_AUDIO_hSai);
00577
00578   /* Update the SAI audio frequency configur
ation */
00579   BSP_AUDIO_hSai.Init.Mckdiv = SAIClockDivid
er(AudioFreq);
00580   HAL_SAI_Init(&BSP_AUDIO_hSai);
00581
```

```
00582   /* Enable SAI peripheral to generate MCLK
*/
00583   __HAL_SAI_ENABLE(&BSP_AUDIO_hSai);
00584
00585   hAudioOut.Frequency = AudioFreq;
00586
00587   return AUDIO_OK;
00588 }
00589
00590 /**
00591   * @brief  Changes the Audio Out Configurat
ion.
00592   * @param  AudioOutOption: specifies the au
dio out new configuration
00593   *         This parameter can be any value
of @ref BSP_Audio_Out_Option
00594   * @note   This API should be called after
the BSP_AUDIO_OUT_Init() to adjust the
00595   *         audio out configuration.
00596   * @retval None
00597   */
00598 void BSP_AUDIO_OUT_ChangeAudioConfig(uint32_
t AudioOutOption)
00599 {
00600   /********** Playback Buffer circular/norma
l mode **********/
00601   if(AudioOutOption & BSP_AUDIO_OUT_CIRCULAR
MODE)
00602   {
00603     /* Deinitialize the Stream to update DMA
 mode */
00604     HAL_DMA_DeInit(BSP_AUDIO_hSai.hdmatx);
00605
00606     /* Update the SAI audio Transfer DMA mod
e */
00607     BSP_AUDIO_hSai.hdmatx->Init.Mode = DMA_C
IRCULAR;
```

```
00608
00609     /* Configure the DMA Stream with new Tra
nsfer DMA mode */
00610     HAL_DMA_Init(BSP_AUDIO_hSai.hdmatx);
00611   }
00612   else /* BSP_AUDIO_OUT_NORMALMODE */
00613   {
00614     /* Deinitialize the Stream to update DMA
 mode */
00615     HAL_DMA_DeInit(BSP_AUDIO_hSai.hdmatx);
00616
00617     /* Update the SAI audio Transfer DMA mod
e */
00618     BSP_AUDIO_hSai.hdmatx->Init.Mode = DMA_N
ORMAL;
00619
00620     /* Configure the DMA Stream with new Tra
nsfer DMA mode */
00621     HAL_DMA_Init(BSP_AUDIO_hSai.hdmatx);
00622   }
00623
00624   /********** Playback Buffer stereo/mono mo
de **********/
00625   if(AudioOutOption & BSP_AUDIO_OUT_STEREOMO
DE)
00626   {
00627     /* Disable SAI peripheral to allow acces
s to SAI internal registers */
00628     __HAL_SAI_DISABLE(&BSP_AUDIO_hSai);
00629
00630     /* Update the SAI audio frame slot confi
guration */
00631     BSP_AUDIO_hSai.Init.MonoStereoMode = SAI
_STEREOMODE;
00632     HAL_SAI_Init(&BSP_AUDIO_hSai);
00633
00634     /* Enable SAI peripheral to generate MCL
```

```
K */
00635     __HAL_SAI_ENABLE(&BSP_AUDIO_hSai);
00636   }
00637   else /* BSP_AUDIO_OUT_MONOMODE */
00638   {
00639     /* Disable SAI peripheral to allow acces
s to SAI internal registers */
00640     __HAL_SAI_DISABLE(&BSP_AUDIO_hSai);
00641
00642     /* Update the SAI audio frame slot confi
guration */
00643     BSP_AUDIO_hSai.Init.MonoStereoMode = SAI
_MONOMODE;
00644     HAL_SAI_Init(&BSP_AUDIO_hSai);
00645
00646     /* Enable SAI peripheral to generate MCL
K */
00647     __HAL_SAI_ENABLE(&BSP_AUDIO_hSai);
00648   }
00649 }
00650
00651 /**
00652   * @brief  register user callback functions
00653   * @param  ErrorCallback: pointer to the er
ror callback function
00654   * @param  HalfTransferCallback: pointer to
 the half transfer callback function
00655   * @param  TransferCompleteCallback: pointe
r to the transfer complete callback function
00656   * @retval None
00657   */
00658 void BSP_AUDIO_OUT_RegisterCallbacks(Audio_C
allbackTypeDef ErrorCallback,
00659                                      Audio_C
allbackTypeDef HalfTransferCallback,
00660                                      Audio_C
allbackTypeDef TransferCompleteCallback)
```

```
00661 {
00662   hAudioOut.CbError            = ErrorCallba
ck;
00663   hAudioOut.CbHalfTransfer     = HalfTransfe
rCallback;
00664   hAudioOut.CbTransferComplete = TransferCom
pleteCallback;
00665 }
00666
00667 /**
00668   * @brief  Tx Transfer completed callbacks.
00669   * @param  hsai: SAI handle
00670   * @retval None
00671   */
00672 void HAL_SAI_TxCpltCallback(SAI_HandleTypeDe
f *hsai)
00673 {
00674   /* Invoke the registered 'TransferComplete
' function (if any) */
00675   if (hAudioOut.CbTransferComplete != (Audio
_CallbackTypeDef)NULL)
00676   {
00677     hAudioOut.CbTransferComplete();
00678   }
00679 }
00680
00681 /**
00682   * @brief  Tx Half Transfer completed callb
acks.
00683   * @param  hsai: SAI handle
00684   * @retval None
00685   */
00686 void HAL_SAI_TxHalfCpltCallback(SAI_HandleTy
peDef *hsai)
00687 {
00688   /* Invoke the registered 'HalfTransfer' ca
llback function (if any) */
```

```
00689   if (hAudioOut.CbHalfTransfer != (Audio_Cal
lbackTypeDef)NULL)
00690   {
00691     hAudioOut.CbHalfTransfer();
00692   }
00693 }
00694
00695 /**
00696   * @brief  SAI error callbacks.
00697   * @param  hsai: SAI handle
00698   * @retval None
00699   */
00700 void HAL_SAI_ErrorCallback(SAI_HandleTypeDef
 *hsai)
00701 {
00702   /* Invoke the registered 'ErrorCallback' c
allback function (if any) */
00703   if (hAudioOut.CbError != (Audio_CallbackTy
peDef)NULL)
00704   {
00705     hAudioOut.CbError();
00706   }
00707 }
00708
00709 /**
00710   * @}
00711   */
00712
00713 /** @addtogroup STM32L4R9I_EVAL_AUDIO_Export
ed_Functions
00714   * @{
00715   */
00716
00717 /**
00718   * @brief  Initializes wave recording.
00719   * @param  AudioFreq: Audio frequency to be
 configured.
```

```
00720   * @param  BitRes: Audio bit resolution to
be configured..
00721   * @param  ChnlNbr: Number of channel to be
 configured.
00722   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00723   */
00724 uint8_t BSP_AUDIO_IN_Init(uint32_t AudioFreq
, uint32_t BitRes, uint32_t ChnlNbr)
00725 {
00726   /* Update the audio input context */
00727   hAudioIn.InputDevice        = INPUT_DEVICE
_DIGITAL_MIC1_MIC2;
00728   hAudioIn.Frequency          = AudioFreq;
00729   hAudioIn.BitResolution      = BitRes;
00730   hAudioIn.ChannelNbr         = ChnlNbr;
00731   hAudioIn.CbError            = (Audio_Callb
ackTypeDef)NULL;
00732   hAudioIn.CbHalfTransfer     = (Audio_Callb
ackTypeDef)NULL;
00733   hAudioIn.CbTransferComplete = (Audio_Callb
ackTypeDef)NULL;
00734
00735   /* Check if output device is currently use
d */
00736   if(hAudioOut.OutputDevice != 0)
00737   {
00738     /* If output device is currently used, S
AI PLL is already initialized */
00739     /* Check that AudioFreq for record and p
layback is the same */
00740     if(hAudioIn.Frequency != hAudioOut.Frequ
ency)
00741     {
00742       return AUDIO_ERROR;
00743     }
00744   }
```

```
00745   else
00746   {
00747     /* Configure the SAI PLL according to th
e requested audio frequency */
00748     if (AUDIO_SAIPLLConfig(AudioFreq) != AUD
IO_OK)
00749     {
00750       return AUDIO_ERROR;
00751     }
00752   }
00753
00754   /* Initializes the Digital Filter for Sigm
a-Delta Modulators interface */
00755   if(AUDIO_DFSDMx_Init(AudioFreq) != AUDIO_OK
)
00756   {
00757     return AUDIO_ERROR;
00758   }
00759
00760   /* Initialise transfer control flag */
00761   DmaLeftRecHalfBuffCplt  = 0;
00762   DmaLeftRecBuffCplt      = 0;
00763   DmaRightRecHalfBuffCplt = 0;
00764   DmaRightRecBuffCplt     = 0;
00765
00766   return AUDIO_OK;
00767 }
00768
00769 /**
00770   * @brief  De-Initializes microphone relate
d peripherals.
00771   * @retval BSP AUDIO status
00772   */
00773 uint8_t BSP_AUDIO_IN_DeInit(void)
00774 {
00775   /* De-initializes the Digital Filter for S
igma-Delta Modulators interface */
```

```
00776   if (AUDIO_DFSDMx_DeInit() != AUDIO_OK)
00777   {
00778     return AUDIO_ERROR;
00779   }
00780
00781   /* Reset the audio input context */
00782   memset(&hAudioIn, 0, sizeof(hAudioIn));
00783
00784   return AUDIO_OK;
00785 }
00786
00787 /**
00788   * @brief  Starts audio recording.
00789   * @param  pbuf: Main buffer pointer for th
e recorded data storing
00790   * @param  size: Current size of the record
ed buffer
00791   * @note   The Right channel is start at fi
rst with synchro on start of Left channel
00792   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00793   */
00794 uint8_t BSP_AUDIO_IN_Record(uint16_t* pbuf,
uint32_t size)
00795 {
00796   hAudioIn.pRecBuf = pbuf;
00797   hAudioIn.RecSize = size;
00798
00799   /* Allocate hAudioIn.LeftRecBuff buffer */
00800 #if defined(BSP_AUDIO_USE_RTOS)
00801   hAudioIn.LeftRecBuff  = (int32_t *)k_mallo
c((size / hAudioIn.ChannelNbr) * sizeof(int32_t));
00802 #else
00803   hAudioIn.LeftRecBuff  = (int32_t *)malloc(
(size / hAudioIn.ChannelNbr) * sizeof(int32_t));
00804 #endif
00805   if(hAudioIn.LeftRecBuff == NULL)
```

```
00806   {
00807     return AUDIO_ERROR;
00808   }
00809
00810   /* Allocate hAudioIn.RightRecBuff buffer */

00811 #if defined(BSP_AUDIO_USE_RTOS)
00812   hAudioIn.RightRecBuff  = (int32_t *)k_mall
oc((size / hAudioIn.ChannelNbr) * sizeof(int32_t))
;
00813 #else
00814   hAudioIn.RightRecBuff  = (int32_t *)malloc
((size / hAudioIn.ChannelNbr) * sizeof(int32_t));
00815 #endif
00816   if(hAudioIn.RightRecBuff == NULL)
00817   {
00818     return AUDIO_ERROR;
00819   }
00820
00821   /* Call the Media layer start function for
 right channel */
00822   if(HAL_DFSDM_FilterRegularStart_DMA(&BSP_A
UDIO_hDfsdmRightFilter,
00823                                       (int32
_t*)hAudioIn.RightRecBuff,
00824                                       (hAudi
oIn.RecSize/hAudioIn.ChannelNbr)) != HAL_OK)
00825   {
00826     return AUDIO_ERROR;
00827   }
00828
00829   /* Call the Media layer start function for
 left channel */
00830   if(HAL_DFSDM_FilterRegularStart_DMA(&BSP_A
UDIO_hDfsdmLeftFilter,
00831                                       (int32
_t*)hAudioIn.LeftRecBuff,
```

```
00832                                                 (hAudi
oIn.RecSize/hAudioIn.ChannelNbr)) != HAL_OK)
00833   {
00834     return AUDIO_ERROR;
00835   }
00836
00837   return AUDIO_OK;
00838 }
00839
00840 /**
00841   * @brief  Updates the audio frequency.
00842   * @param  AudioFreq: Audio frequency used
to record the audio stream.
00843   * @note   This API should be called after
the BSP_AUDIO_IN_Init() to adjust the
00844   *         audio frequency.
00845   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00846   */
00847 uint8_t BSP_AUDIO_IN_SetFrequency(uint32_t A
udioFreq)
00848 {
00849   /* Configure the SAI PLL according to the
requested audio frequency */
00850   if (AUDIO_SAIPLLConfig(AudioFreq) != AUDIO
_OK)
00851   {
00852     return AUDIO_ERROR;
00853   }
00854
00855   /* De-initializes the Digital Filter for S
igma-Delta Modulators interface */
00856   if(AUDIO_DFSDMx_DeInit() != AUDIO_OK)
00857   {
00858     return AUDIO_ERROR;
00859   }
00860
```

```
00861   /* Initializes the Digital Filter for Sigm
a-Delta Modulators interface */
00862   if(AUDIO_DFSDMx_Init(AudioFreq) != AUDIO_OK
)
00863   {
00864     return AUDIO_ERROR;
00865   }
00866
00867   return AUDIO_OK;
00868 }
00869
00870 /**
00871   * @brief  Regular conversion complete call
back.
00872   * @note   In interrupt mode, user has to r
ead conversion value in this function
00873             using HAL_DFSDM_FilterGetRegular
Value.
00874   * @param  hdfsdm_filter : DFSDM filter han
dle.
00875   * @retval None
00876   */
00877 void HAL_DFSDM_FilterRegConvCpltCallback(DFS
DM_Filter_HandleTypeDef *hdfsdm_filter)
00878 {
00879   uint32_t index;
00880   uint32_t recbufsize = (hAudioIn.RecSize/hA
udioIn.ChannelNbr);
00881
00882   if(hdfsdm_filter == &BSP_AUDIO_hDfsdmLeftF
ilter)
00883   {
00884     DmaLeftRecBuffCplt = 1;
00885   }
00886   else
00887   {
00888     DmaRightRecBuffCplt = 1;
```

```
00889   }
00890
00891   if((DmaLeftRecBuffCplt == 1) && (DmaRightR
ecBuffCplt == 1))
00892   {
00893     for(index = (recbufsize/2); index < recb
ufsize; index++)
00894     {
00895       hAudioIn.pRecBuf[2*index]       = (uin
t16_t)(SaturaLH((hAudioIn.LeftRecBuff[index] >> 8)
, -32768, 32767));
00896       hAudioIn.pRecBuf[(2*index) + 1] = (uin
t16_t)(SaturaLH((hAudioIn.RightRecBuff[index] >> 8
), -32768, 32767));
00897     }
00898     DmaLeftRecBuffCplt = DmaRightRecBuffCplt
 = 0;
00899
00900     if (hAudioIn.CbTransferComplete != (Audi
o_CallbackTypeDef)NULL)
00901     {
00902       hAudioIn.CbTransferComplete();
00903     }
00904   }
00905 }
00906
00907 /**
00908   * @brief  Half regular conversion complete
 callback.
00909   * @param  hdfsdm_filter : DFSDM filter han
dle.
00910   * @retval None
00911   */
00912 void HAL_DFSDM_FilterRegConvHalfCpltCallback
(DFSDM_Filter_HandleTypeDef *hdfsdm_filter)
00913 {
00914   uint32_t index;
```

```
00915   uint32_t recbufsize = (hAudioIn.RecSize/hA
udioIn.ChannelNbr);
00916
00917   if(hdfsdm_filter == &BSP_AUDIO_hDfsdmLeftF
ilter)
00918   {
00919     DmaLeftRecHalfBuffCplt = 1;
00920   }
00921   else
00922   {
00923     DmaRightRecHalfBuffCplt = 1;
00924   }
00925
00926   if((DmaLeftRecHalfBuffCplt == 1) && (DmaRi
ghtRecHalfBuffCplt == 1))
00927   {
00928     for(index = 0; index < (recbufsize/2); i
ndex++)
00929     {
00930       hAudioIn.pRecBuf[2*index]       = (uin
t16_t)(SaturaLH((hAudioIn.LeftRecBuff[index] >> 8)
, -32768, 32767));
00931       hAudioIn.pRecBuf[(2*index) + 1] = (uin
t16_t)(SaturaLH((hAudioIn.RightRecBuff[index] >> 8
), -32768, 32767));
00932     }
00933     DmaLeftRecHalfBuffCplt = DmaRightRecHalf
BuffCplt = 0;
00934
00935     if (hAudioIn.CbHalfTransfer != (Audio_Ca
llbackTypeDef)NULL)
00936     {
00937       hAudioIn.CbHalfTransfer();
00938     }
00939   }
00940 }
00941
```

```
00942 /**
00943   * @brief  Error callback.
00944   * @param  hdfsdm_filter : DFSDM filter han
dle.
00945   * @retval None
00946   */
00947 void HAL_DFSDM_FilterErrorCallback(DFSDM_Fil
ter_HandleTypeDef *hdfsdm_filter)
00948 {
00949   /* Invoke the registered 'ErrorCallback' c
allback function (if any) */
00950   if (hAudioIn.CbError != (Audio_CallbackTyp
eDef)NULL)
00951   {
00952     hAudioIn.CbError();
00953   }
00954 }
00955
00956 /**
00957   * @brief  Stops audio recording.
00958   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00959   */
00960 uint8_t BSP_AUDIO_IN_Stop(void)
00961 {
00962   /* Call the Media layer stop function for
right channel */
00963   if(HAL_DFSDM_FilterRegularStop_DMA(&BSP_AU
DIO_hDfsdmRightFilter) != HAL_OK )
00964   {
00965     return AUDIO_ERROR;
00966   }
00967   /* Call the Media layer stop function for
left channel */
00968   if(HAL_DFSDM_FilterRegularStop_DMA(&BSP_AU
DIO_hDfsdmLeftFilter) != HAL_OK )
00969   {
```

```
00970     return AUDIO_ERROR;
00971   }
00972
00973   /* Free hAudioIn.LeftRecBuff buffer */
00974 #if defined(BSP_AUDIO_USE_RTOS)
00975   k_free((void *)hAudioIn.LeftRecBuff);
00976 #else
00977   free((void *)hAudioIn.LeftRecBuff);
00978 #endif
00979   /* Free hAudioIn.RightRecBuff buffer */
00980 #if defined(BSP_AUDIO_USE_RTOS)
00981   k_free((void *)hAudioIn.RightRecBuff);
00982 #else
00983   free((void *)hAudioIn.RightRecBuff);
00984 #endif
00985
00986   return AUDIO_OK;
00987 }
00988
00989 /**
00990   * @brief  Pauses the audio file stream.
00991   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
00992   */
00993 uint8_t BSP_AUDIO_IN_Pause(void)
00994 {
00995   /* Call the Media layer stop function */
00996   if(HAL_DFSDM_FilterRegularStop_DMA(&BSP_AU
DIO_hDfsdmRightFilter) != HAL_OK)
00997   {
00998     return AUDIO_ERROR;
00999   }
01000
01001   if(HAL_DFSDM_FilterRegularStop_DMA(&BSP_AU
DIO_hDfsdmLeftFilter) != HAL_OK)
01002   {
01003     return AUDIO_ERROR;
```

```
01004   }
01005
01006   /* Return AUDIO_OK when all operations are
 correctly done */
01007   return AUDIO_OK;
01008 }
01009
01010 /**
01011   * @brief  Resumes the audio file stream.
01012   * @note   The Right channel is start a fir
st with synchro on start Left channel
01013   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
01014   */
01015 uint8_t BSP_AUDIO_IN_Resume(void)
01016 {
01017   /* Call the Media layer start function for
 right channel */
01018   if(HAL_DFSDM_FilterRegularStart_DMA(&BSP_A
UDIO_hDfsdmRightFilter,
01019                                       (int32
_t*)hAudioIn.RightRecBuff,
01020                                       (hAudi
oIn.RecSize/hAudioIn.ChannelNbr)) != HAL_OK)
01021   {
01022     return AUDIO_ERROR;
01023   }
01024
01025   /* Call the Media layer start function for
 left channel */
01026   if(HAL_DFSDM_FilterRegularStart_DMA(&BSP_A
UDIO_hDfsdmLeftFilter,
01027                                       (int32
_t*)hAudioIn.LeftRecBuff,
01028                                       (hAudi
oIn.RecSize/hAudioIn.ChannelNbr)) != HAL_OK)
01029   {
```

```
01030     return AUDIO_ERROR;
01031   }
01032
01033   /* Return AUDIO_OK when all operations are
 correctly done */
01034   return AUDIO_OK;
01035 }
01036
01037 /**
01038   * @brief  register user callback functions
01039   * @param  ErrorCallback: pointer to the er
ror callback function
01040   * @param  HalfTransferCallback: pointer to
 the half transfer callback function
01041   * @param  TransferCompleteCallback: pointe
r to the transfer complete callback function
01042   * @retval None
01043   */
01044 void BSP_AUDIO_IN_RegisterCallbacks(Audio_Ca
llbackTypeDef ErrorCallback,
01045                                     Audio_Ca
llbackTypeDef HalfTransferCallback,
01046                                     Audio_Ca
llbackTypeDef TransferCompleteCallback)
01047 {
01048   hAudioIn.CbError            = ErrorCallbac
k;
01049   hAudioIn.CbHalfTransfer     = HalfTransfer
Callback;
01050   hAudioIn.CbTransferComplete = TransferComp
leteCallback;
01051 }
01052
01053 /**
01054   * @}
01055   */
01056
```

```
01057 /* Private functions ---------------------
---------------------------------*/
01058 /** @addtogroup STM32L4R9I_EVAL_AUDIO_Privat
e_Functions
01059   * @{
01060   */
01061
01062 /*****************************************
**********************************
01063                                   Static Functions
01064 *****************************************
**********************************/
01065 /**
01066   * @brief  Initializes the Audio Codec audi
o interface (SAI).
01067   * @param  AudioFreq: Audio frequency to be
 configured for the SAI peripheral.
01068   * @param  OutputDevice: OUTPUT_DEVICE_SPEA
KER, OUTPUT_DEVICE_HEADPHONE,
01069   *                          or OUTPUT_DEVICE_B
OTH.
01070   * @note   The default SlotActive configura
tion is set to CODEC_AUDIOFRAME_SLOT_0123
01071   *         and user can update this configu
ration using
01072   * @retval None
01073   */
01074 static uint8_t AUDIO_SAIx_Init(uint32_t Audi
oFreq, uint16_t OutputDevice)
01075 {
01076   /* Initialize the haudio_out_sai Instance
parameter */
01077   BSP_AUDIO_hSai.Instance = AUDIO_SAIx;
01078
01079   /* Disable SAI peripheral to allow access
to SAI internal registers */
01080   __HAL_SAI_DISABLE(&BSP_AUDIO_hSai);
```

```
01081
01082   /* Configure SAI_Block_x
01083   LSBFirst: Disabled
01084   DataSize: 16 */
01085   BSP_AUDIO_hSai.Init.AudioMode          = S
AI_MODEMASTER_TX;
01086   BSP_AUDIO_hSai.Init.Synchro            = S
AI_ASYNCHRONOUS;
01087   BSP_AUDIO_hSai.Init.SynchroExt         = S
AI_SYNCEXT_DISABLE;
01088   BSP_AUDIO_hSai.Init.OutputDrive        = S
AI_OUTPUTDRIVE_ENABLE;
01089   BSP_AUDIO_hSai.Init.NoDivider          = S
AI_MASTERDIVIDER_ENABLE;
01090   BSP_AUDIO_hSai.Init.FIFOThreshold      = S
AI_FIFOTHRESHOLD_1QF;
01091   BSP_AUDIO_hSai.Init.AudioFrequency     = S
AI_AUDIO_FREQUENCY_MCKDIV;
01092   BSP_AUDIO_hSai.Init.Mckdiv             = S
AIClockDivider(AudioFreq);
01093   BSP_AUDIO_hSai.Init.MonoStereoMode     = S
AI_STEREOMODE;
01094   BSP_AUDIO_hSai.Init.CompandingMode     = S
AI_NOCOMPANDING;
01095   BSP_AUDIO_hSai.Init.TriState           = S
AI_OUTPUT_NOTRELEASED;
01096   BSP_AUDIO_hSai.Init.Protocol           = S
AI_FREE_PROTOCOL;
01097   BSP_AUDIO_hSai.Init.DataSize           = S
AI_DATASIZE_16;
01098   BSP_AUDIO_hSai.Init.FirstBit           = S
AI_FIRSTBIT_MSB;
01099   BSP_AUDIO_hSai.Init.ClockStrobing      = S
AI_CLOCKSTROBING_FALLINGEDGE;
01100   BSP_AUDIO_hSai.Init.MckOverSampling    = S
AI_MCK_OVERSAMPLING_DISABLE;
01101   BSP_AUDIO_hSai.Init.PdmInit.Activation = D
```

```
ISABLE;
01102
01103   /* Configure SAI_Block_x Frame
01104   Frame Length: 64
01105   Frame active Length: 32
01106   FS Definition: Start frame + Channel Side
identification
01107   FS Polarity: FS active Low
01108   FS Offset: FS asserted one bit before the
first bit of slot 0 */
01109   BSP_AUDIO_hSai.FrameInit.FrameLength
 = 64;
01110   BSP_AUDIO_hSai.FrameInit.ActiveFrameLength
 = 32;
01111   BSP_AUDIO_hSai.FrameInit.FSDefinition
 = SAI_FS_CHANNEL_IDENTIFICATION;
01112   BSP_AUDIO_hSai.FrameInit.FSPolarity
 = SAI_FS_ACTIVE_LOW;
01113   BSP_AUDIO_hSai.FrameInit.FSOffset
 = SAI_FS_BEFOREFIRSTBIT;
01114
01115   /* Configure SAI Block_x Slot
01116   Slot First Bit Offset: 0
01117   Slot Size  : 16
01118   Slot Number: 4
01119   Slot Active: 0 and 2 */
01120   BSP_AUDIO_hSai.SlotInit.FirstBitOffset = 0
;
01121   BSP_AUDIO_hSai.SlotInit.SlotSize       = S
AI_SLOTSIZE_DATASIZE;
01122   BSP_AUDIO_hSai.SlotInit.SlotNumber     = 4
;
01123   if(OutputDevice == OUTPUT_DEVICE_SPEAKER)
01124   {
01125     BSP_AUDIO_hSai.SlotInit.SlotActive     =
 SAI_SLOTACTIVE_1 | SAI_SLOTACTIVE_3;
01126   }
```

```
01127   else if(OutputDevice == OUTPUT_DEVICE_HEAD
PHONE)
01128   {
01129     BSP_AUDIO_hSai.SlotInit.SlotActive     =
 SAI_SLOTACTIVE_0 | SAI_SLOTACTIVE_2;
01130   }
01131   else /* OUTPUT_DEVICE_BOTH */
01132   {
01133     BSP_AUDIO_hSai.SlotInit.SlotActive     =
 SAI_SLOTACTIVE_0 | SAI_SLOTACTIVE_1 | SAI_SLOTACT
IVE_2 | SAI_SLOTACTIVE_3;
01134   }
01135
01136   /* Initializes the SAI peripheral*/
01137   if (HAL_SAI_Init(&BSP_AUDIO_hSai) != HAL_O
K)
01138   {
01139     return AUDIO_ERROR;
01140   }
01141
01142   /* Enable SAI peripheral to generate MCLK
*/
01143   __HAL_SAI_ENABLE(&BSP_AUDIO_hSai);
01144
01145   return AUDIO_OK;
01146 }
01147
01148 /**
01149   * @brief  De-initializes the Audio Codec a
udio interface (SAI).
01150   * @retval BSP AUDIO status
01151   */
01152 static uint8_t AUDIO_SAIx_DeInit(void)
01153 {
01154   /* Disable the SAI audio block */
01155   __HAL_SAI_DISABLE(&BSP_AUDIO_hSai);
01156
```

```
01157   /* De-initializes the SAI peripheral */
01158   if (HAL_SAI_DeInit(&BSP_AUDIO_hSai) != HAL
_OK)
01159   {
01160     return AUDIO_ERROR;
01161   }
01162
01163   /* Disable SAIx PLL */
01164   if(AUDIO_SAIx_PLL_DISABLE() != AUDIO_OK)
01165   {
01166     return AUDIO_ERROR;
01167   }
01168
01169   return AUDIO_OK;
01170 }
01171
01172 /**
01173   * @brief  SAI MSP Init
01174   * @param  hsai : pointer to a SAI_HandleTy
peDef structure
01175   * @retval None
01176   */
01177 void HAL_SAI_MspInit(SAI_HandleTypeDef *hsai
)
01178 {
01179   GPIO_InitTypeDef  GPIO_InitStruct;
01180
01181   /* Enable SAI clock */
01182   AUDIO_SAIx_CLK_ENABLE();
01183
01184   /* Enable GPIO clock */
01185   AUDIO_SAIx_MCK_CLK_ENABLE();
01186   AUDIO_SAIx_FS_CLK_ENABLE();
01187   AUDIO_SAIx_SCK_CLK_ENABLE();
01188   AUDIO_SAIx_SD_CLK_ENABLE();
01189
01190   /* CODEC_SAI pins configuration: FS, SCK,
```

```
MCK and SD pins ------------------*/
01191   GPIO_InitStruct.Mode      = GPIO_MODE_AF_P
P;
01192   GPIO_InitStruct.Pull      = GPIO_NOPULL;
01193   GPIO_InitStruct.Speed     = GPIO_SPEED_FRE
Q_VERY_HIGH;
01194   GPIO_InitStruct.Alternate = AUDIO_SAIx_MCK
_SCK_SD_FS_AF;
01195   GPIO_InitStruct.Pin       = AUDIO_SAIx_MCK
_PIN;
01196   HAL_GPIO_Init(AUDIO_SAIx_MCK_GPIO_PORT, &G
PIO_InitStruct);
01197   GPIO_InitStruct.Pin       = AUDIO_SAIx_FS_
PIN;
01198   HAL_GPIO_Init(AUDIO_SAIx_FS_GPIO_PORT, &GP
IO_InitStruct);
01199   GPIO_InitStruct.Pin       = AUDIO_SAIx_SCK
_PIN;
01200   HAL_GPIO_Init(AUDIO_SAIx_SCK_GPIO_PORT, &G
PIO_InitStruct);
01201   GPIO_InitStruct.Pin       = AUDIO_SAIx_SD_
PIN;
01202   HAL_GPIO_Init(AUDIO_SAIx_SD_GPIO_PORT, &GP
IO_InitStruct);
01203
01204   /* Enable the DMA clock */
01205   AUDIO_SAIx_DMAx_CLK_ENABLE();
01206   __HAL_RCC_DMAMUX1_CLK_ENABLE();
01207
01208   if(hsai->Instance == AUDIO_SAIx)
01209   {
01210     /* Configure the hDmaSai handle paramete
rs */
01211     hDmaSai.Init.Request             = AUDIO
_SAIx_DMAx_REQUEST;
01212     hDmaSai.Init.Direction           = DMA_M
EMORY_TO_PERIPH;
```

```
01213     hDmaSai.Init.PeriphInc           = DMA_P
INC_DISABLE;
01214     hDmaSai.Init.MemInc              = DMA_M
INC_ENABLE;
01215     hDmaSai.Init.PeriphDataAlignment = AUDIO
_SAIx_DMAx_PERIPH_DATA_SIZE;
01216     hDmaSai.Init.MemDataAlignment    = AUDIO
_SAIx_DMAx_MEM_DATA_SIZE;
01217     hDmaSai.Init.Mode                = DMA_C
IRCULAR;
01218     hDmaSai.Init.Priority            = DMA_P
RIORITY_HIGH;
01219
01220     hDmaSai.Instance = AUDIO_SAIx_DMAx_CHANN
EL;
01221
01222     /* Associate the DMA handle */
01223     __HAL_LINKDMA(hsai, hdmatx, hDmaSai);
01224
01225     /* Deinitialize the Stream for new trans
fer */
01226     HAL_DMA_DeInit(&hDmaSai);
01227
01228     /* Configure the DMA Stream */
01229     HAL_DMA_Init(&hDmaSai);
01230   }
01231
01232   /* SAI DMA IRQ Channel configuration */
01233   HAL_NVIC_SetPriority(AUDIO_SAIx_DMAx_IRQ,
AUDIO_OUT_IRQ_PREPRIO, 0);
01234   HAL_NVIC_EnableIRQ(AUDIO_SAIx_DMAx_IRQ);
01235 }
01236
01237 /**
01238   * @brief  SAI MSP De-init
01239   * @param  hsai : pointer to a SAI_HandleTy
peDef structure
```

```
01240   * @retval None
01241   */
01242 void HAL_SAI_MspDeInit(SAI_HandleTypeDef *hs
ai)
01243 {
01244   /* Disable SAI DMA Channel IRQ  */
01245   HAL_NVIC_DisableIRQ(AUDIO_SAIx_DMAx_IRQ);
01246
01247   /* Reset the DMA Stream configuration*/
01248   HAL_DMA_DeInit(&hDmaSai);
01249
01250   /* Disable the DMA clock */
01251   AUDIO_SAIx_DMAx_CLK_DISABLE();
01252
01253   /* De-initialize FS, SCK, MCK and SD pins*/

01254   HAL_GPIO_DeInit(AUDIO_SAIx_MCK_GPIO_PORT,
AUDIO_SAIx_MCK_PIN);
01255   HAL_GPIO_DeInit(AUDIO_SAIx_FS_GPIO_PORT, A
UDIO_SAIx_FS_PIN);
01256   HAL_GPIO_DeInit(AUDIO_SAIx_SCK_GPIO_PORT,
AUDIO_SAIx_SCK_PIN);
01257   HAL_GPIO_DeInit(AUDIO_SAIx_SD_GPIO_PORT, A
UDIO_SAIx_SD_PIN);
01258
01259   /* Disable GPIO clock */
01260   AUDIO_SAIx_MCK_CLK_DISABLE();
01261   AUDIO_SAIx_FS_CLK_DISABLE();
01262   AUDIO_SAIx_SCK_CLK_DISABLE();
01263   /* Don't disable SD GPIO clock (GPIOC) use
d for TAMPER button */
01264   /* AUDIO_SAIx_SD_CLK_DISABLE(); */
01265
01266   /* Disable SAI clock */
01267   AUDIO_SAIx_CLK_DISABLE();
01268 }
01269
```

```
01270 /**
01271   * @brief  Resets the audio codec. It resto
res the default configuration of the
01272   *         codec (this function shall be ca
lled before initializing the codec).
01273   * @retval None
01274   */
01275 static void AUDIO_CODEC_Reset(void)
01276 {
01277   /* Initialize the audio driver structure */

01278   hAudioOut.AudioDrv = &wm8994_drv;
01279
01280   hAudioOut.AudioDrv->Reset(AUDIO_I2C_ADDRESS
);
01281 }
01282
01283 /**
01284   * @}
01285   */
01286
01287 /** @addtogroup STM32L4R9I_EVAL_AUDIO_Privat
e_Functions
01288   * @{
01289   */
01290
01291 /*****************************************
**********************************
01292                                   Static Functions
01293 *****************************************
**********************************/
01294 /**
01295   * @brief  Initializes the Digital Filter f
or Sigma-Delta Modulators interface (DFSDM).
01296   * @param  AudioFreq: Audio frequency to be
 used to set correctly the DFSDM peripheral.
01297   * @retval AUDIO_OK if correct communicatio
```

```
n, else wrong communication
01298   */
01299 static uint8_t AUDIO_DFSDMx_Init(uint32_t Au
dioFreq)
01300 {
01301   /*####CHANNEL 4####*/
01302   hAudioIn.hDfsdmLeftChannel.Init.OutputCloc
k.Activation   = ENABLE;
01303   hAudioIn.hDfsdmLeftChannel.Init.OutputCloc
k.Selection    = DFSDM_CHANNEL_OUTPUT_CLOCK_AUDIO;
01304   /* Set the DFSDM clock OUT audio frequency
 configuration */
01305   hAudioIn.hDfsdmLeftChannel.Init.OutputCloc
k.Divider      = DFSDMClockDivider(AudioFreq);
01306   hAudioIn.hDfsdmLeftChannel.Init.Input.Mult
iplexer        = DFSDM_CHANNEL_EXTERNAL_INPUTS;
01307   hAudioIn.hDfsdmLeftChannel.Init.Input.Data
Packing        = DFSDM_CHANNEL_STANDARD_MODE;
01308   hAudioIn.hDfsdmLeftChannel.Init.Input.Pins
               = DFSDM_CHANNEL_SAME_CHANNEL_PINS;
01309   /* Request to sample stable data for LEFT
micro on Rising edge */
01310   hAudioIn.hDfsdmLeftChannel.Init.SerialInte
rface.Type     = DFSDM_CHANNEL_SPI_RISING;
01311   hAudioIn.hDfsdmLeftChannel.Init.SerialInte
rface.SpiClock = DFSDM_CHANNEL_SPI_CLOCK_INTERNAL;
01312   hAudioIn.hDfsdmLeftChannel.Init.Awd.Filter
Order          = DFSDM_CHANNEL_SINC1_ORDER;
01313   hAudioIn.hDfsdmLeftChannel.Init.Awd.Oversa
mpling         = 10;
01314   hAudioIn.hDfsdmLeftChannel.Init.Offset
               = 0;
01315   hAudioIn.hDfsdmLeftChannel.Init.RightBitSh
ift            = DFSDMRightBitShift(AudioFreq);
01316   hAudioIn.hDfsdmLeftChannel.Instance
               = AUDIO_DFSDMx_LEFT_CHANNEL;
01317   /* Init the DFSDM Channel */
```

```
01318   if (HAL_DFSDM_ChannelInit(&hAudioIn.hDfsdm
LeftChannel) != HAL_OK)
01319   {
01320     return AUDIO_ERROR;
01321   }
01322
01323   /*####CHANNEL 3####*/
01324   hAudioIn.hDfsdmRightChannel.Init.OutputClo
ck.Activation   = ENABLE;
01325   hAudioIn.hDfsdmRightChannel.Init.OutputClo
ck.Selection    = DFSDM_CHANNEL_OUTPUT_CLOCK_AUDIO
;
01326   /* Set the DFSDM clock OUT audio frequency
 configuration */
01327   hAudioIn.hDfsdmRightChannel.Init.OutputClo
ck.Divider      = DFSDMClockDivider(AudioFreq);
01328   hAudioIn.hDfsdmRightChannel.Init.Input.Mul
tiplexer        = DFSDM_CHANNEL_EXTERNAL_INPUTS;
01329   hAudioIn.hDfsdmRightChannel.Init.Input.Dat
aPacking        = DFSDM_CHANNEL_STANDARD_MODE;
01330   hAudioIn.hDfsdmRightChannel.Init.Input.Pin
s               = DFSDM_CHANNEL_FOLLOWING_CHANNEL_
PINS;
01331   /* Request to sample stable data for RIGHT
 micro on Falling edge */
01332   hAudioIn.hDfsdmRightChannel.Init.SerialInt
erface.Type     = DFSDM_CHANNEL_SPI_FALLING;
01333   hAudioIn.hDfsdmRightChannel.Init.SerialInt
erface.SpiClock = DFSDM_CHANNEL_SPI_CLOCK_INTERNAL
;
01334   hAudioIn.hDfsdmRightChannel.Init.Awd.Filte
rOrder          = DFSDM_CHANNEL_SINC1_ORDER;
01335   hAudioIn.hDfsdmRightChannel.Init.Awd.Overs
ampling         = 10;
01336   hAudioIn.hDfsdmRightChannel.Init.Offset
                = 0;
01337   hAudioIn.hDfsdmRightChannel.Init.RightBitS
```

```
hift             = DFSDMRightBitShift(AudioFreq);
01338   hAudioIn.hDfsdmRightChannel.Instance
                 = AUDIO_DFSDMx_RIGHT_CHANNEL;
01339   /* Init the DFSDM Channel */
01340   if (HAL_DFSDM_ChannelInit(&hAudioIn.hDfsdm
RightChannel) != HAL_OK)
01341   {
01342     return AUDIO_ERROR;
01343   }
01344
01345   /*####FILTER 0####*/
01346   BSP_AUDIO_hDfsdmLeftFilter.Init.RegularPar
am.Trigger          = DFSDM_FILTER_SW_TRIGGER;
01347   BSP_AUDIO_hDfsdmLeftFilter.Init.RegularPar
am.FastMode         = ENABLE;
01348   BSP_AUDIO_hDfsdmLeftFilter.Init.RegularPar
am.DmaMode          = ENABLE;
01349   BSP_AUDIO_hDfsdmLeftFilter.Init.InjectedPa
ram.Trigger         = DFSDM_FILTER_SW_TRIGGER;
01350   BSP_AUDIO_hDfsdmLeftFilter.Init.InjectedPa
ram.ScanMode        = DISABLE;
01351   BSP_AUDIO_hDfsdmLeftFilter.Init.InjectedPa
ram.DmaMode         = DISABLE;
01352   BSP_AUDIO_hDfsdmLeftFilter.Init.InjectedPa
ram.ExtTrigger      = DFSDM_FILTER_EXT_TRIG_TIM8_TR
GO;
01353   BSP_AUDIO_hDfsdmLeftFilter.Init.InjectedPa
ram.ExtTriggerEdge = DFSDM_FILTER_EXT_TRIG_BOTH_ED
GES;
01354   BSP_AUDIO_hDfsdmLeftFilter.Init.FilterPara
m.SincOrder         = DFSDMFilterOrder(AudioFreq);
01355   /* Set the DFSDM Filters Oversampling to h
ave correct sample rate */
01356   BSP_AUDIO_hDfsdmLeftFilter.Init.FilterPara
m.Oversampling      = DFSDMOverSampling(AudioFreq);
01357   BSP_AUDIO_hDfsdmLeftFilter.Init.FilterPara
m.IntOversampling  = 1;
```

```
01358   BSP_AUDIO_hDfsdmLeftFilter.Instance
                    = AUDIO_DFSDMx_LEFT_FILTER;
01359   /* Init the DFSDM Filter */
01360   if (HAL_DFSDM_FilterInit(&BSP_AUDIO_hDfsdm
LeftFilter) != HAL_OK)
01361   {
01362     return AUDIO_ERROR;
01363   }
01364   /* Configure injected channel */
01365   if(HAL_OK != HAL_DFSDM_FilterConfigRegChan
nel(&BSP_AUDIO_hDfsdmLeftFilter, DFSDM_CHANNEL_4,
DFSDM_CONTINUOUS_CONV_ON))
01366   {
01367     return AUDIO_ERROR;
01368   }
01369
01370   /*####FILTER 1####*/
01371   BSP_AUDIO_hDfsdmRightFilter.Init.RegularPa
ram.Trigger          = DFSDM_FILTER_SYNC_TRIGGER;
01372   BSP_AUDIO_hDfsdmRightFilter.Init.RegularPa
ram.FastMode         = ENABLE;
01373   BSP_AUDIO_hDfsdmRightFilter.Init.RegularPa
ram.DmaMode          = ENABLE;
01374   BSP_AUDIO_hDfsdmRightFilter.Init.InjectedP
aram.Trigger         = DFSDM_FILTER_SW_TRIGGER;
01375   BSP_AUDIO_hDfsdmRightFilter.Init.InjectedP
aram.ScanMode        = DISABLE;
01376   BSP_AUDIO_hDfsdmRightFilter.Init.InjectedP
aram.DmaMode         = DISABLE;
01377   BSP_AUDIO_hDfsdmRightFilter.Init.InjectedP
aram.ExtTrigger      = DFSDM_FILTER_EXT_TRIG_TIM8_T
RGO;
01378   BSP_AUDIO_hDfsdmRightFilter.Init.InjectedP
aram.ExtTriggerEdge = DFSDM_FILTER_EXT_TRIG_BOTH_E
DGES;
01379   BSP_AUDIO_hDfsdmRightFilter.Init.FilterPar
am.SincOrder         = DFSDMFilterOrder(AudioFreq);
```

```
01380   /* Set the DFSDM Filters Oversampling to h
ave correct sample rate */
01381   BSP_AUDIO_hDfsdmRightFilter.Init.FilterPar
am.Oversampling     = DFSDMOverSampling(AudioFreq)
;
01382   BSP_AUDIO_hDfsdmRightFilter.Init.FilterPar
am.IntOversampling  = 1;
01383   BSP_AUDIO_hDfsdmRightFilter.Instance
                    = AUDIO_DFSDMx_RIGHT_FILTER;
01384   /* Init the DFSDM Filter */
01385   if (HAL_DFSDM_FilterInit(&BSP_AUDIO_hDfsdm
RightFilter) != HAL_OK)
01386   {
01387     return AUDIO_ERROR;
01388   }
01389   /* Configure injected channel */
01390   if(HAL_OK != HAL_DFSDM_FilterConfigRegChan
nel(&BSP_AUDIO_hDfsdmRightFilter, DFSDM_CHANNEL_3,
 DFSDM_CONTINUOUS_CONV_ON))
01391   {
01392     return AUDIO_ERROR;
01393   }
01394
01395   return AUDIO_OK;
01396 }
01397
01398 /**
01399   * @brief  De-initializes the Digital Filte
r for Sigma-Delta Modulators interface (DFSDM).
01400   * @retval AUDIO_OK if correct communicatio
n, else wrong communication
01401   */
01402 static uint8_t AUDIO_DFSDMx_DeInit(void)
01403 {
01404   /* De-initializes the DFSDM filters to all
ow access to DFSDM internal registers */
01405   if(HAL_OK != HAL_DFSDM_FilterDeInit(&BSP_A
```

```
UDIO_hDfsdmRightFilter))
01406   {
01407     return AUDIO_ERROR;
01408   }
01409   if(HAL_OK != HAL_DFSDM_FilterDeInit(&BSP_A
UDIO_hDfsdmLeftFilter))
01410   {
01411     return AUDIO_ERROR;
01412   }
01413
01414   /* De-initializes the DFSDM channels to al
low access to DFSDM internal registers */
01415   if(HAL_OK != HAL_DFSDM_ChannelDeInit(&hAud
ioIn.hDfsdmRightChannel))
01416   {
01417     return AUDIO_ERROR;
01418   }
01419   if(HAL_OK != HAL_DFSDM_ChannelDeInit(&hAud
ioIn.hDfsdmLeftChannel))
01420   {
01421     return AUDIO_ERROR;
01422   }
01423
01424   /* DFSDM reset */
01425   __HAL_RCC_DFSDM1_FORCE_RESET();
01426   __HAL_RCC_DFSDM1_RELEASE_RESET();
01427
01428   return AUDIO_OK;
01429 }
01430
01431 /**
01432   * @brief  Initializes the DFSDM channel MS
P.
01433   * @param  hdfsdm_channel : DFSDM channel h
andle.
01434   * @retval None
01435   */
```

```
01436 void HAL_DFSDM_ChannelMspInit(DFSDM_Channel_
HandleTypeDef *hdfsdm_channel)
01437 {
01438   GPIO_InitTypeDef  GPIO_InitStruct;
01439
01440   /* Enable DFSDM clock */
01441   AUDIO_DFSDMx_CLK_ENABLE();
01442
01443   /* Enable GPIO clock */
01444   AUDIO_DFSDMx_CKOUT_CLK_ENABLE();
01445   AUDIO_DFSDMx_DATIN_CLK_ENABLE();
01446
01447   /* DFSDM pins configuration: DFSDM_CKOUT,
DMIC_DATIN pins ------------------*/
01448   GPIO_InitStruct.Mode      = GPIO_MODE_AF_P
P;
01449   GPIO_InitStruct.Pull      = GPIO_NOPULL;
01450   GPIO_InitStruct.Speed     = GPIO_SPEED_FRE
Q_VERY_HIGH;
01451   GPIO_InitStruct.Alternate = AUDIO_DFSDMx_C
KOUT_DATIN_AF;
01452   GPIO_InitStruct.Pin       = AUDIO_DFSDMx_C
KOUT_PIN;
01453   HAL_GPIO_Init(AUDIO_DFSDMx_CKOUT_GPIO_PORT
, &GPIO_InitStruct);
01454   GPIO_InitStruct.Pin       = AUDIO_DFSDMx_D
ATIN_PIN;
01455   HAL_GPIO_Init(AUDIO_DFSDMx_DATIN_GPIO_PORT
, &GPIO_InitStruct);
01456 }
01457
01458 /**
01459   * @brief  DeInitializes the DFSDM channel
MSP.
01460   * @param  hdfsdm_channel : DFSDM channel h
andle.
01461   * @retval None
```

```
01462   */
01463 void HAL_DFSDM_ChannelMspDeInit(DFSDM_Channe
l_HandleTypeDef *hdfsdm_channel)
01464 {
01465   /* DeInit DFSDM pins configuration: DFSDM_
CKOUT, DMIC_DATIN pins -----------*/
01466   HAL_GPIO_DeInit(AUDIO_DFSDMx_CKOUT_GPIO_PO
RT, AUDIO_DFSDMx_CKOUT_PIN);
01467   HAL_GPIO_DeInit(AUDIO_DFSDMx_DATIN_GPIO_PO
RT, AUDIO_DFSDMx_DATIN_PIN);
01468   AUDIO_DFSDMx_CKOUT_CLK_DISABLE();
01469   /* Don't disable DATIN GPIO clock (GPIOC)
used for TAMPER button */
01470   /* AUDIO_DFSDMx_DATIN_CLK_DISABLE(); */
01471
01472   /* Disable DFSDM clock */
01473   AUDIO_DFSDMx_CLK_DISABLE();
01474 }
01475
01476 /**
01477   * @brief  Initializes the DFSDM filter MSP.

01478   * @param  hdfsdm_filter : DFSDM filter han
dle.
01479   * @retval None
01480   */
01481 void HAL_DFSDM_FilterMspInit(DFSDM_Filter_Ha
ndleTypeDef *hdfsdm_filter)
01482 {
01483   if(hdfsdm_filter->Instance == AUDIO_DFSDMx
_LEFT_FILTER)
01484   {
01485     /* Enable the DMA clock */
01486     AUDIO_DFSDMx_DMAx_CLK_ENABLE();
01487     __HAL_RCC_DMAMUX1_CLK_ENABLE();
01488
01489     /* Configure the hAudioIn.hDmaDfsdmLeft
```

```
handle parameters */
01490     hAudioIn.hDmaDfsdmLeft.Init.Request
        = AUDIO_DFSDMx_DMAx_LEFT_REQUEST;
01491     hAudioIn.hDmaDfsdmLeft.Init.Direction
        = DMA_PERIPH_TO_MEMORY;
01492     hAudioIn.hDmaDfsdmLeft.Init.PeriphInc
        = DMA_PINC_DISABLE;
01493     hAudioIn.hDmaDfsdmLeft.Init.MemInc
        = DMA_MINC_ENABLE;
01494     hAudioIn.hDmaDfsdmLeft.Init.PeriphDataAl
ignment = AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE;
01495     hAudioIn.hDmaDfsdmLeft.Init.MemDataAlign
ment    = AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE;
01496     hAudioIn.hDmaDfsdmLeft.Init.Mode
        = DMA_CIRCULAR;
01497     hAudioIn.hDmaDfsdmLeft.Init.Priority
        = DMA_PRIORITY_HIGH;
01498     hAudioIn.hDmaDfsdmLeft.Instance
        = AUDIO_DFSDMx_DMAx_LEFT_CHANNEL;
01499
01500     /* Associate the DMA handle */
01501     __HAL_LINKDMA(hdfsdm_filter, hdmaReg, hA
udioIn.hDmaDfsdmLeft);
01502
01503     /* Reset DMA handle state */
01504     __HAL_DMA_RESET_HANDLE_STATE(&hAudioIn.h
DmaDfsdmLeft);
01505
01506     /* Configure the DMA Channel */
01507     HAL_DMA_Init(&hAudioIn.hDmaDfsdmLeft);
01508
01509     /* DMA IRQ Channel configuration */
01510     HAL_NVIC_SetPriority(AUDIO_DFSDMx_DMAx_L
EFT_IRQ, AUDIO_IN_IRQ_PREPRIO, 0);
01511     HAL_NVIC_EnableIRQ(AUDIO_DFSDMx_DMAx_LEF
T_IRQ);
01512   }
```

```
01513   else /* DFSDM1_Filter for right channel */
01514   {
01515     /* DMA clock already enabled */
01516     /* AUDIO_DFSDMx_DMAx_CLK_ENABLE(); */
01517
01518     /* Configure the hAudioIn.hDmaDfsdmRight
 handle parameters */
01519     hAudioIn.hDmaDfsdmRight.Init.Request
         = AUDIO_DFSDMx_DMAx_RIGHT_REQUEST;
01520     hAudioIn.hDmaDfsdmRight.Init.Direction
         = DMA_PERIPH_TO_MEMORY;
01521     hAudioIn.hDmaDfsdmRight.Init.PeriphInc
         = DMA_PINC_DISABLE;
01522     hAudioIn.hDmaDfsdmRight.Init.MemInc
         = DMA_MINC_ENABLE;
01523     hAudioIn.hDmaDfsdmRight.Init.PeriphDataA
lignment = AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE;
01524     hAudioIn.hDmaDfsdmRight.Init.MemDataAlig
nment    = AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE;
01525     hAudioIn.hDmaDfsdmRight.Init.Mode
         = DMA_CIRCULAR;
01526     hAudioIn.hDmaDfsdmRight.Init.Priority
         = DMA_PRIORITY_HIGH;
01527     hAudioIn.hDmaDfsdmRight.Instance
         = AUDIO_DFSDMx_DMAx_RIGHT_CHANNEL;
01528
01529     /* Associate the DMA handle */
01530     __HAL_LINKDMA(hdfsdm_filter, hdmaReg, hA
udioIn.hDmaDfsdmRight);
01531
01532     /* Reset DMA handle state */
01533     __HAL_DMA_RESET_HANDLE_STATE(&hAudioIn.h
DmaDfsdmRight);
01534
01535     /* Configure the DMA Channel */
01536     HAL_DMA_Init(&hAudioIn.hDmaDfsdmRight);
01537
```

```
01538     /* DMA IRQ Channel configuration */
01539     HAL_NVIC_SetPriority(AUDIO_DFSDMx_DMAx_R
IGHT_IRQ, AUDIO_IN_IRQ_PREPRIO, 0);
01540     HAL_NVIC_EnableIRQ(AUDIO_DFSDMx_DMAx_RIG
HT_IRQ);
01541   }
01542 }
01543
01544 /**
01545   * @brief  DeInitializes the DFSDM filter M
SP.
01546   * @param  hdfsdm_filter : DFSDM filter han
dle.
01547   * @retval None
01548   */
01549 void HAL_DFSDM_FilterMspDeInit(DFSDM_Filter_
HandleTypeDef *hdfsdm_filter)
01550 {
01551   if(hdfsdm_filter->Instance == AUDIO_DFSDMx
_LEFT_FILTER)
01552   {
01553     /* Disable DMA  Channel IRQ */
01554     HAL_NVIC_DisableIRQ(AUDIO_DFSDMx_DMAx_LE
FT_IRQ);
01555
01556     /* De-initialize the DMA Channel */
01557     HAL_DMA_DeInit(&hAudioIn.hDmaDfsdmLeft);
01558
01559     /* Disable the DMA clock */
01560     AUDIO_DFSDMx_DMAx_CLK_DISABLE();
01561   }
01562   else /* DFSDM1_Filter for right channel */
01563   {
01564     /* Disable DMA  Channel IRQ */
01565     HAL_NVIC_DisableIRQ(AUDIO_DFSDMx_DMAx_RI
GHT_IRQ);
01566
```

```
01567     /* De-initialize the DMA Channel */
01568     HAL_DMA_DeInit(&hAudioIn.hDmaDfsdmRight)
;
01569
01570     /* Don't disable the DMA clock (filter 1
 deinitialized before filter 0) */
01571     /*__HAL_RCC_DMA1_CLK_DISABLE();*/
01572   }
01573 }
01574
01575 /**
01576   * @brief  Configures the SAI PLL clock acc
ording to the required audio frequency.
01577   * @param  Frequency: Audio frequency.
01578   * @retval BSP AUDIO status
01579   * @note   The SAI PLL input clock must be
configured in the user application.
01580   *         The SAI PLL configuration done w
ithin this function assumes that
01581   *         the SAI PLL input clock runs at
4 MHz.
01582   */
01583 static uint8_t AUDIO_SAIPLLConfig(uint32_t F
requency)
01584 {
01585   RCC_PeriphCLKInitTypeDef RCC_ExCLKInitStru
ct;
01586
01587   /* Retrieve actual RCC configuration */
01588   HAL_RCCEx_GetPeriphCLKConfig(&RCC_ExCLKIni
tStruct);
01589
01590   if (   (Frequency == AUDIO_FREQUENCY_11K)
01591       || (Frequency == AUDIO_FREQUENCY_22K)
01592       || (Frequency == AUDIO_FREQUENCY_44K)
)
01593   {
```

```
01594     /* Configure PLLSAI prescalers */
01595     /* SAI clock config
01596     PLLSAI1_VCO= 4 Mhz * PLLSAI1N = 4 * 48 =
 VCO_192M
01597     SAI_CK_x = PLLSAI1_VCO/PLLSAI1P = 192/17
 = 11.294 Mhz */
01598     RCC_ExCLKInitStruct.PeriphClockSelection
    = RCC_PERIPHCLK_SAI1;
01599     RCC_ExCLKInitStruct.PLLSAI1.PLLSAI1M
    = 1;
01600     RCC_ExCLKInitStruct.PLLSAI1.PLLSAI1N
    = 48;
01601     RCC_ExCLKInitStruct.PLLSAI1.PLLSAI1P
    = 17;
01602     RCC_ExCLKInitStruct.PLLSAI1.PLLSAI1Clock
Out = RCC_PLLSAI1_SAI1CLK;
01603     RCC_ExCLKInitStruct.Sai1ClockSelection
    = RCC_SAI1CLKSOURCE_PLLSAI1;
01604   }
01605   else /* AUDIO_FREQUENCY_8K, AUDIO_FREQUENC
Y_16K, AUDIO_FREQUENCY_48K, AUDIO_FREQUENCY_96K */
01606   {
01607     /* SAI clock config
01608     PLLSAI1_VCO= 4 Mhz * PLLSAI1N = 4 * 86 =
 VCO_344M
01609     SAI_CK_x = PLLSAI1_VCO/PLLSAI1P = 344/7
= 49.142 Mhz */
01610     RCC_ExCLKInitStruct.PeriphClockSelection
    = RCC_PERIPHCLK_SAI1;
01611     RCC_ExCLKInitStruct.PLLSAI1.PLLSAI1M
    = 1;
01612     RCC_ExCLKInitStruct.PLLSAI1.PLLSAI1N
    = 86;
01613     RCC_ExCLKInitStruct.PLLSAI1.PLLSAI1P
    = 7;
01614     RCC_ExCLKInitStruct.PLLSAI1.PLLSAI1Clock
Out = RCC_PLLSAI1_SAI1CLK;
```

```
01615     RCC_ExCLKInitStruct.Sai1ClockSelection
    = RCC_SAI1CLKSOURCE_PLLSAI1;
01616   }
01617
01618   if (HAL_RCCEx_PeriphCLKConfig(&RCC_ExCLKIn
itStruct) != HAL_OK)
01619   {
01620     return AUDIO_ERROR;
01621   }
01622
01623   return AUDIO_OK;
01624
01625 /**
01626   * @}
01627   */
01628
01629 /**
01630   * @}
01631   */
01632
01633 /**
01634   * @}
01635   */
01636
01637 /**
01638   * @}
01639   */
01640
01641 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL DSI LCD

STM32L4R9I_EVAL LCD

Modules

- Private Macros
- Exported Variables
- Private Variables
- Private FunctionPrototypes
- Exported Functions
- Private Functions

STM32L4R9I_EVAL BSP User Manual

Exported Types

STM32L4R9I_EVAL LCD

Data Structures

struct	**LCD_DrawPropTypeDef**
struct	**Point**

Typedefs

typedef **Point** * **pPoint**

Enumerations

enum	**Text_AlignModeTypdef** { **CENTER_MODE** = 0x01, **RIGHT_MODE** = 0x02, **LEFT_MODE** = 0x03 }
enum	**LCD_OrientationTypeDef** { **LCD_ORIENTATION_PORTRAIT** = 0x00, **LCD_ORIENTATION_LANDSCAPE** = 0x01, **LCD_ORIENTATION_INVALID** = 0x02 }

Typedef Documentation

typedef Point* pPoint

Definition at line **166** of file **stm32l4r9i_eval_lcd.h**.

Enumeration Type Documentation

enum LCD_OrientationTypeDef

Enumerator:

LCD_ORIENTATION_PORTRAIT	Portrait orientation choice of LCD screen
LCD_ORIENTATION_LANDSCAPE	Landscape orientation choice of LCD screen
LCD_ORIENTATION_INVALID	Invalid orientation choice of LCD screen

Definition at line **177** of file **stm32l4r9i_eval_lcd.h**.

enum Text_AlignModeTypdef

Enumerator:

CENTER_MODE	Center mode
RIGHT_MODE	Right mode
LEFT_MODE	Left mode

Definition at line **169** of file **stm32l4r9i_eval_lcd.h**.

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_lcd.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_lcd.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains alias for compatibility with DSI and TFT LCD.
00006   ******************************************************************************
00007   * @attention
00008   *
00009   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00010   *
00011   * Redistribution and use in source and binary forms, with or without modification,
00012   * are permitted provided that the following conditions are met:
00013   *   1. Redistributions of source code must retain the above copyright notice,
00014   *      this list of conditions and the following disclaimer.
```

```
00015   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00016   *      this list of conditions and the fol
lowing disclaimer in the documentation
00017   *      and/or other materials provided wit
h the distribution.
00018   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00019   *      may be used to endorse or promote p
roducts derived from this software
00020   *      without specific prior written perm
ission.
00021   *
00022   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00023   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00024   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00025   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00026   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00027   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00028   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00029   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00030   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00031   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00032   *
00033   ****************************************
**********************************
00034   */
```

```
00035
00036 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00037 #ifndef __STM32L4R9I_EVAL_LCD_H
00038 #define __STM32L4R9I_EVAL_LCD_H
00039
00040 #ifdef __cplusplus
00041  extern "C" {
00042 #endif
00043
00044 #include "../../../Utilities/Fonts/fonts.h"
00045
00046 /** @addtogroup BSP
00047   * @{
00048   */
00049
00050 /** @addtogroup STM32L4R9I_EVAL
00051   * @{
00052   */
00053
00054 /** @addtogroup STM32L4R9I_EVAL_LCD STM32L4R
9I_EVAL LCD
00055   * @{
00056   */
00057
00058 /** @defgroup STM32L4R9I_EVAL_LCD_Exported_C
onstants Exported Constants
00059   * @{
00060   */
00061
00062 /* LTDC layer configuration structure */
00063 #define LCD_LayerCfgTypeDef                LT
DC_LayerCfgTypeDef
00064
00065 /* Maximum number of LTDC layers */
00066 #define LTDC_MAX_LAYER_NUMBER              ((
uint32_t) 2)
```

```
00067
00068 /* LTDC Background layer index */
00069 #define LTDC_ACTIVE_LAYER_BACKGROUND      LT
DC_LAYER_1
00070
00071 /* LTDC Foreground layer index : Not used on
 STM32L4R9I_EVAL, only one layer used */
00072 #define LTDC_ACTIVE_LAYER_FOREGROUND      LT
DC_LAYER_2
00073
00074 /* Number of LTDC layers */
00075 #define LTDC_NB_OF_LAYERS                 ((
uint32_t) 2)
00076
00077 /* LTDC Default used layer index */
00078 #define LTDC_DEFAULT_ACTIVE_LAYER         LT
DC_ACTIVE_LAYER_BACKGROUND
00079
00080 /* LCD status */
00081 #define   LCD_OK          0x00
00082 #define   LCD_ERROR       0x01
00083 #define   LCD_TIMEOUT     0x02
00084
00085 /* LCD color definitions values in ARGB8888
format */
00086 /* Blue value in ARGB8888 format */
00087 #define LCD_COLOR_BLUE          ((uint32_t)
0xFF0000FF)
00088 /* Green value in ARGB8888 format */
00089 #define LCD_COLOR_GREEN         ((uint32_t)
0xFF00FF00)
00090 /* Red value in ARGB8888 format */
00091 #define LCD_COLOR_RED           ((uint32_t)
0xFFFF0000)
00092 /* Cyan value in ARGB8888 format */
00093 #define LCD_COLOR_CYAN          ((uint32_t)
0xFF00FFFF)
```

```
00094 /* Magenta value in ARGB8888 format */
00095 #define LCD_COLOR_MAGENTA       ((uint32_t)
0xFFFF00FF)
00096 /* Yellow value in ARGB8888 format */
00097 #define LCD_COLOR_YELLOW        ((uint32_t)
0xFFFFFF00)
00098 /* Light Blue value in ARGB8888 format */
00099 #define LCD_COLOR_LIGHTBLUE     ((uint32_t)
0xFF8080FF)
00100 /* Light Green value in ARGB8888 format */
00101 #define LCD_COLOR_LIGHTGREEN    ((uint32_t)
0xFF80FF80)
00102 /* Light Red value in ARGB8888 format */
00103 #define LCD_COLOR_LIGHTRED      ((uint32_t)
0xFFFF8080)
00104 /* Light Cyan value in ARGB8888 format */
00105 #define LCD_COLOR_LIGHTCYAN     ((uint32_t)
0xFF80FFFF)
00106 /* Light Magenta value in ARGB8888 format */
00107 #define LCD_COLOR_LIGHTMAGENTA  ((uint32_t)
0xFFFF80FF)
00108 /* Light Yellow value in ARGB8888 format */
00109 #define LCD_COLOR_LIGHTYELLOW   ((uint32_t)
0xFFFFFF80)
00110 /* Dark Blue value in ARGB8888 format */
00111 #define LCD_COLOR_DARKBLUE      ((uint32_t)
0xFF000080)
00112 /* Light Dark Green value in ARGB8888 format
 */
00113 #define LCD_COLOR_DARKGREEN     ((uint32_t)
0xFF008000)
00114 /* Light Dark Red value in ARGB8888 format */

00115 #define LCD_COLOR_DARKRED       ((uint32_t)
0xFF800000)
00116 /* Dark Cyan value in ARGB8888 format */
00117 #define LCD_COLOR_DARKCYAN      ((uint32_t)
```

```
0xFF008080)
00118 /* Dark Magenta value in ARGB8888 format */
00119 #define LCD_COLOR_DARKMAGENTA   ((uint32_t)
0xFF800080)
00120 /* Dark Yellow value in ARGB8888 format */
00121 #define LCD_COLOR_DARKYELLOW    ((uint32_t)
0xFF808000)
00122 /* White value in ARGB8888 format */
00123 #define LCD_COLOR_WHITE         ((uint32_t)
0xFFFFFFFF)
00124 /* Light Gray value in ARGB8888 format */
00125 #define LCD_COLOR_LIGHTGRAY     ((uint32_t)
0xFFD3D3D3)
00126 /* Gray value in ARGB8888 format */
00127 #define LCD_COLOR_GRAY          ((uint32_t)
0xFF808080)
00128 /* Dark Gray value in ARGB8888 format */
00129 #define LCD_COLOR_DARKGRAY      ((uint32_t)
0xFF404040)
00130 /* Black value in ARGB8888 format */
00131 #define LCD_COLOR_BLACK         ((uint32_t)
0xFF000000)
00132 /* Brown value in ARGB8888 format */
00133 #define LCD_COLOR_BROWN         ((uint32_t)
0xFFA52A2A)
00134 /* Orange value in ARGB8888 format */
00135 #define LCD_COLOR_ORANGE        ((uint32_t)
0xFFFFA500)
00136 /* Transparent value in ARGB8888 format */
00137 #define LCD_COLOR_TRANSPARENT   ((uint32_t)
0xFF000000)
00138
00139 /* LCD default font */
00140 #define LCD_DEFAULT_FONT        Font24
00141
00142 /**
00143   * @}
```

```
00144   */
00145
00146 /** @defgroup STM32L4R9I_EVAL_LCD_Exported_T
ypes Exported Types
00147   * @{
00148   */
00149
00150 /* LCD Drawing main properties */
00151 typedef struct
00152 {
00153   uint32_t TextColor; /*!< Specifies the col
or of text */
00154   uint32_t BackColor; /*!< Specifies the bac
kground color below the text */
00155   sFONT    *pFont;    /*!< Specifies the fon
t used for the text */
00156 } LCD_DrawPropTypeDef;
00157
00158 /* LCD Drawing point (pixel) geometric defin
ition */
00159 typedef struct
00160 {
00161   int16_t X; /*!< geometric X position of dr
awing */
00162   int16_t Y; /*!< geometric Y position of dr
awing */
00163 } Point;
00164
00165 /* Pointer on LCD Drawing point (pixel) geom
etric definition */
00166 typedef Point * pPoint;
00167
00168 /* LCD drawing Line alignment mode definitio
ns */
00169 typedef enum
00170 {
00171   CENTER_MODE             = 0x01,    /*!< Ce
```

```
nter mode */
00172   RIGHT_MODE              = 0x02,    /*!< Ri
ght mode  */
00173   LEFT_MODE               = 0x03     /*!< Le
ft mode   */
00174 } Text_AlignModeTypdef;
00175
00176 /* LCD_OrientationTypeDef : Possible values
of Display Orientation */
00177 typedef enum
00178 {
00179   LCD_ORIENTATION_PORTRAIT  = 0x00, /*!< Por
trait orientation choice of LCD screen  */
00180   LCD_ORIENTATION_LANDSCAPE = 0x01, /*!< Lan
dscape orientation choice of LCD screen */
00181   LCD_ORIENTATION_INVALID   = 0x02  /*!< Inv
alid orientation choice of LCD screen   */
00182 } LCD_OrientationTypeDef;
00183
00184 /**
00185   * @}
00186   */
00187
00188 /** @addtogroup STM32L4R9I_EVAL_DSI_LCD_Expo
rted_Functions
00189   * @{
00190   */
00191
00192 #if defined(USE_GVO_390x390)
00193
00194 #include "stm32l4r9i_eval_dsi_lcd.h"
00195
00196 #define  BSP_LCD_DMA2D_IRQHandler     BSP_DS
I_LCD_DMA2D_IRQHandler
00197 #define  BSP_LCD_DSI_IRQHandler       BSP_DS
I_LCD_DSI_IRQHandler
00198 #define  BSP_LCD_LTDC_IRQHandler      BSP_DS
```

```
I_LCD_LTDC_IRQHandler
00199 #define  BSP_LCD_LTDC_ER_IRQHandler   BSP_DS
I_LCD_LTDC_ER_IRQHandler
00200
00201 #define  BSP_LCD_Init                 BSP_DS
I_LCD_Init
00202 #define  BSP_LCD_DeInit               BSP_DS
I_LCD_DeInit
00203
00204 #define  BSP_LCD_MspDeInit            BSP_DS
I_LCD_MspDeInit
00205 #define  BSP_LCD_MspInit              BSP_DS
I_LCD_MspInit
00206
00207 #define  BSP_LCD_GetXSize             BSP_DS
I_LCD_GetXSize
00208 #define  BSP_LCD_GetYSize             BSP_DS
I_LCD_GetYSize
00209
00210 #define  BSP_LCD_SetTransparency      BSP_DS
I_LCD_SetTransparency
00211 #define  BSP_LCD_SetColorKeying       BSP_DS
I_LCD_SetColorKeying
00212 #define  BSP_LCD_ResetColorKeying     BSP_DS
I_LCD_ResetColorKeying
00213
00214 #define  BSP_LCD_SelectLayer          BSP_DS
I_LCD_SelectLayer
00215 #define  BSP_LCD_SetLayerVisible      BSP_DS
I_LCD_SetLayerVisible
00216
00217 #define  BSP_LCD_SetTextColor         BSP_DS
I_LCD_SetTextColor
00218 #define  BSP_LCD_GetTextColor         BSP_DS
I_LCD_GetTextColor
00219 #define  BSP_LCD_SetBackColor         BSP_DS
I_LCD_SetBackColor
```

```
00220 #define  BSP_LCD_GetBackColor          BSP_DS
I_LCD_GetBackColor
00221 #define  BSP_LCD_SetFont               BSP_DS
I_LCD_SetFont
00222 #define  BSP_LCD_GetFont               BSP_DS
I_LCD_GetFont
00223
00224 #define  BSP_LCD_ReadPixel             BSP_DS
I_LCD_ReadPixel
00225 #define  BSP_LCD_DrawPixel             BSP_DS
I_LCD_DrawPixel
00226 #define  BSP_LCD_Clear                 BSP_DS
I_LCD_Clear
00227 #define  BSP_LCD_ClearStringLine       BSP_DS
I_LCD_ClearStringLine
00228 #define  BSP_LCD_DisplayStringAtLine   BSP_DS
I_LCD_DisplayStringAtLine
00229 #define  BSP_LCD_DisplayStringAt       BSP_DS
I_LCD_DisplayStringAt
00230 #define  BSP_LCD_DisplayChar           BSP_DS
I_LCD_DisplayChar
00231
00232 #define  BSP_LCD_DrawHLine             BSP_DS
I_LCD_DrawHLine
00233 #define  BSP_LCD_DrawVLine             BSP_DS
I_LCD_DrawVLine
00234 #define  BSP_LCD_DrawLine              BSP_DS
I_LCD_DrawLine
00235 #define  BSP_LCD_DrawRect              BSP_DS
I_LCD_DrawRect
00236 #define  BSP_LCD_DrawCircle            BSP_DS
I_LCD_DrawCircle
00237 #define  BSP_LCD_DrawPolygon           BSP_DS
I_LCD_DrawPolygon
00238 #define  BSP_LCD_DrawEllipse           BSP_DS
I_LCD_DrawEllipse
00239 #define  BSP_LCD_DrawBitmap            BSP_DS
```

```
I_LCD_DrawBitmap
00240
00241 #define  BSP_LCD_FillRect                BSP_DS
I_LCD_FillRect
00242 #define  BSP_LCD_FillCircle              BSP_DS
I_LCD_FillCircle
00243 #define  BSP_LCD_FillPolygon             BSP_DS
I_LCD_FillPolygon
00244 #define  BSP_LCD_FillEllipse             BSP_DS
I_LCD_FillEllipse
00245
00246 #define  BSP_LCD_DisplayOff              BSP_DS
I_LCD_DisplayOff
00247 #define  BSP_LCD_DisplayOn               BSP_DS
I_LCD_DisplayOn
00248
00249 #define  BSP_LCD_Refresh                   BSP
_DSI_LCD_Refresh
00250 #define  BSP_LCD_IsFrameBufferAvailable  BSP
_DSI_LCD_IsFrameBufferAvailable
00251
00252 #define  BSP_LCD_SetBrightness           BSP_DS
I_LCD_SetBrightness
00253
00254 #endif /* USE_GVO_390x390 */
00255
00256 #if defined(USE_ROCKTECH_480x272)
00257
00258 #include "stm32l4r9i_eval_rgb_lcd.h"
00259
00260 #define  BSP_LCD_DMA2D_IRQHandler     BSP_RG
B_LCD_DMA2D_IRQHandler
00261 #define  BSP_LCD_LTDC_IRQHandler      BSP_RG
B_LCD_LTDC_IRQHandler
00262 #define  BSP_LCD_LTDC_ER_IRQHandler   BSP_RG
B_LCD_LTDC_ER_IRQHandler
00263
```

```
00264 #define  BSP_LCD_Init                    BSP_RG
B_LCD_Init
00265 #define  BSP_LCD_DeInit                  BSP_RG
B_LCD_DeInit
00266
00267 #define  BSP_LCD_MspDeInit               BSP_RG
B_LCD_MspDeInit
00268 #define  BSP_LCD_MspInit                 BSP_RG
B_LCD_MspInit
00269
00270 #define  BSP_LCD_GetXSize                BSP_RG
B_LCD_GetXSize
00271 #define  BSP_LCD_GetYSize                BSP_RG
B_LCD_GetYSize
00272
00273 #define  BSP_LCD_SetTransparency         BSP_RG
B_LCD_SetTransparency
00274 #define  BSP_LCD_SetColorKeying          BSP_RG
B_LCD_SetColorKeying
00275 #define  BSP_LCD_ResetColorKeying        BSP_RG
B_LCD_ResetColorKeying
00276
00277 #define  BSP_LCD_LayerDefaultInit        BSP_RG
B_LCD_LayerRgb565Init
00278 #define  BSP_LCD_SelectLayer             BSP_RG
B_LCD_SelectLayer
00279 #define  BSP_LCD_SetLayerVisible         BSP_RG
B_LCD_SetLayerVisible
00280
00281 #define  BSP_LCD_SetTextColor            BSP_RG
B_LCD_SetTextColor
00282 #define  BSP_LCD_GetTextColor            BSP_RG
B_LCD_GetTextColor
00283 #define  BSP_LCD_SetBackColor            BSP_RG
B_LCD_SetBackColor
00284 #define  BSP_LCD_GetBackColor            BSP_RG
B_LCD_GetBackColor
```

```
00285 #define  BSP_LCD_SetFont                BSP_RG
B_LCD_SetFont
00286 #define  BSP_LCD_GetFont                BSP_RG
B_LCD_GetFont
00287
00288 #define  BSP_LCD_ReadPixel              BSP_RG
B_LCD_ReadPixel
00289 #define  BSP_LCD_DrawPixel              BSP_RG
B_LCD_DrawPixel
00290 #define  BSP_LCD_Clear                  BSP_RG
B_LCD_Clear
00291 #define  BSP_LCD_ClearStringLine        BSP_RG
B_LCD_ClearStringLine
00292 #define  BSP_LCD_DisplayStringAtLine    BSP_RG
B_LCD_DisplayStringAtLine
00293 #define  BSP_LCD_DisplayStringAt        BSP_RG
B_LCD_DisplayStringAt
00294 #define  BSP_LCD_DisplayChar            BSP_RG
B_LCD_DisplayChar
00295
00296 #define  BSP_LCD_DrawHLine              BSP_RG
B_LCD_DrawHLine
00297 #define  BSP_LCD_DrawVLine              BSP_RG
B_LCD_DrawVLine
00298 #define  BSP_LCD_DrawLine               BSP_RG
B_LCD_DrawLine
00299 #define  BSP_LCD_DrawRect               BSP_RG
B_LCD_DrawRect
00300 #define  BSP_LCD_DrawCircle             BSP_RG
B_LCD_DrawCircle
00301 #define  BSP_LCD_DrawPolygon            BSP_RG
B_LCD_DrawPolygon
00302 #define  BSP_LCD_DrawEllipse            BSP_RG
B_LCD_DrawEllipse
00303 #define  BSP_LCD_DrawBitmap             BSP_RG
B_LCD_DrawBitmap
00304
```

```
00305 #define  BSP_LCD_FillRect                BSP_RG
B_LCD_FillRect
00306 #define  BSP_LCD_FillCircle              BSP_RG
B_LCD_FillCircle
00307 #define  BSP_LCD_FillPolygon             BSP_RG
B_LCD_FillPolygon
00308 #define  BSP_LCD_FillEllipse             BSP_RG
B_LCD_FillEllipse
00309
00310 #define  BSP_LCD_DisplayOff              BSP_RG
B_LCD_DisplayOff
00311 #define  BSP_LCD_DisplayOn               BSP_RG
B_LCD_DisplayOn
00312
00313 #endif /* USE_ROCKTECH_480x272 */
00314
00315 /**
00316   * @}
00317   */
00318
00319 /**
00320   * @}
00321   */
00322
00323 /**
00324   * @}
00325   */
00326
00327 /**
00328   * @}
00329   */
00330
00331 #ifdef __cplusplus
00332 }
00333 #endif
00334
00335 #endif /* __STM32L4R9I_EVAL_LCD_H */
```

```
00336
00337 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_dsi_lcd.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_dsi_lcd.c
00004   * @author  MCD Application Team
00005   * @brief   This file includes the driver for DSI Liquid Crystal Display (LCD)
00006   *          module mounted on STM32L4R9I_EVAL evaluation board.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* File Info: ------------------------------
-------------------------------------
00038                                         User NOTES

00039 1. How To use this driver:
00040 --------------------------
00041    - This driver is used to drive directly i
n command mode a LCD TFT using the
00042      DSI interface.
00043      The following IPs are implied : DSI Hos
t IP block working
00044      in conjunction to the LTDC controller.
00045    - This driver is linked by construction t
o LCD.
00046
00047 2. Driver description:
00048 ----------------------
00049   + Initialization steps:
00050      o Initialize the LCD using the BSP_DSI_
LCD_Init() function.
00051        Please note that LCD display will be
turned on at the end of this function.
00052      o De-inietialize the LCD using BSP_DSI_
LCD_DeInit() function.
00053
00054   + Options
00055      o Modify in the fly the display propert
ies using the following functions:
00056        - BSP_DSI_LCD_SetTransparency().
00057        - BSP_DSI_LCD_SetColorKeying().
00058        - BSP_DSI_LCD_ResetColorKeying().
00059        - BSP_DSI_LCD_SetFont().
00060        - BSP_DSI_LCD_SetBackColor().
00061        - BSP_DSI_LCD_SetTextColor().
```

```
00062         - BSP_DSI_LCD_SetBrightness().
00063       o You can set display on/off using foll
owing functions:
00064         - BSP_DSI_LCD_DisplayOn().
00065         - BSP_DSI_LCD_DisplayOff().
00066
00067    + Display on LCD
00068       o First, check that frame buffer is ava
ilable using BSP_DSI_LCD_IsFrameBufferAvailable().
00069       o When frame buffer is available, modif
y it using following functions:
00070         - BSP_DSI_LCD_Clear().
00071         - BSP_DSI_LCD_ClearStringLine().
00072         - BSP_DSI_LCD_DisplayChar().
00073         - BSP_DSI_LCD_DisplayStringAt().
00074         - BSP_DSI_LCD_DisplayStringAtLine().
00075         - BSP_DSI_LCD_DrawBitmap().
00076         - BSP_DSI_LCD_DrawCircle().
00077         - BSP_DSI_LCD_DrawEllipse().
00078         - ....
00079       o Call BSP_DSI_LCD_Refresh() to refresh
 LCD display.
00080
00081 ------------------------------------------
----------------------------------*/
00082
00083 /* Includes ---------------------------------
-----------------------------------*/
00084 #include "stm32l4r9i_eval_lcd.h"
00085 #include "stm32l4r9i_eval_dsi_lcd.h"
00086 #include "stm32l4r9i_eval_gfxmmu_lut.h"
00087 #include "stm32l4r9i_eval_io.h"
00088 #include "../../../Utilities/Fonts/fonts.h"
00089 #include "../../../Utilities/Fonts/font24.c"
00090 #include "../../../Utilities/Fonts/font20.c"
00091 #include "../../../Utilities/Fonts/font16.c"
00092 #include "../../../Utilities/Fonts/font12.c"
```

```
00093 #include "../../../Utilities/Fonts/font8.c"
00094
00095 /** @addtogroup BSP
00096   * @{
00097   */
00098
00099 /** @addtogroup STM32L4R9I_EVAL
00100   * @{
00101   */
00102
00103 /** @defgroup STM32L4R9I_EVAL_LCD STM32L4R9I
_EVAL LCD
00104   * @{
00105   */
00106
00107 /** @defgroup STM32L4R9I_EVAL_DSI_LCD STM32L
4R9I_EVAL DSI LCD
00108   * @{
00109   */
00110
00111 /** @defgroup STM32L4R9I_EVAL_DSI_LCD_Privat
e_Macros Private Macros
00112   * @{
00113   */
00114 #define ABS(X)                ((X) > 0 ? (X
) : -(X))
00115
00116 #define POLY_X(Z)             ((int32_t)((P
oints + (Z))->X))
00117 #define POLY_Y(Z)             ((int32_t)((P
oints + (Z))->Y))
00118 /**
00119   * @}
00120   */
00121
00122 /** @defgroup STM32L4R9I_EVAL_DSI_LCD_Export
ed_Variables Exported Variables
```

```
00123   * @{
00124   */
00125 /* DMA2D handle */
00126 DMA2D_HandleTypeDef  hdma2d_eval;
00127
00128 /**
00129   * @}
00130   */
00131
00132 /** @defgroup STM32L4R9I_EVAL_DSI_LCD_Private_Variables Private Variables
00133   * @{
00134   */
00135 /* LCD initialization status */
00136 static uint32_t bsp_lcd_initialized = 0;
00137 /* Flag to indicate if HSE has to be disabled during de-initialization */
00138 static uint32_t bsp_lcd_hse_to_disable = 0;
00139 /* Default Active LTDC Layer in which drawing is made is LTDC Layer Background */
00140 static uint32_t  ActiveLayer = LTDC_ACTIVE_LAYER_BACKGROUND;
00141 /* Current Drawing Layer properties variable */
00142 static LCD_DrawPropTypeDef DrawProp[LTDC_MAX_LAYER_NUMBER];
00143
00144 /* Physical frame buffer for active layer */
00145 /* 390*390 pixels with 32bpp - 20% */
00146 #if defined ( __ICCARM__ )  /* IAR Compiler */
00147   #pragma data_alignment = 16
00148 uint32_t              PhysFrameBuffer[121680];
00149 #elif defined (__GNUC__)    /* GNU Compiler */
00150 uint32_t              PhysFrameBuffer[121680
```

```
] __attribute__ ((aligned (16)));
00151 #else                          /* ARM Compiler
*/
00152 __align(16) uint32_t  PhysFrameBuffer[121680
];
00153 #endif
00154
00155 /* Global variable used to know if frame buf
fer is available (1) or not because refresh is on
going (0) */
00156 __IO uint32_t FrameBufferAvailable = 1;
00157 /* LCD size */
00158 uint32_t lcd_x_size = 390;
00159 uint32_t lcd_y_size = 390;
00160 /* GFXMMU, LTDC and DSI handles */
00161 GFXMMU_HandleTypeDef hgfxmmu_eval;
00162 LTDC_HandleTypeDef   hltdc_eval;
00163 DSI_HandleTypeDef    hdsi_eval;
00164 /**
00165   * @}
00166   */
00167
00168 /** @defgroup STM32L4R9I_EVAL_DSI_LCD_Privat
e_FunctionPrototypes Private FunctionPrototypes
00169   * @{
00170   */
00171 static void LCD_Reset(void);
00172 static void DrawChar(uint16_t Xpos, uint16_t
 Ypos, const uint8_t *c);
00173 static void FillTriangle(uint16_t x1, uint16
_t x2, uint16_t x3, uint16_t y1, uint16_t y2, uint
16_t y3);
00174 static void LL_FillBuffer(uint32_t LayerInde
x, void *pDst, uint32_t xSize, uint32_t ySize, uin
t32_t OffLine, uint32_t ColorIndex);
00175 static void LL_ConvertLineToARGB8888(void *
pSrc, void *pDst, uint32_t xSize, uint32_t ColorMo
```

```
de);
00176 /**
00177   * @}
00178   */
00179
00180 /** @defgroup STM32L4R9I_EVAL_DSI_LCD_Export
ed_Functions Exported Functions
00181   * @{
00182   */
00183
00184 /**
00185   * @brief  Initialize the DSI LCD.
00186   * The initialization is done as below:
00187   *     - GFXMMU initialization
00188   *     - DSI PLL initialization
00189   *     - DSI initialization
00190   *     - LTDC initialization
00191   *     - RM67162 LCD Display IC Driver init
ialization
00192   * @retval LCD state
00193   */
00194 uint8_t BSP_DSI_LCD_Init(void)
00195 {
00196   if(bsp_lcd_initialized == 0)
00197   {
00198     LTDC_LayerCfgTypeDef    LayerCfg;
00199     DSI_PLLInitTypeDef      dsiPllInit;
00200     DSI_PHY_TimerTypeDef    PhyTimings;
00201     DSI_HOST_TimeoutTypeDef HostTimeouts;
00202     DSI_LPCmdTypeDef        LPCmd;
00203     DSI_CmdCfgTypeDef       CmdCfg;
00204
00205     /* Toggle Hardware Reset of the DSI LCD
using its XRES signal (active low) */
00206     LCD_Reset();
00207
00208     /* Call first MSP Initialize
```

```
00209       * This will set IP blocks LTDC, DSI and
 DMA2D
00210       * - out of reset
00211       * - clocked
00212       * - NVIC IRQ related to IP blocks enabl
ed
00213     */
00214     BSP_DSI_LCD_MspInit();
00215
00216     /************************/
00217     /* GFXMMU CONFIGURATION */
00218     /************************/
00219     hgfxmmu_eval.Instance = GFXMMU;
00220     __HAL_GFXMMU_RESET_HANDLE_STATE(&hgfxmmu
_eval);
00221     hgfxmmu_eval.Init.BlocksPerLine
            = GFXMMU_192BLOCKS;
00222     hgfxmmu_eval.Init.DefaultValue
            = 0xFFFFFFFF;
00223     hgfxmmu_eval.Init.Buffers.Buf0Address
            = (uint32_t) PhysFrameBuffer;
00224     hgfxmmu_eval.Init.Buffers.Buf1Address
            = 0; /* NU */
00225     hgfxmmu_eval.Init.Buffers.Buf2Address
            = 0; /* NU */
00226     hgfxmmu_eval.Init.Buffers.Buf3Address
            = 0; /* NU */
00227     hgfxmmu_eval.Init.Interrupts.Activation
            = DISABLE;
00228     hgfxmmu_eval.Init.Interrupts.UsedInterru
pts         = GFXMMU_AHB_MASTER_ERROR_IT; /* NU */
00229     if(HAL_OK != HAL_GFXMMU_Init(&hgfxmmu_ev
al))
00230     {
00231       return(LCD_ERROR);
00232     }
00233
```

```
00234     /* Initialize LUT */
00235     if(HAL_OK != HAL_GFXMMU_ConfigLut(&hgfxm
mu_eval, 0, 390, (uint32_t) gfxmmu_lut_config_argb
8888))
00236     {
00237       return(LCD_ERROR);
00238     }
00239     /* Disable non visible lines : from line
 390 to 1023 */
00240     if(HAL_OK != HAL_GFXMMU_DisableLutLines(
&hgfxmmu_eval, 390, 634))
00241     {
00242       return(LCD_ERROR);
00243     }
00244
00245     /***********************/
00246     /* LTDC CONFIGURATION */
00247     /***********************/
00248
00249     /* LTDC initialization */
00250     hltdc_eval.Instance = LTDC;
00251     __HAL_LTDC_RESET_HANDLE_STATE(&hltdc_eva
l);
00252     hltdc_eval.Init.HSPolarity         = LTD
C_HSPOLARITY_AL;
00253     hltdc_eval.Init.VSPolarity         = LTD
C_VSPOLARITY_AL;
00254     hltdc_eval.Init.DEPolarity         = LTD
C_DEPOLARITY_AL;
00255     hltdc_eval.Init.PCPolarity         = LTD
C_PCPOLARITY_IPC;
00256     hltdc_eval.Init.HorizontalSync     = 0;
  /* HSYNC width - 1 */
00257     hltdc_eval.Init.VerticalSync       = 0;
  /* VSYNC width - 1 */
00258     hltdc_eval.Init.AccumulatedHBP     = 1;
  /* HSYNC width + HBP - 1 */
```

```
00259     hltdc_eval.Init.AccumulatedVBP     = 1;
  /* VSYNC width + VBP - 1 */
00260     hltdc_eval.Init.AccumulatedActiveW = 391
; /* HSYNC width + HBP + Active width - 1 */
00261     hltdc_eval.Init.AccumulatedActiveH = 391
; /* VSYNC width + VBP + Active height - 1 */
00262     hltdc_eval.Init.TotalWidth         = 392
; /* HSYNC width + HBP + Active width + HFP - 1 */
00263     hltdc_eval.Init.TotalHeigh         = 392
; /* VSYNC width + VBP + Active height + VFP - 1 */

00264     hltdc_eval.Init.Backcolor.Red      = 255
;
00265     hltdc_eval.Init.Backcolor.Green    = 255
;
00266     hltdc_eval.Init.Backcolor.Blue     = 0;
00267     hltdc_eval.Init.Backcolor.Reserved = 0xF
F;
00268     if(HAL_LTDC_Init(&hltdc_eval) != HAL_OK)
00269     {
00270       return(LCD_ERROR);
00271     }
00272
00273     /* LTDC layers configuration */
00274     LayerCfg.WindowX0        = 0;
00275     LayerCfg.WindowX1        = 390;
00276     LayerCfg.WindowY0        = 0;
00277     LayerCfg.WindowY1        = 390;
00278     LayerCfg.PixelFormat     = LTDC_PIXEL_FO
RMAT_ARGB8888;
00279     LayerCfg.Alpha           = 0xFF; /* NU d
efault value */
00280     LayerCfg.Alpha0          = 0; /* NU defa
ult value */
00281     LayerCfg.BlendingFactor1 = LTDC_BLENDING
_FACTOR1_PAxCA; /* NU default value */
00282     LayerCfg.BlendingFactor2 = LTDC_BLENDING
```

```
_FACTOR2_PAxCA; /* NU default value */
00283     LayerCfg.FBStartAdress   = GFXMMU_VIRTUA
L_BUFFER0_BASE;
00284     LayerCfg.ImageWidth      = 768; /* virtu
al frame buffer contains 768 pixels per line for 3
2bpp */
00285     LayerCfg.ImageHeight     = 390;
00286     LayerCfg.Backcolor.Red   = 0; /* NU defa
ult value */
00287     LayerCfg.Backcolor.Green = 0; /* NU defa
ult value */
00288     LayerCfg.Backcolor.Blue  = 0; /* NU defa
ult value */
00289     LayerCfg.Backcolor.Reserved = 0xFF;
00290     if(HAL_LTDC_ConfigLayer(&hltdc_eval, &La
yerCfg, 0) != HAL_OK)
00291     {
00292       return(LCD_ERROR);
00293     }
00294
00295     DrawProp[0].BackColor = LCD_COLOR_WHITE;
00296     DrawProp[0].pFont     = &Font24;
00297     DrawProp[0].TextColor = LCD_COLOR_BLACK;
00298
00299     /**********************/
00300     /* DSI CONFIGURATION */
00301     /**********************/
00302
00303     /* DSI initialization */
00304     hdsi_eval.Instance = DSI;
00305     __HAL_DSI_RESET_HANDLE_STATE(&hdsi_eval)
;
00306     hdsi_eval.Init.AutomaticClockLaneControl
 = DSI_AUTO_CLK_LANE_CTRL_DISABLE;
00307     /* We have 1 data lane at 500Mbps => lan
e byte clock at 500/8 = 62,5 MHZ */
00308     /* We want TX escape clock at arround 20
```

```
MHz and under 20MHz so clock division is set to 4
*/
00309     hdsi_eval.Init.TXEscapeCkdiv
 = 4;
00310     hdsi_eval.Init.NumberOfLanes
 = DSI_ONE_DATA_LANE;
00311     /* We have HSE value at 25MHz or 16MHz a
nd we want data lane at 500Mbps */
00312     dsiPllInit.PLLNDIV = (HSE_VALUE == 25000
000) ? 100 : 125;
00313     dsiPllInit.PLLIDF  = (HSE_VALUE == 25000
000) ? DSI_PLL_IN_DIV5 : DSI_PLL_IN_DIV4;
00314     dsiPllInit.PLLODF  = DSI_PLL_OUT_DIV1;
00315     if(HAL_DSI_Init(&hdsi_eval, &dsiPllInit)
 != HAL_OK)
00316     {
00317       return(LCD_ERROR);
00318     }
00319
00320     PhyTimings.ClockLaneHS2LPTime  = 33; /*
Tclk-post + Tclk-trail + Ths-exit = [(60ns + 52xUI
) + (60ns) + (300ns)]/16ns */
00321     PhyTimings.ClockLaneLP2HSTime  = 30; /*
Tlpx + (Tclk-prepare + Tclk-zero) + Tclk-pre = [15
0ns + 300ns + 8xUI]/16ns */
00322     PhyTimings.DataLaneHS2LPTime   = 11; /*
Ths-trail + Ths-exit = [(60ns + 4xUI) + 100ns]/16n
s */
00323     PhyTimings.DataLaneLP2HSTime   = 21; /*
Tlpx + (Ths-prepare + Ths-zero) + Ths-sync = [150n
s + (145ns + 10xUI) + 8xUI]/16ns */
00324     PhyTimings.DataLaneMaxReadTime = 0;
00325     PhyTimings.StopWaitTime        = 7;
00326     if(HAL_DSI_ConfigPhyTimer(&hdsi_eval, &P
hyTimings) != HAL_OK)
00327     {
00328       return(LCD_ERROR);
```

```
00329     }
00330
00331     HostTimeouts.TimeoutCkdiv
  = 1;
00332     HostTimeouts.HighSpeedTransmissionTimeou
t = 0;
00333     HostTimeouts.LowPowerReceptionTimeout
  = 0;
00334     HostTimeouts.HighSpeedReadTimeout
  = 0;
00335     HostTimeouts.LowPowerReadTimeout
  = 0;
00336     HostTimeouts.HighSpeedWriteTimeout
  = 0;
00337     HostTimeouts.HighSpeedWritePrespMode
  = 0;
00338     HostTimeouts.LowPowerWriteTimeout
  = 0;
00339     HostTimeouts.BTATimeout
  = 0;
00340     if(HAL_DSI_ConfigHostTimeouts(&hdsi_eval
, &HostTimeouts) != HAL_OK)
00341     {
00342       return(LCD_ERROR);
00343     }
00344
00345     LPCmd.LPGenShortWriteNoP  = DSI_LP_GSW0P
_ENABLE;
00346     LPCmd.LPGenShortWriteOneP = DSI_LP_GSW1P
_ENABLE;
00347     LPCmd.LPGenShortWriteTwoP = DSI_LP_GSW2P
_ENABLE;
00348     LPCmd.LPGenShortReadNoP   = DSI_LP_GSR0P
_ENABLE;
00349     LPCmd.LPGenShortReadOneP  = DSI_LP_GSR1P
_ENABLE;
00350     LPCmd.LPGenShortReadTwoP  = DSI_LP_GSR2P
```

```
_ENABLE;
00351     LPCmd.LPGenLongWrite      = DSI_LP_GLW_D
ISABLE;
00352     LPCmd.LPDcsShortWriteNoP  = DSI_LP_DSW0P
_ENABLE;
00353     LPCmd.LPDcsShortWriteOneP = DSI_LP_DSW1P
_ENABLE;
00354     LPCmd.LPDcsShortReadNoP   = DSI_LP_DSR0P
_ENABLE;
00355     LPCmd.LPDcsLongWrite      = DSI_LP_DLW_D
ISABLE;
00356     LPCmd.LPMaxReadPacket     = DSI_LP_MRDP_
DISABLE;
00357     LPCmd.AcknowledgeRequest  = DSI_ACKNOWLE
DGE_DISABLE;
00358     if(HAL_DSI_ConfigCommand(&hdsi_eval, &LP
Cmd) != HAL_OK)
00359     {
00360       return(LCD_ERROR);
00361     }
00362
00363     CmdCfg.VirtualChannelID      = 0;
00364     CmdCfg.ColorCoding           = DSI_RGB88
8;
00365     CmdCfg.CommandSize           = 390;
00366     CmdCfg.TearingEffectSource   = DSI_TE_DS
ILINK;
00367     CmdCfg.TearingEffectPolarity = DSI_TE_FA
LLING_EDGE;
00368     CmdCfg.HSPolarity            = DSI_HSYNC
_ACTIVE_LOW;
00369     CmdCfg.VSPolarity            = DSI_VSYNC
_ACTIVE_LOW;
00370     CmdCfg.DEPolarity            = DSI_DATA_
ENABLE_ACTIVE_HIGH;
00371     CmdCfg.VSyncPol              = DSI_VSYNC
_FALLING;
```

```
00372     CmdCfg.AutomaticRefresh      = DSI_AR_EN
ABLE;
00373     CmdCfg.TEAcknowledgeRequest  = DSI_TE_AC
KNOWLEDGE_ENABLE;
00374     if(HAL_DSI_ConfigAdaptedCommandMode(&hds
i_eval, &CmdCfg) != HAL_OK)
00375     {
00376       return(LCD_ERROR);
00377     }
00378
00379     /* Disable the Tearing Effect interrupt
activated by default on previous function */
00380     __HAL_DSI_DISABLE_IT(&hdsi_eval, DSI_IT_
TE);
00381
00382     if(HAL_DSI_ConfigFlowControl(&hdsi_eval,
 DSI_FLOW_CONTROL_BTA) != HAL_OK)
00383     {
00384       return(LCD_ERROR);
00385     }
00386
00387     /* Enable DSI */
00388     __HAL_DSI_ENABLE(&hdsi_eval);
00389
00390     /*************************/
00391     /* LCD POWER ON SEQUENCE */
00392     /*************************/
00393     /* Step 1 */
00394     /* Go to command 2 */
00395     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0xFE, 0x01);
00396     /* IC Frame rate control, set power, sw
mapping, mux swithc timing command */
00397     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x06, 0x62);
00398     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x0E, 0x80);
```

```
00399     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x0F, 0x80);
00400     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x10, 0x71);
00401     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x13, 0x81);
00402     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x14, 0x81);
00403     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x15, 0x82);
00404     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x16, 0x82);
00405     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x18, 0x88);
00406     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x19, 0x55);
00407     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1A, 0x10);
00408     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1C, 0x99);
00409     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1D, 0x03);
00410     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1E, 0x03);
00411     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1F, 0x03);
00412     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x20, 0x03);
00413     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x25, 0x03);
00414     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x26, 0x8D);
00415     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x2A, 0x03);
00416     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x2B, 0x8D);
00417     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
```

```
S_SHORT_PKT_WRITE_P1, 0x36, 0x00);
00418     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x37, 0x10);
00419     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x3A, 0x00);
00420     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x3B, 0x00);
00421     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x3D, 0x20);
00422     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x3F, 0x3A);
00423     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x40, 0x30);
00424     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x41, 0x1A);
00425     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x42, 0x33);
00426     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x43, 0x22);
00427     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x44, 0x11);
00428     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x45, 0x66);
00429     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x46, 0x55);
00430     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x47, 0x44);
00431     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x4C, 0x33);
00432     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x4D, 0x22);
00433     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x4E, 0x11);
00434     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x4F, 0x66);
00435     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x50, 0x55);
```

```
00436     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x51, 0x44);
00437     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x57, 0x33);
00438     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x6B, 0x1B);
00439     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x70, 0x55);
00440     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x74, 0x0C);
00441
00442     /* Go to command 3 */
00443     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0xFE, 0x02);
00444     /* Set the VGMP/VGSP coltage control */
00445     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x9B, 0x40);
00446     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x9C, 0x00);
00447     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x9D, 0x20);
00448
00449     /* Go to command 4 */
00450     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0xFE, 0x03);
00451     /* Set the VGMP/VGSP coltage control */
00452     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x9B, 0x40);
00453     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x9C, 0x00);
00454     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x9D, 0x20);
00455
00456
00457     /* Go to command 5 */
00458     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0xFE, 0x04);
```

```
00459     /* VSR command */
00460     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x5D, 0x10);
00461     /* VSR1 timing set */
00462     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x00, 0x8D);
00463     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x01, 0x00);
00464     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x02, 0x01);
00465     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x03, 0x01);
00466     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x04, 0x10);
00467     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x05, 0x01);
00468     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x06, 0xA7);
00469     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x07, 0x20);
00470     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x08, 0x00);
00471     /* VSR2 timing set */
00472     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x09, 0xC2);
00473     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x0A, 0x00);
00474     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x0B, 0x02);
00475     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x0C, 0x01);
00476     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x0D, 0x40);
00477     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x0E, 0x06);
00478     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x0F, 0x01);
```

```
00479     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x10, 0xA7);
00480     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x11, 0x00);
00481     /* VSR3 timing set */
00482     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x12, 0xC2);
00483     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x13, 0x00);
00484     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x14, 0x02);
00485     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x15, 0x01);
00486     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x16, 0x40);
00487     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x17, 0x07);
00488     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x18, 0x01);
00489     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x19, 0xA7);
00490     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1A, 0x00);
00491     /* VSR4 timing set */
00492     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1B, 0x82);
00493     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1C, 0x00);
00494     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1D, 0xFF);
00495     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1E, 0x05);
00496     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x1F, 0x60);
00497     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x20, 0x02);
00498     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
```

```
S_SHORT_PKT_WRITE_P1, 0x21, 0x01);
00499     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x22, 0x7C);
00500     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x23, 0x00);
00501     /* VSR5 timing set */
00502     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x24, 0xC2);
00503     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x25, 0x00);
00504     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x26, 0x04);
00505     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x27, 0x02);
00506     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x28, 0x70);
00507     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x29, 0x05);
00508     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x2A, 0x74);
00509     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x2B, 0x8D);
00510     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x2D, 0x00);
00511     /* VSR6 timing set */
00512     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x2F, 0xC2);
00513     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x30, 0x00);
00514     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x31, 0x04);
00515     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x32, 0x02);
00516     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x33, 0x70);
00517     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x34, 0x07);
```

```
00518     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x35, 0x74);
00519     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x36, 0x8D);
00520     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x37, 0x00);
00521     /* VSR marping command */
00522     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x5E, 0x20);
00523     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x5F, 0x31);
00524     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x60, 0x54);
00525     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x61, 0x76);
00526     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x62, 0x98);
00527
00528     /* Go to command 6 */
00529     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0xFE, 0x05);
00530     /* Set the ELVSS voltage */
00531     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x05, 0x17);
00532     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x2A, 0x04);
00533     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0x91, 0x00);
00534
00535     /* Go back in standard commands */
00536     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0xFE, 0x00);
00537
00538     /* Set tear off */
00539     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, DSI_SET_TEAR_OFF, 0x0);
00540
```

```
00541     /* Set DSI mode to internal timing added
 vs ORIGINAL for Command mode */
00542     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P1, 0xC2, 0x0);
00543
00544     /* Set memory address MODIFIED vs ORIGIN
AL */
00545     uint8_t InitParam1[4]= {0x00, 0x04, 0x01
, 0x89}; // MODIF OFe: adjusted w/ real image
00546     HAL_DSI_LongWrite(&hdsi_eval, 0, DSI_DCS
_LONG_PKT_WRITE, 4, DSI_SET_COLUMN_ADDRESS, InitPa
ram1);
00547     uint8_t InitParam2[4]= {0x00, 0x00, 0x01
, 0x85};
00548     HAL_DSI_LongWrite(&hdsi_eval, 0, DSI_DCS
_LONG_PKT_WRITE, 4, DSI_SET_PAGE_ADDRESS, InitPara
m2);
00549
00550     /* Sleep out */
00551     HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DC
S_SHORT_PKT_WRITE_P0, DSI_EXIT_SLEEP_MODE, 0x0);
00552
00553     HAL_Delay(120);
00554
00555     /* Set display on */
00556     if(HAL_DSI_ShortWrite(&hdsi_eval,
00557                           0,
00558                           DSI_DCS_SHORT_PKT_
WRITE_P0,
00559                           DSI_SET_DISPLAY_ON
,
00560                           0x0) != HAL_OK)
00561     {
00562       return(LCD_ERROR);
00563     }
00564
00565     /* Enable DSI Wrapper */
```

```
00566     __HAL_DSI_WRAPPER_ENABLE(&hdsi_eval);
00567
00568     /* Initialize the font */
00569     BSP_DSI_LCD_SetFont(&LCD_DEFAULT_FONT);
00570
00571     bsp_lcd_initialized = 1;
00572   }
00573
00574   return(LCD_OK);
00575 }
00576
00577 /**
00578   * @brief  De-initialize the DSI LCD.
00579   * @retval LCD state
00580   */
00581 uint8_t BSP_DSI_LCD_DeInit(void)
00582 {
00583   if(bsp_lcd_initialized == 1)
00584   {
00585     /* Disable DSI wrapper */
00586     __HAL_DSI_WRAPPER_DISABLE(&hdsi_eval);
00587
00588     /* Set display off */
00589     if(HAL_DSI_ShortWrite(&hdsi_eval,
00590                           0,
00591                           DSI_DCS_SHORT_PKT_
WRITE_P0,
00592                           DSI_SET_DISPLAY_OF
F,
00593                           0x0) != HAL_OK)
00594     {
00595       return(LCD_ERROR);
00596     }
00597
00598     /* Wait before entering in sleep mode */
00599     HAL_Delay(2000);
00600
```

```
00601     /* Put LCD in sleep mode */
00602     if(HAL_DSI_ShortWrite(&hdsi_eval,
00603                           0,
00604                           DSI_DCS_SHORT_PKT_
WRITE_P0,
00605                           DSI_ENTER_SLEEP_MO
DE,
00606                           0x0) != HAL_OK)
00607     {
00608       return(LCD_ERROR);
00609     }
00610
00611     HAL_Delay(120);
00612
00613     /* De-initialize DSI */
00614     if(HAL_DSI_DeInit(&hdsi_eval) != HAL_OK)
00615     {
00616       return(LCD_ERROR);
00617     }
00618
00619     /* De-initialize LTDC */
00620     if(HAL_LTDC_DeInit(&hltdc_eval) != HAL_O
K)
00621     {
00622       return(LCD_ERROR);
00623     }
00624
00625     /* De-initialize GFXMMU */
00626     if(HAL_GFXMMU_DeInit(&hgfxmmu_eval) != H
AL_OK)
00627     {
00628       return(LCD_ERROR);
00629     }
00630
00631     /* Call MSP de-initialize function */
00632     BSP_DSI_LCD_MspDeInit();
00633
```

```
00634     bsp_lcd_initialized = 0;
00635   }
00636
00637   return(LCD_OK);
00638 }
00639
00640 /**
00641   * @brief  Gets the LCD X size.
00642   * @retval Used LCD X size
00643   */
00644 uint32_t BSP_DSI_LCD_GetXSize(void)
00645 {
00646   return (lcd_x_size);
00647 }
00648
00649 /**
00650   * @brief  Gets the LCD Y size.
00651   * @retval Used LCD Y size
00652   */
00653 uint32_t BSP_DSI_LCD_GetYSize(void)
00654 {
00655   return (lcd_y_size);
00656 }
00657
00658 /**
00659   * @brief  Selects the LCD Layer.
00660   * @param  LayerIndex: Layer foreground (1)
 or background (0)
00661   * @note : Only backgroung layer can be use
d.
00662   * @retval LCD state
00663   */
00664 uint8_t BSP_DSI_LCD_SelectLayer(uint32_t Lay
erIndex)
00665 {
00666   uint8_t status = LCD_OK;
00667
```

```
00668   if(LayerIndex == LTDC_ACTIVE_LAYER_BACKGRO
UND)
00669   {
00670     ActiveLayer = LayerIndex;
00671   }
00672   else
00673   {
00674     status = LCD_ERROR;
00675   }
00676   return(status);
00677 }
00678
00679 /**
00680   * @brief  Sets an LCD Layer visible
00681   * @param  LayerIndex: Visible Layer
00682   * @param  State: New state of the specifie
d layer
00683   *          This parameter can be one of th
e following values:
00684   *            @arg  ENABLE
00685   *            @arg  DISABLE
00686   * @note : Only backgroung layer can be use
d.
00687   * @retval LCD state
00688   */
00689 uint8_t BSP_DSI_LCD_SetLayerVisible(uint32_t
 LayerIndex, FunctionalState State)
00690 {
00691   uint8_t status = LCD_OK;
00692
00693   if(LayerIndex == LTDC_ACTIVE_LAYER_BACKGRO
UND)
00694   {
00695     if(State == ENABLE)
00696     {
00697       __HAL_LTDC_LAYER_ENABLE(&(hltdc_eval),
 LayerIndex);
```

```
00698     }
00699     else
00700     {
00701       __HAL_LTDC_LAYER_DISABLE(&(hltdc_eval)
, LayerIndex);
00702     }
00703     __HAL_LTDC_RELOAD_IMMEDIATE_CONFIG(&(hlt
dc_eval));
00704   }
00705   else
00706   {
00707     status = LCD_ERROR;
00708   }
00709   return(status);
00710 }
00711
00712 /**
00713   * @brief  Configures the transparency.
00714   * @param  LayerIndex: Layer foreground or
background.
00715   * @param  Transparency: Transparency
00716   *           This parameter must be a numbe
r between Min_Data = 0x00 and Max_Data = 0xFF
00717   * @note : Only backgroung layer can be use
d.
00718   * @retval LCD state
00719   */
00720 uint8_t BSP_DSI_LCD_SetTransparency(uint32_t
 LayerIndex, uint8_t Transparency)
00721 {
00722   uint8_t status = LCD_OK;
00723
00724   if(LayerIndex == LTDC_ACTIVE_LAYER_BACKGRO
UND)
00725   {
00726     HAL_LTDC_SetAlpha(&(hltdc_eval), Transpa
rency, LayerIndex);
```

```
00727   }
00728   else
00729   {
00730     status = LCD_ERROR;
00731   }
00732   return(status);
00733 }
00734
00735 /**
00736   * @brief  Configures and sets the color ke
ying.
00737   * @param  LayerIndex: Layer foreground (1)
 or background (0)
00738   * @param  RGBValue: Color reference
00739   * @note : Only backgroung layer can be use
d.
00740   * @retval LCD state
00741   */
00742 uint8_t BSP_DSI_LCD_SetColorKeying(uint32_t
LayerIndex, uint32_t RGBValue)
00743 {
00744   uint8_t status = LCD_OK;
00745
00746   if(LayerIndex == LTDC_ACTIVE_LAYER_BACKGRO
UND)
00747   {
00748     /* Configure and Enable the color Keying
 for LCD Layer */
00749     HAL_LTDC_ConfigColorKeying(&(hltdc_eval)
, RGBValue, LayerIndex);
00750     HAL_LTDC_EnableColorKeying(&(hltdc_eval)
, LayerIndex);
00751   }
00752   else
00753   {
00754     status = LCD_ERROR;
00755   }
```

```
00756   return(status);
00757 }
00758
00759 /**
00760   * @brief  Disables the color keying.
00761   * @param  LayerIndex: Layer foreground (1)
 or background (0)
00762   * @note : Only backgroung layer can be use
d.
00763   * @retval LCD state
00764   */
00765 uint8_t BSP_DSI_LCD_ResetColorKeying(uint32_
t LayerIndex)
00766 {
00767   uint8_t status = LCD_OK;
00768
00769   if(LayerIndex == LTDC_ACTIVE_LAYER_BACKGRO
UND)
00770   {
00771     /* Disable the color Keying for LCD Laye
r */
00772     HAL_LTDC_DisableColorKeying(&(hltdc_eval
), LayerIndex);
00773   }
00774   else
00775   {
00776     status = LCD_ERROR;
00777   }
00778   return(status);
00779 }
00780
00781 /**
00782   * @brief  Sets the LCD text color.
00783   * @param  Color: Text color code ARGB8888
00784   */
00785 void BSP_DSI_LCD_SetTextColor(uint32_t Color
)
```

```
00786 {
00787   DrawProp[ActiveLayer].TextColor = Color;
00788 }
00789
00790 /**
00791   * @brief  Gets the LCD text color.
00792   * @retval Used text color.
00793   */
00794 uint32_t BSP_DSI_LCD_GetTextColor(void)
00795 {
00796   return DrawProp[ActiveLayer].TextColor;
00797 }
00798
00799 /**
00800   * @brief  Sets the LCD background color.
00801   * @param  Color: Layer background color co
de ARGB8888
00802   */
00803 void BSP_DSI_LCD_SetBackColor(uint32_t Color
)
00804 {
00805   DrawProp[ActiveLayer].BackColor = Color;
00806 }
00807
00808 /**
00809   * @brief  Gets the LCD background color.
00810   * @retval Used background color
00811   */
00812 uint32_t BSP_DSI_LCD_GetBackColor(void)
00813 {
00814   return DrawProp[ActiveLayer].BackColor;
00815 }
00816
00817 /**
00818   * @brief  Sets the LCD text font.
00819   * @param  fonts: Layer font to be used
00820   */
```

```
00821 void BSP_DSI_LCD_SetFont(sFONT *fonts)
00822 {
00823   DrawProp[ActiveLayer].pFont = fonts;
00824 }
00825
00826 /**
00827   * @brief  Gets the LCD text font.
00828   * @retval Used layer font
00829   */
00830 sFONT *BSP_DSI_LCD_GetFont(void)
00831 {
00832   return DrawProp[ActiveLayer].pFont;
00833 }
00834
00835 /**
00836   * @brief  Reads an LCD pixel.
00837   * @param  Xpos: X position
00838   * @param  Ypos: Y position
00839   * @retval ARGB8888 pixel color
00840   */
00841 uint32_t BSP_DSI_LCD_ReadPixel(uint16_t Xpos
, uint16_t Ypos)
00842 {
00843   uint32_t ret = 0;
00844
00845   /* Read value of corresponding pixel */
00846   /* We have 768 pixels per line and 4 bytes
 per pixel so 3072 bytes per line */
00847   ret = *(__IO uint32_t*) (hltdc_eval.LayerC
fg[ActiveLayer].FBStartAdress + (4*(Ypos*768 + Xpo
s)));
00848
00849   /* Return pixel value */
00850   return ret;
00851 }
00852
00853 /**
```

```
00854   * @brief  Clears the whole currently activ
e layer of LTDC.
00855   * @param  Color: Color of the background (
in ARGB8888 format)
00856   */
00857 void BSP_DSI_LCD_Clear(uint32_t Color)
00858 {
00859   /* Clear the LCD */
00860   /* Offset line is 768 - 390 = 378 */
00861   LL_FillBuffer(ActiveLayer, (uint32_t *)(hl
tdc_eval.LayerCfg[ActiveLayer].FBStartAdress), 390
, 390, 378, Color);
00862 }
00863
00864 /**
00865   * @brief  Clears the selected line in curr
ently active layer.
00866   * @param  Line: Line to be cleared
00867   */
00868 void BSP_DSI_LCD_ClearStringLine(uint32_t Li
ne)
00869 {
00870   uint32_t color_backup = DrawProp[ActiveLay
er].TextColor;
00871   DrawProp[ActiveLayer].TextColor = DrawProp[
ActiveLayer].BackColor;
00872
00873   /* Draw rectangle with background color */
00874   BSP_DSI_LCD_FillRect(0, (Line * DrawProp[A
ctiveLayer].pFont->Height), 390, DrawProp[ActiveLa
yer].pFont->Height);
00875
00876   DrawProp[ActiveLayer].TextColor = color_ba
ckup;
00877   BSP_DSI_LCD_SetTextColor(DrawProp[ActiveLa
yer].TextColor);
00878 }
```

```
00879
00880 /**
00881   * @brief  Displays one character in curren
tly active layer.
00882   * @param  Xpos: Start column address
00883   * @param  Ypos: Line where to display the
character shape.
00884   * @param  Ascii: Character ascii code
00885   *           This parameter must be a numbe
r between Min_Data = 0x20 and Max_Data = 0x7E
00886   */
00887 void BSP_DSI_LCD_DisplayChar(uint16_t Xpos,
uint16_t Ypos, uint8_t Ascii)
00888 {
00889   DrawChar(Xpos, Ypos, &DrawProp[ActiveLayer
].pFont->table[(Ascii-' ') * DrawProp[ActiveLayer].
pFont->Height * ((DrawProp[ActiveLayer].pFont->Wid
th + 7) / 8)]);
00890 }
00891
00892 /**
00893   * @brief  Displays characters in currently
 active layer.
00894   * @param  Xpos: X position (in pixel)
00895   * @param  Ypos: Y position (in pixel)
00896   * @param  Text: Pointer to string to displ
ay on LCD
00897   * @param  Mode: Display mode
00898   *          This parameter can be one of th
e following values:
00899   *            @arg  CENTER_MODE
00900   *            @arg  RIGHT_MODE
00901   *            @arg  LEFT_MODE
00902   */
00903 void BSP_DSI_LCD_DisplayStringAt(uint16_t Xp
os, uint16_t Ypos, uint8_t *Text, Text_AlignModeTy
pdef Mode)
```

```
00904 {
00905   uint16_t refcolumn = 1, i = 0;
00906   uint32_t size = 0, xsize = 0;
00907   uint8_t  *ptr = Text;
00908
00909   /* Get the text size */
00910   while (*ptr++) size ++ ;
00911
00912   /* Characters number per line */
00913   xsize = (390/DrawProp[ActiveLayer].pFont->
Width);
00914
00915   switch (Mode)
00916   {
00917   case CENTER_MODE:
00918     {
00919       refcolumn = Xpos + ((xsize - size)* Dr
awProp[ActiveLayer].pFont->Width) / 2;
00920       break;
00921     }
00922   case LEFT_MODE:
00923     {
00924       refcolumn = Xpos;
00925       break;
00926     }
00927   case RIGHT_MODE:
00928     {
00929       refcolumn = - Xpos + ((xsize - size)*D
rawProp[ActiveLayer].pFont->Width);
00930       break;
00931     }
00932   default:
00933     {
00934       refcolumn = Xpos;
00935       break;
00936     }
00937   }
```

```
00938
00939   /* Check that the Start column is located
in the screen */
00940   if ((refcolumn < 1) || (refcolumn >= 0x800
0))
00941   {
00942     refcolumn = 1;
00943   }
00944
00945   /* Send the string character by character
on LCD */
00946   while ((*Text != 0) & (((390 - (i*DrawProp[
ActiveLayer].pFont->Width)) & 0xFFFF) >= DrawProp[
ActiveLayer].pFont->Width))
00947   {
00948     /* Display one character on LCD */
00949     BSP_DSI_LCD_DisplayChar(refcolumn, Ypos,
 *Text);
00950     /* Decrement the column position by 16 */

00951     refcolumn += DrawProp[ActiveLayer].pFont
->Width;
00952
00953     /* Point on the next character */
00954     Text++;
00955     i++;
00956   }
00957
00958 }
00959
00960 /**
00961   * @brief  Displays string on specified lin
e of the LCD.
00962   * @param  Line: Line where to display the
character shape
00963   * @param  ptr: Pointer to string to displa
y on LCD
```

```
00964   */
00965 void BSP_DSI_LCD_DisplayStringAtLine(uint16_
t Line, uint8_t *ptr)
00966 {
00967   BSP_DSI_LCD_DisplayStringAt(0, Line * (((s
FONT *)BSP_DSI_LCD_GetFont())->Height), ptr, CENTE
R_MODE);
00968 }
00969
00970 /**
00971   * @brief  Draws an horizontal line in curr
ently active layer.
00972   * @param  Xpos: X position
00973   * @param  Ypos: Y position
00974   * @param  Length: Line length
00975   */
00976 void BSP_DSI_LCD_DrawHLine(uint16_t Xpos, ui
nt16_t Ypos, uint16_t Length)
00977 {
00978   uint32_t  Xaddress = 0;
00979
00980   /* Get the line address */
00981   Xaddress = (hltdc_eval.LayerCfg[ActiveLayer
].FBStartAdress) + 4*(768*Ypos + Xpos);
00982
00983   /* Write line */
00984   LL_FillBuffer(ActiveLayer, (uint32_t *)Xad
dress, Length, 1, (768 - Length), DrawProp[ActiveL
ayer].TextColor);
00985 }
00986
00987 /**
00988   * @brief  Draws a vertical line in current
ly active layer.
00989   * @param  Xpos: X position
00990   * @param  Ypos: Y position
00991   * @param  Length: Line length
```

```
00992   */
00993 void BSP_DSI_LCD_DrawVLine(uint16_t Xpos, ui
nt16_t Ypos, uint16_t Length)
00994 {
00995   uint32_t  Xaddress = 0;
00996
00997   /* Get the line address */
00998   Xaddress = (hltdc_eval.LayerCfg[ActiveLayer
].FBStartAdress) + 4*(768*Ypos + Xpos);
00999
01000   /* Write line */
01001   LL_FillBuffer(ActiveLayer, (uint32_t *)Xad
dress, 1, Length, (768 - 1), DrawProp[ActiveLayer]
.TextColor);
01002 }
01003
01004 /**
01005   * @brief  Draws an uni-line (between two p
oints) in currently active layer.
01006   * @param  x1: Point 1 X position
01007   * @param  y1: Point 1 Y position
01008   * @param  x2: Point 2 X position
01009   * @param  y2: Point 2 Y position
01010   */
01011 void BSP_DSI_LCD_DrawLine(uint16_t x1, uint1
6_t y1, uint16_t x2, uint16_t y2)
01012 {
01013   int16_t deltax = 0, deltay = 0, x = 0, y =
 0, xinc1 = 0, xinc2 = 0,
01014   yinc1 = 0, yinc2 = 0, den = 0, num = 0, nu
madd = 0, numpixels = 0,
01015   curpixel = 0;
01016
01017   deltax = ABS(x2 - x1);        /* The diffe
rence between the x's */
01018   deltay = ABS(y2 - y1);        /* The diffe
rence between the y's */
```

```
01019   x = x1;                         /* Start x o
ff at the first pixel */
01020   y = y1;                         /* Start y o
ff at the first pixel */
01021
01022   if (x2 >= x1)                   /* The x-val
ues are increasing */
01023   {
01024     xinc1 = 1;
01025     xinc2 = 1;
01026   }
01027   else                            /* The x-val
ues are decreasing */
01028   {
01029     xinc1 = -1;
01030     xinc2 = -1;
01031   }
01032
01033   if (y2 >= y1)                   /* The y-val
ues are increasing */
01034   {
01035     yinc1 = 1;
01036     yinc2 = 1;
01037   }
01038   else                            /* The y-val
ues are decreasing */
01039   {
01040     yinc1 = -1;
01041     yinc2 = -1;
01042   }
01043
01044   if (deltax >= deltay)           /* There is
at least one x-value for every y-value */
01045   {
01046     xinc1 = 0;                    /* Don't cha
nge the x when numerator >= denominator */
01047     yinc2 = 0;                    /* Don't cha
```

```
nge the y for every iteration */
01048     den = deltax;
01049     num = deltax / 2;
01050     numadd = deltay;
01051     numpixels = deltax;         /* There are
 more x-values than y-values */
01052   }
01053   else                          /* There is
at least one y-value for every x-value */
01054   {
01055     xinc2 = 0;                  /* Don't cha
nge the x for every iteration */
01056     yinc1 = 0;                  /* Don't cha
nge the y when numerator >= denominator */
01057     den = deltay;
01058     num = deltay / 2;
01059     numadd = deltax;
01060     numpixels = deltay;         /* There are
 more y-values than x-values */
01061   }
01062
01063   for (curpixel = 0; curpixel <= numpixels;
curpixel++)
01064   {
01065     BSP_DSI_LCD_DrawPixel(x, y, DrawProp[Act
iveLayer].TextColor);   /* Draw the current pixel
*/
01066     num += numadd;
  /* Increase the numerator by the top of the frac
tion */
01067     if (num >= den)
  /* Check if numerator >= denominator */
01068     {
01069       num -= den;
  /* Calculate the new numerator value */
01070       x += xinc1;
  /* Change the x as appropriate */
```

```
01071       y += yinc1;
  /* Change the y as appropriate */
01072     }
01073     x += xinc2;
  /* Change the x as appropriate */
01074     y += yinc2;
  /* Change the y as appropriate */
01075   }
01076 }
01077
01078 /**
01079   * @brief  Draws a rectangle in currently a
ctive layer.
01080   * @param  Xpos: X position
01081   * @param  Ypos: Y position
01082   * @param  Width: Rectangle width
01083   * @param  Height: Rectangle height
01084   */
01085 void BSP_DSI_LCD_DrawRect(uint16_t Xpos, uin
t16_t Ypos, uint16_t Width, uint16_t Height)
01086 {
01087   /* Draw horizontal lines */
01088   BSP_DSI_LCD_DrawHLine(Xpos, Ypos, Width);
01089   BSP_DSI_LCD_DrawHLine(Xpos, (Ypos+ Height)
, Width);
01090
01091   /* Draw vertical lines */
01092   BSP_DSI_LCD_DrawVLine(Xpos, Ypos, Height);
01093   BSP_DSI_LCD_DrawVLine((Xpos + Width), Ypos
, Height);
01094 }
01095
01096 /**
01097   * @brief  Draws a circle in currently acti
ve layer.
01098   * @param  Xpos: X position
01099   * @param  Ypos: Y position
```

```
01100   * @param  Radius: Circle radius
01101   */
01102 void BSP_DSI_LCD_DrawCircle(uint16_t Xpos, u
int16_t Ypos, uint16_t Radius)
01103 {
01104   int32_t   D;    /* Decision Variable */
01105   uint32_t  CurX; /* Current X Value */
01106   uint32_t  CurY; /* Current Y Value */
01107
01108   D = 3 - (Radius << 1);
01109   CurX = 0;
01110   CurY = Radius;
01111
01112   while (CurX <= CurY)
01113   {
01114     BSP_DSI_LCD_DrawPixel((Xpos + CurX), (Yp
os - CurY), DrawProp[ActiveLayer].TextColor);
01115
01116     BSP_DSI_LCD_DrawPixel((Xpos - CurX), (Yp
os - CurY), DrawProp[ActiveLayer].TextColor);
01117
01118     BSP_DSI_LCD_DrawPixel((Xpos + CurY), (Yp
os - CurX), DrawProp[ActiveLayer].TextColor);
01119
01120     BSP_DSI_LCD_DrawPixel((Xpos - CurY), (Yp
os - CurX), DrawProp[ActiveLayer].TextColor);
01121
01122     BSP_DSI_LCD_DrawPixel((Xpos + CurX), (Yp
os + CurY), DrawProp[ActiveLayer].TextColor);
01123
01124     BSP_DSI_LCD_DrawPixel((Xpos - CurX), (Yp
os + CurY), DrawProp[ActiveLayer].TextColor);
01125
01126     BSP_DSI_LCD_DrawPixel((Xpos + CurY), (Yp
os + CurX), DrawProp[ActiveLayer].TextColor);
01127
01128     BSP_DSI_LCD_DrawPixel((Xpos - CurY), (Yp
```

```
os + CurX), DrawProp[ActiveLayer].TextColor);
01129
01130     if (D < 0)
01131     {
01132       D += (CurX << 2) + 6;
01133     }
01134     else
01135     {
01136       D += ((CurX - CurY) << 2) + 10;
01137       CurY--;
01138     }
01139     CurX++;
01140   }
01141 }
01142
01143 /**
01144   * @brief  Draws an poly-line (between many
 points) in currently active layer.
01145   * @param  Points: Pointer to the points ar
ray
01146   * @param  PointCount: Number of points
01147   */
01148 void BSP_DSI_LCD_DrawPolygon(pPoint Points,
uint16_t PointCount)
01149 {
01150   int16_t X = 0, Y = 0;
01151
01152   if(PointCount < 2)
01153   {
01154     return;
01155   }
01156
01157   BSP_DSI_LCD_DrawLine(Points->X, Points->Y,
 (Points+PointCount-1)->X, (Points+PointCount-1)->
Y);
01158
01159   while(--PointCount)
```

```
01160   {
01161     X = Points->X;
01162     Y = Points->Y;
01163     Points++;
01164     BSP_DSI_LCD_DrawLine(X, Y, Points->X, Po
ints->Y);
01165   }
01166 }
01167
01168 /**
01169   * @brief  Draws an ellipse on LCD in curre
ntly active layer.
01170   * @param  Xpos: X position
01171   * @param  Ypos: Y position
01172   * @param  XRadius: Ellipse X radius
01173   * @param  YRadius: Ellipse Y radius
01174   */
01175 void BSP_DSI_LCD_DrawEllipse(int Xpos, int Y
pos, int XRadius, int YRadius)
01176 {
01177   int x = 0, y = -YRadius, err = 2-2*XRadius
, e2;
01178   float K = 0, rad1 = 0, rad2 = 0;
01179
01180   rad1 = XRadius;
01181   rad2 = YRadius;
01182
01183   K = (float)(rad2/rad1);
01184
01185   do {
01186     BSP_DSI_LCD_DrawPixel((Xpos-(uint16_t)(x
/K)), (Ypos+y), DrawProp[ActiveLayer].TextColor);
01187     BSP_DSI_LCD_DrawPixel((Xpos+(uint16_t)(x
/K)), (Ypos+y), DrawProp[ActiveLayer].TextColor);
01188     BSP_DSI_LCD_DrawPixel((Xpos+(uint16_t)(x
/K)), (Ypos-y), DrawProp[ActiveLayer].TextColor);
01189     BSP_DSI_LCD_DrawPixel((Xpos-(uint16_t)(x
```

```
/K)), (Ypos-y), DrawProp[ActiveLayer].TextColor);
01190
01191     e2 = err;
01192     if (e2 <= x) {
01193       err += ++x*2+1;
01194       if (-y == x && e2 <= y) e2 = 0;
01195     }
01196     if (e2 > y) err += ++y*2+1;
01197   }
01198   while (y <= 0);
01199 }
01200
01201 /**
01202   * @brief  Draws a bitmap picture loaded in
 the internal Flash (32 bpp) in currently active l
ayer.
01203   * @param  Xpos: Bmp X position in the LCD
01204   * @param  Ypos: Bmp Y position in the LCD
01205   * @param  pbmp: Pointer to Bmp picture add
ress in the internal Flash
01206   */
01207 void BSP_DSI_LCD_DrawBitmap(uint32_t Xpos, u
int32_t Ypos, uint8_t *pbmp)
01208 {
01209   uint32_t index = 0, width = 0, height = 0,
 bit_pixel = 0;
01210   uint32_t Address;
01211   uint32_t InputColorMode = 0;
01212
01213   /* Get bitmap data address offset */
01214   index = *(__IO uint16_t *) (pbmp + 10);
01215   index |= (*(__IO uint16_t *) (pbmp + 12))
<< 16;
01216
01217   /* Read bitmap width */
01218   width = *(uint16_t *) (pbmp + 18);
01219   width |= (*(uint16_t *) (pbmp + 20)) << 16
```

```
;
01220
01221   /* Read bitmap height */
01222   height = *(uint16_t *) (pbmp + 22);
01223   height |= (*(uint16_t *) (pbmp + 24)) << 1
6;
01224
01225   /* Read bit/pixel */
01226   bit_pixel = *(uint16_t *) (pbmp + 28);
01227
01228   /* Set the address */
01229   Address = hltdc_eval.LayerCfg[ActiveLayer]
.FBStartAdress + (((768*Ypos) + Xpos)*(4));
01230
01231   /* Get the layer pixel format */
01232   if ((bit_pixel/8) == 4)
01233   {
01234     InputColorMode = DMA2D_INPUT_ARGB8888;
01235   }
01236   else if ((bit_pixel/8) == 2)
01237   {
01238     InputColorMode = DMA2D_INPUT_RGB565;
01239   }
01240   else
01241   {
01242     InputColorMode = DMA2D_INPUT_RGB888;
01243   }
01244
01245   /* Bypass the bitmap header */
01246   pbmp += (index + (width * (height - 1) * (
bit_pixel/8)));
01247
01248   /* Convert picture to ARGB8888 pixel forma
t */
01249   for(index=0; index < height; index++)
01250   {
01251     /* Pixel format conversion */
```

```
01252     LL_ConvertLineToARGB8888((uint32_t *)pbm
p, (uint32_t *)Address, width, InputColorMode);
01253
01254     /* Increment the source and destination
buffers */
01255     Address+=  (768*4);
01256     pbmp -= width*(bit_pixel/8);
01257   }
01258 }
01259
01260 /**
01261   * @brief  Draws a full rectangle in curren
tly active layer.
01262   * @param  Xpos: X position
01263   * @param  Ypos: Y position
01264   * @param  Width: Rectangle width
01265   * @param  Height: Rectangle height
01266   */
01267 void BSP_DSI_LCD_FillRect(uint16_t Xpos, uin
t16_t Ypos, uint16_t Width, uint16_t Height)
01268 {
01269   uint32_t  Xaddress = 0;
01270
01271   /* Set the text color */
01272   BSP_DSI_LCD_SetTextColor(DrawProp[ActiveLa
yer].TextColor);
01273
01274   /* Get the rectangle start address */
01275   Xaddress = (hltdc_eval.LayerCfg[ActiveLayer
].FBStartAdress) + 4*(768*Ypos + Xpos);
01276
01277   /* Fill the rectangle */
01278   LL_FillBuffer(ActiveLayer, (uint32_t *)Xad
dress, Width, Height, (768 - Width), DrawProp[Acti
veLayer].TextColor);
01279 }
01280
```

```
01281 /**
01282   * @brief  Draws a full circle in currently
 active layer.
01283   * @param  Xpos: X position
01284   * @param  Ypos: Y position
01285   * @param  Radius: Circle radius
01286   */
01287 void BSP_DSI_LCD_FillCircle(uint16_t Xpos, u
int16_t Ypos, uint16_t Radius)
01288 {
01289   int32_t  D;     /* Decision Variable */
01290   uint32_t  CurX; /* Current X Value */
01291   uint32_t  CurY; /* Current Y Value */
01292
01293   D = 3 - (Radius << 1);
01294
01295   CurX = 0;
01296   CurY = Radius;
01297
01298   BSP_DSI_LCD_SetTextColor(DrawProp[ActiveLa
yer].TextColor);
01299
01300   while (CurX <= CurY)
01301   {
01302     if(CurY > 0)
01303     {
01304       BSP_DSI_LCD_DrawHLine(Xpos - CurY, Ypo
s + CurX, 2*CurY);
01305       BSP_DSI_LCD_DrawHLine(Xpos - CurY, Ypo
s - CurX, 2*CurY);
01306     }
01307
01308     if(CurX > 0)
01309     {
01310       BSP_DSI_LCD_DrawHLine(Xpos - CurX, Ypo
s - CurY, 2*CurX);
01311       BSP_DSI_LCD_DrawHLine(Xpos - CurX, Ypo
```

```
s + CurY, 2*CurX);
01312     }
01313     if (D < 0)
01314     {
01315       D += (CurX << 2) + 6;
01316     }
01317     else
01318     {
01319       D += ((CurX - CurY) << 2) + 10;
01320       CurY--;
01321     }
01322     CurX++;
01323   }
01324
01325   BSP_DSI_LCD_SetTextColor(DrawProp[ActiveLa
yer].TextColor);
01326   BSP_DSI_LCD_DrawCircle(Xpos, Ypos, Radius)
;
01327 }
01328
01329 /**
01330   * @brief  Draws a full poly-line (between
many points) in currently active layer.
01331   * @param  Points: Pointer to the points ar
ray
01332   * @param  PointCount: Number of points
01333   */
01334 void BSP_DSI_LCD_FillPolygon(pPoint Points,
uint16_t PointCount)
01335 {
01336   int16_t X = 0, Y = 0, X2 = 0, Y2 = 0, X_ce
nter = 0, Y_center = 0, X_first = 0, Y_first = 0,
pixelX = 0, pixelY = 0, counter = 0;
01337   uint16_t  IMAGE_LEFT = 0, IMAGE_RIGHT = 0,
 IMAGE_TOP = 0, IMAGE_BOTTOM = 0;
01338
01339   IMAGE_LEFT = IMAGE_RIGHT = Points->X;
```

```
01340   IMAGE_TOP= IMAGE_BOTTOM = Points->Y;
01341
01342   for(counter = 1; counter < PointCount; cou
nter++)
01343   {
01344     pixelX = POLY_X(counter);
01345     if(pixelX < IMAGE_LEFT)
01346     {
01347       IMAGE_LEFT = pixelX;
01348     }
01349     if(pixelX > IMAGE_RIGHT)
01350     {
01351       IMAGE_RIGHT = pixelX;
01352     }
01353
01354     pixelY = POLY_Y(counter);
01355     if(pixelY < IMAGE_TOP)
01356     {
01357       IMAGE_TOP = pixelY;
01358     }
01359     if(pixelY > IMAGE_BOTTOM)
01360     {
01361       IMAGE_BOTTOM = pixelY;
01362     }
01363   }
01364
01365   if(PointCount < 2)
01366   {
01367     return;
01368   }
01369
01370   X_center = (IMAGE_LEFT + IMAGE_RIGHT)/2;
01371   Y_center = (IMAGE_BOTTOM + IMAGE_TOP)/2;
01372
01373   X_first = Points->X;
01374   Y_first = Points->Y;
01375
```

```
01376   while(--PointCount)
01377   {
01378     X = Points->X;
01379     Y = Points->Y;
01380     Points++;
01381     X2 = Points->X;
01382     Y2 = Points->Y;
01383 
01384     FillTriangle(X, X2, X_center, Y, Y2, Y_c
enter);
01385     FillTriangle(X, X_center, X2, Y, Y_cente
r, Y2);
01386     FillTriangle(X_center, X2, X, Y_center, 
Y2, Y);
01387   }
01388 
01389   FillTriangle(X_first, X2, X_center, Y_firs
t, Y2, Y_center);
01390   FillTriangle(X_first, X_center, X2, Y_firs
t, Y_center, Y2);
01391   FillTriangle(X_center, X2, X_first, Y_cent
er, Y2, Y_first);
01392 }
01393 
01394 /**
01395   * @brief  Draws a full ellipse in currentl
y active layer.
01396   * @param  Xpos: X position
01397   * @param  Ypos: Y position
01398   * @param  XRadius: Ellipse X radius
01399   * @param  YRadius: Ellipse Y radius
01400   */
01401 void BSP_DSI_LCD_FillEllipse(int Xpos, int Y
pos, int XRadius, int YRadius)
01402 {
01403   int x = 0, y = -YRadius, err = 2-2*XRadius
, e2;
```

```
01404   float K = 0, rad1 = 0, rad2 = 0;
01405
01406   rad1 = XRadius;
01407   rad2 = YRadius;
01408
01409   K = (float)(rad2/rad1);
01410
01411   do
01412   {
01413     BSP_DSI_LCD_DrawHLine((Xpos-(uint16_t)(x
/K)), (Ypos+y), (2*(uint16_t)(x/K) + 1));
01414     BSP_DSI_LCD_DrawHLine((Xpos-(uint16_t)(x
/K)), (Ypos-y), (2*(uint16_t)(x/K) + 1));
01415
01416     e2 = err;
01417     if (e2 <= x)
01418     {
01419       err += ++x*2+1;
01420       if (-y == x && e2 <= y) e2 = 0;
01421     }
01422     if (e2 > y) err += ++y*2+1;
01423   }
01424   while (y <= 0);
01425 }
01426
01427 /**
01428   * @brief  Draws a pixel on LCD.
01429   * @param  Xpos: X position
01430   * @param  Ypos: Y position
01431   * @param  RGB_Code: Pixel color in ARGB mo
de (8-8-8-8)
01432   */
01433 void BSP_DSI_LCD_DrawPixel(uint16_t Xpos, ui
nt16_t Ypos, uint32_t RGB_Code)
01434 {
01435   /* Write pixel */
01436   *(__IO uint32_t*) (hltdc_eval.LayerCfg[Act
```

```
iveLayer].FBStartAdress + (4*(Ypos*768 + Xpos))) =
 RGB_Code;
01437 }
01438
01439 /**
01440   * @brief  Switch back on the display if wa
s switched off by previous call of BSP_DSI_LCD_Dis
playOff().
01441   */
01442 void BSP_DSI_LCD_DisplayOn(void)
01443 {
01444   /* Send Display on DCS command to display
*/
01445   HAL_DSI_ShortWrite(&hdsi_eval,
01446                      0,
01447                      DSI_DCS_SHORT_PKT_WRITE
_P0,
01448                      DSI_SET_DISPLAY_ON,
01449                      0x0);
01450 }
01451
01452 /**
01453   * @brief  Switch Off the display.
01454   */
01455 void BSP_DSI_LCD_DisplayOff(void)
01456 {
01457   /* Send Display off DCS Command to display
 */
01458   HAL_DSI_ShortWrite(&hdsi_eval,
01459                      0,
01460                      DSI_DCS_SHORT_PKT_WRITE
_P0,
01461                      DSI_SET_DISPLAY_OFF,
01462                      0x0);
01463 }
01464
01465 /**
```

```
01466   * @brief  Refresh the display.
01467   */
01468 void BSP_DSI_LCD_Refresh(void)
01469 {
01470   /* Set frame buffer busy */
01471   FrameBufferAvailable = 0;
01472
01473   /* Set tear on */
01474   HAL_DSI_ShortWrite(&hdsi_eval, 0, DSI_DCS_
SHORT_PKT_WRITE_P1, DSI_SET_TEAR_ON, 0x0);
01475 }
01476
01477 /**
01478   * @brief  Check if frame buffer is availab
le.
01479   * @retval LCD_OK if frame buffer is availa
ble else LCD_ERROR (frame buffer busy)
01480   */
01481 uint8_t BSP_DSI_LCD_IsFrameBufferAvailable(v
oid)
01482 {
01483   uint8_t status;
01484   /* Check frame buffer status */
01485   status = (FrameBufferAvailable == 1) ? LCD
_OK : LCD_ERROR;
01486
01487   return(status);
01488 }
01489
01490 /**
01491   * @brief  Set the brightness value
01492   * @param  BrightnessValue: [0% Min (black)
, 100% Max]
01493   */
01494 void BSP_DSI_LCD_SetBrightness(uint8_t Brigh
tnessValue)
01495 {
```

```
01496   /* Send Display on DCS command to display
*/
01497   HAL_DSI_ShortWrite(&hdsi_eval,
01498                      0,
01499                      DSI_DCS_SHORT_PKT_WRITE
_P1,
01500                      0x51, (uint16_t)(Bright
nessValue * 255)/100);
01501 }
01502
01503 /**************************************************
**********************************
01504                            LTDC, DMA2D and DSI B
SP Routines
01505 **************************************************
**********************************/
01506 /**
01507   * @brief  Handles DMA2D interrupt request.
01508   * @note   Application can surcharge if nee
ded this function implementation.
01509   */
01510 __weak void BSP_DSI_LCD_DMA2D_IRQHandler(void
)
01511 {
01512   HAL_DMA2D_IRQHandler(&hdma2d_eval);
01513 }
01514
01515 /**
01516   * @brief  Handles DSI interrupt request.
01517   * @note   Application can surcharge if nee
ded this function implementation.
01518   */
01519 __weak void BSP_DSI_LCD_DSI_IRQHandler(void)
01520 {
01521   HAL_DSI_IRQHandler(&(hdsi_eval));
01522 }
01523
```

```
01524
01525 /**
01526   * @brief  Handles LTDC interrupt request.
01527   * @note   Application can surcharge if nee
ded this function implementation.
01528   */
01529 __weak void BSP_DSI_LCD_LTDC_IRQHandler(void
)
01530 {
01531   HAL_LTDC_IRQHandler(&(hltdc_eval));
01532 }
01533
01534 /**
01535   * @brief  This function handles LTDC Error
 interrupt Handler.
01536   * @note   Application can surcharge if nee
ded this function implementation.
01537   */
01538
01539 __weak void BSP_DSI_LCD_LTDC_ER_IRQHandler(v
oid)
01540 {
01541   HAL_LTDC_IRQHandler(&(hltdc_eval));
01542 }
01543
01544 /**
01545   * @brief  De-Initializes the BSP LCD Msp.
01546   * @note   Application can surcharge if nee
ded this function implementation.
01547   */
01548 __weak void BSP_DSI_LCD_MspDeInit(void)
01549 {
01550   /* Disable IRQ of LTDC IP */
01551   HAL_NVIC_DisableIRQ(LTDC_IRQn);
01552   HAL_NVIC_DisableIRQ(LTDC_ER_IRQn);
01553
01554   /* Disable IRQ of DMA2D IP */
```

```
01555   HAL_NVIC_DisableIRQ(DMA2D_IRQn);
01556
01557   /* Disable IRQ of DSI IP */
01558   HAL_NVIC_DisableIRQ(DSI_IRQn);
01559
01560   /* Disable HSE used for DSI PLL */
01561   if(bsp_lcd_hse_to_disable == 1)
01562   {
01563     RCC_OscInitTypeDef RCC_OscInitStruct;
01564     RCC_OscInitStruct.OscillatorType = RCC_O
SCILLATORTYPE_HSE;
01565     RCC_OscInitStruct.HSEState       = RCC_H
SE_OFF;
01566     RCC_OscInitStruct.PLL.PLLState   = RCC_P
LL_NONE;
01567     if (HAL_RCC_OscConfig(&RCC_OscInitStruct
) != HAL_OK)
01568     {
01569       while(1);
01570     }
01571     bsp_lcd_hse_to_disable = 0;
01572   }
01573
01574   /* Force and let in reset state GFXMMU, LT
DC, DMA2D and DSI Host + Wrapper IPs */
01575   __HAL_RCC_GFXMMU_FORCE_RESET();
01576   __HAL_RCC_LTDC_FORCE_RESET();
01577   __HAL_RCC_DMA2D_FORCE_RESET();
01578   __HAL_RCC_DSI_FORCE_RESET();
01579
01580   /* Disable the GFXMMU, LTDC, DMA2D and DSI
 Host and Wrapper clocks */
01581   __HAL_RCC_GFXMMU_CLK_DISABLE();
01582   __HAL_RCC_LTDC_CLK_DISABLE();
01583   __HAL_RCC_DMA2D_CLK_DISABLE();
01584   __HAL_RCC_DSI_CLK_DISABLE();
01585 }
```

```
01586
01587 /**
01588   * @brief  Initialize the BSP LCD Msp.
01589   * @note   Application can surcharge if nee
ded this function implementation.
01590   */
01591 __weak void BSP_DSI_LCD_MspInit(void)
01592 {
01593   /* Enable the GFXMMU clock */
01594   __HAL_RCC_GFXMMU_CLK_ENABLE();
01595
01596   /* Reset of GFXMMU IP */
01597   __HAL_RCC_GFXMMU_FORCE_RESET();
01598   __HAL_RCC_GFXMMU_RELEASE_RESET();
01599
01600   /* Enable the LTDC clock */
01601   __HAL_RCC_LTDC_CLK_ENABLE();
01602
01603   /* Reset of LTDC IP */
01604   __HAL_RCC_LTDC_FORCE_RESET();
01605   __HAL_RCC_LTDC_RELEASE_RESET();
01606
01607   /* Enable the DMA2D clock */
01608   __HAL_RCC_DMA2D_CLK_ENABLE();
01609
01610   /* Reset of DMA2D IP */
01611   __HAL_RCC_DMA2D_FORCE_RESET();
01612   __HAL_RCC_DMA2D_RELEASE_RESET();
01613
01614   /* Enable DSI Host and wrapper clocks */
01615   __HAL_RCC_DSI_CLK_ENABLE();
01616
01617   /* Reset the DSI Host and wrapper */
01618   __HAL_RCC_DSI_FORCE_RESET();
01619   __HAL_RCC_DSI_RELEASE_RESET();
01620
01621   /* Configure the clock for the LTDC */
```

```
01622   /* We want DSI PHI at 500MHz */
01623   /* We have only one line => 500Mbps */
01624   /* With 24bits per pixel, equivalent PCLK
is 500/24 = 20.8MHz */
01625   /* We will set PCLK at 15MHz */
01626   /* Following values are OK with MSI = 4MHz
 */
01627   /* (4*60)/(1*4*4) = 15MHz */
01628   RCC_PeriphCLKInitTypeDef  PeriphClkInit;
01629   PeriphClkInit.PeriphClockSelection = RCC_P
ERIPHCLK_LTDC;
01630   PeriphClkInit.PLLSAI2.PLLSAI2Source = RCC_
PLLSOURCE_MSI;
01631   PeriphClkInit.PLLSAI2.PLLSAI2M = 1;
01632   PeriphClkInit.PLLSAI2.PLLSAI2N = 60;
01633   PeriphClkInit.PLLSAI2.PLLSAI2R = RCC_PLLR_
DIV4;
01634   PeriphClkInit.LtdcClockSelection = RCC_LTD
CCLKSOURCE_PLLSAI2_DIV4;
01635   PeriphClkInit.PLLSAI2.PLLSAI2ClockOut = RC
C_PLLSAI2_LTDCCLK;
01636   if(HAL_RCCEx_PeriphCLKConfig(&PeriphClkIni
t) != HAL_OK)
01637   {
01638     while(1);
01639   }
01640
01641   /* Enable HSE used for DSI PLL */
01642   RCC_OscInitTypeDef RCC_OscInitStruct;
01643   HAL_RCC_GetOscConfig(&RCC_OscInitStruct);
01644   if(RCC_OscInitStruct.HSEState == RCC_HSE_O
FF)
01645   {
01646     RCC_OscInitStruct.OscillatorType = RCC_O
SCILLATORTYPE_HSE;
01647     RCC_OscInitStruct.HSEState       = RCC_H
SE_ON;
```

```
01648     RCC_OscInitStruct.PLL.PLLState   = RCC_P
LL_NONE;
01649     if (HAL_RCC_OscConfig(&RCC_OscInitStruct
) != HAL_OK)
01650     {
01651       while(1);
01652     }
01653     bsp_lcd_hse_to_disable = 1;
01654   }
01655
01656   /* NVIC configuration for LTDC interrupts
that are now enabled */
01657   HAL_NVIC_SetPriority(LTDC_IRQn, 3, 0);
01658   HAL_NVIC_EnableIRQ(LTDC_IRQn);
01659   HAL_NVIC_SetPriority(LTDC_ER_IRQn, 3, 0);
01660   HAL_NVIC_EnableIRQ(LTDC_ER_IRQn);
01661
01662   /* NVIC configuration for DMA2D interrupt
that is now enabled */
01663   HAL_NVIC_SetPriority(DMA2D_IRQn, 3, 0);
01664   HAL_NVIC_EnableIRQ(DMA2D_IRQn);
01665
01666   /* NVIC configuration for DSI interrupt th
at is now enabled */
01667   HAL_NVIC_SetPriority(DSI_IRQn, 3, 0);
01668   HAL_NVIC_EnableIRQ(DSI_IRQn);
01669 }
01670
01671 /**
01672   * @}
01673   */
01674
01675 /** @defgroup STM32L4R9I_EVAL_DSI_LCD_Privat
e_Functions Private Functions
01676   * @{
01677   */
01678
```

```
01679 /**
01680   * @brief  End of Refresh DSI callback.
01681   * @param  hdsi: pointer to a DSI_HandleTyp
eDef structure that contains
01682   *               the configuration informat
ion for the DSI.
01683   * @retval None
01684   */
01685 void HAL_DSI_EndOfRefreshCallback(DSI_Handle
TypeDef *hdsi)
01686 {
01687   /* Clear pending tearing effect flag */
01688   __HAL_DSI_CLEAR_FLAG(hdsi, DSI_FLAG_TE);
01689
01690   /* Set frame buffer available */
01691   FrameBufferAvailable = 1;
01692 }
01693
01694 /**
01695   * @brief  LCD Reset
01696   *         Hw reset the LCD DSI activating
its XRES signal (active low for some time)
01697   *         and desactivating it later.
01698   */
01699 static void LCD_Reset(void)
01700 {
01701   /* Reset the LCD by activation of XRES (ac
tive low) */
01702   BSP_IO_Init();
01703
01704   /* Configure the GPIO connected to XRES si
gnal */
01705   BSP_IO_ConfigPin(IO_PIN_9, IO_MODE_OUTPUT)
;
01706
01707   /* Activate XRES (active low) */
01708   BSP_IO_WritePin(IO_PIN_9, GPIO_PIN_RESET);
```

```
01709
01710   /* Wait at least 1 ms (reset low pulse wid
th) */
01711   HAL_Delay(2);
01712
01713   /* Desactivate XRES */
01714   BSP_IO_WritePin(IO_PIN_9, GPIO_PIN_SET);
01715
01716   /* Wait reset complete time (maximum time
is 5ms when LCD in sleep mode and 120ms when LCD i
s not in sleep mode) */
01717   HAL_Delay(120);
01718 }
01719
01720 /**
01721   * @brief  Draws a character on LCD.
01722   * @param  Xpos: Line where to display the
character shape
01723   * @param  Ypos: Start column address
01724   * @param  c: Pointer to the character data
01725   */
01726 static void DrawChar(uint16_t Xpos, uint16_t
 Ypos, const uint8_t *c)
01727 {
01728   uint32_t i = 0, j = 0;
01729   uint16_t height, width;
01730   uint8_t  offset;
01731   uint8_t  *pchar;
01732   uint32_t line;
01733
01734   height = DrawProp[ActiveLayer].pFont->Heig
ht;
01735   width  = DrawProp[ActiveLayer].pFont->Widt
h;
01736
01737   offset =  8 *((width + 7)/8) -  width ;
01738
```

```
01739   for(i = 0; i < height; i++)
01740   {
01741     pchar = ((uint8_t *)c + (width + 7)/8 *
i);
01742
01743     switch(((width + 7)/8))
01744     {
01745
01746     case 1:
01747       line =  pchar[0];
01748       break;
01749
01750     case 2:
01751       line =  (pchar[0]<< 8) | pchar[1];
01752       break;
01753
01754     case 3:
01755     default:
01756       line =  (pchar[0]<< 16) | (pchar[1]<<
8) | pchar[2];
01757       break;
01758     }
01759
01760     for (j = 0; j < width; j++)
01761     {
01762       if(line & (1 << (width- j + offset- 1)
))
01763       {
01764         BSP_DSI_LCD_DrawPixel((Xpos + j), Yp
os, DrawProp[ActiveLayer].TextColor);
01765       }
01766       else
01767       {
01768         BSP_DSI_LCD_DrawPixel((Xpos + j), Yp
os, DrawProp[ActiveLayer].BackColor);
01769       }
01770     }
```

```
01771     Ypos++;
01772   }
01773 }
01774
01775 /**
01776   * @brief  Fills a triangle (between 3 poin
ts).
01777   * @param  x1: Point 1 X position
01778   * @param  y1: Point 1 Y position
01779   * @param  x2: Point 2 X position
01780   * @param  y2: Point 2 Y position
01781   * @param  x3: Point 3 X position
01782   * @param  y3: Point 3 Y position
01783   */
01784 static void FillTriangle(uint16_t x1, uint16
_t x2, uint16_t x3, uint16_t y1, uint16_t y2, uint
16_t y3)
01785 {
01786   int16_t deltax = 0, deltay = 0, x = 0, y =
 0, xinc1 = 0, xinc2 = 0,
01787   yinc1 = 0, yinc2 = 0, den = 0, num = 0, nu
madd = 0, numpixels = 0,
01788   curpixel = 0;
01789
01790   deltax = ABS(x2 - x1);        /* The diffe
rence between the x's */
01791   deltay = ABS(y2 - y1);        /* The diffe
rence between the y's */
01792   x = x1;                       /* Start x o
ff at the first pixel */
01793   y = y1;                       /* Start y o
ff at the first pixel */
01794
01795   if (x2 >= x1)                 /* The x-val
ues are increasing */
01796   {
01797     xinc1 = 1;
```

```
01798     xinc2 = 1;
01799   }
01800   else                            /* The x-val
ues are decreasing */
01801   {
01802     xinc1 = -1;
01803     xinc2 = -1;
01804   }
01805
01806   if (y2 >= y1)                   /* The y-val
ues are increasing */
01807   {
01808     yinc1 = 1;
01809     yinc2 = 1;
01810   }
01811   else                            /* The y-val
ues are decreasing */
01812   {
01813     yinc1 = -1;
01814     yinc2 = -1;
01815   }
01816
01817   if (deltax >= deltay)           /* There is
at least one x-value for every y-value */
01818   {
01819     xinc1 = 0;                    /* Don't cha
nge the x when numerator >= denominator */
01820     yinc2 = 0;                    /* Don't cha
nge the y for every iteration */
01821     den = deltax;
01822     num = deltax / 2;
01823     numadd = deltay;
01824     numpixels = deltax;           /* There are
 more x-values than y-values */
01825   }
01826   else                            /* There is
at least one y-value for every x-value */
```

```
01827   {
01828     xinc2 = 0;                  /* Don't cha
nge the x for every iteration */
01829     yinc1 = 0;                  /* Don't cha
nge the y when numerator >= denominator */
01830     den = deltay;
01831     num = deltay / 2;
01832     numadd = deltax;
01833     numpixels = deltay;         /* There are
 more y-values than x-values */
01834   }
01835
01836   for (curpixel = 0; curpixel <= numpixels;
curpixel++)
01837   {
01838     BSP_DSI_LCD_DrawLine(x, y, x3, y3);
01839
01840     num += numadd;              /* Increase
the numerator by the top of the fraction */
01841     if (num >= den)             /* Check if
numerator >= denominator */
01842     {
01843       num -= den;               /* Calculate
 the new numerator value */
01844       x += xinc1;               /* Change th
e x as appropriate */
01845       y += yinc1;               /* Change th
e y as appropriate */
01846     }
01847     x += xinc2;                 /* Change th
e x as appropriate */
01848     y += yinc2;                 /* Change th
e y as appropriate */
01849   }
01850 }
01851
01852 /**
```

```
01853   * @brief  Fills a buffer.
01854   * @param  LayerIndex: Layer index
01855   * @param  pDst: Pointer to destination buf
fer
01856   * @param  xSize: Buffer width
01857   * @param  ySize: Buffer height
01858   * @param  OffLine: Offset
01859   * @param  ColorIndex: Color index
01860   */
01861 static void LL_FillBuffer(uint32_t LayerInde
x, void *pDst, uint32_t xSize, uint32_t ySize, uin
t32_t OffLine, uint32_t ColorIndex)
01862 {
01863   /* Register to memory mode with ARGB8888 a
s color Mode */
01864   hdma2d_eval.Init.Mode          = DMA2D_R2M
;
01865   hdma2d_eval.Init.ColorMode     = DMA2D_OUT
PUT_ARGB8888;
01866   hdma2d_eval.Init.OutputOffset  = OffLine;
01867   hdma2d_eval.Init.AlphaInverted = DMA2D_REG
ULAR_ALPHA;
01868   hdma2d_eval.Init.RedBlueSwap   = DMA2D_RB_
REGULAR;
01869
01870   hdma2d_eval.Instance = DMA2D;
01871
01872   /* DMA2D Initialization */
01873   if(HAL_DMA2D_Init(&hdma2d_eval) == HAL_OK)
01874   {
01875     if(HAL_DMA2D_ConfigLayer(&hdma2d_eval, L
ayerIndex) == HAL_OK)
01876     {
01877       if (HAL_DMA2D_Start(&hdma2d_eval, Colo
rIndex, (uint32_t)pDst, xSize, ySize) == HAL_OK)
01878       {
01879         /* Polling For DMA transfer */
```

```
01880         HAL_DMA2D_PollForTransfer(&hdma2d_ev
al, 10);
01881       }
01882     }
01883   }
01884 }
01885
01886 /**
01887   * @brief  Converts a line to an ARGB8888 p
ixel format.
01888   * @param  pSrc: Pointer to source buffer
01889   * @param  pDst: Output color
01890   * @param  xSize: Buffer width
01891   * @param  ColorMode: Input color mode
01892   */
01893 static void LL_ConvertLineToARGB8888(void *p
Src, void *pDst, uint32_t xSize, uint32_t ColorMod
e)
01894 {
01895   /* Configure the DMA2D Mode, Color Mode an
d output offset */
01896   hdma2d_eval.Init.Mode           = DMA2D_M2
M_PFC;
01897   hdma2d_eval.Init.ColorMode      = DMA2D_OU
TPUT_ARGB8888;
01898   hdma2d_eval.Init.OutputOffset   = 0;
01899   hdma2d_eval.Init.LineOffsetMode = DMA2D_LO
M_PIXELS;
01900   hdma2d_eval.Init.BytesSwap      = DMA2D_BY
TES_REGULAR;
01901   hdma2d_eval.Init.AlphaInverted  = DMA2D_RE
GULAR_ALPHA;
01902   hdma2d_eval.Init.RedBlueSwap    = DMA2D_RB
_REGULAR;
01903
01904   /* Foreground Configuration */
01905   hdma2d_eval.LayerCfg[1].AlphaMode      = D
```

```
MA2D_NO_MODIF_ALPHA;
01906   hdma2d_eval.LayerCfg[1].InputAlpha     = 0
xFF;
01907   hdma2d_eval.LayerCfg[1].InputColorMode = C
olorMode;
01908   hdma2d_eval.LayerCfg[1].InputOffset    = 0
;
01909   hdma2d_eval.LayerCfg[1].AlphaInverted  = D
MA2D_REGULAR_ALPHA;
01910   hdma2d_eval.LayerCfg[1].RedBlueSwap    = D
MA2D_RB_REGULAR;
01911
01912   hdma2d_eval.Instance = DMA2D;
01913
01914   /* DMA2D Initialization */
01915   if(HAL_DMA2D_Init(&hdma2d_eval) == HAL_OK)
01916   {
01917     if(HAL_DMA2D_ConfigLayer(&hdma2d_eval, 1
) == HAL_OK)
01918     {
01919       if (HAL_DMA2D_Start(&hdma2d_eval, (uin
t32_t)pSrc, (uint32_t)pDst, xSize, 1) == HAL_OK)
01920       {
01921         /* Polling For DMA transfer */
01922         HAL_DMA2D_PollForTransfer(&hdma2d_ev
al, 10);
01923       }
01924     }
01925   }
01926 }
01927
01928 /**
01929   * @}
01930   */
01931
01932 /**
01933   * @}
```

```
01934   */
01935
01936 /**
01937   * @}
01938   */
01939
01940 /**
01941   * @}
01942   */
01943
01944 /**
01945   * @}
01946   */
01947
01948 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

Exported Types

STM32L4R9I_EVAL TS

Data Structures

struct	**TS_StateTypeDef** **TS_StateTypeDef** Define TS State structure. More...

Enumerations

enum	**TS_StatusTypeDef** { **TS_OK** = 0x00, **TS_ERROR** = 0x01, **TS_TIMEOUT** = 0x02, **TS_DEVICE_NOT_FOUND** = 0x03 } TS_StatusTypeDef Define BSP_TS_xxx() functions possible return value, when status is returned by those functions. More...

enum	**TS_GestureIdTypeDef** {
	GEST_ID_NO_GESTURE = 0x00, **GEST_ID_MOVE_UP** = 0x01, **GEST_ID_MOVE_RIGHT** = 0x02,
	GEST_ID_MOVE_DOWN = 0x03,
	GEST_ID_MOVE_LEFT = 0x04, **GEST_ID_ZOOM_IN** = 0x05, **GEST_ID_ZOOM_OUT** = 0x06,
	GEST_ID_SINGLE_CLICK = 0x07,
	GEST_ID_DOUBLE_CLICK = 0x08, **GEST_ID_NB_MAX** = 0x09
	}
	TS_GestureIdTypeDef Define Possible managed gesture identification values returned by touch screen driver. More...

enum	**TS_TouchEventTypeDef** {
	TOUCH_EVENT_NO_EVT = 0x00,
	TOUCH_EVENT_PRESS_DOWN = 0x01,
	TOUCH_EVENT_LIFT_UP = 0x02,
	TOUCH_EVENT_CONTACT = 0x03,
	TOUCH_EVENT_NB_MAX = 0x04
	}
	TS_TouchEventTypeDef Define Possible touch events kind as returned values by touch screen IC Driver. More...

Enumeration Type Documentation

enum TS_GestureIdTypeDef

TS_GestureIdTypeDef Define Possible managed gesture identification values returned by touch screen driver.

Enumerator:

GEST_ID_NO_GESTURE	Gesture not defined / recognized
GEST_ID_MOVE_UP	Gesture Move Up
GEST_ID_MOVE_RIGHT	Gesture Move Right
GEST_ID_MOVE_DOWN	Gesture Move Down
GEST_ID_MOVE_LEFT	Gesture Move Left
GEST_ID_ZOOM_IN	Gesture Zoom In
GEST_ID_ZOOM_OUT	Gesture Zoom Out
GEST_ID_SINGLE_CLICK	Gesture Single Click
GEST_ID_DOUBLE_CLICK	Gesture Double Click
GEST_ID_NB_MAX	max number of gesture id

Definition at line **139** of file **stm32l4r9i_eval_ts.h**.

enum TS_StatusTypeDef

TS_StatusTypeDef Define BSP_TS_xxx() functions possible return value, when status is returned by those functions.

Enumerator:

TS_OK	Touch Ok
TS_ERROR	Touch Error
TS_TIMEOUT	Touch Timeout
TS_DEVICE_NOT_FOUND	Touchscreen device not found

Definition at line **126** of file **stm32l4r9i_eval_ts.h**.

enum TS_TouchEventTypeDef

TS_TouchEventTypeDef Define Possible touch events kind as returned values by touch screen IC Driver.

Enumerator:

TOUCH_EVENT_NO_EVT	Touch Event : undetermined
TOUCH_EVENT_PRESS_DOWN	Touch Event Press Down

TOUCH_EVENT_LIFT_UP	Touch Event Lift Up
TOUCH_EVENT_CONTACT	Touch Event Contact
TOUCH_EVENT_NB_MAX	max number of touch events kind

Definition at line **158** of file **stm32l4r9i_eval_ts.h**.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ts.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_ts.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_ts.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_TS_H
00039 #define __STM32L4R9I_EVAL_TS_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-----------------------------------*/
00046 #include "stm32l4r9i_eval.h"
00047 #include "stm32l4r9i_eval_lcd.h"
00048 #include "stm32l4r9i_eval_io.h"
00049
00050 /* Include TouchScreen component driver */
00051 #if defined(USE_GVO_390x390)
00052 /* Include touch screen FT3X67 component Dri
ver */
00053 #include "../Components/ft3x67/ft3x67.h"
00054 #endif
00055
00056 #if defined(USE_ROCKTECH_480x272)
00057 /* Include touch screen FT5336 component Dri
ver */
00058 #include "../Components/ft5336/ft5336.h"
00059 #endif
00060
00061 /** @addtogroup BSP
00062   * @{
00063   */
00064
00065 /** @addtogroup STM32L4R9I_EVAL
00066   * @{
```

```
00067   */
00068
00069 /** @defgroup STM32L4R9I_EVAL_TS STM32L4R9I_
EVAL TS
00070   * @{
00071   */
00072
00073  /** @defgroup STM32L4R9I_EVAL_TS_Exported_C
onstants Exported Constants
00074    * @{
00075    */
00076 #if defined(USE_GVO_390x390)
00077 /** @brief With FT3X67 : maximum 2 touches d
etected simultaneously
00078   */
00079 #define TS_MAX_NB_TOUCH                 ((ui
nt32_t) FT3X67_MAX_DETECTABLE_TOUCH)
00080 #endif
00081
00082 #if defined(USE_ROCKTECH_480x272)
00083 /** @brief With FT5336 : maximum 5 touches d
etected simultaneously
00084   */
00085 #define TS_MAX_NB_TOUCH                 ((ui
nt32_t) FT5336_MAX_DETECTABLE_TOUCH)
00086 #endif
00087
00088 #define TS_NO_IRQ_PENDING               ((ui
nt8_t) 0)
00089 #define TS_IRQ_PENDING                  ((ui
nt8_t) 1)
00090
00091 #define TS_SWAP_NONE                    ((ui
nt8_t) 0x01)
00092 #define TS_SWAP_X                       ((ui
nt8_t) 0x02)
00093 #define TS_SWAP_Y                       ((ui
```

```
nt8_t) 0x04)
00094 #define TS_SWAP_XY                      ((ui
nt8_t) 0x08)
00095
00096 #define TS_ORIENTATION_PORTRAIT         ((ui
nt8_t) 0)
00097 #define TS_ORIENTATION_LANDSCAPE        ((ui
nt8_t) 1)
00098
00099 /**
00100   * @}
00101   */
00102
00103 /** @defgroup STM32L4R9I_EVAL_TS_Exported_Ty
pes  Exported Types
00104   * @{
00105   */
00106 /**
00107 *  @brief TS_StateTypeDef
00108 *  Define TS State structure
00109 */
00110 typedef struct
00111 {
00112   uint8_t  touchDetected;                 /*
!< Total number of active touches detected at last
 scan */
00113   uint16_t touchX[TS_MAX_NB_TOUCH];       /*
!< Touch X[0], X[1] coordinates on 12 bits */
00114   uint16_t touchY[TS_MAX_NB_TOUCH];       /*
!< Touch Y[0], Y[1] coordinates on 12 bits */
00115   uint8_t  touchWeight[TS_MAX_NB_TOUCH];  /*
!< Touch_Weight[0], Touch_Weight[1] : weight prope
rty of touches */
00116   uint8_t  touchEventId[TS_MAX_NB_TOUCH]; /*
!< Touch_EventId[0], Touch_EventId[1] : take value
 of type @ref TS_TouchEventTypeDef */
00117   uint8_t  touchArea[TS_MAX_NB_TOUCH];    /*
```

```
!< Touch_Area[0], Touch_Area[1] : touch area of ea
ch touch */
00118   uint32_t gestureId;                     /*
!< type of gesture detected : take value of type @
ref TS_GestureIdTypeDef */
00119 } TS_StateTypeDef;
00120
00121 /**
00122  *  @brief TS_StatusTypeDef
00123  *  Define BSP_TS_xxx() functions possible r
eturn value,
00124  *  when status is returned by those functio
ns.
00125  */
00126 typedef enum
00127 {
00128   TS_OK               = 0x00, /*!< Touch Ok
 */
00129   TS_ERROR            = 0x01, /*!< Touch Er
ror */
00130   TS_TIMEOUT          = 0x02, /*!< Touch Ti
meout */
00131   TS_DEVICE_NOT_FOUND  = 0x03  /*!< Touchscr
een device not found */
00132 } TS_StatusTypeDef;
00133
00134 /**
00135  *  @brief TS_GestureIdTypeDef
00136  *  Define Possible managed gesture identifi
cation values returned by touch screen
00137  *  driver.
00138  */
00139 typedef enum
00140 {
00141   GEST_ID_NO_GESTURE   = 0x00, /*!< Gesture
not defined / recognized */
00142   GEST_ID_MOVE_UP      = 0x01, /*!< Gesture
```

```
Move Up */
00143   GEST_ID_MOVE_RIGHT    = 0x02, /*!< Gesture
Move Right */
00144   GEST_ID_MOVE_DOWN     = 0x03, /*!< Gesture
Move Down */
00145   GEST_ID_MOVE_LEFT     = 0x04, /*!< Gesture
Move Left */
00146   GEST_ID_ZOOM_IN       = 0x05, /*!< Gesture
Zoom In */
00147   GEST_ID_ZOOM_OUT      = 0x06, /*!< Gesture
Zoom Out */
00148   GEST_ID_SINGLE_CLICK = 0x07, /*!< Gesture
Single Click */
00149   GEST_ID_DOUBLE_CLICK = 0x08, /*!< Gesture
Double Click */
00150   GEST_ID_NB_MAX        = 0x09 /*!< max numbe
r of gesture id */
00151 } TS_GestureIdTypeDef;
00152
00153 /**
00154  *  @brief TS_TouchEventTypeDef
00155  *  Define Possible touch events kind as ret
urned values
00156  *  by touch screen IC Driver.
00157  */
00158 typedef enum
00159 {
00160   TOUCH_EVENT_NO_EVT         = 0x00, /*!< Tou
ch Event : undetermined */
00161   TOUCH_EVENT_PRESS_DOWN     = 0x01, /*!< Tou
ch Event Press Down */
00162   TOUCH_EVENT_LIFT_UP        = 0x02, /*!< Tou
ch Event Lift Up */
00163   TOUCH_EVENT_CONTACT        = 0x03, /*!< Tou
ch Event Contact */
00164   TOUCH_EVENT_NB_MAX         = 0x04  /*!< max
 number of touch events kind */
```

```
00165 } TS_TouchEventTypeDef;
00166
00167 /**
00168   * @}
00169   */
00170
00171 /** @defgroup STM32L4R9I_EVAL_TS_Exported_Fu
nctions Exported Functions
00172   * @{
00173   */
00174
00175 #if defined(USE_GVO_390x390)
00176
00177 #include "stm32l4r9i_eval_dsi_ts.h"
00178
00179 #define BSP_TS_Init               BSP_DSI_T
S_Init
00180 #define BSP_TS_DeInit             BSP_DSI_T
S_DeInit
00181
00182 #define BSP_TS_GetState           BSP_DSI_T
S_GetState
00183 #define BSP_TS_GestureConfig      BSP_DSI_T
S_GestureConfig
00184 #define BSP_TS_Get_GestureId      BSP_DSI_T
S_Get_GestureId
00185 #define BSP_TS_ResetTouchData     BSP_DSI_T
S_ResetTouchData
00186
00187 #define BSP_TS_ITConfig           BSP_DSI_T
S_ITConfig
00188 #define BSP_TS_ITGetStatus        BSP_DSI_T
S_ITGetStatus
00189 #define BSP_TS_ITClear            BSP_DSI_T
S_ITClear
00190
00191 #endif /* USE_GVO_390x390 */
```

```
00192
00193
00194 #if defined(USE_ROCKTECH_480x272)
00195
00196 #include "stm32l4r9i_eval_rgb_ts.h"
00197
00198 #define BSP_TS_Init                BSP_RGB_T
S_Init
00199 #define BSP_TS_DeInit              BSP_RGB_T
S_DeInit
00200
00201 #define BSP_TS_GetState            BSP_RGB_T
S_GetState
00202 #define BSP_TS_Get_GestureId       BSP_RGB_T
S_Get_GestureId
00203 #define BSP_TS_ResetTouchData      BSP_RGB_T
S_ResetTouchData
00204
00205 #define BSP_TS_ITConfig            BSP_RGB_T
S_ITConfig
00206 #define BSP_TS_ITGetStatus         BSP_RGB_T
S_ITGetStatus
00207 #define BSP_TS_ITClear             BSP_RGB_T
S_ITClear
00208
00209 #endif /* USE_ROCKTECH_480x272 */
00210
00211 /**
00212   * @}
00213   */
00214
00215 /**
00216   * @}
00217   */
00218
00219 /**
00220   * @}
```

```
00221   */
00222
00223 /**
00224   * @}
00225   */
00226
00227
00228 #ifdef __cplusplus
00229 }
00230 #endif
00231
00232 #endif /* __STM32L4R9I_EVAL_TS_H */
00233
00234 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL TS

STM32L4R9I_EVAL

Modules

- **STM32L4R9I_EVAL DSI TS**
- **STM32L4R9I_EVAL RGB TS**
- **Exported Constants**
- **Exported Types**
- **Exported Functions**

STM32L4R9I_EVAL BSP User Manual

Exported Types

STM32L4R9I_EVAL OSPI NOR

Data Structures

struct	**OSPI_NOR_Info**

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ospi_nor.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_ospi_nor.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_ospi_nor.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_OSPI_NOR_H
00039 #define __STM32L4R9I_EVAL_OSPI_NOR_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-----------------------------------*/
00046 #include "stm32l4xx_hal.h"
00047 #include "../Components/mx25lm51245g/mx25lm5
1245g.h"
00048
00049 /** @addtogroup BSP
00050   * @{
00051   */
00052
00053 /** @addtogroup STM32L4R9I_EVAL
00054   * @{
00055   */
00056
00057 /** @addtogroup STM32L4R9I_EVAL_OSPI_NOR
00058   * @{
00059   */
00060
00061 /* Exported constants --------------------
-----------------------------------*/
00062 /** @defgroup STM32L4R9I_EVAL_OSPI_NOR_Expor
ted_Constants Exported Constants
00063   * @{
00064   */
00065 /* OSPI Error codes */
```

```
00066 #define OSPI_NOR_OK             ((uint8_t)0x0
0)
00067 #define OSPI_NOR_ERROR          ((uint8_t)0x0
1)
00068 #define OSPI_NOR_BUSY           ((uint8_t)0x0
2)
00069 #define OSPI_NOR_NOT_SUPPORTED ((uint8_t)0x0
4)
00070 #define OSPI_NOR_SUSPENDED      ((uint8_t)0x0
8)
00071
00072 /**
00073   * @}
00074   */
00075
00076 /* Exported types ---------------------------
-----------------------------------*/
00077 /** @defgroup STM32L4R9I_EVAL_OSPI_NOR_Expor
ted_Types Exported Types
00078   * @{
00079   */
00080 /* OSPI Info */
00081 typedef struct {
00082   uint32_t FlashSize;          /*!< Size of
the flash */
00083   uint32_t EraseSectorSize;    /*!< Size of
sectors for the erase operation */
00084   uint32_t EraseSectorsNumber; /*!< Number o
f sectors for the erase operation */
00085   uint32_t ProgPageSize;       /*!< Size of
pages for the program operation */
00086   uint32_t ProgPagesNumber;    /*!< Number o
f pages for the program operation */
00087 } OSPI_NOR_Info;
00088
00089 /**
00090   * @}
```

```
00091   */
00092
00093 /* Exported functions ---------------------
---------------------------------*/
00094 /** @defgroup STM32L4R9I_EVAL_OSPI_NOR_Expor
ted_Functions Exported Functions
00095   * @{
00096   */
00097 uint8_t BSP_OSPI_NOR_Init                  (
void);
00098 uint8_t BSP_OSPI_NOR_DeInit                (
void);
00099 uint8_t BSP_OSPI_NOR_Read                  (
uint8_t* pData, uint32_t ReadAddr, uint32_t Size);
00100 uint8_t BSP_OSPI_NOR_Write                 (
uint8_t* pData, uint32_t WriteAddr, uint32_t Size)
;
00101 uint8_t BSP_OSPI_NOR_Erase_Block          (
uint32_t BlockAddress);
00102 uint8_t BSP_OSPI_NOR_Erase_Sector         (
uint32_t Sector);
00103 uint8_t BSP_OSPI_NOR_Erase_Chip           (
void);
00104 uint8_t BSP_OSPI_NOR_GetStatus            (
void);
00105 uint8_t BSP_OSPI_NOR_GetInfo              (
OSPI_NOR_Info* pInfo);
00106 uint8_t BSP_OSPI_NOR_EnableMemoryMappedMode(
void);
00107 uint8_t BSP_OSPI_NOR_SuspendErase         (
void);
00108 uint8_t BSP_OSPI_NOR_ResumeErase          (
void);
00109 uint8_t BSP_OSPI_NOR_EnterDeepPowerDown   (
void);
00110 uint8_t BSP_OSPI_NOR_LeaveDeepPowerDown   (
void);
```

```
00111
00112 /**
00113   * @}
00114   */
00115
00116 /**
00117   * @}
00118   */
00119
00120 /**
00121   * @}
00122   */
00123
00124 /**
00125   * @}
00126   */
00127
00128 #ifdef __cplusplus
00129 }
00130 #endif
00131
00132 #endif /* __STM32L4R9I_EVAL_OSPI_NOR_H */
00133
00134 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ospi_nor.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_ospi_nor.c
00004   * @author  MCD Application Team
00005   * @brief   This file includes a standard driver for the MX25UM51245G OSPI
00006   *          memory mounted on STM32L4R9I-EVAL board.
00007   @verbatim
00008   ==============================================================================
00009                     ##### How to use this driver #####
00010   ==============================================================================
00011   [..]
00012    (#) This driver is used to drive the MX25UM51245G OSPI external
00013        memory mounted on STM32L4R9I-EVAL evaluation board.
00014
```

```
00015    (#) This driver need a specific component
 driver (MX25UM51245G) to be included with.
00016
00017    (#) Initialization steps:
00018        (++) Initialize the OPSI external mem
ory using the BSP_OSPI_NOR_Init() function. This
00019             function includes the MSP layer
hardware resources initialization and the
00020             OSPI interface with the external
 memory.
00021
00022    (#) OSPI memory operations
00023        (++) OSPI memory can be accessed with
 read/write operations once it is
00024             initialized.
00025             Read/write operation can be perf
ormed with AHB access using the functions
00026             BSP_OSPI_NOR_Read()/BSP_OSPI_NOR
_Write().
00027        (++) The function BSP_OSPI_NOR_GetInf
o() returns the configuration of the OSPI memory.
00028             (see the OSPI memory data sheet)
00029        (++) Perform erase block operation us
ing the function BSP_OSPI_NOR_Erase_Block() and by
00030             specifying the block address. Yo
u can perform an erase operation of the whole
00031             chip by calling the function BSP
_OSPI_NOR_Erase_Chip().
00032        (++) The function BSP_OSPI_NOR_GetSta
tus() returns the current status of the OSPI memor
y.
00033             (see the OSPI memory data sheet)
00034   @endverbatim
00035   ******************************************
**********************************
00036   * @attention
00037   *
```

```
00038   * <h2><center>&copy; COPYRIGHT(c) 2017 STM
icroelectronics</center></h2>
00039   *
00040   * Redistribution and use in source and bin
ary forms, with or without modification,
00041   * are permitted provided that the followin
g conditions are met:
00042   *   1. Redistributions of source code must
 retain the above copyright notice,
00043   *      this list of conditions and the fol
lowing disclaimer.
00044   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00045   *      this list of conditions and the fol
lowing disclaimer in the documentation
00046   *      and/or other materials provided wit
h the distribution.
00047   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00048   *      may be used to endorse or promote p
roducts derived from this software
00049   *      without specific prior written perm
ission.
00050   *
00051   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00052   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00053   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00054   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00055   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00056   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00057   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
```

```
 OR BUSINESS INTERRUPTION) HOWEVER
00058   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00059   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00060   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00061   *
00062   ****************************************
**********************************
00063   */
00064
00065 /* Includes ------------------------------
-----------------------------------*/
00066 #include "stm32l4r9i_eval_ospi_nor.h"
00067
00068 /** @addtogroup BSP
00069   * @{
00070   */
00071
00072 /** @addtogroup STM32L4R9I_EVAL
00073   * @{
00074   */
00075
00076 /** @defgroup STM32L4R9I_EVAL_OSPI_NOR STM32
L4R9I_EVAL OSPI NOR
00077   * @{
00078   */
00079
00080 /* Private constants ---------------------
-----------------------------------*/
00081 /** @defgroup STM32L4R9I_EVAL_OSPI_NOR_Priva
te_Constants Private Constants
00082   * @{
00083   */
00084 #define OSPI_NOR_OCTAL_DISABLE      0x0
00085 #define OSPI_NOR_OCTAL_ENABLE       0x1
```

```
00086
00087 #define DUMMY_CYCLES_READ_OCTAL     MX25LM51
245G_DUMMY_CYCLES_READ_OCTAL_66M
00088 #define DUMMY_CYCLES_CR_CFG         MX25LM51
245G_CR2_DC_66M
00089 /**
00090   * @}
00091   */
00092
00093 /* Uncomment this line to use the memory in
SDR mode */
00094 //#define BSP_OSPI_NOR_DDR_MODE_DEACTIVATED
00095
00096 /* Private variables ----------------------
-----------------------------------*/
00097
00098 /** @defgroup STM32L4R9I_EVAL_OSPI_NOR_Priva
te_Variables Private Variables
00099   * @{
00100   */
00101 OSPI_HandleTypeDef OSPINORHandle;
00102
00103 /**
00104   * @}
00105   */
00106
00107
00108 /* Private functions ----------------------
-----------------------------------*/
00109
00110 /** @defgroup STM32L4R9I_EVAL_OSPI_NOR_Priva
te_Functions Private Functions
00111   * @{
00112   */
00113 static void    OSPI_NOR_MspInit             (
void);
00114 static void    OSPI_NOR_MspDeInit           (
```

```
void);
00115 static uint8_t OSPI_NOR_ResetMemory        (
OSPI_HandleTypeDef *hospi);
00116 static uint8_t OSPI_NOR_WriteEnable        (
OSPI_HandleTypeDef *hospi);
00117 static uint8_t OSPI_NOR_AutoPollingMemReady(
OSPI_HandleTypeDef *hospi, uint32_t Timeout);
00118 static uint8_t OSPI_NOR_OctalMode          (
OSPI_HandleTypeDef *hospi, uint8_t Operation);
00119 /**
00120   * @}
00121   */
00122
00123 /* Exported functions ---------------------
-----------------------------------*/
00124
00125 /** @addtogroup STM32L4R9I_EVAL_OSPI_NOR_Exp
orted_Functions
00126   * @{
00127   */
00128
00129 /**
00130   * @brief  Initializes the OSPI interface.
00131   * @retval OSPI memory status
00132   */
00133 uint8_t BSP_OSPI_NOR_Init(void)
00134 {
00135   OSPINORHandle.Instance = OCTOSPI1;
00136
00137   /* Call the DeInit function to reset the d
river */
00138   if (HAL_OSPI_DeInit(&OSPINORHandle) != HAL
_OK)
00139   {
00140     return OSPI_NOR_ERROR;
00141   }
00142
```

```
00143   /* System level initialization */
00144   OSPI_NOR_MspInit();
00145
00146   /* OctoSPI initialization */
00147   OSPINORHandle.Init.FifoThreshold         =
 4;
00148   OSPINORHandle.Init.DualQuad              =
 HAL_OSPI_DUALQUAD_DISABLE;
00149   OSPINORHandle.Init.DeviceSize            =
 POSITION_VAL(MX25LM51245G_FLASH_SIZE); /* 512 MBi
ts */
00150   OSPINORHandle.Init.ChipSelectHighTime    =
 2;
00151   OSPINORHandle.Init.FreeRunningClock      =
 HAL_OSPI_FREERUNCLK_DISABLE;
00152   OSPINORHandle.Init.ClockMode             =
 HAL_OSPI_CLOCK_MODE_0;
00153   OSPINORHandle.Init.WrapSize              =
 HAL_OSPI_WRAP_NOT_SUPPORTED;
00154   OSPINORHandle.Init.ClockPrescaler        =
 2; /* OctoSPI clock = 110MHz / ClockPrescaler = 5
5MHz */
00155   OSPINORHandle.Init.SampleShifting        =
 HAL_OSPI_SAMPLE_SHIFTING_NONE;
00156 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00157   OSPINORHandle.Init.MemoryType            =
 HAL_OSPI_MEMTYPE_MICRON;
00158   OSPINORHandle.Init.DelayHoldQuarterCycle =
 HAL_OSPI_DHQC_DISABLE;
00159 #else
00160   OSPINORHandle.Init.MemoryType            =
 HAL_OSPI_MEMTYPE_MACRONIX;
00161   OSPINORHandle.Init.DelayHoldQuarterCycle =
 HAL_OSPI_DHQC_ENABLE;
00162 #endif
00163   OSPINORHandle.Init.ChipSelectBoundary    =
```

```
 0;
00164
00165   if (HAL_OSPI_Init(&OSPINORHandle) != HAL_O
K)
00166   {
00167     return OSPI_NOR_ERROR;
00168   }
00169
00170   /* OctoSPI memory reset */
00171   if (OSPI_NOR_ResetMemory(&OSPINORHandle) !
= OSPI_NOR_OK)
00172   {
00173     return OSPI_NOR_NOT_SUPPORTED;
00174   }
00175
00176   /* OctoSPI octal enable */
00177   if (OSPI_NOR_OctalMode(&OSPINORHandle, OSP
I_NOR_OCTAL_ENABLE) != OSPI_NOR_OK)
00178   {
00179     return OSPI_NOR_ERROR;
00180   }
00181
00182   return OSPI_NOR_OK;
00183 }
00184
00185 /**
00186   * @brief  De-Initializes the OSPI interfac
e.
00187   * @retval OSPI memory status
00188   */
00189 uint8_t BSP_OSPI_NOR_DeInit(void)
00190 {
00191   OSPINORHandle.Instance = OCTOSPI1;
00192
00193   /* Abort any activity on OctoSPI (eg Memor
y-mapped mode) */
00194   if (HAL_OSPI_GetState(&OSPINORHandle) != H
```

```
AL_OSPI_STATE_READY)
00195   {
00196     if (HAL_OSPI_Abort(&OSPINORHandle) != HA
L_OK)
00197     {
00198       return OSPI_NOR_ERROR;
00199     }
00200   }
00201
00202   /* OctoSPI octal mode disable */
00203   if (OSPI_NOR_OctalMode(&OSPINORHandle, OSP
I_NOR_OCTAL_DISABLE) != OSPI_NOR_OK)
00204   {
00205     return OSPI_NOR_ERROR;
00206   }
00207
00208   /* Call the DeInit function to reset the d
river */
00209   if (HAL_OSPI_DeInit(&OSPINORHandle) != HAL
_OK)
00210   {
00211     return OSPI_NOR_ERROR;
00212   }
00213
00214   /* System level De-initialization */
00215   OSPI_NOR_MspDeInit();
00216
00217   return OSPI_NOR_OK;
00218 }
00219
00220 /**
00221   * @brief  Reads an amount of data from the
 OSPI memory.
00222   * @param  pData: Pointer to data to be read

00223   * @param  ReadAddr: Read start address
00224   * @param  Size: Size of data to read
```

```
00225   * @retval OSPI memory status
00226   */
00227 uint8_t BSP_OSPI_NOR_Read(uint8_t* pData, ui
nt32_t ReadAddr, uint32_t Size)
00228 {
00229   OSPI_RegularCmdTypeDef sCommand;
00230
00231   /* Initialize the read command */
00232   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
00233   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
00234   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_8_LINES;
00235   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_16_BITS;
00236   sCommand.Address               = ReadAddr;
00237   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_8_LINES;
00238   sCommand.AddressSize           = HAL_OSPI_
ADDRESS_32_BITS;
00239   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00240   sCommand.DataMode              = HAL_OSPI_
DATA_8_LINES;
00241   sCommand.NbData                = Size;
00242   sCommand.DummyCycles           = DUMMY_CYC
LES_READ_OCTAL;
00243   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00244
00245 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00246   sCommand.Instruction           = OCTAL_IO_
READ_CMD;
00247   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
```

```
00248   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_DISABLE;
00249   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_DISABLE;
00250   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
00251 #else
00252   sCommand.Instruction           = OCTAL_IO_
DTR_READ_CMD;
00253   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00254   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_ENABLE;
00255   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_ENABLE;
00256   sCommand.DQSMode               = HAL_OSPI_
DQS_ENABLE;
00257 #endif
00258
00259   /* Configure the command */
00260   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00261   {
00262     return OSPI_NOR_ERROR;
00263   }
00264
00265   /* Reception of the data */
00266   if (HAL_OSPI_Receive(&OSPINORHandle, pData
, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00267   {
00268     return OSPI_NOR_ERROR;
00269   }
00270
00271   return OSPI_NOR_OK;
00272 }
00273
00274 /**
```

```
00275   * @brief  Writes an amount of data to the
OSPI memory.
00276   * @param  pData: Pointer to data to be wri
tten
00277   * @param  WriteAddr: Write start address
00278   * @param  Size: Size of data to write
00279   * @retval OSPI memory status
00280   */
00281 uint8_t BSP_OSPI_NOR_Write(uint8_t* pData, u
int32_t WriteAddr, uint32_t Size)
00282 {
00283   OSPI_RegularCmdTypeDef sCommand;
00284   uint32_t end_addr, current_size, current_a
ddr;
00285
00286   /* Calculation of the size between the wri
te address and the end of the page */
00287   current_size = MX25LM51245G_PAGE_SIZE - (W
riteAddr % MX25LM51245G_PAGE_SIZE);
00288
00289   /* Check if the size of the data is less t
han the remaining place in the page */
00290   if (current_size > Size)
00291   {
00292     current_size = Size;
00293   }
00294
00295   /* Initialize the adress variables */
00296   current_addr = WriteAddr;
00297   end_addr = WriteAddr + Size;
00298
00299   /* Initialize the program command */
00300   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
00301   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
00302   sCommand.Instruction           = OCTAL_PAG
```

```
E_PROG_CMD;
00303   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_8_LINES;
00304   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_16_BITS;
00305   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_8_LINES;
00306   sCommand.AddressSize           = HAL_OSPI_
ADDRESS_32_BITS;
00307   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00308   sCommand.DataMode              = HAL_OSPI_
DATA_8_LINES;
00309   sCommand.DummyCycles           = 0;
00310   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00311
00312 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00313   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
00314   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_DISABLE;
00315   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_DISABLE;
00316   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
00317 #else
00318   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00319   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_ENABLE;
00320   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_ENABLE;
00321   sCommand.DQSMode               = HAL_OSPI_
DQS_ENABLE;
00322 #endif
```

```
00323
00324   /* Perform the write page by page */
00325   do
00326   {
00327     sCommand.Address = current_addr;
00328     sCommand.NbData  = current_size;
00329
00330     /* Enable write operations */
00331     if (OSPI_NOR_WriteEnable(&OSPINORHandle)
 != OSPI_NOR_OK)
00332     {
00333       return OSPI_NOR_ERROR;
00334     }
00335
00336     /* Configure the command */
00337     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00338     {
00339       return OSPI_NOR_ERROR;
00340     }
00341
00342     /* Transmission of the data */
00343     if (HAL_OSPI_Transmit(&OSPINORHandle, pD
ata, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00344     {
00345       return OSPI_NOR_ERROR;
00346     }
00347
00348     /* Configure automatic polling mode to w
ait for end of program */
00349     if (OSPI_NOR_AutoPollingMemReady(&OSPINO
RHandle, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != OSPI_N
OR_OK)
00350     {
00351       return OSPI_NOR_ERROR;
00352     }
00353
```

```
00354     /* Update the address and size variables
 for next page programming */
00355     current_addr += current_size;
00356     pData += current_size;
00357     current_size = ((current_addr + MX25LM51
245G_PAGE_SIZE) > end_addr) ? (end_addr - current_
addr) : MX25LM51245G_PAGE_SIZE;
00358   } while (current_addr < end_addr);
00359
00360   return OSPI_NOR_OK;
00361 }
00362
00363 /**
00364   * @brief  Erases the specified block of th
e OSPI memory.
00365   * @param  BlockAddress: Block address to e
rase
00366   * @retval OSPI memory status
00367   */
00368 uint8_t BSP_OSPI_NOR_Erase_Block(uint32_t Bl
ockAddress)
00369 {
00370   OSPI_RegularCmdTypeDef sCommand;
00371
00372   /* Initialize the erase command */
00373   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
00374   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
00375   sCommand.Instruction           = OCTAL_BLO
CK_ERASE_CMD;
00376   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_8_LINES;
00377   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_16_BITS;
00378   sCommand.Address               = BlockAddr
ess;
```

```
00379   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_8_LINES;
00380   sCommand.AddressSize           = HAL_OSPI_
ADDRESS_32_BITS;
00381   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00382   sCommand.DataMode              = HAL_OSPI_
DATA_NONE;
00383   sCommand.DummyCycles           = 0;
00384   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
00385   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00386
00387 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00388   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
00389   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_DISABLE;
00390 #else
00391   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00392   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_ENABLE;
00393 #endif
00394
00395   /* Enable write operations */
00396   if (OSPI_NOR_WriteEnable(&OSPINORHandle) !
= OSPI_NOR_OK)
00397   {
00398     return OSPI_NOR_ERROR;
00399   }
00400
00401   /* Send the command */
00402   if (HAL_OSPI_Command(&OSPINORHandle, &Com
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
```

```
00403   {
00404     return OSPI_NOR_ERROR;
00405   }
00406
00407   /* Configure automatic polling mode to wai
t for end of erase */
00408   if (OSPI_NOR_AutoPollingMemReady(&OSPINORH
andle, MX25LM51245G_BLOCK_ERASE_MAX_TIME) != OSPI_
NOR_OK)
00409   {
00410     return OSPI_NOR_ERROR;
00411   }
00412
00413   return OSPI_NOR_OK;
00414 }
00415
00416 /**
00417   * @brief  Erases the specified sector of t
he OSPI memory.
00418   * @param  Sector: Sector address to erase
(0 to 255)
00419   * @retval OSPI memory status
00420   * @note This function is non blocking mean
ing that sector erase
00421   *       operation is started but not compl
eted when the function
00422   *       returns. Application has to call B
SP_QSPI_GetStatus()
00423   *       to know when the device is availab
le again (i.e. erase operation
00424   *       completed).
00425   */
00426 uint8_t BSP_OSPI_NOR_Erase_Sector(uint32_t S
ector)
00427 {
00428   OSPI_RegularCmdTypeDef sCommand;
00429
```

```
00430   if (Sector >= (uint32_t)(MX25LM51245G_FLAS
H_SIZE/MX25LM51245G_SECTOR_SIZE))
00431   {
00432     return OSPI_NOR_ERROR;
00433   }
00434
00435   /* Initialize the erase command */
00436   sCommand.OperationType          = HAL_OSPI_
OPTYPE_COMMON_CFG;
00437   sCommand.FlashId                = HAL_OSPI_
FLASH_ID_1;
00438   sCommand.Instruction            = OCTAL_SEC
TOR_ERASE_CMD;
00439   sCommand.InstructionMode        = HAL_OSPI_
INSTRUCTION_8_LINES;
00440   sCommand.InstructionSize        = HAL_OSPI_
INSTRUCTION_16_BITS;
00441   sCommand.Address                = (Sector *
 MX25LM51245G_SECTOR_SIZE);
00442   sCommand.AddressMode            = HAL_OSPI_
ADDRESS_8_LINES;
00443   sCommand.AddressSize            = HAL_OSPI_
ADDRESS_32_BITS;
00444   sCommand.AlternateBytesMode     = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00445   sCommand.DataMode               = HAL_OSPI_
DATA_NONE;
00446   sCommand.DummyCycles            = 0;
00447   sCommand.DQSMode                = HAL_OSPI_
DQS_DISABLE;
00448   sCommand.SIOOMode               = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00449
00450 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00451   sCommand.InstructionDtrMode     = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
```

```
00452   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_DISABLE;
00453 #else
00454   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00455   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_ENABLE;
00456 #endif
00457
00458   /* Enable write operations */
00459   if (OSPI_NOR_WriteEnable(&OSPINORHandle) !
= OSPI_NOR_OK)
00460   {
00461     return OSPI_NOR_ERROR;
00462   }
00463
00464   /* Send the command */
00465   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00466   {
00467     return OSPI_NOR_ERROR;
00468   }
00469
00470   return OSPI_NOR_OK;
00471 }
00472
00473 /**
00474   * @brief  Erases the entire OSPI memory.
00475   * @retval OSPI memory status
00476   */
00477 uint8_t BSP_OSPI_NOR_Erase_Chip(void)
00478 {
00479   OSPI_RegularCmdTypeDef sCommand;
00480
00481   /* Initialize the erase command */
00482   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
```

```
00483   sCommand.FlashId                = HAL_OSPI_
FLASH_ID_1;
00484   sCommand.Instruction            = OCTAL_CHI
P_ERASE_CMD;
00485   sCommand.InstructionMode        = HAL_OSPI_
INSTRUCTION_8_LINES;
00486   sCommand.InstructionSize        = HAL_OSPI_
INSTRUCTION_16_BITS;
00487   sCommand.AddressMode            = HAL_OSPI_
ADDRESS_NONE;
00488   sCommand.AlternateBytesMode     = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00489   sCommand.DataMode               = HAL_OSPI_
DATA_NONE;
00490   sCommand.DummyCycles            = 0;
00491   sCommand.DQSMode                = HAL_OSPI_
DQS_DISABLE;
00492   sCommand.SIOOMode               = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00493
00494 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00495   sCommand.InstructionDtrMode     = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
00496 #else
00497   sCommand.InstructionDtrMode     = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00498 #endif
00499
00500   /* Enable write operations */
00501   if (OSPI_NOR_WriteEnable(&OSPINORHandle) !
= OSPI_NOR_OK)
00502   {
00503     return OSPI_NOR_ERROR;
00504   }
00505
00506   /* Send the command */
```

```
00507   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00508   {
00509     return OSPI_NOR_ERROR;
00510   }
00511
00512   /* Configure automatic polling mode to wai
t for end of erase */
00513   if (OSPI_NOR_AutoPollingMemReady(&OSPINORH
andle, MX25LM51245G_CHIP_ERASE_MAX_TIME) != OSPI_N
OR_OK)
00514   {
00515     return OSPI_NOR_ERROR;
00516   }
00517
00518   return OSPI_NOR_OK;
00519 }
00520
00521 /**
00522   * @brief  Reads current status of the OSPI
 memory.
00523   * @retval OSPI memory status
00524   */
00525 uint8_t BSP_OSPI_NOR_GetStatus(void)
00526 {
00527   OSPI_RegularCmdTypeDef sCommand;
00528   uint8_t reg[2];
00529
00530   /* Initialize the read security register c
ommand */
00531   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
00532   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
00533   sCommand.Instruction           = OCTAL_REA
D_SECURITY_REG_CMD;
00534   sCommand.InstructionMode       = HAL_OSPI_
```

```
INSTRUCTION_8_LINES;
00535   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_16_BITS;
00536   sCommand.Address               = 0;
00537   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_8_LINES;
00538   sCommand.AddressSize           = HAL_OSPI_
ADDRESS_32_BITS;
00539   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00540   sCommand.DataMode              = HAL_OSPI_
DATA_8_LINES;
00541   sCommand.NbData                = 2;
00542   sCommand.DummyCycles           = DUMMY_CYC
LES_READ_OCTAL;
00543   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00544
00545 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00546   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
00547   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_DISABLE;
00548   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_DISABLE;
00549   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
00550 #else
00551   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00552   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_ENABLE;
00553   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_ENABLE;
00554   sCommand.DQSMode               = HAL_OSPI_
DQS_ENABLE;
```

```
00555 #endif
00556
00557   /* Configure the command */
00558   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00559   {
00560     return OSPI_NOR_ERROR;
00561   }
00562
00563   /* Reception of the data */
00564   if (HAL_OSPI_Receive(&OSPINORHandle, reg,
HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00565   {
00566     return OSPI_NOR_ERROR;
00567   }
00568
00569   /* Check the value of the register */
00570   if ((reg[0] & (MX25LM51245G_SECR_P_FAIL |
MX25LM51245G_SECR_E_FAIL)) != 0)
00571   {
00572     return OSPI_NOR_ERROR;
00573   }
00574   else if ((reg[0] & (MX25LM51245G_SECR_PSB
| MX25LM51245G_SECR_ESB)) != 0)
00575   {
00576     return OSPI_NOR_SUSPENDED;
00577   }
00578
00579   /* Initialize the read status register com
mand */
00580   sCommand.Instruction = OCTAL_READ_STATUS_R
EG_CMD;
00581
00582   /* Configure the command */
00583   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00584   {
```

```
00585     return OSPI_NOR_ERROR;
00586   }
00587
00588   /* Reception of the data */
00589   if (HAL_OSPI_Receive(&OSPINORHandle, reg,
HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00590   {
00591     return OSPI_NOR_ERROR;
00592   }
00593
00594   /* Check the value of the register */
00595   if ((reg[0] & MX25LM51245G_SR_WIP) != 0)
00596   {
00597     return OSPI_NOR_BUSY;
00598   }
00599   else
00600   {
00601     return OSPI_NOR_OK;
00602   }
00603 }
00604
00605 /**
00606   * @brief  Return the configuration of the
OSPI memory.
00607   * @param  pInfo: pointer on the configurat
ion structure
00608   * @retval OSPI memory status
00609   */
00610 uint8_t BSP_OSPI_NOR_GetInfo(OSPI_NOR_Info*
pInfo)
00611 {
00612   /* Configure the structure with the memory
 configuration */
00613   pInfo->FlashSize          = MX25LM51245G_F
LASH_SIZE;
00614   pInfo->EraseSectorSize    = MX25LM51245G_B
LOCK_SIZE;
```

```
00615   pInfo->EraseSectorsNumber = (MX25LM51245G_
FLASH_SIZE/MX25LM51245G_BLOCK_SIZE);
00616   pInfo->ProgPageSize       = MX25LM51245G_P
AGE_SIZE;
00617   pInfo->ProgPagesNumber    = (MX25LM51245G_
FLASH_SIZE/MX25LM51245G_PAGE_SIZE);
00618
00619   return OSPI_NOR_OK;
00620 }
00621
00622 /**
00623   * @brief  Configure the OSPI in memory-map
ped mode
00624   * @retval OSPI memory status
00625   */
00626 uint8_t BSP_OSPI_NOR_EnableMemoryMappedMode(
void)
00627 {
00628   OSPI_RegularCmdTypeDef   sCommand;
00629   OSPI_MemoryMappedTypeDef sMemMappedCfg;
00630
00631   /* Enable write operations */
00632   if (OSPI_NOR_WriteEnable(&OSPINORHandle) !
= OSPI_NOR_OK)
00633   {
00634     return OSPI_NOR_ERROR;
00635   }
00636
00637   /* Configure the command for the read inst
ruction */
00638   sCommand.OperationType         = HAL_OSPI_
OPTYPE_READ_CFG;
00639   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
00640   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_8_LINES;
00641   sCommand.InstructionSize       = HAL_OSPI_
```

```
INSTRUCTION_16_BITS;
00642   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_8_LINES;
00643   sCommand.AddressSize           = HAL_OSPI_
ADDRESS_32_BITS;
00644   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00645   sCommand.DataMode              = HAL_OSPI_
DATA_8_LINES;
00646   sCommand.DummyCycles           = DUMMY_CYC
LES_READ_OCTAL;
00647   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00648
00649 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00650   sCommand.Instruction           = OCTAL_IO_
READ_CMD;
00651   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
00652   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_DISABLE;
00653   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_DISABLE;
00654   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
00655 #else
00656   sCommand.Instruction           = OCTAL_IO_
DTR_READ_CMD;
00657   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00658   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_ENABLE;
00659   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_ENABLE;
00660   sCommand.DQSMode               = HAL_OSPI_
DQS_ENABLE;
```

```
00661 #endif
00662
00663   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00664   {
00665     return OSPI_NOR_ERROR;
00666   }
00667
00668   /* Configure the command for the write ins
truction */
00669   sCommand.OperationType         = HAL_OSPI_
OPTYPE_WRITE_CFG;
00670   sCommand.Instruction           = OCTAL_PAG
E_PROG_CMD;
00671   sCommand.DummyCycles           = 0;
00672
00673   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00674   {
00675     return OSPI_NOR_ERROR;
00676   }
00677
00678    /* Configure the memory mapped mode */
00679   sMemMappedCfg.TimeOutActivation = HAL_OSPI
_TIMEOUT_COUNTER_DISABLE;
00680
00681   if (HAL_OSPI_MemoryMapped(&OSPINORHandle,
&sMemMappedCfg) != HAL_OK)
00682   {
00683     return OSPI_NOR_ERROR;
00684   }
00685
00686   return OSPI_NOR_OK;
00687 }
00688
00689 /**
00690   * @brief  This function suspends an ongoin
```

```
g erase command.
00691   * @retval OSPI memory status
00692   */
00693 uint8_t BSP_OSPI_NOR_SuspendErase(void)
00694 {
00695   OSPI_RegularCmdTypeDef sCommand;
00696
00697   /* Check whether the device is busy (erase
 operation is
00698   in progress).
00699   */
00700   if (BSP_OSPI_NOR_GetStatus() == OSPI_NOR_B
USY)
00701   {
00702     /* Initialize the suspend command */
00703     sCommand.OperationType         = HAL_OSP
I_OPTYPE_COMMON_CFG;
00704     sCommand.FlashId               = HAL_OSP
I_FLASH_ID_1;
00705     sCommand.Instruction           = OCTAL_P
ROG_ERASE_SUSPEND_CMD;
00706     sCommand.InstructionMode       = HAL_OSP
I_INSTRUCTION_8_LINES;
00707     sCommand.InstructionSize       = HAL_OSP
I_INSTRUCTION_16_BITS;
00708     sCommand.AddressMode           = HAL_OSP
I_ADDRESS_NONE;
00709     sCommand.AlternateBytesMode    = HAL_OSP
I_ALTERNATE_BYTES_NONE;
00710     sCommand.DataMode              = HAL_OSP
I_DATA_NONE;
00711     sCommand.DummyCycles           = 0;
00712     sCommand.DQSMode               = HAL_OSP
I_DQS_DISABLE;
00713     sCommand.SIOOMode              = HAL_OSP
I_SIOO_INST_EVERY_CMD;
00714
```

```
00715 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00716     sCommand.InstructionDtrMode    = HAL_OSP
I_INSTRUCTION_DTR_DISABLE;
00717 #else
00718     sCommand.InstructionDtrMode    = HAL_OSP
I_INSTRUCTION_DTR_ENABLE;
00719 #endif
00720
00721   /* Send the command */
00722     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00723     {
00724       return OSPI_NOR_ERROR;
00725     }
00726
00727     if (BSP_OSPI_NOR_GetStatus() == OSPI_NOR
_SUSPENDED)
00728     {
00729       return OSPI_NOR_OK;
00730     }
00731
00732     return OSPI_NOR_ERROR;
00733   }
00734
00735   return OSPI_NOR_OK;
00736 }
00737
00738 /**
00739   * @brief  This function resumes a paused e
rase command.
00740   * @retval OSPI memory status
00741   */
00742 uint8_t BSP_OSPI_NOR_ResumeErase(void)
00743 {
00744   OSPI_RegularCmdTypeDef sCommand;
00745
```

```
00746   /* Check whether the device is in suspende
d state */
00747   if (BSP_OSPI_NOR_GetStatus() == OSPI_NOR_S
USPENDED)
00748   {
00749     /* Initialize the resume command */
00750     sCommand.OperationType         = HAL_OSP
I_OPTYPE_COMMON_CFG;
00751     sCommand.FlashId               = HAL_OSP
I_FLASH_ID_1;
00752     sCommand.Instruction           = OCTAL_P
ROG_ERASE_RESUME_CMD;
00753     sCommand.InstructionMode       = HAL_OSP
I_INSTRUCTION_8_LINES;
00754     sCommand.InstructionSize       = HAL_OSP
I_INSTRUCTION_16_BITS;
00755     sCommand.AddressMode           = HAL_OSP
I_ADDRESS_NONE;
00756     sCommand.AlternateBytesMode    = HAL_OSP
I_ALTERNATE_BYTES_NONE;
00757     sCommand.DataMode              = HAL_OSP
I_DATA_NONE;
00758     sCommand.DummyCycles           = 0;
00759     sCommand.DQSMode               = HAL_OSP
I_DQS_DISABLE;
00760     sCommand.SIOOMode              = HAL_OSP
I_SIOO_INST_EVERY_CMD;
00761
00762 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00763     sCommand.InstructionDtrMode    = HAL_OSP
I_INSTRUCTION_DTR_DISABLE;
00764 #else
00765     sCommand.InstructionDtrMode    = HAL_OSP
I_INSTRUCTION_DTR_ENABLE;
00766 #endif
00767
```

```
00768     /* Send the command */
00769     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00770     {
00771       return OSPI_NOR_ERROR;
00772     }
00773
00774     /*
00775     When this command is executed, the statu
s register write in progress bit is set to 1, and
00776     the flag status register program erase c
ontroller bit is set to 0. This command is ignored
00777     if the device is not in a suspended stat
e.
00778     */
00779
00780     if (BSP_OSPI_NOR_GetStatus() == OSPI_NOR
_BUSY)
00781     {
00782       return OSPI_NOR_OK;
00783     }
00784
00785     return OSPI_NOR_ERROR;
00786   }
00787
00788   return OSPI_NOR_OK;
00789 }
00790
00791 /**
00792   * @brief  This function enter the OSPI mem
ory in deep power down mode.
00793   * @retval OSPI memory status
00794   */
00795 uint8_t BSP_OSPI_NOR_EnterDeepPowerDown(void
)
00796 {
00797   OSPI_RegularCmdTypeDef sCommand;
```

```
00798
00799   /* Initialize the deep power down command
*/
00800   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
00801   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
00802   sCommand.Instruction           = OCTAL_ENT
ER_DEEP_POWER_DOWN_CMD;
00803   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_8_LINES;
00804   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_16_BITS;
00805   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_NONE;
00806   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00807   sCommand.DataMode              = HAL_OSPI_
DATA_NONE;
00808   sCommand.DummyCycles           = 0;
00809   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
00810   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00811
00812 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00813   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
00814 #else
00815   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00816 #endif
00817
00818   /* Send the command */
00819   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
```

```
00820   {
00821     return OSPI_NOR_ERROR;
00822   }
00823
00824   /* ---          Memory takes 10us max to e
nter deep power down          --- */
00825
00826   return OSPI_NOR_OK;
00827 }
00828
00829 /**
00830   * @brief  This function leave the OSPI mem
ory from deep power down mode.
00831   * @retval OSPI memory status
00832   */
00833 uint8_t BSP_OSPI_NOR_LeaveDeepPowerDown(void
)
00834 {
00835   OSPI_RegularCmdTypeDef sCommand;
00836
00837   /* Initialize the nop command */
00838   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
00839   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
00840   sCommand.Instruction           = OCTAL_NO_
OPERATION_CMD;
00841   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_8_LINES;
00842   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_16_BITS;
00843   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_NONE;
00844   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00845   sCommand.DataMode              = HAL_OSPI_
DATA_NONE;
```

```
00846   sCommand.DummyCycles            = 0;
00847   sCommand.DQSMode                = HAL_OSPI_
DQS_DISABLE;
00848   sCommand.SIOOMode               = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00849
00850 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

00851   sCommand.InstructionDtrMode     = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
00852 #else
00853   sCommand.InstructionDtrMode     = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
00854 #endif
00855
00856   /* Send the command */
00857   if (HAL_OSPI_Command(&OSPINORHandle, &Com
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00858   {
00859     return OSPI_NOR_ERROR;
00860   }
00861
00862   /* --- A NOP command is sent to the memory
, as the nCS should be low for at least 20 ns ---
*/
00863   /* ---                    Memory takes 30us
min to leave deep power down                    ---
*/
00864
00865   return OSPI_NOR_OK;
00866 }
00867 /**
00868   * @}
00869   */
00870
00871 /** @addtogroup STM32L4R9I_EVAL_OSPI_NOR_Pri
vate_Functions
```

```
00872   * @{
00873   */
00874
00875 /**
00876   * @brief  Initializes the OSPI MSP.
00877   * @retval None
00878   */
00879 static void OSPI_NOR_MspInit(void)
00880 {
00881   GPIO_InitTypeDef GPIO_InitStruct;
00882
00883   /* Enable the OctoSPI memory interface clo
ck */
00884   __HAL_RCC_OSPIM_CLK_ENABLE();
00885   __HAL_RCC_OSPI1_CLK_ENABLE();
00886
00887   /* Reset the OctoSPI memory interface */
00888   __HAL_RCC_OSPI1_FORCE_RESET();
00889   __HAL_RCC_OSPI1_RELEASE_RESET();
00890
00891   /* Enable GPIO clocks */
00892   /* IOSV bit MUST be set to access GPIO por
t G[2:15] */
00893   __HAL_RCC_PWR_CLK_ENABLE();
00894   SET_BIT(PWR->CR2, PWR_CR2_IOSV);
00895
00896   __HAL_RCC_GPIOA_CLK_ENABLE();
00897   __HAL_RCC_GPIOB_CLK_ENABLE();
00898   __HAL_RCC_GPIOC_CLK_ENABLE();
00899   __HAL_RCC_GPIOG_CLK_ENABLE();
00900   __HAL_RCC_GPIOH_CLK_ENABLE();
00901
00902   /* OctoSPI CS GPIO pin configuration  */
00903   GPIO_InitStruct.Pin       = GPIO_PIN_2;
00904   GPIO_InitStruct.Mode      = GPIO_MODE_AF_P
P;
00905   GPIO_InitStruct.Pull      = GPIO_PULLUP;
```

```
00906   GPIO_InitStruct.Speed     = GPIO_SPEED_FRE
Q_VERY_HIGH;
00907   GPIO_InitStruct.Alternate = GPIO_AF10_OCTO
SPIM_P1;
00908   HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
00909
00910   /* OctoSPI DQS GPIO pins configuration  */
00911   GPIO_InitStruct.Pin       = GPIO_PIN_2;
00912   HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
00913
00914   /* OctoSPI CLK, D2 and D3 GPIO pins config
uration  */
00915   GPIO_InitStruct.Pin       = (GPIO_PIN_3 |
GPIO_PIN_6 | GPIO_PIN_7);
00916   GPIO_InitStruct.Pull      = GPIO_NOPULL;
00917   HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
00918
00919   /* OctoSPI D0 and D1 GPIO pins configurati
on  */
00920   GPIO_InitStruct.Pin       = (GPIO_PIN_0 |
GPIO_PIN_1);
00921   HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
00922
00923   /* OctoSPI D6 and D7 GPIO pins configurati
on  */
00924   GPIO_InitStruct.Pin       = (GPIO_PIN_3 |
GPIO_PIN_4);
00925   HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
00926
00927   /* OctoSPI D5 GPIO pins configuration  */
00928   GPIO_InitStruct.Pin       = (GPIO_PIN_11);
00929   GPIO_InitStruct.Alternate = GPIO_AF3_OCTOS
PIM_P1;
00930   HAL_GPIO_Init(GPIOG, &GPIO_InitStruct);
00931
00932   /* OctoSPI D4 GPIO pins configuration  */
00933   GPIO_InitStruct.Pin       = (GPIO_PIN_2);
```

```
00934   HAL_GPIO_Init(GPIOH, &GPIO_InitStruct);
00935 }
00936
00937 /**
00938   * @brief  De-Initializes the OSPI MSP.
00939   * @retval None
00940   */
00941 static void OSPI_NOR_MspDeInit(void)
00942 {
00943   GPIO_InitTypeDef GPIO_InitStruct;
00944
00945   /* OctoSPI CLK, CS, D0-D7, DQS GPIO pins de-configuration  */
00946   HAL_GPIO_DeInit(GPIOA, (GPIO_PIN_6 | GPIO_PIN_7));
00947   HAL_GPIO_DeInit(GPIOB, (GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2));
00948   HAL_GPIO_DeInit(GPIOC, (GPIO_PIN_3 | GPIO_PIN_4));
00949   HAL_GPIO_DeInit(GPIOG, GPIO_PIN_11);
00950   HAL_GPIO_DeInit(GPIOH, GPIO_PIN_2);
00951
00952   /* Set GPIOA pin 2 in pull up mode (optimum default setting) */
00953   GPIO_InitStruct.Mode      = GPIO_MODE_INPUT;
00954   GPIO_InitStruct.Pin       = GPIO_PIN_2;
00955   GPIO_InitStruct.Pull      = GPIO_NOPULL;
00956   GPIO_InitStruct.Speed     = GPIO_SPEED_FREQ_LOW;
00957   HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
00958
00959   /* Set GPIOA pin 3 in no pull, low state (optimum default setting) */
00960   GPIO_InitStruct.Mode      = GPIO_MODE_OUTPUT_PP;
00961   GPIO_InitStruct.Pull      = GPIO_NOPULL;
```

```
00962   GPIO_InitStruct.Pin       = GPIO_PIN_3;
00963   HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
00964   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_
PIN_RESET);
00965
00966   /* Reset the OctoSPI memory interface */
00967   __HAL_RCC_OSPI1_FORCE_RESET();
00968   __HAL_RCC_OSPI1_RELEASE_RESET();
00969
00970   /* Disable the OctoSPI memory interface cl
ock */
00971   __HAL_RCC_OSPI1_CLK_DISABLE();
00972 }
00973
00974 /**
00975   * @brief  This function reset the OSPI mem
ory.
00976   * @param  hospi: OSPI handle
00977   * @retval None
00978   */
00979 static uint8_t OSPI_NOR_ResetMemory(OSPI_Han
dleTypeDef *hospi)
00980 {
00981   OSPI_RegularCmdTypeDef sCommand;
00982   OSPI_AutoPollingTypeDef sConfig;
00983
00984   /* Initialize the reset enable command */
00985   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
00986   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
00987   sCommand.Instruction           = RESET_ENA
BLE_CMD;
00988   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_1_LINE;
00989   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_8_BITS;
```

```
00990   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
00991   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_NONE;
00992   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
00993   sCommand.DataMode              = HAL_OSPI_
DATA_NONE;
00994   sCommand.DummyCycles           = 0;
00995   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
00996   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
00997
00998   /* Send the command */
00999   if (HAL_OSPI_Command(&OSPINORHandle, &Com
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01000   {
01001     return OSPI_NOR_ERROR;
01002   }
01003
01004   /* Send the reset memory command */
01005   sCommand.Instruction = RESET_MEMORY_CMD;
01006   if (HAL_OSPI_Command(&OSPINORHandle, &Com
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01007   {
01008     return OSPI_NOR_ERROR;
01009   }
01010
01011   /* Configure automatic polling mode to wai
t the memory is ready */
01012   sCommand.Instruction  = READ_STATUS_REG_CM
D;
01013   sCommand.DataMode     = HAL_OSPI_DATA_1_LI
NE;
01014   sCommand.NbData       = 1;
01015   sCommand.DataDtrMode  = HAL_OSPI_DATA_DTR_
```

```
DISABLE;
01016
01017   sConfig.Match         = 0;
01018   sConfig.Mask          = MX25LM51245G_SR_WI
P;
01019   sConfig.MatchMode     = HAL_OSPI_MATCH_MOD
E_AND;
01020   sConfig.Interval      = 0x10;
01021   sConfig.AutomaticStop = HAL_OSPI_AUTOMATIC
_STOP_ENABLE;
01022
01023   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01024   {
01025     return OSPI_NOR_ERROR;
01026   }
01027
01028   if (HAL_OSPI_AutoPolling(&OSPINORHandle, &
sConfig, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK
)
01029   {
01030     return OSPI_NOR_ERROR;
01031   }
01032
01033   return OSPI_NOR_OK;
01034 }
01035
01036 /**
01037   * @brief  This function send a Write Enabl
e and wait it is effective.
01038   * @param  hospi: OSPI handle
01039   * @retval None
01040   */
01041 static uint8_t OSPI_NOR_WriteEnable(OSPI_Han
dleTypeDef *hospi)
01042 {
01043   OSPI_RegularCmdTypeDef  sCommand;
```

```
01044   OSPI_AutoPollingTypeDef sConfig;
01045   uint8_t reg[2];
01046
01047   /* Enable write operations */
01048   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
01049   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
01050   sCommand.Instruction           = OCTAL_WRI
TE_ENABLE_CMD;
01051   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_8_LINES;
01052   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_16_BITS;
01053   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_NONE;
01054   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
01055   sCommand.DataMode              = HAL_OSPI_
DATA_NONE;
01056   sCommand.DummyCycles           = 0;
01057   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
01058   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
01059
01060 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

01061   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
01062 #else
01063   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
01064 #endif
01065
01066   if (HAL_OSPI_Command(&OSPINORHandle, &sCom
mand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
```

```
01067   {
01068     return OSPI_NOR_ERROR;
01069   }
01070
01071   /* Configure automatic polling mode to wai
t for write enabling */
01072   sConfig.Match           = MX25LM51245G_SR_
WEL;
01073   sConfig.Mask            = MX25LM51245G_SR_
WEL;
01074   sConfig.MatchMode       = HAL_OSPI_MATCH_M
ODE_AND;
01075   sConfig.Interval        = 0x10;
01076   sConfig.AutomaticStop   = HAL_OSPI_AUTOMAT
IC_STOP_ENABLE;
01077
01078   sCommand.Instruction    = OCTAL_READ_STATU
S_REG_CMD;
01079   sCommand.Address        = 0x0;
01080   sCommand.AddressMode    = HAL_OSPI_ADDRESS
_8_LINES;
01081   sCommand.AddressSize    = HAL_OSPI_ADDRESS
_32_BITS;
01082   sCommand.DataMode       = HAL_OSPI_DATA_8_
LINES;
01083   sCommand.NbData         = 2;
01084
01085 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

01086   sCommand.AddressDtrMode = HAL_OSPI_ADDRESS
_DTR_DISABLE;
01087   sCommand.DataDtrMode    = HAL_OSPI_DATA_DT
R_DISABLE;
01088   sCommand.DQSMode        = HAL_OSPI_DQS_DIS
ABLE;
01089   sCommand.DummyCycles    = DUMMY_CYCLES_REA
D_OCTAL;
```

```
01090 #else
01091   sCommand.AddressDtrMode = HAL_OSPI_ADDRESS
_DTR_ENABLE;
01092   sCommand.DataDtrMode    = HAL_OSPI_DATA_DT
R_ENABLE;
01093   sCommand.DQSMode        = HAL_OSPI_DQS_ENA
BLE;
01094   sCommand.DummyCycles    = 4;
01095 #endif
01096
01097   do
01098   {
01099     HAL_Delay(1); /* sConfig.Interval(0x10)
/ Clock(55 MHz) = 0.29 ms */
01100
01101     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01102     {
01103       return OSPI_NOR_ERROR;
01104     }
01105
01106     if (HAL_OSPI_Receive(&OSPINORHandle, reg
, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01107     {
01108       return OSPI_NOR_ERROR;
01109     }
01110   }while((reg[0] & sConfig.Mask ) != sConfig
.Match);
01111
01112   return OSPI_NOR_OK;
01113 }
01114
01115 /**
01116   * @brief  This function read the SR of the
 memory and wait the EOP.
01117   * @param  hospi: OSPI handle
01118   * @param  Timeout: Timeout for auto-polling
```

```
01119   * @retval None
01120   */
01121 static uint8_t OSPI_NOR_AutoPollingMemReady(
OSPI_HandleTypeDef *hospi, uint32_t Timeout)
01122 {
01123   OSPI_RegularCmdTypeDef  sCommand;
01124   OSPI_AutoPollingTypeDef sConfig;
01125   uint8_t reg[2];
01126
01127   /* Configure automatic polling mode to wai
t for memory ready */
01128   sCommand.OperationType         = HAL_OSPI_
OPTYPE_COMMON_CFG;
01129   sCommand.FlashId               = HAL_OSPI_
FLASH_ID_1;
01130   sCommand.Instruction           = OCTAL_REA
D_STATUS_REG_CMD;
01131   sCommand.InstructionMode       = HAL_OSPI_
INSTRUCTION_8_LINES;
01132   sCommand.InstructionSize       = HAL_OSPI_
INSTRUCTION_16_BITS;
01133   sCommand.Address               = 0;
01134   sCommand.AddressMode           = HAL_OSPI_
ADDRESS_8_LINES;
01135   sCommand.AddressSize           = HAL_OSPI_
ADDRESS_32_BITS;
01136   sCommand.AlternateBytesMode    = HAL_OSPI_
ALTERNATE_BYTES_NONE;
01137   sCommand.DataMode              = HAL_OSPI_
DATA_8_LINES;
01138   sCommand.NbData                = 2;
01139   sCommand.DummyCycles           = DUMMY_CYC
LES_READ_OCTAL;
01140   sCommand.SIOOMode              = HAL_OSPI_
SIOO_INST_EVERY_CMD;
01141
```

```
01142 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

01143   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_DISABLE;
01144   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_DISABLE;
01145   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_DISABLE;
01146   sCommand.DQSMode               = HAL_OSPI_
DQS_DISABLE;
01147 #else
01148   sCommand.InstructionDtrMode    = HAL_OSPI_
INSTRUCTION_DTR_ENABLE;
01149   sCommand.AddressDtrMode        = HAL_OSPI_
ADDRESS_DTR_ENABLE;
01150   sCommand.DataDtrMode           = HAL_OSPI_
DATA_DTR_ENABLE;
01151   sCommand.DQSMode               = HAL_OSPI_
DQS_ENABLE;
01152 #endif
01153
01154   sConfig.Match           = 0;
01155   sConfig.Mask            = MX25LM51245G_SR_
WIP;
01156   sConfig.MatchMode       = HAL_OSPI_MATCH_M
ODE_AND;
01157   sConfig.Interval        = 0x10;
01158   sConfig.AutomaticStop   = HAL_OSPI_AUTOMAT
IC_STOP_ENABLE;
01159
01160   do
01161   {
01162     HAL_Delay(1); /* sConfig.Interval(0x10)
/ Clock(55 MHz) = 0.29 ms */
01163
01164     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
```

```
01165     {
01166       return OSPI_NOR_ERROR;
01167     }
01168
01169     if (HAL_OSPI_Receive(&OSPINORHandle, reg
, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01170     {
01171       return OSPI_NOR_ERROR;
01172     }
01173   }while((reg[0] & sConfig.Mask ) != sConfig
.Match);
01174
01175   return OSPI_NOR_OK;
01176 }
01177
01178 /**
01179   * @brief  This function enables/disables t
he octal mode of the memory.
01180   * @param  hospi: OSPI handle
01181   * @param  Operation: Octal operation mode
enable/disable
01182   * @retval None
01183   */
01184 static uint8_t OSPI_NOR_OctalMode(OSPI_Handl
eTypeDef *hospi, uint8_t Operation)
01185 {
01186   OSPI_RegularCmdTypeDef  sCommand;
01187   OSPI_AutoPollingTypeDef sConfig;
01188   uint8_t reg[2];
01189
01190   sCommand.OperationType      = HAL_OSPI_OPT
YPE_COMMON_CFG;
01191   sCommand.FlashId            = HAL_OSPI_FLA
SH_ID_1;
01192   sCommand.InstructionDtrMode = HAL_OSPI_INS
TRUCTION_DTR_DISABLE;
01193   sCommand.AddressSize        = HAL_OSPI_ADD
```

```
RESS_32_BITS;
01194   sCommand.AddressDtrMode     = HAL_OSPI_ADD
RESS_DTR_DISABLE;
01195   sCommand.AlternateBytesMode = HAL_OSPI_ALT
ERNATE_BYTES_NONE;
01196   sCommand.DataDtrMode        = HAL_OSPI_DAT
A_DTR_DISABLE;
01197   sCommand.DQSMode            = HAL_OSPI_DQS
_DISABLE;
01198   sCommand.SIOOMode           = HAL_OSPI_SIO
O_INST_EVERY_CMD;
01199
01200   sConfig.MatchMode     = HAL_OSPI_MATCH_MOD
E_AND;
01201   sConfig.Interval      = 0x10;
01202   sConfig.AutomaticStop = HAL_OSPI_AUTOMATIC
_STOP_ENABLE;
01203
01204   /* Activate/deactivate the Octal mode */
01205   if (Operation == OSPI_NOR_OCTAL_ENABLE)
01206   {
01207     /* Enable write operations */
01208     sCommand.Instruction     = WRITE_ENABLE_
CMD;
01209     sCommand.InstructionMode = HAL_OSPI_INST
RUCTION_1_LINE;
01210     sCommand.InstructionSize = HAL_OSPI_INST
RUCTION_8_BITS;
01211     sCommand.AddressMode     = HAL_OSPI_ADDR
ESS_NONE;
01212     sCommand.DataMode        = HAL_OSPI_DATA
_NONE;
01213     sCommand.DummyCycles     = 0;
01214
01215     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01216     {
```

```
01217       return OSPI_NOR_ERROR;
01218     }
01219
01220     /* Configure automatic polling mode to w
ait for write enabling */
01221     sConfig.Match = MX25LM51245G_SR_WEL;
01222     sConfig.Mask  = MX25LM51245G_SR_WEL;
01223
01224     sCommand.Instruction = READ_STATUS_REG_C
MD;
01225     sCommand.DataMode    = HAL_OSPI_DATA_1_L
INE;
01226     sCommand.NbData      = 1;
01227
01228     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01229     {
01230       return OSPI_NOR_ERROR;
01231     }
01232
01233     if (HAL_OSPI_AutoPolling(&OSPINORHandle,
 &sConfig, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_
OK)
01234     {
01235       return OSPI_NOR_ERROR;
01236     }
01237
01238     /* Write Configuration register 2 (with
new dummy cycles) */
01239     sCommand.Instruction = WRITE_CFG_REG_2_C
MD;
01240     sCommand.Address     = MX25LM51245G_CR2_
REG3_ADDR;
01241     sCommand.AddressMode = HAL_OSPI_ADDRESS_
1_LINE;
01242     reg[0] = DUMMY_CYCLES_CR_CFG;
01243
```

```
01244     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01245     {
01246       return OSPI_NOR_ERROR;
01247     }
01248
01249     if (HAL_OSPI_Transmit(&OSPINORHandle, re
g, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01250     {
01251       return OSPI_NOR_ERROR;
01252     }
01253
01254     /* Enable write operations */
01255     sCommand.Instruction = WRITE_ENABLE_CMD;
01256     sCommand.AddressMode = HAL_OSPI_ADDRESS_
NONE;
01257     sCommand.DataMode    = HAL_OSPI_DATA_NON
E;
01258
01259     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01260     {
01261       return OSPI_NOR_ERROR;
01262     }
01263
01264     /* Configure automatic polling mode to w
ait for write enabling */
01265     sCommand.Instruction = READ_STATUS_REG_C
MD;
01266     sCommand.DataMode    = HAL_OSPI_DATA_1_L
INE;
01267
01268     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01269     {
01270       return OSPI_NOR_ERROR;
01271     }
```

```
01272
01273     if (HAL_OSPI_AutoPolling(&OSPINORHandle,
 &sConfig, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_
OK)
01274     {
01275       return OSPI_NOR_ERROR;
01276     }
01277
01278     /* Write Configuration register 2 (with
Octal I/O SPI protocol) */
01279     sCommand.Instruction = WRITE_CFG_REG_2_C
MD;
01280     sCommand.Address     = MX25LM51245G_CR2_
REG1_ADDR;
01281     sCommand.AddressMode = HAL_OSPI_ADDRESS_
1_LINE;
01282 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

01283     reg[0] = MX25LM51245G_CR2_SOPI;
01284 #else
01285     reg[0] = MX25LM51245G_CR2_DOPI;
01286 #endif
01287
01288     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01289     {
01290       return OSPI_NOR_ERROR;
01291     }
01292
01293     if (HAL_OSPI_Transmit(&OSPINORHandle, re
g, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01294     {
01295       return OSPI_NOR_ERROR;
01296     }
01297
01298     /* Wait that the configuration is effect
ive and check that memory is ready */
```

```
01299     HAL_Delay(MX25LM51245G_WRITE_REG_MAX_TIM
E);
01300
01301     if (OSPI_NOR_AutoPollingMemReady(&OSPINO
RHandle, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != OSPI_N
OR_OK)
01302     {
01303       return OSPI_NOR_ERROR;
01304     }
01305
01306     /* Check the configuration has been corr
ectly done */
01307     sCommand.Instruction     = OCTAL_READ_CF
G_REG_2_CMD;
01308     sCommand.InstructionMode = HAL_OSPI_INST
RUCTION_8_LINES;
01309     sCommand.InstructionSize = HAL_OSPI_INST
RUCTION_16_BITS;
01310     sCommand.AddressMode     = HAL_OSPI_ADDR
ESS_8_LINES;
01311     sCommand.DataMode        = HAL_OSPI_DATA
_8_LINES;
01312     sCommand.DummyCycles     = DUMMY_CYCLES_
READ_OCTAL;
01313     sCommand.NbData          = 2;
01314     reg[0] = 0;
01315
01316 #if !defined BSP_OSPI_NOR_DDR_MODE_DEACTIVAT
ED
01317     sCommand.InstructionDtrMode = HAL_OSPI_I
NSTRUCTION_DTR_ENABLE;
01318     sCommand.AddressDtrMode     = HAL_OSPI_A
DDRESS_DTR_ENABLE;
01319     sCommand.DataDtrMode        = HAL_OSPI_D
ATA_DTR_ENABLE;
01320     sCommand.DQSMode            = HAL_OSPI_D
QS_ENABLE;
```

```
01321 #endif
01322
01323     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01324     {
01325       return OSPI_NOR_ERROR;
01326     }
01327
01328     if (HAL_OSPI_Receive(&OSPINORHandle, reg
, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01329     {
01330       return OSPI_NOR_ERROR;
01331     }
01332
01333 #if defined BSP_OSPI_NOR_DDR_MODE_DEACTIVATED

01334     if (reg[0] != MX25LM51245G_CR2_SOPI)
01335 #else
01336     if (reg[0] != MX25LM51245G_CR2_DOPI)
01337 #endif
01338     {
01339       return OSPI_NOR_ERROR;
01340     }
01341   }
01342   else
01343   {
01344     /* Enable write operations */
01345     if (OSPI_NOR_WriteEnable(&OSPINORHandle)
 != OSPI_NOR_OK)
01346     {
01347       return OSPI_NOR_ERROR;
01348     }
01349
01350     /* Write Configuration register 2 (with
Octal I/O SPI protocol) */
01351     sCommand.Instruction     = OCTAL_WRITE_C
FG_REG_2_CMD;
```

```
01352     sCommand.InstructionMode = HAL_OSPI_INST
RUCTION_8_LINES;
01353     sCommand.InstructionSize = HAL_OSPI_INST
RUCTION_16_BITS;
01354     sCommand.Address         = MX25LM51245G_
CR2_REG1_ADDR;
01355     sCommand.AddressMode     = HAL_OSPI_ADDR
ESS_8_LINES;
01356     sCommand.DataMode        = HAL_OSPI_DATA
_8_LINES;
01357     sCommand.NbData          = 2;
01358     sCommand.DummyCycles     = 0;
01359     reg[0] = 0;
01360     reg[1] = 0;
01361
01362 #if !defined BSP_OSPI_NOR_DDR_MODE_DEACTIVAT
ED
01363     sCommand.InstructionDtrMode = HAL_OSPI_I
NSTRUCTION_DTR_ENABLE;
01364     sCommand.AddressDtrMode     = HAL_OSPI_A
DDRESS_DTR_ENABLE;
01365     sCommand.DataDtrMode        = HAL_OSPI_D
ATA_DTR_ENABLE;
01366 #endif
01367
01368     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01369     {
01370       return OSPI_NOR_ERROR;
01371     }
01372
01373     if (HAL_OSPI_Transmit(&OSPINORHandle, re
g, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01374     {
01375       return OSPI_NOR_ERROR;
01376     }
01377
```

```
01378     /* Wait that the configuration is effect
ive and check that memory is ready */
01379     HAL_Delay(MX25LM51245G_WRITE_REG_MAX_TIM
E);
01380
01381     sCommand.Instruction     = READ_STATUS_R
EG_CMD;
01382     sCommand.InstructionMode = HAL_OSPI_INST
RUCTION_1_LINE;
01383     sCommand.InstructionSize = HAL_OSPI_INST
RUCTION_8_BITS;
01384     sCommand.AddressMode     = HAL_OSPI_ADDR
ESS_NONE;
01385     sCommand.DataMode        = HAL_OSPI_DATA
_1_LINE;
01386     sCommand.NbData          = 1;
01387     sCommand.DummyCycles     = 0;
01388
01389 #if !defined BSP_OSPI_NOR_DDR_MODE_DEACTIVAT
ED
01390     sCommand.InstructionDtrMode = HAL_OSPI_I
NSTRUCTION_DTR_DISABLE;
01391     sCommand.AddressDtrMode     = HAL_OSPI_A
DDRESS_DTR_DISABLE;
01392     sCommand.DataDtrMode        = HAL_OSPI_D
ATA_DTR_DISABLE;
01393 #endif
01394
01395     sConfig.Match = 0;
01396     sConfig.Mask  = MX25LM51245G_SR_WIP;
01397
01398     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01399     {
01400       return OSPI_NOR_ERROR;
01401     }
01402
```

```
01403     if (HAL_OSPI_AutoPolling(&OSPINORHandle,
 &sConfig, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_
OK)
01404     {
01405       return OSPI_NOR_ERROR;
01406     }
01407
01408     /* Check the configuration has been corr
ectly done */
01409     sCommand.Instruction = READ_CFG_REG_2_CM
D;
01410     sCommand.Address     = MX25LM51245G_CR2_
REG1_ADDR;
01411     sCommand.AddressMode = HAL_OSPI_ADDRESS_
1_LINE;
01412
01413     if (HAL_OSPI_Command(&OSPINORHandle, &sC
ommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01414     {
01415       return OSPI_NOR_ERROR;
01416     }
01417
01418     if (HAL_OSPI_Receive(&OSPINORHandle, reg
, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
01419     {
01420       return OSPI_NOR_ERROR;
01421     }
01422
01423     if (reg[0] != 0)
01424     {
01425       return OSPI_NOR_ERROR;
01426     }
01427   }
01428
01429   return OSPI_NOR_OK;
01430 }
01431
```

```
01432 /**
01433   * @}
01434   */
01435
01436 /**
01437   * @}
01438   */
01439
01440 /**
01441   * @}
01442   */
01443
01444 /**
01445   * @}
01446   */
01447
01448 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
01449
```

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL SRAM

Modules

- STM32L476G_EVAL_SRAM_Private_Functions

Functions

uint8_t	**BSP_SRAM_Init** (void) Initializes the SRAM device.
uint8_t	**BSP_SRAM_ReadData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the SRAM device in polling mode.
uint8_t	**BSP_SRAM_ReadData_DMA** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the SRAM device in DMA mode.
uint8_t	**BSP_SRAM_WriteData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data from the SRAM device in polling mode.
uint8_t	**BSP_SRAM_WriteData_DMA** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data from the SRAM device in DMA mode.
void	**BSP_SRAM_DMA_IRQHandler** (void) Handles SRAM DMA transfer interrupt request.

Function Documentation

void BSP_SRAM_DMA_IRQHandler (void)

Handles SRAM DMA transfer interrupt request.

Return values:
None

Definition at line **254** of file **stm32l4r9i_eval_sram.c**.

References **sramHandle**.

uint8_t BSP_SRAM_Init (void)

Initializes the SRAM device.

Return values:
SRAM status

Definition at line **133** of file **stm32l4r9i_eval_sram.c**.

References **SRAM_ADDR_SETUP_TIME**, **SRAM_DATA_HOLD_TIME**, **SRAM_DATA_SETUP_TIME**, **SRAM_ERROR**, **SRAM_MspInit()**, **SRAM_OK**, **SRAM_TURN_AROUND_TIME**, **sramHandle**, and **Timing**.

uint8_t BSP_SRAM_ReadData (uint32_t uwStartAddress, uint16_t * pData, uint32_t uwDataSize)

Reads an amount of data from the SRAM device in polling mode.

Parameters:

uwStartAddress,:	Read start address
pData,:	Pointer to data to be read
uwDataSize,:	Size of read data from the memory

Return values:

SRAM status

Definition at line **181** of file **stm32l4r9i_eval_sram.c**.

References **SRAM_ERROR**, **SRAM_OK**, and **sramHandle**.

uint8_t BSP_SRAM_ReadData_DMA (uint32_t uwStartAddress,
uint16_t * pData,
uint32_t uwDataSize
)

Reads an amount of data from the SRAM device in DMA mode.

Parameters:

uwStartAddress,:	Read start address
pData,:	Pointer to data to be read
uwDataSize,:	Size of read data from the memory

Return values:

SRAM status

Definition at line **200** of file **stm32l4r9i_eval_sram.c**.

References **SRAM_ERROR**, **SRAM_OK**, and **sramHandle**.

uint8_t BSP_SRAM_WriteData (uint32_t uwStartAddress,
uint16_t * pData,
uint32_t uwDataSize

)

Writes an amount of data from the SRAM device in polling mode.

Parameters:

uwStartAddress,:	Write start address
pData,:	Pointer to data to be written
uwDataSize,:	Size of written data from the memory

Return values:

SRAM status

Definition at line **219** of file **stm32l4r9i_eval_sram.c**.

References **SRAM_ERROR**, **SRAM_OK**, and **sramHandle**.

uint8_t BSP_SRAM_WriteData_DMA (uint32_t uwStartAddress,
uint16_t * pData,
uint32_t uwDataSize
)

Writes an amount of data from the SRAM device in DMA mode.

Parameters:

uwStartAddress,:	Write start address
pData,:	Pointer to data to be written
uwDataSize,:	Size of written data from the memory

Return values:

SRAM status

Definition at line **238** of file **stm32l4r9i_eval_sram.c**.

References **SRAM_ERROR**, **SRAM_OK**, and **sramHandle**.

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by doxygen 1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Firmware Directory Reference

Directories

directory	**Drivers**

STM32L4R9I_EVAL BSP User Manual

Drivers Directory Reference

Directories

directory	**BSP**

STM32L4R9I_EVAL BSP User Manual

BSP Directory Reference

Directories

directory	**STM32L4R9I_EVAL**

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL Directory Reference

Files

file **stm32l4r9i_eval.c** [code]

This file provides a set of firmware functions to manage Leds, push-button and COM ports for STM32L4R9I_EVAL.

file **stm32l4r9i_eval.h** [code]

This file contains definitions for STM32L4R9I_EVAL's LEDs, push-buttons and COM ports hardware resources.

file **stm32l4r9i_eval_audio.c** [code]

This file provides the Audio driver for the STM32L4R9I-EVAL evaluation board.

file **stm32l4r9i_eval_audio.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_audio.c** driver.

file **stm32l4r9i_eval_dsi_lcd.c** [code]

This file includes the driver for DSI Liquid Crystal Display (LCD) module mounted on STM32L4R9I_EVAL evaluation board.

file **stm32l4r9i_eval_dsi_lcd.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_dsi_lcd.c** driver.

file **stm32l4r9i_eval_dsi_ts.c** [code]

This file provides a set of functions needed to manage the Touch Screen on DSI LCD of STM32L4R9I-EVAL evaluation board.

file **stm32l4r9i_eval_dsi_ts.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_dsi_ts.c** driver.

file **stm32l4r9i_eval_eeprom.c** [code]

This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory.

file **stm32l4r9i_eval_eeprom.h** [code]

This file contains all the functions prototypes for the **stm32l4r9i_eval_eeprom.c** firmware driver.

file **stm32l4r9i_eval_gfxmmu_lut.h** [code]

This file contains GFXMMU look-up-table for DSI round LCD.

file **stm32l4r9i_eval_idd.c** [code]

This file provides a set of firmware functions to manage the Idd measurement driver for STM32L4R9I_EVAL board.

file **stm32l4r9i_eval_idd.h** [code]

Header file for **stm32l4r9i_eval_idd.c** module.

file **stm32l4r9i_eval_io.c** [code]

This file provides a set of functions needed to manage the IO pins on STM32L4R9I_EVAL evaluation board.

file **stm32l4r9i_eval_io.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_io.c** driver.

file **stm32l4r9i_eval_lcd.h** [code]

This file contains alias for compatibility with DSI and TFT LCD.

file **stm32l4r9i_eval_nor.c** [code]

This file includes a standard driver for the M29W128GL70ZA6E NOR memories mounted on STM32L4R9I-EVAL board.

file **stm32l4r9i_eval_nor.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_nor.c** driver.

file **stm32l4r9i_eval_ospi_nor.c** [code]

This file includes a standard driver for the MX25UM51245G OSPI memory mounted on STM32L4R9I-EVAL board.

file **stm32l4r9i_eval_ospi_nor.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_ospi_nor.c** driver.

file **stm32l4r9i_eval_ospi_ram.c** [code]

This file includes a standard driver for the ISS66WVH8M8 HyperRAM memory mounted on STM32L4R9I-EVAL board.

file **stm32l4r9i_eval_ospi_ram.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_ospi_ram.c** driver.

file **stm32l4r9i_eval_rgb_lcd.c** [code]

This file includes the driver for Liquid Crystal Display (LCD) module mounted on STM32L4R9I_EVAL board.

file **stm32l4r9i_eval_rgb_lcd.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_rgb_lcd.c** driver.

file **stm32l4r9i_eval_rgb_ts.c** [code]

This file provides a set of functions needed to manage the Touch Screen on STM32746G-Discovery board.

file **stm32l4r9i_eval_rgb_ts.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_rgb_ts.c** driver.

file **stm32l4r9i_eval_sd.c** [code]

This file includes the uSD card driver.

file **stm32l4r9i_eval_sd.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_sd.c** driver.

file **stm32l4r9i_eval_sram.c** [code]

This file provides a set of functions needed to drive the IS61WV102416BLL-10MLI SRAM memory mounted on STM32L4R9I-EVAL board.

file **stm32l4r9i_eval_sram.h** [code]

This file contains the common defines and functions prototypes for the **stm32l4r9i_eval_sram.c** driver.

file **stm32l4r9i_eval_ts.h** [code]

This file contains the common defines and functions prototypes for the stm32l4r9i_eval_ts.c driver.

User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_dsi_ts.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_dsi_ts.c
00004   * @author  MCD Application Team
00005   * @brief   This file provides a set of functions needed to manage the Touch
00006   *          Screen on DSI LCD of STM32L4R9I-EVAL evaluation board.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* File Info : ------------------------------
-------------------------------------
00038                                         User NOTES

00039 1. How To use this driver:
00040 --------------------------
00041    - This driver is used to drive the touch
screen module of the STM32L4R9I-EVAL
00042      evaluation board on the DSI LCD mounted
 on MB1314 daughter board.
00043      The touch screen driver IC is a FT3X67
type.
00044
00045 2. Driver description:
00046 ---------------------
00047   + Initialization steps:
00048      o Initialize the TS module using the BS
P_TS_Init() function. This
00049        function includes the MSP layer hardw
are resources initialization and the
00050        communication layer configuration to
start the TS use. The LCD size properties
00051        (x and y) are passed as parameters.
00052      o If TS interrupt mode is desired, you
must configure the TS interrupt mode
00053        by calling the function BSP_TS_ITConf
ig(). The TS interrupt mode is generated
00054        as an external interrupt whenever a t
ouch is detected.
00055
00056   + Touch screen use
00057      o The touch screen state is captured wh
enever the function BSP_TS_GetState() is
00058        used. This function returns informati
```

```
on about the last LCD touch occurred
00059        in the TS_StateTypeDef structure.
00060      o FT3X67 doesn't support weight and are
a features (always 0 value returned).
00061
00062 ----------------------------------------
---------------------------------*/
00063
00064 /* Includes ------------------------------
---------------------------------*/
00065 #include "stm32l4r9i_eval_ts.h"
00066 #include "math.h"
00067
00068 /** @addtogroup BSP
00069   * @{
00070   */
00071
00072 /** @addtogroup STM32L4R9I_EVAL
00073   * @{
00074   */
00075
00076 #if defined(USE_GVO_390x390)
00077
00078 /** @addtogroup STM32L4R9I_EVAL_DSI_TS STM32
L4R9I_EVAL DSI TS
00079   * @{
00080   */
00081
00082 /** @defgroup STM32L4R9I_EVAL_DSI_TS_Private
_Variables Private Variables
00083   * @{
00084   */
00085
00086 static TS_DrvTypeDef *tsDriver;
00087 static uint8_t        I2C_Address   = 0;
00088 static uint8_t        HwRotation    = 0;
00089
```

```
00090 /**
00091   * @}
00092   */
00093
00094 /** @addtogroup STM32L4R9I_EVAL_DSI_TS_Expor
ted_Functions Exported Functions
00095   * @{
00096   */
00097
00098 /**
00099   * @brief  Initializes and configures the t
ouch screen functionalities and
00100   *         configures all necessary hardwar
e resources (GPIOs, I2C, clocks..).
00101   * @param  ts_SizeX : Maximum X size of the
 TS area on LCD
00102   * @param  ts_SizeY : Maximum Y size of the
 TS area on LCD
00103   * @retval TS_OK if all initializations are
 OK. Other value if error.
00104   */
00105 uint8_t BSP_DSI_TS_Init(uint16_t ts_SizeX, u
int16_t ts_SizeY)
00106 {
00107   uint8_t  status  = TS_OK;
00108
00109   /* Initialize the communication channel to
 sensor (I2C) if necessary */
00110   /* that is initialization is done only onc
e after a power up         */
00111   ft3x67_ts_drv.Init(TS_DSI_I2C_ADDRESS);
00112
00113   /* Scan FT3X67 TouchScreen IC controller I
D register by I2C Read */
00114   /* Verify this is a FT3X67, otherwise this
 is an error case      */
00115   if(ft3x67_ts_drv.ReadID(TS_DSI_I2C_ADDRESS
```

```
) == FT3X67_ID_VALUE)
00116   {
00117     /* Found FT3X67 : Initialize the TS driv
er structure */
00118     tsDriver    = &ft3x67_ts_drv;
00119     I2C_Address = TS_DSI_I2C_ADDRESS;
00120
00121     /* Calibrate, Configure and Start the To
uchScreen driver */
00122     tsDriver->Start(I2C_Address);
00123
00124     /* Read firmware register to know if HW
rotation is implemented */
00125     if(TS_IO_Read(I2C_Address, FT3X67_FIRMID
_REG) != 0x01)
00126     {
00127       HwRotation = 1;
00128     }
00129   }
00130   else
00131   {
00132     status = TS_DEVICE_NOT_FOUND;
00133   }
00134   return status;
00135 }
00136
00137 /**
00138   * @brief  DeInitializes the TouchScreen.
00139   * @retval TS state
00140   */
00141 uint8_t BSP_DSI_TS_DeInit(void)
00142 {
00143   /* Actually ts_driver does not provide a D
eInit function */
00144   return TS_OK;
00145 }
00146
```

```
00147 /**
00148   * @brief  Returns status and positions of
the touch screen.
00149   * @param  TS_State: Pointer to touch scree
n current state structure
00150   * @retval TS_OK if OK. Other value if erro
r.
00151   * @note   FT3X67 doesn't support weight an
d area features (always 0 value returned).
00152   */
00153 uint8_t BSP_DSI_TS_GetState(TS_StateTypeDef
*TS_State)
00154 {
00155   static uint32_t xf[TS_MAX_NB_TOUCH] = {0,
0}; /* Final x value */
00156   static uint32_t yf[TS_MAX_NB_TOUCH] = {0,
0}; /* Final y value */
00157   uint8_t  ts_status = TS_OK;
00158   uint16_t x[TS_MAX_NB_TOUCH]; /* Initial x
value */
00159   uint16_t y[TS_MAX_NB_TOUCH]; /* Initial y
value */
00160   int16_t  xc, yc;   /* Coordinates transfer
ed to center screen */
00161   int16_t  xr, yr;   /* Coordinates after ro
tation */
00162   uint32_t index;
00163   uint32_t weight = 0;
00164   uint32_t area   = 0;
00165   uint32_t event  = 0;
00166
00167   /* Check and update the number of touches
active detected */
00168   TS_State->touchDetected = tsDriver->Detect
Touch(I2C_Address);
00169   if(TS_State->touchDetected)
00170   {
```

```
00171     for(index = 0; index < TS_State->touchDe
tected; index++)
00172     {
00173       /* Get each touch coordinates */
00174       tsDriver->GetXY(I2C_Address, &(x[index
]), &(y[index]));
00175
00176       if(HwRotation == 0)
00177       {
00178         /* First translate coordonates to ce
nter screen */
00179         xc = x[index] - 195;
00180         yc = y[index] - 195;
00181
00182         /* Apply rotation of 45 */
00183         xr = (int16_t) (sqrt(2) * (xc - yc)
/ 2);
00184         yr = (int16_t) (sqrt(2) * (xc + yc)
/ 2);
00185
00186         /* Revert the initial translation */
00187         xf[index] = xr + 195;
00188         yf[index] = yr + 195;
00189
00190         TS_State->touchX[index] = xf[index];
00191         TS_State->touchY[index] = yf[index];
00192       }
00193       else
00194       {
00195         TS_State->touchX[index] = x[index];
00196         TS_State->touchY[index] = y[index];
00197       }
00198
00199       /* Get touch info related to the curre
nt touch */
00200       ft3x67_TS_GetTouchInfo(I2C_Address, in
dex, &weight, &area, &event);
```

```
00201
00202       /* Update TS_State structure */
00203       TS_State->touchWeight[index] = weight;
00204       TS_State->touchArea[index]   = area;
00205
00206       /* Remap touch event */
00207       switch(event)
00208       {
00209         case FT3X67_TOUCH_EVT_FLAG_PRESS_DOW
N  :
00210           TS_State->touchEventId[index] = TO
UCH_EVENT_PRESS_DOWN;
00211           break;
00212         case FT3X67_TOUCH_EVT_FLAG_LIFT_UP :
00213           TS_State->touchEventId[index] = TO
UCH_EVENT_LIFT_UP;
00214           break;
00215         case FT3X67_TOUCH_EVT_FLAG_CONTACT :
00216           TS_State->touchEventId[index] = TO
UCH_EVENT_CONTACT;
00217           break;
00218         case FT3X67_TOUCH_EVT_FLAG_NO_EVENT
:
00219           TS_State->touchEventId[index] = TO
UCH_EVENT_NO_EVT;
00220           break;
00221         default :
00222           ts_status = TS_ERROR;
00223           break;
00224       } /* of switch(event) */
00225
00226     } /* of for(index=0; index < TS_State->t
ouchDetected; index++) */
00227   } /* end of if(TS_State->touchDetected !=
0) */
00228
00229   return (ts_status);
```

```
00230 }
00231
00232 /**
00233   * @brief  Configure gesture feature.
00234   * @param  State: Enable/Disable gesture fe
ature.
00235   * @retval TS_OK if OK. Other value if erro
r.
00236   */
00237 uint8_t BSP_DSI_TS_GestureConfig(FunctionalS
tate State)
00238 {
00239   uint8_t  ts_status = TS_OK;
00240   uint32_t Activation;
00241
00242   /* Configure gesture feature */
00243   Activation = (State == ENABLE) ? FT3X67_GE
STURE_ENABLE : FT3X67_GESTURE_DISABLE;
00244   ft3x67_TS_GestureConfig(I2C_Address, Activ
ation);
00245
00246   return(ts_status);
00247 }
00248
00249 /**
00250   * @brief  Update gesture Id following a to
uch detected.
00251   * @param  TS_State: Pointer to touch scree
n current state structure
00252   * @retval TS_OK if OK. Other value if erro
r.
00253   */
00254 uint8_t BSP_DSI_TS_Get_GestureId(TS_StateTyp
eDef *TS_State)
00255 {
00256   uint32_t gestureId = 0;
00257   uint8_t  ts_status = TS_OK;
```

```
00258
00259   /* Get gesture Id */
00260   ft3x67_TS_GetGestureID(I2C_Address, &gestu
reId);
00261
00262   /* Remap gesture Id to a TS_GestureIdTypeD
ef value */
00263   switch(gestureId)
00264   {
00265     case FT3X67_GEST_ID_NO_GESTURE :
00266       TS_State->gestureId = GEST_ID_NO_GESTU
RE;
00267       break;
00268     case FT3X67_GEST_ID_MOVE_UP :
00269       TS_State->gestureId = GEST_ID_MOVE_UP;
00270       break;
00271     case FT3X67_GEST_ID_MOVE_RIGHT :
00272       TS_State->gestureId = GEST_ID_MOVE_RIG
HT;
00273       break;
00274     case FT3X67_GEST_ID_MOVE_DOWN :
00275       TS_State->gestureId = GEST_ID_MOVE_DOWN
;
00276       break;
00277     case FT3X67_GEST_ID_MOVE_LEFT :
00278       TS_State->gestureId = GEST_ID_MOVE_LEFT
;
00279       break;
00280     case FT3X67_GEST_ID_DOUBLE_CLICK :
00281       TS_State->gestureId = GEST_ID_DOUBLE_C
LICK;
00282       break;
00283     default :
00284       ts_status = TS_ERROR;
00285       break;
00286   } /* of switch(gestureId) */
00287
```

```
00288   return(ts_status);
00289 }
00290
00291 /**
00292   * @brief  Function used to reset all touch
 data before a new acquisition
00293   *         of touch information.
00294   * @param  TS_State: Pointer to touch scree
n current state structure
00295   * @retval TS_OK if OK, TS_ERROR if problem
 found.
00296   */
00297 uint8_t BSP_DSI_TS_ResetTouchData(TS_StateTy
peDef *TS_State)
00298 {
00299   uint8_t  ts_status = TS_ERROR;
00300   uint32_t index;
00301
00302   if (TS_State != (TS_StateTypeDef *)NULL)
00303   {
00304     TS_State->gestureId     = GEST_ID_NO_GES
TURE;
00305     TS_State->touchDetected = 0;
00306
00307     for(index = 0; index < TS_MAX_NB_TOUCH;
index++)
00308     {
00309       TS_State->touchX[index]       = 0;
00310       TS_State->touchY[index]       = 0;
00311       TS_State->touchArea[index]    = 0;
00312       TS_State->touchEventId[index] = TOUCH_
EVENT_NO_EVT;
00313       TS_State->touchWeight[index]  = 0;
00314     }
00315
00316     ts_status = TS_OK;
00317   } /* of if (TS_State != (TS_StateTypeDef *
```

```
)NULL) */
00318
00319   return (ts_status);
00320 }
00321
00322 /**
00323   * @brief  Configures and enables the touch
 screen interrupts.
00324   * @retval TS_OK if OK. Other value if erro
r.
00325   */
00326 uint8_t BSP_DSI_TS_ITConfig(void)
00327 {
00328   GPIO_InitTypeDef gpio_init_structure;
00329
00330   /* Configure Interrupt mode for TS detecti
on pin */
00331   gpio_init_structure.Pin = TS_DSI_INT_PIN;
00332   gpio_init_structure.Pull = GPIO_PULLUP;
00333   gpio_init_structure.Speed = GPIO_SPEED_FAS
T;
00334   gpio_init_structure.Mode = GPIO_MODE_IT_FA
LLING;
00335   HAL_GPIO_Init(TS_DSI_INT_GPIO_PORT, &gpio_
init_structure);
00336
00337   /* Enable and set Touch screen EXTI Interr
upt to the lowest priority */
00338   HAL_NVIC_SetPriority((IRQn_Type)(TS_DSI_IN
T_EXTI_IRQn), 0x0F, 0x00);
00339   HAL_NVIC_EnableIRQ((IRQn_Type)(TS_DSI_INT_
EXTI_IRQn));
00340
00341   /* Enable the TS ITs */
00342   tsDriver->EnableIT(I2C_Address);
00343
00344   return TS_OK;
```

```
00345 }
00346
00347 /**
00348   * @brief  Clears all touch screen interrup
ts.
00349   */
00350 void BSP_DSI_TS_ITClear(void)
00351 {
00352   /* Empty function on component FT3X67 */
00353 }
00354
00355 /**
00356   * @brief  Gets the touch screen interrupt
status.
00357   * @retval TS_OK if OK. Other value if erro
r.
00358   */
00359 uint8_t BSP_DSI_TS_ITGetStatus(void)
00360 {
00361   /* Return the TS_OK because feature not av
ailable on FT3X67 */
00362   return TS_OK;
00363 }
00364
00365 /**
00366   * @}
00367   */
00368
00369 /**
00370   * @}
00371   */
00372
00373 #endif /* USE_GVO_390x390 */
00374
00375 /**
00376   * @}
00377   */
```

```
00378
00379 /**
00380   * @}
00381   */
00382
00383 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_rgb_lcd.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_rgb_lcd.c
00004   * @author  MCD Application Team
00005   * @brief   This file includes the driver for Liquid Crystal Display (LCD) module
00006   *          mounted on STM32L4R9I_EVAL board.
00007   @verbatim
00008   1. How To use this driver:
00009   --------------------------
00010      - This driver is used to drive directly an LCD TFT using the LTDC controller.
00011      - This driver uses timing and setting for RK043FN48H LCD.
00012
00013   2. Driver description:
00014   ---------------------
00015     + Initialization steps:
00016        o Initialize the LCD using the BSP_RGB_LCD_Init() function.
```

```
00017         o Apply the Layer configuration using
 the BSP_RGB_LCD_LayerDefaultInit() function.
00018         o Select the LCD layer to be used usi
ng the BSP_RGB_LCD_SelectLayer() function.
00019         o Enable the LCD display using the BS
P_RGB_LCD_DisplayOn() function.
00020
00021     + Options
00022         o Configure and enable the color keyi
ng functionality using the
00023           BSP_RGB_LCD_SetColorKeying() functi
on.
00024         o Modify in the fly the transparency
and/or the frame buffer address
00025           using the following functions:
00026           - BSP_RGB_LCD_SetTransparency()
00027           - BSP_RGB_LCD_SetLayerAddress()
00028
00029     + Display on LCD
00030         o Clear the hole LCD using BSP_RGB_LC
D_Clear() function or only one specified string
00031           line using the BSP_RGB_LCD_ClearStr
ingLine() function.
00032         o Display a character on the specifie
d line and column using the BSP_RGB_LCD_DisplayCha
r()
00033           function or a complete string line
using the BSP_RGB_LCD_DisplayStringAtLine() functi
on.
00034         o Display a string line on the specif
ied position (x,y in pixel) and align mode
00035           using the BSP_RGB_LCD_DisplayString
AtLine() function.
00036         o Draw and fill a basic shapes (dot,
line, rectangle, circle, ellipse, .. bitmap)
00037           on LCD using the available set of f
unctions.
```

```
00038   @endverbatim
00039   ******************************************************************************
00040   * @attention
00041   *
00042   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00043   *
00044   * Redistribution and use in source and binary forms, with or without modification,
00045   * are permitted provided that the following conditions are met:
00046   *   1. Redistributions of source code must retain the above copyright notice,
00047   *      this list of conditions and the following disclaimer.
00048   *   2. Redistributions in binary form must reproduce the above copyright notice,
00049   *      this list of conditions and the following disclaimer in the documentation
00050   *      and/or other materials provided with the distribution.
00051   *   3. Neither the name of STMicroelectronics nor the names of its contributors
00052   *      may be used to endorse or promote products derived from this software
00053   *      without specific prior written permission.
00054   *
00055   * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
00056   * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
00057   * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
00058   * DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
```

```
00059   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00060   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00061   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00062   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00063   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00064   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00065   *
00066   ****************************************
**********************************
00067   */
00068
00069 /* Includes ------------------------------
-----------------------------------*/
00070 #include "stm32l4r9i_eval_lcd.h"
00071 #include "../../../Utilities/Fonts/fonts.h"
00072 #include "../../../Utilities/Fonts/font24.c"
00073 #include "../../../Utilities/Fonts/font20.c"
00074 #include "../../../Utilities/Fonts/font16.c"
00075 #include "../../../Utilities/Fonts/font12.c"
00076 #include "../../../Utilities/Fonts/font8.c"
00077
00078 /** @addtogroup BSP
00079   * @{
00080   */
00081
00082 /** @addtogroup STM32L4R9I_EVAL
00083   * @{
00084   */
00085
00086 /** @defgroup STM32L4R9I_EVAL_LCD STM32L4R9I
_EVAL LCD
```

```
00087   * @{
00088   */
00089
00090 /** @defgroup STM32L4R9I_EVAL_RGB_LCD STM32L
4R9I_EVAL RGB LCD
00091   * @{
00092   */
00093
00094 #if defined(USE_ROCKTECH_480x272)
00095
00096 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Privat
e_Defines Private Defines
00097   * @{
00098   */
00099 #define POLY_X(Z)              ((int32_t)((P
oints + Z)->X))
00100 #define POLY_Y(Z)              ((int32_t)((P
oints + Z)->Y))
00101
00102 /**
00103   * @}
00104   */
00105
00106 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Privat
e_Macros Private Macros
00107   * @{
00108   */
00109 #define ABS(X)  ((X) > 0 ? (X) : -(X))
00110 /**
00111   * @}
00112   */
00113
00114 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Export
ed_Variables Exported Variables
00115   * @{
00116   */
00117 /* DMA2D handle */
```

```
00118 DMA2D_HandleTypeDef  hdma2d_eval;
00119 /**
00120   * @}
00121   */
00122
00123 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Privat
e_Variables Private Variables
00124   * @{
00125   */
00126 /* Physical frame buffer for active layer */
00127 uint32_t  PhysFrameBuffer[RK043FN48H_HEIGHT
* RK043FN48H_WIDTH]; /* 480*272 pixels with 32bpp
*/
00128
00129 /* LTDC handle */
00130 LTDC_HandleTypeDef  hltdc_eval;
00131
00132 /* Default LCD configuration with LCD Layer
1 */
00133 static uint32_t            ActiveLayer = LTD
C_DEFAULT_ACTIVE_LAYER;
00134 static LCD_DrawPropTypeDef DrawProp[LTDC_MAX
_LAYER_NUMBER];
00135 /**
00136   * @}
00137   */
00138
00139 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Privat
e_FunctionPrototypes Private Function Prototypes
00140   * @{
00141   */
00142 static void DrawChar(uint16_t Xpos, uint16_t
 Ypos, const uint8_t *c);
00143 static void FillTriangle(uint16_t x1, uint16
_t x2, uint16_t x3, uint16_t y1, uint16_t y2, uint
16_t y3);
00144 static void LL_FillBuffer(uint32_t LayerInde
```

```
x, void *pDst, uint32_t xSize, uint32_t ySize, uin
t32_t OffLine, uint32_t ColorIndex);
00145 static void LL_ConvertLineToARGB8888(void *
pSrc, void *pDst, uint32_t xSize, uint32_t ColorMo
de);
00146 /**
00147   * @}
00148   */
00149
00150 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Export
ed_Functions Exported Functions
00151   * @{
00152   */
00153
00154 /**
00155   * @brief  Initializes the LCD.
00156   * @retval LCD state
00157   */
00158 uint8_t BSP_RGB_LCD_Init(void)
00159 {
00160   /* Select the used LCD */
00161
00162   /* The RK043FN48H LCD 480x272 is selected
*/
00163   /* Timing Configuration */
00164   hltdc_eval.Init.HorizontalSync = (RK043FN4
8H_HSYNC - 1);
00165   hltdc_eval.Init.VerticalSync = (RK043FN48H
_VSYNC - 1);
00166   hltdc_eval.Init.AccumulatedHBP = (RK043FN4
8H_HSYNC + RK043FN48H_HBP - 1);
00167   hltdc_eval.Init.AccumulatedVBP = (RK043FN4
8H_VSYNC + RK043FN48H_VBP - 1);
00168   hltdc_eval.Init.AccumulatedActiveH = (RK04
3FN48H_HEIGHT + RK043FN48H_VSYNC + RK043FN48H_VBP
- 1);
00169   hltdc_eval.Init.AccumulatedActiveW = (RK04
```

```
3FN48H_WIDTH + RK043FN48H_HSYNC + RK043FN48H_HBP -
 1);
00170   hltdc_eval.Init.TotalHeigh = (RK043FN48H_H
EIGHT + RK043FN48H_VSYNC + RK043FN48H_VBP + RK043F
N48H_VFP - 1);
00171   hltdc_eval.Init.TotalWidth = (RK043FN48H_W
IDTH + RK043FN48H_HSYNC + RK043FN48H_HBP + RK043FN
48H_HFP - 1);
00172
00173   /* LCD clock configuration */
00174   BSP_RGB_LCD_ClockConfig(&hltdc_eval, NULL)
;
00175
00176   /* Initialize the LCD pixel width and pixe
l height */
00177   hltdc_eval.LayerCfg->ImageWidth  = RK043FN
48H_WIDTH;
00178   hltdc_eval.LayerCfg->ImageHeight = RK043FN
48H_HEIGHT;
00179
00180   /* Background value */
00181   hltdc_eval.Init.Backcolor.Blue = 0;
00182   hltdc_eval.Init.Backcolor.Green = 0;
00183   hltdc_eval.Init.Backcolor.Red = 0;
00184
00185   /* Polarity */
00186   hltdc_eval.Init.HSPolarity = LTDC_HSPOLARI
TY_AL;
00187   hltdc_eval.Init.VSPolarity = LTDC_VSPOLARI
TY_AL;
00188   hltdc_eval.Init.DEPolarity = LTDC_DEPOLARI
TY_AL;
00189   hltdc_eval.Init.PCPolarity = LTDC_PCPOLARI
TY_IPC;
00190   hltdc_eval.Instance = LTDC;
00191
00192   if(HAL_LTDC_GetState(&hltdc_eval) == HAL_L
```

```
TDC_STATE_RESET)
00193   {
00194     /* Initialize the LCD Msp: this __weak f
unction can be rewritten by the application */
00195     BSP_RGB_LCD_MspInit(&hltdc_eval, NULL);
00196   }
00197   HAL_LTDC_Init(&hltdc_eval);
00198
00199   /* Screen initialization */
00200   BSP_IO_WritePin(LCD_RGB_RST_PIN, IO_PIN_RE
SET );
00201   HAL_Delay(2);
00202   BSP_IO_WritePin(LCD_RGB_RST_PIN, IO_PIN_SE
T );
00203
00204   /* Assert display enable LCD_DISP pin via
MFX expander */
00205   BSP_IO_WritePin(LCD_RGB_DISP_PIN, IO_PIN_S
ET);
00206
00207   /* Assert backlight LCD_BL_CTRL pin */
00208   HAL_GPIO_WritePin(LCD_RGB_BL_CTRL_GPIO_PORT
, LCD_RGB_BL_CTRL_PIN, GPIO_PIN_SET);
00209
00210   /* Initialize the font */
00211   BSP_RGB_LCD_SetFont(&LCD_DEFAULT_FONT);
00212
00213   return LCD_OK;
00214 }
00215
00216 /**
00217   * @brief  DeInitializes the LCD.
00218   * @retval LCD state
00219   */
00220 uint8_t BSP_RGB_LCD_DeInit(void)
00221 {
00222   /* Initialize the hltdc_eval Instance para
```

```
meter */
00223   hltdc_eval.Instance = LTDC;
00224
00225  /* Disable LTDC block */
00226   __HAL_LTDC_DISABLE(&hltdc_eval);
00227
00228   /* DeInit the LTDC */
00229   HAL_LTDC_DeInit(&hltdc_eval);
00230
00231   /* DeInit the LTDC MSP : this __weak funct
ion can be rewritten by the application */
00232   BSP_RGB_LCD_MspDeInit(&hltdc_eval, NULL);
00233
00234   return LCD_OK;
00235 }
00236
00237 /**
00238   * @brief  Gets the LCD X size.
00239   * @retval Used LCD X size
00240   */
00241 uint32_t BSP_RGB_LCD_GetXSize(void)
00242 {
00243   return hltdc_eval.LayerCfg[ActiveLayer].Im
ageWidth;
00244 }
00245
00246 /**
00247   * @brief  Gets the LCD Y size.
00248   * @retval Used LCD Y size
00249   */
00250 uint32_t BSP_RGB_LCD_GetYSize(void)
00251 {
00252   return hltdc_eval.LayerCfg[ActiveLayer].Im
ageHeight;
00253 }
00254
00255 /**
```

```
00256   * @brief  Set the LCD X size.
00257   * @param  imageWidthPixels : image width i
n pixels unit
00258   * @retval None
00259   */
00260 void BSP_RGB_LCD_SetXSize(uint32_t imageWidt
hPixels)
00261 {
00262   hltdc_eval.LayerCfg[ActiveLayer].ImageWidt
h = imageWidthPixels;
00263 }
00264
00265 /**
00266   * @brief  Set the LCD Y size.
00267   * @param  imageHeightPixels : image height
 in lines unit
00268   * @retval None
00269   */
00270 void BSP_RGB_LCD_SetYSize(uint32_t imageHeig
htPixels)
00271 {
00272   hltdc_eval.LayerCfg[ActiveLayer].ImageHeig
ht = imageHeightPixels;
00273 }
00274
00275 /**
00276   * @brief  Initializes the LCD layer in ARG
B8888 format (32 bits per pixel).
00277   * @param  LayerIndex: Layer foreground or
background
00278   * @param  FB_Address: Layer frame buffer
00279   * @retval None
00280   */
00281 void BSP_RGB_LCD_LayerDefaultInit(uint16_t L
ayerIndex, uint32_t FB_Address)
00282 {
00283   LCD_LayerCfgTypeDef  layer_cfg;
```

```
00284
00285   /* Layer Init */
00286   layer_cfg.WindowX0 = 0;
00287   layer_cfg.WindowX1 = BSP_RGB_LCD_GetXSize(
);
00288   layer_cfg.WindowY0 = 0;
00289   layer_cfg.WindowY1 = BSP_RGB_LCD_GetYSize(
);
00290   layer_cfg.PixelFormat = LTDC_PIXEL_FORMAT_
ARGB8888;
00291   layer_cfg.FBStartAdress = FB_Address;
00292   layer_cfg.Alpha = 255;
00293   layer_cfg.Alpha0 = 0;
00294   layer_cfg.Backcolor.Blue = 0;
00295   layer_cfg.Backcolor.Green = 0;
00296   layer_cfg.Backcolor.Red = 0;
00297   layer_cfg.BlendingFactor1 = LTDC_BLENDING_
FACTOR1_PAxCA;
00298   layer_cfg.BlendingFactor2 = LTDC_BLENDING_
FACTOR2_PAxCA;
00299   layer_cfg.ImageWidth = BSP_RGB_LCD_GetXSize
();
00300   layer_cfg.ImageHeight = BSP_RGB_LCD_GetYSi
ze();
00301
00302   HAL_LTDC_ConfigLayer(&hltdc_eval, &layer_c
fg, LayerIndex);
00303
00304   DrawProp[LayerIndex].BackColor = LCD_COLOR
_WHITE;
00305   DrawProp[LayerIndex].pFont     = &Font24;
00306   DrawProp[LayerIndex].TextColor = LCD_COLOR
_BLACK;
00307 }
00308
00309 /**
00310   * @brief  Initializes the LCD layer in RGB
```

```
565 format (16 bits per pixel).
00311   * @param  LayerIndex: Layer foreground or
background
00312   * @param  FB_Address: Layer frame buffer
00313   * @retval None
00314   */
00315 void BSP_RGB_LCD_LayerRgb565Init(uint16_t La
yerIndex, uint32_t FB_Address)
00316 {
00317   LCD_LayerCfgTypeDef  layer_cfg;
00318
00319   /* Layer Init */
00320   layer_cfg.WindowX0 = 0;
00321   layer_cfg.WindowX1 = BSP_RGB_LCD_GetXSize(
);
00322   layer_cfg.WindowY0 = 0;
00323   layer_cfg.WindowY1 = BSP_RGB_LCD_GetYSize(
);
00324   layer_cfg.PixelFormat = LTDC_PIXEL_FORMAT_
RGB565;
00325   layer_cfg.FBStartAdress = FB_Address;
00326   layer_cfg.Alpha = 255;
00327   layer_cfg.Alpha0 = 0;
00328   layer_cfg.Backcolor.Blue = 0;
00329   layer_cfg.Backcolor.Green = 0;
00330   layer_cfg.Backcolor.Red = 0;
00331   layer_cfg.BlendingFactor1 = LTDC_BLENDING_
FACTOR1_PAxCA;
00332   layer_cfg.BlendingFactor2 = LTDC_BLENDING_
FACTOR2_PAxCA;
00333   layer_cfg.ImageWidth = BSP_RGB_LCD_GetXSize
();
00334   layer_cfg.ImageHeight = BSP_RGB_LCD_GetYSi
ze();
00335
00336   HAL_LTDC_ConfigLayer(&hltdc_eval, &layer_c
fg, LayerIndex);
```

```
00337
00338   DrawProp[LayerIndex].BackColor = LCD_COLOR
_WHITE;
00339   DrawProp[LayerIndex].pFont     = &Font24;
00340   DrawProp[LayerIndex].TextColor = LCD_COLOR
_BLACK;
00341 }
00342
00343 /**
00344   * @brief  Selects the LCD Layer.
00345   * @param  LayerIndex: Layer foreground or
background
00346   * @retval None
00347   */
00348 void BSP_RGB_LCD_SelectLayer(uint32_t LayerI
ndex)
00349 {
00350   ActiveLayer = LayerIndex;
00351 }
00352
00353 /**
00354   * @brief  Sets an LCD Layer visible
00355   * @param  LayerIndex: Visible Layer
00356   * @param  State: New state of the specifie
d layer
00357   *          This parameter can be one of th
e following values:
00358   *            @arg  ENABLE
00359   *            @arg  DISABLE
00360   * @retval None
00361   */
00362 void BSP_RGB_LCD_SetLayerVisible(uint32_t La
yerIndex, FunctionalState State)
00363 {
00364   if(State == ENABLE)
00365   {
00366     __HAL_LTDC_LAYER_ENABLE(&hltdc_eval, Lay
```

```
erIndex);
00367   }
00368   else
00369   {
00370     __HAL_LTDC_LAYER_DISABLE(&hltdc_eval, La
yerIndex);
00371   }
00372   __HAL_LTDC_RELOAD_IMMEDIATE_CONFIG(&hltdc_
eval);
00373 }
00374
00375 /**
00376   * @brief  Sets an LCD Layer visible withou
t reloading.
00377   * @param  LayerIndex: Visible Layer
00378   * @param  State: New state of the specifie
d layer
00379   *          This parameter can be one of th
e following values:
00380   *            @arg  ENABLE
00381   *            @arg  DISABLE
00382   * @retval None
00383   */
00384 void BSP_RGB_LCD_SetLayerVisible_NoReload(ui
nt32_t LayerIndex, FunctionalState State)
00385 {
00386   if(State == ENABLE)
00387   {
00388     __HAL_LTDC_LAYER_ENABLE(&hltdc_eval, Lay
erIndex);
00389   }
00390   else
00391   {
00392     __HAL_LTDC_LAYER_DISABLE(&hltdc_eval, La
yerIndex);
00393   }
00394   /* Do not set the Reload  */
```

```
00395 }
00396
00397 /**
00398   * @brief  Configures the transparency.
00399   * @param  LayerIndex: Layer foreground or
background.
00400   * @param  Transparency: Transparency
00401   *           This parameter must be a numbe
r between Min_Data = 0x00 and Max_Data = 0xFF
00402   * @retval None
00403   */
00404 void BSP_RGB_LCD_SetTransparency(uint32_t La
yerIndex, uint8_t Transparency)
00405 {
00406   HAL_LTDC_SetAlpha(&hltdc_eval, Transparenc
y, LayerIndex);
00407 }
00408
00409 /**
00410   * @brief  Configures the transparency with
out reloading.
00411   * @param  LayerIndex: Layer foreground or
background.
00412   * @param  Transparency: Transparency
00413   *           This parameter must be a numbe
r between Min_Data = 0x00 and Max_Data = 0xFF
00414   * @retval None
00415   */
00416 void BSP_RGB_LCD_SetTransparency_NoReload(ui
nt32_t LayerIndex, uint8_t Transparency)
00417 {
00418   HAL_LTDC_SetAlpha_NoReload(&hltdc_eval, Tr
ansparency, LayerIndex);
00419 }
00420
00421 /**
00422   * @brief  Sets an LCD layer frame buffer a
```

```
ddress.
00423   * @param  LayerIndex: Layer foreground or
background
00424   * @param  Address: New LCD frame buffer va
lue
00425   * @retval None
00426   */
00427 void BSP_RGB_LCD_SetLayerAddress(uint32_t La
yerIndex, uint32_t Address)
00428 {
00429   HAL_LTDC_SetAddress(&hltdc_eval, Address,
LayerIndex);
00430 }
00431
00432 /**
00433   * @brief  Sets an LCD layer frame buffer a
ddress without reloading.
00434   * @param  LayerIndex: Layer foreground or
background
00435   * @param  Address: New LCD frame buffer va
lue
00436   * @retval None
00437   */
00438 void BSP_RGB_LCD_SetLayerAddress_NoReload(ui
nt32_t LayerIndex, uint32_t Address)
00439 {
00440   HAL_LTDC_SetAddress_NoReload(&hltdc_eval,
Address, LayerIndex);
00441 }
00442
00443 /**
00444   * @brief  Sets display window.
00445   * @param  LayerIndex: Layer index
00446   * @param  Xpos: LCD X position
00447   * @param  Ypos: LCD Y position
00448   * @param  Width: LCD window width
00449   * @param  Height: LCD window height
```

```
00450   * @retval None
00451   */
00452 void BSP_RGB_LCD_SetLayerWindow(uint16_t Lay
erIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Wi
dth, uint16_t Height)
00453 {
00454   /* Reconfigure the layer size */
00455   HAL_LTDC_SetWindowSize(&hltdc_eval, Width,
 Height, LayerIndex);
00456
00457   /* Reconfigure the layer position */
00458   HAL_LTDC_SetWindowPosition(&hltdc_eval, Xp
os, Ypos, LayerIndex);
00459 }
00460
00461 /**
00462   * @brief  Sets display window without relo
ading.
00463   * @param  LayerIndex: Layer index
00464   * @param  Xpos: LCD X position
00465   * @param  Ypos: LCD Y position
00466   * @param  Width: LCD window width
00467   * @param  Height: LCD window height
00468   * @retval None
00469   */
00470 void BSP_RGB_LCD_SetLayerWindow_NoReload(uin
t16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, ui
nt16_t Width, uint16_t Height)
00471 {
00472   /* Reconfigure the layer size */
00473   HAL_LTDC_SetWindowSize_NoReload(&hltdc_eval
, Width, Height, LayerIndex);
00474
00475   /* Reconfigure the layer position */
00476   HAL_LTDC_SetWindowPosition_NoReload(&hltdc
_eval, Xpos, Ypos, LayerIndex);
00477 }
```

```
00478
00479 /**
00480   * @brief  Configures and sets the color ke
ying.
00481   * @param  LayerIndex: Layer foreground or
background
00482   * @param  RGBValue: Color reference
00483   * @retval None
00484   */
00485 void BSP_RGB_LCD_SetColorKeying(uint32_t Lay
erIndex, uint32_t RGBValue)
00486 {
00487   /* Configure and Enable the color Keying f
or LCD Layer */
00488   HAL_LTDC_ConfigColorKeying(&hltdc_eval, RG
BValue, LayerIndex);
00489   HAL_LTDC_EnableColorKeying(&hltdc_eval, La
yerIndex);
00490 }
00491
00492 /**
00493   * @brief  Configures and sets the color ke
ying without reloading.
00494   * @param  LayerIndex: Layer foreground or
background
00495   * @param  RGBValue: Color reference
00496   * @retval None
00497   */
00498 void BSP_RGB_LCD_SetColorKeying_NoReload(uin
t32_t LayerIndex, uint32_t RGBValue)
00499 {
00500   /* Configure and Enable the color Keying f
or LCD Layer */
00501   HAL_LTDC_ConfigColorKeying_NoReload(&hltdc
_eval, RGBValue, LayerIndex);
00502   HAL_LTDC_EnableColorKeying_NoReload(&hltdc
_eval, LayerIndex);
```

```
00503 }
00504
00505 /**
00506   * @brief  Disables the color keying.
00507   * @param  LayerIndex: Layer foreground or
background
00508   * @retval None
00509   */
00510 void BSP_RGB_LCD_ResetColorKeying(uint32_t L
ayerIndex)
00511 {
00512   /* Disable the color Keying for LCD Layer
*/
00513   HAL_LTDC_DisableColorKeying(&hltdc_eval, L
ayerIndex);
00514 }
00515
00516 /**
00517   * @brief  Disables the color keying withou
t reloading.
00518   * @param  LayerIndex: Layer foreground or
background
00519   * @retval None
00520   */
00521 void BSP_RGB_LCD_ResetColorKeying_NoReload(u
int32_t LayerIndex)
00522 {
00523   /* Disable the color Keying for LCD Layer
*/
00524   HAL_LTDC_DisableColorKeying_NoReload(&hltd
c_eval, LayerIndex);
00525 }
00526
00527 /**
00528   * @brief  Disables the color keying withou
t reloading.
00529   * @param  ReloadType: can be one of the fo
```

```
llowing values
00530   *           - LCD_RELOAD_IMMEDIATE
00531   *           - LCD_RELOAD_VERTICAL_BLANKING
00532   * @retval None
00533   */
00534 void BSP_RGB_LCD_Reload(uint32_t ReloadType)
00535 {
00536   HAL_LTDC_Reload (&hltdc_eval, ReloadType);
00537 }
00538
00539 /**
00540   * @brief  Sets the LCD text color.
00541   * @param  Color: Text color code ARGB(8-8-
8-8)
00542   * @retval None
00543   */
00544 void BSP_RGB_LCD_SetTextColor(uint32_t Color
)
00545 {
00546   DrawProp[ActiveLayer].TextColor = Color;
00547 }
00548
00549 /**
00550   * @brief  Gets the LCD text color.
00551   * @retval Used text color.
00552   */
00553 uint32_t BSP_RGB_LCD_GetTextColor(void)
00554 {
00555   return DrawProp[ActiveLayer].TextColor;
00556 }
00557
00558 /**
00559   * @brief  Sets the LCD background color.
00560   * @param  Color: Layer background color co
de ARGB(8-8-8-8)
00561   * @retval None
00562   */
```

```
00563 void BSP_RGB_LCD_SetBackColor(uint32_t Color
)
00564 {
00565   DrawProp[ActiveLayer].BackColor = Color;
00566 }
00567
00568 /**
00569   * @brief  Gets the LCD background color.
00570   * @retval Used background colour
00571   */
00572 uint32_t BSP_RGB_LCD_GetBackColor(void)
00573 {
00574   return DrawProp[ActiveLayer].BackColor;
00575 }
00576
00577 /**
00578   * @brief  Sets the LCD text font.
00579   * @param  fonts: Layer font to be used
00580   * @retval None
00581   */
00582 void BSP_RGB_LCD_SetFont(sFONT *fonts)
00583 {
00584   DrawProp[ActiveLayer].pFont = fonts;
00585 }
00586
00587 /**
00588   * @brief  Gets the LCD text font.
00589   * @retval Used layer font
00590   */
00591 sFONT *BSP_RGB_LCD_GetFont(void)
00592 {
00593   return DrawProp[ActiveLayer].pFont;
00594 }
00595
00596 /**
00597   * @brief  Reads an LCD pixel.
00598   * @param  Xpos: X position
```

```
00599   * @param  Ypos: Y position
00600   * @retval RGB pixel color
00601   */
00602 uint32_t BSP_RGB_LCD_ReadPixel(uint16_t Xpos
, uint16_t Ypos)
00603 {
00604   uint32_t ret = 0;
00605
00606   if(hltdc_eval.LayerCfg[ActiveLayer].PixelF
ormat == LTDC_PIXEL_FORMAT_ARGB8888)
00607   {
00608     /* Read data value from SDRAM memory */
00609     ret = *(__IO uint32_t*) (hltdc_eval.Laye
rCfg[ActiveLayer].FBStartAdress + (2*(Ypos*BSP_RGB
_LCD_GetXSize() + Xpos)));
00610   }
00611   else if(hltdc_eval.LayerCfg[ActiveLayer].P
ixelFormat == LTDC_PIXEL_FORMAT_RGB888)
00612   {
00613     /* Read data value from SDRAM memory */
00614     ret = (*(__IO uint32_t*) (hltdc_eval.Lay
erCfg[ActiveLayer].FBStartAdress + (2*(Ypos*BSP_RG
B_LCD_GetXSize() + Xpos))) & 0x00FFFFFF);
00615   }
00616   else if((hltdc_eval.LayerCfg[ActiveLayer].
PixelFormat == LTDC_PIXEL_FORMAT_RGB565) || \
00617           (hltdc_eval.LayerCfg[ActiveLayer].
PixelFormat == LTDC_PIXEL_FORMAT_ARGB4444) || \
00618           (hltdc_eval.LayerCfg[ActiveLayer].
PixelFormat == LTDC_PIXEL_FORMAT_AL88))
00619   {
00620     /* Read data value from SDRAM memory */
00621     ret = *(__IO uint16_t*) (hltdc_eval.Laye
rCfg[ActiveLayer].FBStartAdress + (2*(Ypos*BSP_RGB
_LCD_GetXSize() + Xpos)));
00622   }
00623   else
```

```
00624   {
00625     /* Read data value from SDRAM memory */
00626     ret = *(__IO uint8_t*) (hltdc_eval.Layer
Cfg[ActiveLayer].FBStartAdress + (2*(Ypos*BSP_RGB_
LCD_GetXSize() + Xpos)));
00627   }
00628
00629   return ret;
00630 }
00631
00632 /**
00633   * @brief  Clears the hole LCD.
00634   * @param  Color: Color of the background
00635   * @retval None
00636   */
00637 void BSP_RGB_LCD_Clear(uint32_t Color)
00638 {
00639   /* Clear the LCD */
00640   LL_FillBuffer(ActiveLayer, (uint32_t *)(hl
tdc_eval.LayerCfg[ActiveLayer].FBStartAdress), BSP
_RGB_LCD_GetXSize(), BSP_RGB_LCD_GetYSize(), 0, Co
lor);
00641 }
00642
00643 /**
00644   * @brief  Clears the selected line.
00645   * @param  Line: Line to be cleared
00646   * @retval None
00647   */
00648 void BSP_RGB_LCD_ClearStringLine(uint32_t Li
ne)
00649 {
00650   uint32_t color_backup = DrawProp[ActiveLay
er].TextColor;
00651   DrawProp[ActiveLayer].TextColor = DrawProp[
ActiveLayer].BackColor;
00652
```

```
00653   /* Draw rectangle with background color */
00654   BSP_RGB_LCD_FillRect(0, (Line * DrawProp[A
ctiveLayer].pFont->Height), BSP_RGB_LCD_GetXSize()
, DrawProp[ActiveLayer].pFont->Height);
00655
00656   DrawProp[ActiveLayer].TextColor = color_ba
ckup;
00657   BSP_RGB_LCD_SetTextColor(DrawProp[ActiveLa
yer].TextColor);
00658 }
00659
00660 /**
00661   * @brief  Displays one character.
00662   * @param  Xpos: Start column address
00663   * @param  Ypos: Line where to display the
character shape.
00664   * @param  Ascii: Character ascii code
00665   *           This parameter must be a numbe
r between Min_Data = 0x20 and Max_Data = 0x7E
00666   * @retval None
00667   */
00668 void BSP_RGB_LCD_DisplayChar(uint16_t Xpos,
uint16_t Ypos, uint8_t Ascii)
00669 {
00670   DrawChar(Xpos, Ypos, &DrawProp[ActiveLayer
].pFont->table[(Ascii-' ') *\
00671     DrawProp[ActiveLayer].pFont->Height * ((
DrawProp[ActiveLayer].pFont->Width + 7) / 8)]);
00672 }
00673
00674 /**
00675   * @brief  Displays characters on the LCD.
00676   * @param  Xpos: X position (in pixel)
00677   * @param  Ypos: Y position (in pixel)
00678   * @param  Text: Pointer to string to displ
ay on LCD
00679   * @param  Mode: Display mode
```

```
00680   *           This parameter can be one of th
e following values:
00681   *             @arg  CENTER_MODE
00682   *             @arg  RIGHT_MODE
00683   *             @arg  LEFT_MODE
00684   * @retval None
00685   */
00686 void BSP_RGB_LCD_DisplayStringAt(uint16_t Xp
os, uint16_t Ypos, uint8_t *Text, Text_AlignModeTy
pdef Mode)
00687 {
00688   uint16_t ref_column = 1, i = 0;
00689   uint32_t size = 0, xsize = 0;
00690   uint8_t  *ptr = Text;
00691
00692   /* Get the text size */
00693   while (*ptr++) size ++ ;
00694
00695   /* Characters number per line */
00696   xsize = (BSP_RGB_LCD_GetXSize()/DrawProp[A
ctiveLayer].pFont->Width);
00697
00698   switch (Mode)
00699   {
00700   case CENTER_MODE:
00701     {
00702       ref_column = Xpos + ((xsize - size)* D
rawProp[ActiveLayer].pFont->Width) / 2;
00703       break;
00704     }
00705   case LEFT_MODE:
00706     {
00707       ref_column = Xpos;
00708       break;
00709     }
00710   case RIGHT_MODE:
00711     {
```

```
00712       ref_column = - Xpos + ((xsize - size)*
DrawProp[ActiveLayer].pFont->Width);
00713       break;
00714     }
00715   default:
00716     {
00717       ref_column = Xpos;
00718       break;
00719     }
00720   }
00721
00722   /* Check that the Start column is located
in the screen */
00723   if ((ref_column < 1) || (ref_column >= 0x8
000))
00724   {
00725     ref_column = 1;
00726   }
00727
00728   /* Send the string character by character
on LCD */
00729   while ((*Text != 0) & (((BSP_RGB_LCD_GetXS
ize() - (i*DrawProp[ActiveLayer].pFont->Width)) &
0xFFFF) >= DrawProp[ActiveLayer].pFont->Width))
00730   {
00731     /* Display one character on LCD */
00732     BSP_RGB_LCD_DisplayChar(ref_column, Ypos
, *Text);
00733     /* Decrement the column position by 16 */

00734     ref_column += DrawProp[ActiveLayer].pFont
->Width;
00735     /* Point on the next character */
00736     Text++;
00737     i++;
00738   }
00739 }
```

```
00740
00741 /**
00742   * @brief  Displays a maximum of 60 charact
ers on the LCD.
00743   * @param  Line: Line where to display the
character shape
00744   * @param  ptr: Pointer to string to displa
y on LCD
00745   * @retval None
00746   */
00747 void BSP_RGB_LCD_DisplayStringAtLine(uint16_
t Line, uint8_t *ptr)
00748 {
00749   BSP_RGB_LCD_DisplayStringAt(0, LINE(Line),
 ptr, LEFT_MODE);
00750 }
00751
00752 /**
00753   * @brief  Draws an horizontal line.
00754   * @param  Xpos: X position
00755   * @param  Ypos: Y position
00756   * @param  Length: Line length
00757   * @retval None
00758   */
00759 void BSP_RGB_LCD_DrawHLine(uint16_t Xpos, ui
nt16_t Ypos, uint16_t Length)
00760 {
00761   uint32_t  Xaddress = 0;
00762
00763   /* Get the line address */
00764   if(hltdc_eval.LayerCfg[ActiveLayer].PixelF
ormat == LTDC_PIXEL_FORMAT_RGB565)
00765   { /* RGB565 format */
00766     Xaddress = (hltdc_eval.LayerCfg[ActiveLa
yer].FBStartAdress) + 2*(BSP_RGB_LCD_GetXSize()*Yp
os + Xpos);
00767   }
```

```
00768   else
00769   { /* ARGB8888 format */
00770     Xaddress = (hltdc_eval.LayerCfg[ActiveLa
yer].FBStartAdress) + 4*(BSP_RGB_LCD_GetXSize()*Yp
os + Xpos);
00771   }
00772
00773   /* Write line */
00774   LL_FillBuffer(ActiveLayer, (uint32_t *)Xad
dress, Length, 1, 0, DrawProp[ActiveLayer].TextCol
or);
00775 }
00776
00777 /**
00778   * @brief  Draws a vertical line.
00779   * @param  Xpos: X position
00780   * @param  Ypos: Y position
00781   * @param  Length: Line length
00782   * @retval None
00783   */
00784 void BSP_RGB_LCD_DrawVLine(uint16_t Xpos, ui
nt16_t Ypos, uint16_t Length)
00785 {
00786   uint32_t  Xaddress = 0;
00787
00788   /* Get the line address */
00789   if(hltdc_eval.LayerCfg[ActiveLayer].PixelF
ormat == LTDC_PIXEL_FORMAT_RGB565)
00790   { /* RGB565 format */
00791     Xaddress = (hltdc_eval.LayerCfg[ActiveLa
yer].FBStartAdress) + 2*(BSP_RGB_LCD_GetXSize()*Yp
os + Xpos);
00792   }
00793   else
00794   { /* ARGB8888 format */
00795     Xaddress = (hltdc_eval.LayerCfg[ActiveLa
yer].FBStartAdress) + 4*(BSP_RGB_LCD_GetXSize()*Yp
```

```
os + Xpos);
00796   }
00797
00798   /* Write line */
00799   LL_FillBuffer(ActiveLayer, (uint32_t *)Xad
dress, 1, Length, (BSP_RGB_LCD_GetXSize() - 1), Dr
awProp[ActiveLayer].TextColor);
00800 }
00801
00802 /**
00803   * @brief  Draws an uni-line (between two p
oints).
00804   * @param  x1: Point 1 X position
00805   * @param  y1: Point 1 Y position
00806   * @param  x2: Point 2 X position
00807   * @param  y2: Point 2 Y position
00808   * @retval None
00809   */
00810 void BSP_RGB_LCD_DrawLine(uint16_t x1, uint1
6_t y1, uint16_t x2, uint16_t y2)
00811 {
00812   int16_t deltax = 0, deltay = 0, x = 0, y =
 0, xinc1 = 0, xinc2 = 0,
00813   yinc1 = 0, yinc2 = 0, den = 0, num = 0, nu
m_add = 0, num_pixels = 0,
00814   curpixel = 0;
00815
00816   deltax = ABS(x2 - x1);        /* The diffe
rence between the x's */
00817   deltay = ABS(y2 - y1);        /* The diffe
rence between the y's */
00818   x = x1;                       /* Start x o
ff at the first pixel */
00819   y = y1;                       /* Start y o
ff at the first pixel */
00820
00821   if (x2 >= x1)                 /* The x-val
```

```
ues are increasing */
00822   {
00823     xinc1 = 1;
00824     xinc2 = 1;
00825   }
00826   else                              /* The x-val
ues are decreasing */
00827   {
00828     xinc1 = -1;
00829     xinc2 = -1;
00830   }
00831
00832   if (y2 >= y1)                     /* The y-val
ues are increasing */
00833   {
00834     yinc1 = 1;
00835     yinc2 = 1;
00836   }
00837   else                              /* The y-val
ues are decreasing */
00838   {
00839     yinc1 = -1;
00840     yinc2 = -1;
00841   }
00842
00843   if (deltax >= deltay)             /* There is
at least one x-value for every y-value */
00844   {
00845     xinc1 = 0;                      /* Don't cha
nge the x when numerator >= denominator */
00846     yinc2 = 0;                      /* Don't cha
nge the y for every iteration */
00847     den = deltax;
00848     num = deltax / 2;
00849     num_add = deltay;
00850     num_pixels = deltax;          /* There ar
e more x-values than y-values */
```

```
00851   }
00852   else                            /* There is
at least one y-value for every x-value */
00853   {
00854     xinc2 = 0;                    /* Don't cha
nge the x for every iteration */
00855     yinc1 = 0;                    /* Don't cha
nge the y when numerator >= denominator */
00856     den = deltay;
00857     num = deltay / 2;
00858     num_add = deltax;
00859     num_pixels = deltay;          /* There ar
e more y-values than x-values */
00860   }
00861
00862   for (curpixel = 0; curpixel <= num_pixels;
 curpixel++)
00863   {
00864     BSP_RGB_LCD_DrawPixel(x, y, DrawProp[Act
iveLayer].TextColor);   /* Draw the current pixel
*/
00865     num += num_add;
   /* Increase the numerator by the top of the fra
ction */
00866     if (num >= den)
  /* Check if numerator >= denominator */
00867     {
00868       num -= den;
  /* Calculate the new numerator value */
00869       x += xinc1;
  /* Change the x as appropriate */
00870       y += yinc1;
  /* Change the y as appropriate */
00871     }
00872     x += xinc2;
  /* Change the x as appropriate */
00873     y += yinc2;
```

```
  /* Change the y as appropriate */
00874   }
00875 }
00876
00877 /**
00878   * @brief  Draws a rectangle.
00879   * @param  Xpos: X position
00880   * @param  Ypos: Y position
00881   * @param  Width: Rectangle width
00882   * @param  Height: Rectangle height
00883   * @retval None
00884   */
00885 void BSP_RGB_LCD_DrawRect(uint16_t Xpos, uin
t16_t Ypos, uint16_t Width, uint16_t Height)
00886 {
00887   /* Draw horizontal lines */
00888   BSP_RGB_LCD_DrawHLine(Xpos, Ypos, Width);
00889   BSP_RGB_LCD_DrawHLine(Xpos, (Ypos+ Height)
, Width);
00890
00891   /* Draw vertical lines */
00892   BSP_RGB_LCD_DrawVLine(Xpos, Ypos, Height);
00893   BSP_RGB_LCD_DrawVLine((Xpos + Width), Ypos
, Height);
00894 }
00895
00896 /**
00897   * @brief  Draws a circle.
00898   * @param  Xpos: X position
00899   * @param  Ypos: Y position
00900   * @param  Radius: Circle radius
00901   * @retval None
00902   */
00903 void BSP_RGB_LCD_DrawCircle(uint16_t Xpos, u
int16_t Ypos, uint16_t Radius)
00904 {
00905   int32_t   decision;    /* Decision Variabl
```

```
e */
00906   uint32_t  current_x;   /* Current X Value
*/
00907   uint32_t  current_y;   /* Current Y Value
*/
00908
00909   decision = 3 - (Radius << 1);
00910   current_x = 0;
00911   current_y = Radius;
00912
00913   while (current_x <= current_y)
00914   {
00915     BSP_RGB_LCD_DrawPixel((Xpos + current_x)
, (Ypos - current_y), DrawProp[ActiveLayer].TextCo
lor);
00916
00917     BSP_RGB_LCD_DrawPixel((Xpos - current_x)
, (Ypos - current_y), DrawProp[ActiveLayer].TextCo
lor);
00918
00919     BSP_RGB_LCD_DrawPixel((Xpos + current_y)
, (Ypos - current_x), DrawProp[ActiveLayer].TextCo
lor);
00920
00921     BSP_RGB_LCD_DrawPixel((Xpos - current_y)
, (Ypos - current_x), DrawProp[ActiveLayer].TextCo
lor);
00922
00923     BSP_RGB_LCD_DrawPixel((Xpos + current_x)
, (Ypos + current_y), DrawProp[ActiveLayer].TextCo
lor);
00924
00925     BSP_RGB_LCD_DrawPixel((Xpos - current_x)
, (Ypos + current_y), DrawProp[ActiveLayer].TextCo
lor);
00926
00927     BSP_RGB_LCD_DrawPixel((Xpos + current_y)
```

```
, (Ypos + current_x), DrawProp[ActiveLayer].TextCo
lor);
00928
00929     BSP_RGB_LCD_DrawPixel((Xpos - current_y)
, (Ypos + current_x), DrawProp[ActiveLayer].TextCo
lor);
00930
00931     if (decision < 0)
00932     {
00933       decision += (current_x << 2) + 6;
00934     }
00935     else
00936     {
00937       decision += ((current_x - current_y) <
< 2) + 10;
00938       current_y--;
00939     }
00940     current_x++;
00941   }
00942 }
00943
00944 /**
00945   * @brief  Draws an poly-line (between many
 points).
00946   * @param  Points: Pointer to the points ar
ray
00947   * @param  PointCount: Number of points
00948   * @retval None
00949   */
00950 void BSP_RGB_LCD_DrawPolygon(pPoint Points,
uint16_t PointCount)
00951 {
00952   int16_t x = 0, y = 0;
00953
00954   if(PointCount < 2)
00955   {
00956     return;
```

```
00957   }
00958
00959   BSP_RGB_LCD_DrawLine(Points->X, Points->Y,
 (Points+PointCount-1)->X, (Points+PointCount-1)->
Y);
00960
00961   while(--PointCount)
00962   {
00963     x = Points->X;
00964     y = Points->Y;
00965     Points++;
00966     BSP_RGB_LCD_DrawLine(x, y, Points->X, Po
ints->Y);
00967   }
00968 }
00969
00970 /**
00971   * @brief  Draws an ellipse on LCD.
00972   * @param  Xpos: X position
00973   * @param  Ypos: Y position
00974   * @param  XRadius: Ellipse X radius
00975   * @param  YRadius: Ellipse Y radius
00976   * @retval None
00977   */
00978 void BSP_RGB_LCD_DrawEllipse(int Xpos, int Y
pos, int XRadius, int YRadius)
00979 {
00980   int x = 0, y = -YRadius, err = 2-2*XRadius
, e2;
00981   float k = 0, rad1 = 0, rad2 = 0;
00982
00983   rad1 = XRadius;
00984   rad2 = YRadius;
00985
00986   k = (float)(rad2/rad1);
00987
00988   do {
```

```
00989     BSP_RGB_LCD_DrawPixel((Xpos-(uint16_t)(x
/k)), (Ypos+y), DrawProp[ActiveLayer].TextColor);
00990     BSP_RGB_LCD_DrawPixel((Xpos+(uint16_t)(x
/k)), (Ypos+y), DrawProp[ActiveLayer].TextColor);
00991     BSP_RGB_LCD_DrawPixel((Xpos+(uint16_t)(x
/k)), (Ypos-y), DrawProp[ActiveLayer].TextColor);
00992     BSP_RGB_LCD_DrawPixel((Xpos-(uint16_t)(x
/k)), (Ypos-y), DrawProp[ActiveLayer].TextColor);
00993
00994     e2 = err;
00995     if (e2 <= x) {
00996       err += ++x*2+1;
00997       if (-y == x && e2 <= y) e2 = 0;
00998     }
00999     if (e2 > y) err += ++y*2+1;
01000   }
01001   while (y <= 0);
01002 }
01003
01004 /**
01005   * @brief  Draws a pixel on LCD.
01006   * @param  Xpos: X position
01007   * @param  Ypos: Y position
01008   * @param  RGB_Code: Pixel color in ARGB mo
de (8-8-8-8)
01009   * @retval None
01010   */
01011 void BSP_RGB_LCD_DrawPixel(uint16_t Xpos, ui
nt16_t Ypos, uint32_t RGB_Code)
01012 {
01013   /* Write data value to all SDRAM memory */
01014   if(hltdc_eval.LayerCfg[ActiveLayer].PixelF
ormat == LTDC_PIXEL_FORMAT_RGB565)
01015   { /* RGB565 format */
01016     *(__IO uint16_t*) (hltdc_eval.LayerCfg[A
ctiveLayer].FBStartAdress + (2*(Ypos*BSP_RGB_LCD_G
etXSize() + Xpos))) = (uint16_t)RGB_Code;
```

```
01017   }
01018   else
01019   { /* ARGB8888 format */
01020     *(__IO uint32_t*) (hltdc_eval.LayerCfg[A
ctiveLayer].FBStartAdress + (4*(Ypos*BSP_RGB_LCD_G
etXSize() + Xpos))) = RGB_Code;
01021   }
01022 }
01023
01024 /**
01025   * @brief  Draws a bitmap picture loaded in
 the internal Flash in ARGB888 format (32 bits per
 pixel).
01026   * @param  Xpos: Bmp X position in the LCD
01027   * @param  Ypos: Bmp Y position in the LCD
01028   * @param  pbmp: Pointer to Bmp picture add
ress in the internal Flash
01029   * @retval None
01030   */
01031 void BSP_RGB_LCD_DrawBitmap(uint32_t Xpos, u
int32_t Ypos, uint8_t *pbmp)
01032 {
01033   uint32_t index = 0, width = 0, height = 0,
 bit_pixel = 0;
01034   uint32_t address;
01035   uint32_t input_color_mode = 0;
01036
01037   /* Get bitmap data address offset */
01038   index = *(__IO uint16_t *) (pbmp + 10);
01039   index |= (*(__IO uint16_t *) (pbmp + 12))
<< 16;
01040
01041   /* Read bitmap width */
01042   width = *(uint16_t *) (pbmp + 18);
01043   width |= (*(uint16_t *) (pbmp + 20)) << 16
;
01044
```

```
01045   /* Read bitmap height */
01046   height = *(uint16_t *) (pbmp + 22);
01047   height |= (*(uint16_t *) (pbmp + 24)) << 1
6;
01048
01049   /* Read bit/pixel */
01050   bit_pixel = *(uint16_t *) (pbmp + 28);
01051
01052   /* Set the address */
01053   address = hltdc_eval.LayerCfg[ActiveLayer]
.FBStartAdress + (((BSP_RGB_LCD_GetXSize()*Ypos) +
 Xpos)*(4));
01054
01055   /* Get the layer pixel format */
01056   if ((bit_pixel/8) == 4)
01057   {
01058     input_color_mode = DMA2D_INPUT_ARGB8888;
01059   }
01060   else if ((bit_pixel/8) == 2)
01061   {
01062     input_color_mode = DMA2D_INPUT_RGB565;
01063   }
01064   else
01065   {
01066     input_color_mode = DMA2D_INPUT_RGB888;
01067   }
01068
01069   /* Bypass the bitmap header */
01070   pbmp += (index + (width * (height - 1) * (
bit_pixel/8)));
01071
01072   /* Convert picture to ARGB8888 pixel forma
t */
01073   for(index=0; index < height; index++)
01074   {
01075     /* Pixel format conversion */
01076     LL_ConvertLineToARGB8888((uint32_t *)pbm
```

```
p, (uint32_t *)address, width, input_color_mode);
01077
01078     /* Increment the source and destination
buffers */
01079     address+=  (BSP_RGB_LCD_GetXSize()*4);
01080     pbmp -= width*(bit_pixel/8);
01081   }
01082 }
01083
01084 /**
01085   * @brief  Draws a full rectangle.
01086   * @param  Xpos: X position
01087   * @param  Ypos: Y position
01088   * @param  Width: Rectangle width
01089   * @param  Height: Rectangle height
01090   * @retval None
01091   */
01092 void BSP_RGB_LCD_FillRect(uint16_t Xpos, uin
t16_t Ypos, uint16_t Width, uint16_t Height)
01093 {
01094   uint32_t  x_address = 0;
01095
01096   /* Set the text color */
01097   BSP_RGB_LCD_SetTextColor(DrawProp[ActiveLa
yer].TextColor);
01098
01099   /* Get the rectangle start address */
01100   if(hltdc_eval.LayerCfg[ActiveLayer].PixelF
ormat == LTDC_PIXEL_FORMAT_RGB565)
01101   { /* RGB565 format */
01102     x_address = (hltdc_eval.LayerCfg[ActiveL
ayer].FBStartAdress) + 2*(BSP_RGB_LCD_GetXSize()*Y
pos + Xpos);
01103   }
01104   else
01105   { /* ARGB8888 format */
01106     x_address = (hltdc_eval.LayerCfg[ActiveL
```

```
ayer].FBStartAdress) + 4*(BSP_RGB_LCD_GetXSize()*Y
pos + Xpos);
01107   }
01108   /* Fill the rectangle */
01109   LL_FillBuffer(ActiveLayer, (uint32_t *)x_a
ddress, Width, Height, (BSP_RGB_LCD_GetXSize() - W
idth), DrawProp[ActiveLayer].TextColor);
01110 }
01111
01112 /**
01113   * @brief  Draws a full circle.
01114   * @param  Xpos: X position
01115   * @param  Ypos: Y position
01116   * @param  Radius: Circle radius
01117   * @retval None
01118   */
01119 void BSP_RGB_LCD_FillCircle(uint16_t Xpos, u
int16_t Ypos, uint16_t Radius)
01120 {
01121   int32_t  decision;     /* Decision Variabl
e */
01122   uint32_t  current_x;   /* Current X Value
*/
01123   uint32_t  current_y;   /* Current Y Value
*/
01124
01125   decision = 3 - (Radius << 1);
01126
01127   current_x = 0;
01128   current_y = Radius;
01129
01130   BSP_RGB_LCD_SetTextColor(DrawProp[ActiveLa
yer].TextColor);
01131
01132   while (current_x <= current_y)
01133   {
01134     if(current_y > 0)
```

```
01135     {
01136       BSP_RGB_LCD_DrawHLine(Xpos - current_y
, Ypos + current_x, 2*current_y);
01137       BSP_RGB_LCD_DrawHLine(Xpos - current_y
, Ypos - current_x, 2*current_y);
01138     }
01139
01140     if(current_x > 0)
01141     {
01142       BSP_RGB_LCD_DrawHLine(Xpos - current_x
, Ypos - current_y, 2*current_x);
01143       BSP_RGB_LCD_DrawHLine(Xpos - current_x
, Ypos + current_y, 2*current_x);
01144     }
01145     if (decision < 0)
01146     {
01147       decision += (current_x << 2) + 6;
01148     }
01149     else
01150     {
01151       decision += ((current_x - current_y) <
< 2) + 10;
01152       current_y--;
01153     }
01154     current_x++;
01155   }
01156
01157   BSP_RGB_LCD_SetTextColor(DrawProp[ActiveLa
yer].TextColor);
01158   BSP_RGB_LCD_DrawCircle(Xpos, Ypos, Radius)
;
01159 }
01160
01161 /**
01162   * @brief  Draws a full poly-line (between
many points).
01163   * @param  Points: Pointer to the points ar
```

```
ray
01164   * @param  PointCount: Number of points
01165   * @retval None
01166   */
01167 void BSP_RGB_LCD_FillPolygon(pPoint Points,
uint16_t PointCount)
01168 {
01169   int16_t X = 0, Y = 0, X2 = 0, Y2 = 0, X_ce
nter = 0, Y_center = 0, X_first = 0, Y_first = 0,
pixelX = 0, pixelY = 0, counter = 0;
01170   uint16_t  image_left = 0, image_right = 0,
 image_top = 0, image_bottom = 0;
01171
01172   image_left = image_right = Points->X;
01173   image_top= image_bottom = Points->Y;
01174
01175   for(counter = 1; counter < PointCount; cou
nter++)
01176   {
01177     pixelX = POLY_X(counter);
01178     if(pixelX < image_left)
01179     {
01180       image_left = pixelX;
01181     }
01182     if(pixelX > image_right)
01183     {
01184       image_right = pixelX;
01185     }
01186
01187     pixelY = POLY_Y(counter);
01188     if(pixelY < image_top)
01189     {
01190       image_top = pixelY;
01191     }
01192     if(pixelY > image_bottom)
01193     {
01194       image_bottom = pixelY;
```

```
01195     }
01196   }
01197
01198   if(PointCount < 2)
01199   {
01200     return;
01201   }
01202
01203   X_center = (image_left + image_right)/2;
01204   Y_center = (image_bottom + image_top)/2;
01205
01206   X_first = Points->X;
01207   Y_first = Points->Y;
01208
01209   while(--PointCount)
01210   {
01211     X = Points->X;
01212     Y = Points->Y;
01213     Points++;
01214     X2 = Points->X;
01215     Y2 = Points->Y;
01216
01217     FillTriangle(X, X2, X_center, Y, Y2, Y_c
enter);
01218     FillTriangle(X, X_center, X2, Y, Y_cente
r, Y2);
01219     FillTriangle(X_center, X2, X, Y_center,
Y2, Y);
01220   }
01221
01222   FillTriangle(X_first, X2, X_center, Y_firs
t, Y2, Y_center);
01223   FillTriangle(X_first, X_center, X2, Y_firs
t, Y_center, Y2);
01224   FillTriangle(X_center, X2, X_first, Y_cent
er, Y2, Y_first);
01225 }
```

```
01226
01227 /**
01228   * @brief  Draws a full ellipse.
01229   * @param  Xpos: X position
01230   * @param  Ypos: Y position
01231   * @param  XRadius: Ellipse X radius
01232   * @param  YRadius: Ellipse Y radius
01233   * @retval None
01234   */
01235 void BSP_RGB_LCD_FillEllipse(int Xpos, int Y
pos, int XRadius, int YRadius)
01236 {
01237   int x = 0, y = -YRadius, err = 2-2*XRadius
, e2;
01238   float k = 0, rad1 = 0, rad2 = 0;
01239
01240   rad1 = XRadius;
01241   rad2 = YRadius;
01242
01243   k = (float)(rad2/rad1);
01244
01245   do
01246   {
01247     BSP_RGB_LCD_DrawHLine((Xpos-(uint16_t)(x
/k)), (Ypos+y), (2*(uint16_t)(x/k) + 1));
01248     BSP_RGB_LCD_DrawHLine((Xpos-(uint16_t)(x
/k)), (Ypos-y), (2*(uint16_t)(x/k) + 1));
01249
01250     e2 = err;
01251     if (e2 <= x)
01252     {
01253       err += ++x*2+1;
01254       if (-y == x && e2 <= y) e2 = 0;
01255     }
01256     if (e2 > y) err += ++y*2+1;
01257   }
01258   while (y <= 0);
```

```
01259 }
01260
01261 /**
01262   * @brief  Enables the display.
01263   * @retval None
01264   */
01265 void BSP_RGB_LCD_DisplayOn(void)
01266 {
01267   /* Display On */
01268   __HAL_LTDC_ENABLE(&hltdc_eval);
01269   BSP_IO_WritePin(LCD_RGB_DISP_PIN, GPIO_PIN
_SET);                                    /* Assert
LCD_RGB_DISP pin */
01270   HAL_GPIO_WritePin(LCD_RGB_BL_CTRL_GPIO_PORT
, LCD_RGB_BL_CTRL_PIN, GPIO_PIN_SET);  /* Assert L
CD_BL_CTRL pin */
01271 }
01272
01273 /**
01274   * @brief  Disables the display.
01275   * @retval None
01276   */
01277 void BSP_RGB_LCD_DisplayOff(void)
01278 {
01279   /* Display Off */
01280   __HAL_LTDC_DISABLE(&hltdc_eval);
01281   BSP_IO_WritePin(LCD_RGB_DISP_PIN, GPIO_PIN
_RESET);                                  /* De-asse
rt LCD_DISP pin */
01282   HAL_GPIO_WritePin(LCD_RGB_BL_CTRL_GPIO_PORT
, LCD_RGB_BL_CTRL_PIN, GPIO_PIN_RESET);/* De-asser
t LCD_BL_CTRL pin */
01283 }
01284
01285 /**
01286   * @brief  Initializes the LTDC MSP.
01287   * @param  hltdc: LTDC handle
```

```
01288   * @param  Params
01289   * @retval None
01290   */
01291 __weak void BSP_RGB_LCD_MspInit(LTDC_HandleT
ypeDef *hltdc, void *Params)
01292 {
01293   GPIO_InitTypeDef gpio_init_structure;
01294
01295   /* IO expander init */
01296   BSP_IO_Init();
01297
01298   /* Enable the LTDC and DMA2D clocks */
01299   __HAL_RCC_LTDC_CLK_ENABLE();
01300   __HAL_RCC_DMA2D_CLK_ENABLE();
01301
01302   /* Enable GPIOs clock */
01303   __HAL_RCC_GPIOD_CLK_ENABLE();
01304   __HAL_RCC_GPIOE_CLK_ENABLE();
01305   __HAL_RCC_GPIOF_CLK_ENABLE();
01306   LCD_RGB_BL_CTRL_GPIO_CLK_ENABLE();
01307
01308   /*** LTDC Pins configuration ***/
01309   /* GPIOD configuration */
01310   gpio_init_structure.Pin       = GPIO_PIN_0
  | GPIO_PIN_1  | GPIO_PIN_3  | GPIO_PIN_8  | GPIO
_PIN_9  | \
01311                                   GPIO_PIN_1
0 | GPIO_PIN_11 | GPIO_PIN_12 | GPIO_PIN_14 | GPIO
_PIN_15;
01312   gpio_init_structure.Mode      = GPIO_MODE_
AF_PP;
01313   gpio_init_structure.Pull      = GPIO_NOPUL
L;
01314   gpio_init_structure.Speed     = GPIO_SPEED
_FAST;
01315   gpio_init_structure.Alternate = GPIO_AF11_
LTDC;
```

```
01316   HAL_GPIO_Init(GPIOD, &gpio_init_structure)
;
01317
01318   /* GPIOE configuration */
01319   gpio_init_structure.Pin       = GPIO_PIN_0
  | GPIO_PIN_1  | GPIO_PIN_2  | GPIO_PIN_3  | GPIO
_PIN_4  | \
01320                                   GPIO_PIN_7
  | GPIO_PIN_8  | GPIO_PIN_9  | GPIO_PIN_10 | GPIO
_PIN_11 | \
01321                                   GPIO_PIN_1
2 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15;
01322   gpio_init_structure.Mode      = GPIO_MODE_
AF_PP;
01323   gpio_init_structure.Pull      = GPIO_NOPUL
L;
01324   gpio_init_structure.Speed     = GPIO_SPEED
_FAST;
01325   gpio_init_structure.Alternate = GPIO_AF11_
LTDC;
01326   HAL_GPIO_Init(GPIOE, &gpio_init_structure)
;
01327
01328   /* GPIOF configuration */
01329   gpio_init_structure.Pin       = GPIO_PIN_1
3 | GPIO_PIN_14 | GPIO_PIN_15;
01330   gpio_init_structure.Mode      = GPIO_MODE_
AF_PP;
01331   gpio_init_structure.Alternate = GPIO_AF11_
LTDC;
01332   HAL_GPIO_Init(GPIOF, &gpio_init_structure)
;
01333
01334   gpio_init_structure.Pin       = GPIO_PIN_1
1;
01335   gpio_init_structure.Alternate = GPIO_AF9_L
TDC;
```

```
01336   HAL_GPIO_Init(GPIOF, &gpio_init_structure)
;
01337
01338   /* LCD_DISP & LCD_RST pins configuration (
over MFX IO Expander) */
01339   BSP_IO_ConfigPin(LCD_RGB_DISP_PIN | LCD_RG
B_RST_PIN, IO_MODE_OUTPUT);
01340
01341   /* LCD_BL_CTRL GPIO configuration */
01342   gpio_init_structure.Pin       = LCD_RGB_BL
_CTRL_PIN;  /* LCD_BL_CTRL pin has to be manually
controlled */
01343   gpio_init_structure.Mode      = GPIO_MODE_
OUTPUT_PP;
01344   HAL_GPIO_Init(LCD_RGB_BL_CTRL_GPIO_PORT, &
gpio_init_structure);
01345 }
01346
01347 /**
01348   * @brief  DeInitializes BSP_RGB_LCD_ MSP.
01349   * @param  hltdc: LTDC handle
01350   * @param  Params
01351   * @retval None
01352   */
01353 __weak void BSP_RGB_LCD_MspDeInit(LTDC_Handl
eTypeDef *hltdc, void *Params)
01354 {
01355   GPIO_InitTypeDef  gpio_init_structure;
01356
01357   /* Disable LTDC block */
01358   __HAL_LTDC_DISABLE(hltdc);
01359
01360   /* LTDC Pins deactivation */
01361
01362   /* GPIOD deactivation */
01363   gpio_init_structure.Pin       = GPIO_PIN_0
  | GPIO_PIN_1  | GPIO_PIN_3  | GPIO_PIN_8  | GPIO
```

```
_PIN_9  | \
01364                                           GPIO_PIN_1
0 | GPIO_PIN_11 | GPIO_PIN_12 | GPIO_PIN_14 | GPIO
_PIN_15;
01365   HAL_GPIO_DeInit(GPIOD, gpio_init_structure
.Pin);
01366
01367   /* GPIOE deactivation */
01368   gpio_init_structure.Pin       = GPIO_PIN_0
  | GPIO_PIN_1  | GPIO_PIN_2  | GPIO_PIN_3  | GPIO
_PIN_4  | \
01369                                           GPIO_PIN_7
  | GPIO_PIN_8  | GPIO_PIN_9  | GPIO_PIN_10 | GPIO
_PIN_11 | \
01370                                           GPIO_PIN_1
2 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15;
01371   HAL_GPIO_DeInit(GPIOE, gpio_init_structure
.Pin);
01372
01373   /* GPIOF configuration */
01374   gpio_init_structure.Pin       = GPIO_PIN_1
1 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15;
01375   HAL_GPIO_DeInit(GPIOF, gpio_init_structure
.Pin);
01376
01377   /* Disable LTDC clock */
01378   __HAL_RCC_LTDC_CLK_DISABLE();
01379
01380   /* GPIO pins clock can be shut down in the
 application
01381      by surcharging this __weak function */
01382 }
01383
01384 /**
01385   * @brief  Clock Config.
01386   * @param  hltdc: LTDC handle
01387   * @param  Params
```

```
01388   * @note   This API is called by BSP_RGB_LCD_Init()
01389   *         Being __weak it can be overwritten by the application
01390   * @retval None
01391   */
01392 __weak void BSP_RGB_LCD_ClockConfig(LTDC_HandleTypeDef *hltdc, void *Params)
01393 {
01394   RCC_PeriphCLKInitTypeDef  PeriphClkInit;
01395
01396   /* RK043FN48H LCD clock configuration */
01397   /* PLLSAI2_VCO Input = MSI_VALUE/PLLSAI2_M = 4 Mhz */
01398   /* PLLSAI2_VCO Output = PLLSAI2_VCO Input * PLLSAI2N = 192 Mhz */
01399   /* PLLLCDCLK = PLLSAI2_VCO Output/PLLSAIR = 160/4 = 40 Mhz */
01400   /* LTDC clock frequency = PLLLCDCLK / LTDC_PLLSAI_DIVR_4 = 40/4 = 10Mhz */
01401   PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_LTDC;
01402   PeriphClkInit.PLLSAI2.PLLSAI2Source = RCC_PLLSOURCE_MSI;
01403   PeriphClkInit.PLLSAI2.PLLSAI2M = 1;
01404   PeriphClkInit.PLLSAI2.PLLSAI2N = 40;
01405   PeriphClkInit.PLLSAI2.PLLSAI2R = 4;
01406   PeriphClkInit.PLLSAI2.PLLSAI2ClockOut = RCC_PLLSAI2_LTDCCLK;
01407   PeriphClkInit.LtdcClockSelection = RCC_LTDCCLKSOURCE_PLLSAI2_DIV4;
01408   if(HAL_RCCEx_PeriphCLKConfig(&PeriphClkInit) != HAL_OK)
01409   {
01410     while(1);
01411   }
01412 }
```

```
/*****************************************************************************
                        LTDC and DMA2D BSP Routines
*****************************************************************************/
/**
  * @brief  Handles LTDC interrupt request.
  * @note : Can be surcharged by application code implementation of the function.
  */
__weak void BSP_RGB_LCD_LTDC_IRQHandler(void)
{
  HAL_LTDC_IRQHandler(&(hltdc_eval));
}

/**
  * @brief  This function handles LTDC Error interrupt Handler.
  * @note : Can be surcharged by application code implementation of the function.
  */

__weak void BSP_RGB_LCD_LTDC_ER_IRQHandler(void)
{
  HAL_LTDC_IRQHandler(&(hltdc_eval));
}

/**
  * @brief  Handles DMA2D interrupt request.
  * @note : Can be surcharged by application code implementation of the function.
  */
__weak void BSP_RGB_LCD_DMA2D_IRQHandler(void
```

```
)
{
  HAL_DMA2D_IRQHandler(&hdma2d_eval);
}

/**
  * @}
  */

/** @defgroup STM32L4R9I_EVAL_RGB_LCD_Private_Functions Private Functions
  * @{
  */

/*****************************************************************************
                                Static Functions
*****************************************************************************/

/**
  * @brief  Draws a character on LCD.
  * @param  Xpos: Line where to display the character shape
  * @param  Ypos: Start column address
  * @param  c: Pointer to the character data
  * @retval None
  */
static void DrawChar(uint16_t Xpos, uint16_t Ypos, const uint8_t *c)
{
  uint32_t i = 0, j = 0;
  uint16_t height, width;
  uint8_t  offset;
  uint8_t  *pchar;
  uint32_t line;

```

```
01472   height = DrawProp[ActiveLayer].pFont->Heig
ht;
01473   width  = DrawProp[ActiveLayer].pFont->Widt
h;
01474
01475   offset =  8 *((width + 7)/8) -  width ;
01476
01477   for(i = 0; i < height; i++)
01478   {
01479     pchar = ((uint8_t *)c + (width + 7)/8 *
i);
01480
01481     switch(((width + 7)/8))
01482     {
01483
01484     case 1:
01485       line =  pchar[0];
01486       break;
01487
01488     case 2:
01489       line =  (pchar[0]<< 8) | pchar[1];
01490       break;
01491
01492     case 3:
01493     default:
01494       line =  (pchar[0]<< 16) | (pchar[1]<<
8) | pchar[2];
01495       break;
01496     }
01497
01498     for (j = 0; j < width; j++)
01499     {
01500       if(line & (1 << (width- j + offset- 1)
))
01501       {
01502         BSP_RGB_LCD_DrawPixel((Xpos + j), Yp
os, DrawProp[ActiveLayer].TextColor);
```

```
01503       }
01504       else
01505       {
01506         BSP_RGB_LCD_DrawPixel((Xpos + j), Yp
os, DrawProp[ActiveLayer].BackColor);
01507       }
01508     }
01509     Ypos++;
01510   }
01511 }
01512
01513 /**
01514   * @brief  Fills a triangle (between 3 poin
ts).
01515   * @param  x1: Point 1 X position
01516   * @param  y1: Point 1 Y position
01517   * @param  x2: Point 2 X position
01518   * @param  y2: Point 2 Y position
01519   * @param  x3: Point 3 X position
01520   * @param  y3: Point 3 Y position
01521   * @retval None
01522   */
01523 static void FillTriangle(uint16_t x1, uint16
_t x2, uint16_t x3, uint16_t y1, uint16_t y2, uint
16_t y3)
01524 {
01525   int16_t deltax = 0, deltay = 0, x = 0, y =
 0, xinc1 = 0, xinc2 = 0,
01526   yinc1 = 0, yinc2 = 0, den = 0, num = 0, nu
m_add = 0, num_pixels = 0,
01527   curpixel = 0;
01528
01529   deltax = ABS(x2 - x1);        /* The diffe
rence between the x's */
01530   deltay = ABS(y2 - y1);        /* The diffe
rence between the y's */
01531   x = x1;                       /* Start x o
```

```
ff at the first pixel */
01532   y = y1;                         /* Start y o
ff at the first pixel */
01533
01534   if (x2 >= x1)                   /* The x-val
ues are increasing */
01535   {
01536     xinc1 = 1;
01537     xinc2 = 1;
01538   }
01539   else                            /* The x-val
ues are decreasing */
01540   {
01541     xinc1 = -1;
01542     xinc2 = -1;
01543   }
01544
01545   if (y2 >= y1)                   /* The y-val
ues are increasing */
01546   {
01547     yinc1 = 1;
01548     yinc2 = 1;
01549   }
01550   else                            /* The y-val
ues are decreasing */
01551   {
01552     yinc1 = -1;
01553     yinc2 = -1;
01554   }
01555
01556   if (deltax >= deltay)           /* There is
at least one x-value for every y-value */
01557   {
01558     xinc1 = 0;                    /* Don't cha
nge the x when numerator >= denominator */
01559     yinc2 = 0;                    /* Don't cha
nge the y for every iteration */
```

```
01560     den = deltax;
01561     num = deltax / 2;
01562     num_add = deltay;
01563     num_pixels = deltax;          /* There ar
e more x-values than y-values */
01564   }
01565   else                           /* There is
at least one y-value for every x-value */
01566   {
01567     xinc2 = 0;                   /* Don't cha
nge the x for every iteration */
01568     yinc1 = 0;                   /* Don't cha
nge the y when numerator >= denominator */
01569     den = deltay;
01570     num = deltay / 2;
01571     num_add = deltax;
01572     num_pixels = deltay;          /* There ar
e more y-values than x-values */
01573   }
01574
01575   for (curpixel = 0; curpixel <= num_pixels;
 curpixel++)
01576   {
01577     BSP_RGB_LCD_DrawLine(x, y, x3, y3);
01578
01579     num += num_add;               /* Increase
 the numerator by the top of the fraction */
01580     if (num >= den)              /* Check if
numerator >= denominator */
01581     {
01582       num -= den;                /* Calculate
 the new numerator value */
01583       x += xinc1;                /* Change th
e x as appropriate */
01584       y += yinc1;                /* Change th
e y as appropriate */
01585     }
```

```
01586     x += xinc2;                    /* Change th
e x as appropriate */
01587     y += yinc2;                    /* Change th
e y as appropriate */
01588   }
01589 }
01590
01591 /**
01592   * @brief  Fills a buffer.
01593   * @param  LayerIndex: Layer index
01594   * @param  pDst: Pointer to destination buf
fer
01595   * @param  xSize: Buffer width
01596   * @param  ySize: Buffer height
01597   * @param  OffLine: Offset
01598   * @param  ColorIndex: Color index
01599   * @retval None
01600   */
01601 static void LL_FillBuffer(uint32_t LayerInde
x, void *pDst, uint32_t xSize, uint32_t ySize, uin
t32_t OffLine, uint32_t ColorIndex)
01602 {
01603   /* Register to memory mode with ARGB8888 a
s color Mode */
01604   hdma2d_eval.Init.Mode         = DMA2D_R2M;
01605   if(hltdc_eval.LayerCfg[ActiveLayer].PixelF
ormat == LTDC_PIXEL_FORMAT_RGB565)
01606   { /* RGB565 format */
01607     hdma2d_eval.Init.ColorMode    = DMA2D_OU
TPUT_RGB565;
01608   }
01609   else
01610   { /* ARGB8888 format */
01611     hdma2d_eval.Init.ColorMode    = DMA2D_OU
TPUT_ARGB8888;
01612   }
01613   hdma2d_eval.Init.OutputOffset = OffLine;
```

```
01614
01615   hdma2d_eval.Instance = DMA2D;
01616
01617   /* DMA2D Initialization */
01618   if(HAL_DMA2D_Init(&hdma2d_eval) == HAL_OK)
01619   {
01620     if(HAL_DMA2D_ConfigLayer(&hdma2d_eval, L
ayerIndex) == HAL_OK)
01621     {
01622       if (HAL_DMA2D_Start(&hdma2d_eval, Colo
rIndex, (uint32_t)pDst, xSize, ySize) == HAL_OK)
01623       {
01624         /* Polling For DMA transfer */
01625         HAL_DMA2D_PollForTransfer(&hdma2d_ev
al, 10);
01626       }
01627     }
01628   }
01629 }
01630
01631 /**
01632   * @brief  Converts a line to an ARGB8888 p
ixel format.
01633   * @param  pSrc: Pointer to source buffer
01634   * @param  pDst: Output color
01635   * @param  xSize: Buffer width
01636   * @param  ColorMode: Input color mode
01637   * @retval None
01638   */
01639 static void LL_ConvertLineToARGB8888(void *p
Src, void *pDst, uint32_t xSize, uint32_t ColorMod
e)
01640 {
01641   /* Configure the DMA2D Mode, Color Mode an
d output offset */
01642   hdma2d_eval.Init.Mode         = DMA2D_M2M_
PFC;
```

```
01643   hdma2d_eval.Init.ColorMode    = DMA2D_OUTP
UT_ARGB8888;
01644   hdma2d_eval.Init.OutputOffset = 0;
01645
01646   /* Foreground Configuration */
01647   hdma2d_eval.LayerCfg[1].AlphaMode = DMA2D_
NO_MODIF_ALPHA;
01648   hdma2d_eval.LayerCfg[1].InputAlpha = 0xFF;
01649   hdma2d_eval.LayerCfg[1].InputColorMode = C
olorMode;
01650   hdma2d_eval.LayerCfg[1].InputOffset = 0;
01651
01652   hdma2d_eval.Instance = DMA2D;
01653
01654   /* DMA2D Initialization */
01655   if(HAL_DMA2D_Init(&hdma2d_eval) == HAL_OK)
01656   {
01657     if(HAL_DMA2D_ConfigLayer(&hdma2d_eval, 1
) == HAL_OK)
01658     {
01659       if (HAL_DMA2D_Start(&hdma2d_eval, (uin
t32_t)pSrc, (uint32_t)pDst, xSize, 1) == HAL_OK)
01660       {
01661         /* Polling For DMA transfer */
01662         HAL_DMA2D_PollForTransfer(&hdma2d_ev
al, 10);
01663       }
01664     }
01665   }
01666 }
01667
01668 /**
01669   * @}
01670   */
01671
01672 #endif /* USE_ROCKTECH_480x272 */
01673
```

```
01674 /**
01675   * @}
01676   */
01677
01678 /**
01679   * @}
01680   */
01681
01682 /**
01683   * @}
01684   */
01685
01686 /**
01687   * @}
01688   */
01689
01690 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_rgb_ts.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_rgb_ts.c
00004   * @author  MCD Application Team
00005   * @brief   This file provides a set of functions needed to manage the Touch
00006   *          Screen on STM32746G-Discovery board.
00007   @verbatim
00008    1. How To use this driver:
00009    --------------------------
00010       - This driver is used to drive the touch screen module of the STM32L4R9I_EVAL
00011         board on the RK043FN48H-CT672B 480x272 LCD screen with capacitive touch screen.
00012       - The FT5336 component driver must be included in project files according to
00013         the touch screen driver present on this board.
00014
00015    2. Driver description:
```

```
00016    ----------------------
00017      + Initialization steps:
00018         o Initialize the TS module using the
 BSP_RGB_TS_Init() function. This
00019           function includes the MSP layer ha
rdware resources initialization and the
00020           communication layer configuration
to start the TS use. The LCD size properties
00021           (x and y) are passed as parameters.

00022         o If TS interrupt mode is desired, y
ou must configure the TS interrupt mode
00023           by calling the function BSP_RGB_TS
_ITConfig(). The TS interrupt mode is generated
00024           as an external interrupt whenever
a touch is detected.
00025           The interrupt mode internally uses
 the IO functionalities driver driven by
00026           the IO expander, to configure the
IT line.
00027
00028      + Touch screen use
00029         o The touch screen state is captured
 whenever the function BSP_RGB_TS_GetState() is
00030           used. This function returns inform
ation about the last LCD touch occurred
00031           in the TS_StateTypeDef structure.
00032         o If TS interrupt mode is used, the
function BSP_RGB_TS_ITGetStatus() is needed to get
00033           the interrupt status. To clear the
 IT pending bits, you should call the
00034           function BSP_RGB_TS_ITClear().
00035         o The IT is handled using the corres
ponding external interrupt IRQ handler,
00036           the user IT callback treatment is
implemented on the same external interrupt
00037           callback.
```

```
00038   @endverbatim
00039   ******************************************************************************
00040   * @attention
00041   *
00042   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00043   *
00044   * Redistribution and use in source and binary forms, with or without modification,
00045   * are permitted provided that the following conditions are met:
00046   *   1. Redistributions of source code must retain the above copyright notice,
00047   *      this list of conditions and the following disclaimer.
00048   *   2. Redistributions in binary form must reproduce the above copyright notice,
00049   *      this list of conditions and the following disclaimer in the documentation
00050   *      and/or other materials provided with the distribution.
00051   *   3. Neither the name of STMicroelectronics nor the names of its contributors
00052   *      may be used to endorse or promote products derived from this software
00053   *      without specific prior written permission.
00054   *
00055   * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
00056   * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
00057   * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
00058   * DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
```

```
00059   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00060   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00061   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00062   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00063   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00064   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00065   *
00066   ****************************************
**********************************
00067   */
00068
00069 /* Includes ------------------------------
-----------------------------------*/
00070 #include "stm32l4r9i_eval_ts.h"
00071
00072 /** @addtogroup BSP
00073   * @{
00074   */
00075
00076 /** @addtogroup STM32L4R9I_EVAL
00077   * @{
00078   */
00079
00080 #if defined(USE_ROCKTECH_480x272)
00081
00082 /** @defgroup STM32L4R9I_EVAL_RGB_TS STM32L4
R9I_EVAL RGB TS
00083   * @{
00084   */
00085
00086 /** @defgroup STM32L4R9I_EVAL_RGB_TS_Private
```

```
_Variables Private Variables
00087   * @{
00088   */
00089 static TS_DrvTypeDef *tsDriver;
00090 static uint16_t tsXBoundary, tsYBoundary;
00091 static uint8_t  tsOrientation;
00092 static uint8_t  I2cAddress;
00093 /**
00094   * @}
00095   */
00096
00097 /** @defgroup STM32L4R9I_EVAL_RGB_TS_Exporte
d_Functions Exported Functions
00098   * @{
00099   */
00100
00101 /**
00102   * @brief  Initializes and configures the t
ouch screen functionalities and
00103   *         configures all necessary hardwar
e resources (GPIOs, I2C, clocks..).
00104   * @param  ts_SizeX: Maximum X size of the
TS area on LCD
00105   * @param  ts_SizeY: Maximum Y size of the
TS area on LCD
00106   * @retval TS_OK if all initializations are
 OK. Other value if error.
00107   */
00108 uint8_t BSP_RGB_TS_Init(uint16_t ts_SizeX, u
int16_t ts_SizeY)
00109 {
00110   uint8_t status = TS_OK;
00111   tsXBoundary = ts_SizeX;
00112   tsYBoundary = ts_SizeY;
00113
00114   /* Read ID and verify if the touch screen
driver is ready */
```

```
00115   ft5336_ts_drv.Init(TS_RGB_I2C_ADDRESS);
00116   if(ft5336_ts_drv.ReadID(TS_RGB_I2C_ADDRESS
) == FT5336_ID_VALUE)
00117   {
00118     /* Initialize the TS driver structure */
00119     tsDriver = &ft5336_ts_drv;
00120     I2cAddress = TS_RGB_I2C_ADDRESS;
00121     tsOrientation = TS_SWAP_XY;
00122
00123     /* Initialize the TS driver */
00124     tsDriver->Start(I2cAddress);
00125   }
00126   else
00127   {
00128     status = TS_DEVICE_NOT_FOUND;
00129   }
00130
00131   return status;
00132 }
00133
00134 /**
00135   * @brief  DeInitializes the TouchScreen.
00136   * @retval TS state
00137   */
00138 uint8_t BSP_RGB_TS_DeInit(void)
00139 {
00140   /* Actually ts_driver does not provide a D
eInit function */
00141   return TS_OK;
00142 }
00143
00144 /**
00145   * @brief  Configures and enables the touch
 screen interrupts.
00146   * @retval TS_OK if all initializations are
 OK. Other value if error.
00147   */
```

```
00148 uint8_t BSP_RGB_TS_ITConfig(void)
00149 {
00150   GPIO_InitTypeDef gpio_init_structure;
00151
00152   /* Configure Interrupt mode for TS detecti
on pin */
00153   gpio_init_structure.Pin = TS_RGB_INT_PIN;
00154   gpio_init_structure.Pull = GPIO_NOPULL;
00155   gpio_init_structure.Speed = GPIO_SPEED_FAS
T;
00156   gpio_init_structure.Mode = GPIO_MODE_IT_RI
SING;
00157   HAL_GPIO_Init(TS_RGB_INT_GPIO_PORT, &gpio_
init_structure);
00158
00159   /* Enable and set Touch screen EXTI Interr
upt to the lowest priority */
00160   HAL_NVIC_SetPriority((IRQn_Type)(TS_RGB_IN
T_EXTI_IRQn), 0x0F, 0x00);
00161   HAL_NVIC_EnableIRQ((IRQn_Type)(TS_RGB_INT_
EXTI_IRQn));
00162
00163   /* Enable the TS ITs */
00164   tsDriver->EnableIT(I2cAddress);
00165
00166   return TS_OK;
00167 }
00168
00169 /**
00170   * @brief  Gets the touch screen interrupt
status.
00171   * @retval TS_OK if all initializations are
 OK. Other value if error.
00172   */
00173 uint8_t BSP_RGB_TS_ITGetStatus(void)
00174 {
00175   /* Return the TS IT status */
```

```
00176   return (tsDriver->GetITStatus(I2cAddress))
;
00177 }
00178
00179 /**
00180   * @brief  Returns status and positions of
the touch screen.
00181   * @param  TS_State: Pointer to touch scree
n current state structure
00182   * @retval TS_OK if all initializations are
 OK. Other value if error.
00183   */
00184 uint8_t BSP_RGB_TS_GetState(TS_StateTypeDef
*TS_State)
00185 {
00186   static uint32_t _x[TS_MAX_NB_TOUCH] = {0,
0};
00187   static uint32_t _y[TS_MAX_NB_TOUCH] = {0,
0};
00188   uint8_t ts_status = TS_OK;
00189   uint16_t x[TS_MAX_NB_TOUCH];
00190   uint16_t y[TS_MAX_NB_TOUCH];
00191   uint16_t brute_x[TS_MAX_NB_TOUCH];
00192   uint16_t brute_y[TS_MAX_NB_TOUCH];
00193   uint16_t x_diff;
00194   uint16_t y_diff;
00195   uint32_t index;
00196 #if (TS_MULTI_TOUCH_SUPPORTED == 1)
00197   uint32_t weight = 0;
00198   uint32_t area = 0;
00199   uint32_t event = 0;
00200 #endif /* TS_MULTI_TOUCH_SUPPORTED == 1 */
00201
00202   /* Check and update the number of touches
active detected */
00203   TS_State->touchDetected = tsDriver->Detect
Touch(I2cAddress);
```

```
00204
00205   if(TS_State->touchDetected)
00206   {
00207     for(index=0; index < TS_State->touchDete
cted; index++)
00208     {
00209       /* Get each touch coordinates */
00210       tsDriver->GetXY(I2cAddress, &(brute_x[
index]), &(brute_y[index]));
00211
00212       if(tsOrientation == TS_SWAP_NONE)
00213       {
00214         x[index] = brute_x[index];
00215         y[index] = brute_y[index];
00216       }
00217
00218       if(tsOrientation & TS_SWAP_X)
00219       {
00220         x[index] = 4096 - brute_x[index];
00221       }
00222
00223       if(tsOrientation & TS_SWAP_Y)
00224       {
00225         y[index] = 4096 - brute_y[index];
00226       }
00227
00228       if(tsOrientation & TS_SWAP_XY)
00229       {
00230         y[index] = brute_x[index];
00231         x[index] = brute_y[index];
00232       }
00233
00234       x_diff = x[index] > _x[index]? (x[inde
x] - _x[index]): (_x[index] - x[index]);
00235       y_diff = y[index] > _y[index]? (y[inde
x] - _y[index]): (_y[index] - y[index]);
00236
```

```
00237       if ((x_diff + y_diff) > 5)
00238       {
00239         _x[index] = x[index];
00240         _y[index] = y[index];
00241       }
00242
00243       if(I2cAddress == FT5336_I2C_SLAVE_ADDR
ESS)
00244       {
00245         TS_State->touchX[index] = x[index];
00246         TS_State->touchY[index] = y[index];
00247       }
00248       else
00249       {
00250         /* 2^12 = 4096 : indexes are express
ed on a dynamic of 4096 */
00251         TS_State->touchX[index] = (tsXBounda
ry * _x[index]) >> 12;
00252         TS_State->touchY[index] = (tsYBounda
ry * _y[index]) >> 12;
00253       }
00254
00255 #if (TS_MULTI_TOUCH_SUPPORTED == 1)
00256
00257       /* Get touch info related to the curre
nt touch */
00258       ft5336_TS_GetTouchInfo(I2cAddress, ind
ex, &weight, &area, &event);
00259
00260       /* Update TS_State structure */
00261       TS_State->touchWeight[index] = weight;
00262       TS_State->touchArea[index]   = area;
00263
00264       /* Remap touch event */
00265       switch(event)
00266       {
00267         case FT5336_TOUCH_EVT_FLAG_PRESS_DOW
```

```
N   :
00268           TS_State->touchEventId[index] = TO
UCH_EVENT_PRESS_DOWN;
00269           break;
00270         case FT5336_TOUCH_EVT_FLAG_LIFT_UP :
00271           TS_State->touchEventId[index] = TO
UCH_EVENT_LIFT_UP;
00272           break;
00273         case FT5336_TOUCH_EVT_FLAG_CONTACT :
00274           TS_State->touchEventId[index] = TO
UCH_EVENT_CONTACT;
00275           break;
00276         case FT5336_TOUCH_EVT_FLAG_NO_EVENT
:
00277           TS_State->touchEventId[index] = TO
UCH_EVENT_NO_EVT;
00278           break;
00279         default :
00280           ts_status = TS_ERROR;
00281           break;
00282       } /* of switch(event) */
00283
00284 #endif /* TS_MULTI_TOUCH_SUPPORTED == 1 */
00285
00286     } /* of for(index=0; index < TS_State->t
ouchDetected; index++) */
00287
00288 #if (TS_MULTI_TOUCH_SUPPORTED == 1)
00289     /* Get gesture Id */
00290     ts_status = BSP_RGB_TS_Get_GestureId(TS_
State);
00291 #endif /* TS_MULTI_TOUCH_SUPPORTED == 1 */
00292
00293   } /* end of if(TS_State->touchDetected !=
0) */
00294
00295   return (ts_status);
```

```
00296 }
00297
00298 #if (TS_MULTI_TOUCH_SUPPORTED == 1)
00299 /**
00300   * @brief  Update gesture Id following a to
uch detected.
00301   * @param  TS_State: Pointer to touch scree
n current state structure
00302   * @retval TS_OK if all initializations are
 OK. Other value if error.
00303   */
00304 uint8_t BSP_RGB_TS_Get_GestureId(TS_StateTyp
eDef *TS_State)
00305 {
00306   uint32_t gestureId = 0;
00307   uint8_t  ts_status = TS_OK;
00308
00309   /* Get gesture Id */
00310   ft5336_TS_GetGestureID(I2cAddress, &gestur
eId);
00311
00312   /* Remap gesture Id to a TS_GestureIdTypeD
ef value */
00313   switch(gestureId)
00314   {
00315     case FT5336_GEST_ID_NO_GESTURE :
00316       TS_State->gestureId = GEST_ID_NO_GESTU
RE;
00317       break;
00318     case FT5336_GEST_ID_MOVE_UP :
00319       TS_State->gestureId = GEST_ID_MOVE_UP;
00320       break;
00321     case FT5336_GEST_ID_MOVE_RIGHT :
00322       TS_State->gestureId = GEST_ID_MOVE_RIG
HT;
00323       break;
00324     case FT5336_GEST_ID_MOVE_DOWN :
```

```
00325       TS_State->gestureId = GEST_ID_MOVE_DOWN
;
00326       break;
00327     case FT5336_GEST_ID_MOVE_LEFT :
00328       TS_State->gestureId = GEST_ID_MOVE_LEFT
;
00329       break;
00330     case FT5336_GEST_ID_ZOOM_IN :
00331       TS_State->gestureId = GEST_ID_ZOOM_IN;
00332       break;
00333     case FT5336_GEST_ID_ZOOM_OUT :
00334       TS_State->gestureId = GEST_ID_ZOOM_OUT
;
00335       break;
00336     default :
00337       ts_status = TS_ERROR;
00338       break;
00339   } /* of switch(gestureId) */
00340
00341   return(ts_status);
00342 }
00343 #endif /* TS_MULTI_TOUCH_SUPPORTED == 1 */
00344
00345 /**
00346   * @brief  Clears all touch screen interrup
ts.
00347   */
00348 void BSP_RGB_TS_ITClear(void)
00349 {
00350   /* Clear TS IT pending bits */
00351   tsDriver->ClearIT(I2cAddress);
00352 }
00353
00354 /**
00355   * @brief  Function used to reset all touch
 data before a new acquisition
00356   *         of touch information.
```

```
00357   * @param  TS_State: Pointer to touch scree
n current state structure
00358   * @retval TS_OK if OK, TE_ERROR if problem
 found.
00359   */
00360 uint8_t BSP_RGB_TS_ResetTouchData(TS_StateTy
peDef *TS_State)
00361 {
00362   uint8_t ts_status = TS_ERROR;
00363   uint32_t index;
00364
00365   if (TS_State != (TS_StateTypeDef *)NULL)
00366   {
00367     TS_State->gestureId = GEST_ID_NO_GESTURE
;
00368     TS_State->touchDetected = 0;
00369
00370     for(index = 0; index < TS_MAX_NB_TOUCH;
index++)
00371     {
00372       TS_State->touchX[index]       = 0;
00373       TS_State->touchY[index]       = 0;
00374       TS_State->touchArea[index]    = 0;
00375       TS_State->touchEventId[index] = TOUCH_
EVENT_NO_EVT;
00376       TS_State->touchWeight[index]  = 0;
00377     }
00378
00379     ts_status = TS_OK;
00380
00381   } /* of if (TS_State != (TS_StateTypeDef *
)NULL) */
00382
00383   return (ts_status);
00384 }
00385
00386 /**
```

```
00387   * @}
00388   */
00389
00390 /**
00391   * @}
00392   */
00393
00394 #endif /* USE_ROCKTECH_480x272 */
00395
00396 /**
00397   * @}
00398   */
00399
00400 /**
00401   * @}
00402   */
00403
00404 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

Exported Defines

STM32L4R9I_EVAL Common

Defines

#define	**__STM32L4R9I_EVAL_BSP_VERSION_MAIN** (0x01) STM32L4R9I EVAL BSP Driver version number.
#define	**__STM32L4R9I_EVAL_BSP_VERSION_SUB1** (0x00)
#define	**__STM32L4R9I_EVAL_BSP_VERSION_SUB2** (0x00)
#define	**__STM32L4R9I_EVAL_BSP_VERSION_RC** (0x00)
#define	**__STM32L4R9I_EVAL_BSP_VERSION**
#define	**BSP_I2C_NO_USER** 0x00000000U BSP I2C users.
#define	**BSP_I2C_AUDIO_USER** 0x00000001U
#define	**BSP_I2C_TS_USER** 0x00000002U
#define	**BSP_I2C_MFX_USER** 0x00000004U
#define	**BSP_I2C_EEPROM_USER** 0x00000008U
#define	**BSP_I2C_ALL_USERS** 0x0000000FU

Define Documentation

#define __STM32L4R9I_EVAL_BSP_VERSION

Value:

```
((__STM32L4R9I_EVAL_BSP_VERSION_MAIN << 24)\
|(__STM32L4R9I_EVAL_BSP_VERSION_SUB1 << 16)\
|(__STM32L4R9I_EVAL_BSP_VERSION_SUB2 << 8 )\
|(__STM32L4R9I_EVAL_BSP_VERSION_RC))
```

Definition at line **75** of file **stm32l4r9i_eval.c**.

Referenced by **BSP_GetVersion()**.

#define __STM32L4R9I_EVAL_BSP_VERSION_MAIN (0x01)

STM32L4R9I EVAL BSP Driver version number.

[31:24] main version

Definition at line **71** of file **stm32l4r9i_eval.c**.

#define __STM32L4R9I_EVAL_BSP_VERSION_RC (0x00)

[7:0] release candidate

Definition at line **74** of file **stm32l4r9i_eval.c**.

#define __STM32L4R9I_EVAL_BSP_VERSION_SUB1 (0x00)

[23:16] sub1 version

Definition at line **72** of file **stm32l4r9i_eval.c**.

#define __STM32L4R9I_EVAL_BSP_VERSION_SUB2 (0x00)

[15:8] sub2 version

Definition at line **73** of file **stm32l4r9i_eval.c**.

#define BSP_I2C_ALL_USERS 0x0000000FU

Definition at line **89** of file **stm32l4r9i_eval.c**.

Referenced by **I2Cx_Error()**.

#define BSP_I2C_AUDIO_USER 0x00000001U

Definition at line **85** of file **stm32l4r9i_eval.c**.

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

#define BSP_I2C_EEPROM_USER 0x00000008U

Definition at line **88** of file **stm32l4r9i_eval.c**.

Referenced by **EEPROM_IO_Init()**.

#define BSP_I2C_MFX_USER 0x00000004U

Definition at line **87** of file **stm32l4r9i_eval.c**.

Referenced by **MFX_IO_DeInit()**, and **MFX_IO_Init()**.

#define BSP_I2C_NO_USER 0x00000000U

BSP I2C users.

Definition at line **84** of file **stm32l4r9i_eval.c**.

Referenced by **I2Cx_DeInit()**.

#define BSP_I2C_TS_USER 0x00000002U

Definition at line **86** of file **stm32l4r9i_eval.c**.

Referenced by **TS_IO_Init()**.

STM32L4R9I_EVAL BSP User Manual

Private Macros

STM32L4R9I_EVAL DSI LCD

Defines

```
#define  ABS(X)   ((X) > 0 ? (X) : -(X))
#define  POLY_X(Z)   ((int32_t)((Points + (Z))->X))
#define  POLY_Y(Z)   ((int32_t)((Points + (Z))->Y))
```

Define Documentation

#define ABS (X) ((X) > 0 ? (X) : -(X))

Definition at line 114 of file stm32l4r9i_eval_dsi_lcd.c.

Referenced by BSP_DSI_LCD_DrawLine(), and FillTriangle().

#define POLY_X (Z) ((int32_t)((Points + (Z))->X))

Definition at line 116 of file stm32l4r9i_eval_dsi_lcd.c.

Referenced by BSP_DSI_LCD_FillPolygon().

#define POLY_Y (Z) ((int32_t)((Points + (Z))->Y))

Definition at line 117 of file stm32l4r9i_eval_dsi_lcd.c.

Referenced by BSP_DSI_LCD_FillPolygon().

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL DSI LCD

Variables

static uint32_t	**bsp_lcd_initialized** = 0
static uint32_t	**bsp_lcd_hse_to_disable** = 0
static uint32_t	**ActiveLayer** = **LTDC_ACTIVE_LAYER_BACKGROU**
static **LCD_DrawPropTypeDef**	**DrawProp** [**LTDC_MAX_LAYER_NUMBER**]

Variable Documentation

uint32_t ActiveLayer = LTDC_ACTIVE_LAYER_BACKGROUND [sta

Definition at line 140 of file stm32l4r9i_eval_dsi_lcd.c.

Referenced by BSP_DSI_LCD_Clear(), BSP_DSI_LCD_ClearStringLine(), BSP_DSI_LCD_DisplayChar(), BSP_DSI_LCD_DisplayStringAt(), BSP_DSI_LCD_DrawBitmap(), BSP_DSI_LCD_DrawCircle(), BSP_DSI_LCD_DrawEllipse(), BSP_DSI_LCD_DrawHLine(), BSP_DSI_LCD_DrawLine(), BSP_DSI_LCD_DrawPixel(), BSP_DSI_LCD_DrawVLine(), BSP_DSI_LCD_FillCircle(), BSP_DSI_LCD_FillRect(), BSP_DSI_LCD_GetBackColor(), BSP_DSI_LCD_GetFont(), BSP_DSI_LCD_GetTextColor(), BSP_DSI_LCD_ReadPixel(), BSP_DSI_LCD_SelectLayer(), BSP_DSI_LCD_SetBackColor(), BSP_DSI_LCD_SetFont(), BSP_DSI_LCD_SetTextColor(), and DrawChar().

uint32_t bsp_lcd_hse_to_disable = 0 [static]

Definition at line 138 of file stm32l4r9i_eval_dsi_lcd.c.

Referenced by BSP_DSI_LCD_MspDeInit(), and BSP_DSI_LCD_MspInit().

uint32_t bsp_lcd_initialized = 0 [static]

Definition at line 136 of file stm32l4r9i_eval_dsi_lcd.c.

Referenced by BSP_DSI_LCD_DeInit().

LCD_DrawPropTypeDef DrawProp[LTDC_MAX_LAYER_NUMBER]

Definition at line **142** of file **stm32l4r9i_eval_dsi_lcd.c**.

STM32L4R9I_EVAL BSP User Manual

POTENTIOMETER Constants

Exported Constants

Defines

#define	**BSP_POTENTIOMETER_OK** 0x00 Potentiometer Status values.
#define	**BSP_POTENTIOMETER_ERROR** 0xFF
#define	**ADCx** ADC1 Definition for Potentiometer, connected to ADC1.
#define	**ADCx_CLK_ENABLE**() __HAL_RCC_ADC_CLK_ENABLE()
#define	**ADCx_CHANNEL_GPIO_CLK_ENABLE**() __HAL_RCC_GP
#define	**ADCx_FORCE_RESET**() __HAL_RCC_ADC_FORCE_RESE
#define	**ADCx_RELEASE_RESET**() __HAL_RCC_ADC_RELEASE_I
#define	**ADCx_CHANNEL_PIN** GPIO_PIN_0
#define	**ADCx_CHANNEL_GPIO_PORT** GPIOA
#define	**ADCx_CHANNEL** ADC_CHANNEL_5
#define	**SAMPLINGTIME** ADC_SAMPLETIME_3CYCLES
#define	**ADCx_POLL_TIMEOUT** 10
#define	**OPAMPx_INP_PIN** GPIO_PIN_0 Definition for Potentiometer connected to OPAMP.
#define	**OPAMPx_INP_GPIO_PORT** GPIOA
#define	**OPAMPx_INM_PIN** GPIO_PIN_1
#define	**OPAMPx_INM_GPIO_PORT** GPIOA
#define	**OPAMPx_VOUT_PIN** GPIO_PIN_3
#define	**OPAMPx_VOUT_GPIO_PORT** GPIOA
#define	**OPAMPx** OPAMP
#define	**OPAMPx_CLK_ENABLE**() __HAL_RCC_OPAMP_CLK_ENA
#define	**OPAMPx_CLK_DISABLE**() __HAL_RCC_OPAMP_CLK_DIS
#define	**OPAMPx_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOA_CLK
#define	**OPAMPx_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOA_CL
#define	**OPAMPx_FORCE_RESET**() __HAL_RCC_OPAMP_FORCE_
#define	**OPAMPx_RELEASE_RESET**() __HAL_RCC_OPAMP_RELE
#define	**DSI_RST_PIN** **IO_PIN_9** Eval Pins definition connected to MFX.
#define	**EXT_RST_PIN** **IO_PIN_8**

```
#define LCD_DISP_PIN   IO_PIN_11
#define LCD_RST_PIN   IO_PIN_12
#define OTG_FS_POWER_SWITCH_PIN   IO_PIN_6
#define OTG_FS_OVER_CURRENT_PIN   IO_PIN_7
#define SD_DETECT_PIN   IO_PIN_5
#define SWIRE_PIN   IO_PIN_13
```

Define Documentation

#define ADCx ADC1

Definition for Potentiometer, connected to ADC1.

Definition at line 324 of file stm32l4r9i_eval.h.

Referenced by BSP_POTENTIOMETER_Init().

#define ADCx_CHANNEL ADC_CHANNEL_5

Definition at line 336 of file stm32l4r9i_eval.h.

Referenced by BSP_POTENTIOMETER_Init().

#define ADCx_CHANNEL_GPIO_CLK_ENABLE ( ) __HAL_RCC_G

Definition at line 326 of file stm32l4r9i_eval.h.

Referenced by BSP_POTENTIOMETER_Init().

#define ADCx_CHANNEL_GPIO_PORT GPIOA

Definition at line 333 of file stm32l4r9i_eval.h.

Referenced by BSP_POTENTIOMETER_Init().

#define ADCx_CHANNEL_PIN GPIO_PIN_0

Definition at line 332 of file stm32l4r9i_eval.h.

Referenced by **BSP_POTENTIOMETER_Init()**.

#define ADCx_CLK_ENABLE ( ) __HAL_RCC_ADC_CLK_ENABLE

Definition at line **325** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_POTENTIOMETER_Init()**.

#define ADCx_FORCE_RESET ( ) __HAL_RCC_ADC_FORCE_RES

Definition at line **328** of file **stm32l4r9i_eval.h**.

#define ADCx_POLL_TIMEOUT 10

Definition at line **338** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_POTENTIOMETER_GetLevel()**.

#define ADCx_RELEASE_RESET ( ) __HAL_RCC_ADC_RELEASE

Definition at line **329** of file **stm32l4r9i_eval.h**.

#define BSP_POTENTIOMETER_ERROR 0xFF

Definition at line **319** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_POTENTIOMETER_DeInit()**, **BSP_POTENTIOMETER_GetLevel()**, and **BSP_POTENTIOMETER_Init()**.

#define BSP_POTENTIOMETER_OK 0x00

Potentiometer Status values.

Definition at line **318** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_POTENTIOMETER_DeInit()**, and **BSP_POTENTIOMETER_Init()**.

#define DSI_RST_PIN IO_PIN_9

Eval Pins definition connected to MFX.

Definition at line **366** of file **stm32l4r9i_eval.h**.

#define EXT_RST_PIN IO_PIN_8

Definition at line **367** of file **stm32l4r9i_eval.h**.

#define LCD_DISP_PIN IO_PIN_11

Definition at line **368** of file **stm32l4r9i_eval.h**.

#define LCD_RST_PIN IO_PIN_12

Definition at line **369** of file **stm32l4r9i_eval.h**.

#define OPAMPx OPAMP

Definition at line **353** of file **stm32l4r9i_eval.h**.

#define OPAMPx_CLK_DISABLE ( ) __HAL_RCC_OPAMP_CLK_D

Definition at line **355** of file **stm32l4r9i_eval.h**.

#define OPAMPx_CLK_ENABLE ( ) __HAL_RCC_OPAMP_CLK_EI

Definition at line **354** of file **stm32l4r9i_eval.h**.

#define OPAMPx_FORCE_RESET ( ) __HAL_RCC_OPAMP_FORC

Definition at line **359** of file **stm32l4r9i_eval.h**.

#define OPAMPx_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOA_C

Definition at line **357** of file **stm32l4r9i_eval.h**.

#define OPAMPx_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOA_CI

Definition at line **356** of file **stm32l4r9i_eval.h**.

#define OPAMPx_INM_GPIO_PORT GPIOA

Definition at line **349** of file **stm32l4r9i_eval.h**.

#define OPAMPx_INM_PIN GPIO_PIN_1

Definition at line **348** of file **stm32l4r9i_eval.h**.

#define OPAMPx_INP_GPIO_PORT GPIOA

Definition at line **347** of file **stm32l4r9i_eval.h**.

#define OPAMPx_INP_PIN GPIO_PIN_0

Definition for Potentiometer connected to OPAMP.

Definition at line 346 of file stm32l4r9i_eval.h.

#define OPAMPx_RELEASE_RESET ( ) __HAL_RCC_OPAMP_REI

Definition at line 360 of file stm32l4r9i_eval.h.

#define OPAMPx_VOUT_GPIO_PORT GPIOA

Definition at line 351 of file stm32l4r9i_eval.h.

#define OPAMPx_VOUT_PIN GPIO_PIN_3

Definition at line 350 of file stm32l4r9i_eval.h.

#define OTG_FS_OVER_CURRENT_PIN IO_PIN_7

Definition at line 371 of file stm32l4r9i_eval.h.

#define OTG_FS_POWER_SWITCH_PIN IO_PIN_6

Definition at line 370 of file stm32l4r9i_eval.h.

#define SAMPLINGTIME ADC_SAMPLETIME_3CYCLES

Definition at line 337 of file stm32l4r9i_eval.h.

#define SD_DETECT_PIN IO_PIN_5

Definition at line 372 of file stm32l4r9i_eval.h.

Referenced by BSP_SD_Init(), BSP_SD_IsDetected(), and BSP_SD_ITConfig().

#define SWIRE_PIN IO_PIN_13

Definition at line 373 of file stm32l4r9i_eval.h.

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL IO

Defines

#define	**IO_PIN_0** ((uint32_t)0x00000001)
#define	**IO_PIN_1** ((uint32_t)0x00000002)
#define	**IO_PIN_2** ((uint32_t)0x00000004)
#define	**IO_PIN_3** ((uint32_t)0x00000008)
#define	**IO_PIN_4** ((uint32_t)0x00000010)
#define	**IO_PIN_5** ((uint32_t)0x00000020)
#define	**IO_PIN_6** ((uint32_t)0x00000040)
#define	**IO_PIN_7** ((uint32_t)0x00000080)
#define	**IO_PIN_8** ((uint32_t)0x00000100)
#define	**IO_PIN_9** ((uint32_t)0x00000200)
#define	**IO_PIN_10** ((uint32_t)0x00000400)
#define	**IO_PIN_11** ((uint32_t)0x00000800)
#define	**IO_PIN_12** ((uint32_t)0x00001000)
#define	**IO_PIN_13** ((uint32_t)0x00002000)
#define	**IO_PIN_14** ((uint32_t)0x00004000)
#define	**IO_PIN_15** ((uint32_t)0x00008000)
#define	**AGPIO_PIN_0** ((uint32_t)0x00010000)
#define	**AGPIO_PIN_1** ((uint32_t)0x00020000)
#define	**IO_PIN_ALL** ((uint32_t)0x0003FFFF)

Define Documentation

#define AGPIO_PIN_0 ((uint32_t)0x00010000)

Definition at line 95 of file stm32l4r9i_eval_io.h.

#define AGPIO_PIN_1 ((uint32_t)0x00020000)

Definition at line 96 of file stm32l4r9i_eval_io.h.

#define IO_PIN_0 ((uint32_t)0x00000001)

Definition at line 79 of file stm32l4r9i_eval_io.h.

#define IO_PIN_1 ((uint32_t)0x00000002)

Definition at line 80 of file stm32l4r9i_eval_io.h.

#define IO_PIN_10 ((uint32_t)0x00000400)

Definition at line 89 of file stm32l4r9i_eval_io.h.

#define IO_PIN_11 ((uint32_t)0x00000800)

Definition at line 90 of file stm32l4r9i_eval_io.h.

#define IO_PIN_12 ((uint32_t)0x00001000)

Definition at line 91 of file stm32l4r9i_eval_io.h.

#define IO_PIN_13 ((uint32_t)0x00002000)

Definition at line 92 of file stm32l4r9i_eval_io.h.

#define IO_PIN_14 ((uint32_t)0x00004000)

Definition at line 93 of file stm32l4r9i_eval_io.h.

#define IO_PIN_15 ((uint32_t)0x00008000)

Definition at line 94 of file stm32l4r9i_eval_io.h.

#define IO_PIN_2 ((uint32_t)0x00000004)

Definition at line 81 of file stm32l4r9i_eval_io.h.

#define IO_PIN_3 ((uint32_t)0x00000008)

Definition at line 82 of file stm32l4r9i_eval_io.h.

#define IO_PIN_4 ((uint32_t)0x00000010)

Definition at line 83 of file stm32l4r9i_eval_io.h.

#define IO_PIN_5 ((uint32_t)0x00000020)

Definition at line 84 of file stm32l4r9i_eval_io.h.

#define IO_PIN_6 ((uint32_t)0x00000040)

Definition at line 85 of file stm32l4r9i_eval_io.h.

#define IO_PIN_7 ((uint32_t)0x00000080)

Definition at line 86 of file stm32l4r9i_eval_io.h.

#define IO_PIN_8 ((uint32_t)0x00000100)

Definition at line 87 of file stm32l4r9i_eval_io.h.

#define IO_PIN_9 ((uint32_t)0x00000200)

Definition at line 88 of file stm32l4r9i_eval_io.h.

Referenced by LCD_Reset().

#define IO_PIN_ALL ((uint32_t)0x0003FFFF)

Definition at line 97 of file stm32l4r9i_eval_io.h.

Referenced by BSP_IO_Init().

STM32L4R9I_EVAL BSP User Manual

Private Functions

STM32L4R9I_EVAL AUDIO

Functions

static void	**AUDIO_CODEC_Reset** (void) Resets the audio codec.
static uint8_t	**AUDIO_SAIx_Init** (uint32_t AudioFreq, uint16_t OutputDevice) Initializes the Audio Codec audio interface (SAI).
static uint8_t	**AUDIO_SAIx_DeInit** (void) De-initializes the Audio Codec audio interface (SAI).
static uint8_t	**AUDIO_DFSDMx_Init** (uint32_t AudioFreq) Initializes the Digital Filter for Sigma-Delta Modulators interface (DFSDM).
static uint8_t	**AUDIO_DFSDMx_DeInit** (void) De-initializes the Digital Filter for Sigma-Delta Modulators interface (DFSDM).
static uint8_t	**AUDIO_SAIPLLConfig** (uint32_t Frequency) Configures the SAI PLL clock according to the required audio frequency.
void	**HAL_SAI_MspInit** (SAI_HandleTypeDef *hsai) SAI MSP Init.
void	**HAL_SAI_MspDeInit** (SAI_HandleTypeDef *hsai) SAI MSP De-init.
void	**HAL_DFSDM_ChannelMspInit** (DFSDM_Channel_HandleTypeDef *hdfsdm_channel) Initializes the DFSDM channel MSP.
void	**HAL_DFSDM_ChannelMspDeInit** (DFSDM_Channel_HandleTypeDef *hdfsdm_channel) DeInitializes the DFSDM channel MSP.
void	**HAL_DFSDM_FilterMspInit** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter) Initializes the DFSDM filter MSP.
void	**HAL_DFSDM_FilterMspDeInit** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter)

DeInitializes the DFSDM filter MSP.

Function Documentation

static void AUDIO_CODEC_Reset (void) [static]

Resets the audio codec.

It restores the default configuration of the codec (this function shall be called before initializing the codec).

Return values:

None

Definition at line **1275** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_I2C_ADDRESS**, and **AUDIO_OUT_TypeDef::AudioDrv**.

Referenced by **BSP_AUDIO_OUT_Init()**.

static uint8_t AUDIO_DFSDMx_DeInit (void) [static]

De-initializes the Digital Filter for Sigma-Delta Modulators interface (DFSDM).

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **1402** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **BSP_AUDIO_hDfsdmLeftFilter**, **BSP_AUDIO_hDfsdmRightFilter**, **AUDIO_IN_TypeDef::hDfsdmLeftChannel**, and **AUDIO_IN_TypeDef::hDfsdmRightChannel**.

Referenced by **BSP_AUDIO_IN_DeInit()**, and

BSP_AUDIO_IN_SetFrequency().

static uint8_t AUDIO_DFSDMx_Init (uint32_t AudioFreq) [static]

Initializes the Digital Filter for Sigma-Delta Modulators interface (DFSDM).

Parameters:

AudioFreq,: Audio frequency to be used to set correctly the DFSDM peripheral.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **1299** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_DFSDMx_LEFT_CHANNEL**, **AUDIO_DFSDMx_LEFT_FILTER**, **AUDIO_DFSDMx_RIGHT_CHANNEL**, **AUDIO_DFSDMx_RIGHT_FILTER**, **AUDIO_ERROR**, **AUDIO_OK**, **BSP_AUDIO_hDfsdmLeftFilter**, **BSP_AUDIO_hDfsdmRightFilter**, **DFSDMClockDivider**, **DFSDMFilterOrder**, **DFSDMOverSampling**, **DFSDMRightBitShift**, **AUDIO_IN_TypeDef::hDfsdmLeftChannel**, and **AUDIO_IN_TypeDef::hDfsdmRightChannel**.

Referenced by **BSP_AUDIO_IN_Init()**, and **BSP_AUDIO_IN_SetFrequency()**.

static uint8_t AUDIO_SAIPLLConfig (uint32_t Frequency) [static]

Configures the SAI PLL clock according to the required audio frequency.

Parameters:

Frequency,: Audio frequency.

Return values:

BSP AUDIO status

Note:

The SAI PLL input clock must be configured in the user application. The SAI PLL configuration done within this function assumes that the SAI PLL input clock runs at 4 MHz.

Definition at line **1583** of file **stm32l4r9i_eval_audio.c**.

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

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

static uint8_t AUDIO_SAIx_DeInit (void) [static]

De-initializes the Audio Codec audio interface (SAI).

Return values:

BSP AUDIO status

Definition at line **1152** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **AUDIO_SAIx_PLL_DISABLE**, and **BSP_AUDIO_hSai**.

Referenced by **BSP_AUDIO_OUT_DeInit()**.

static uint8_t AUDIO_SAIx_Init (uint32_t AudioFreq, uint16_t OutputDevice) [static]

Initializes the Audio Codec audio interface (SAI).

Parameters:

AudioFreq,:	Audio frequency to be configured for the SAI peripheral.
OutputDevice,:	OUTPUT_DEVICE_SPEAKER, OUTPUT_DEVICE_HEADPHONE, or OUTPUT_DEVICE_BOTH.

Note:
The default SlotActive configuration is set to CODEC_AUDIOFRAME_SLOT_0123 and user can update this configuration using

Return values:
None

Definition at line **1074** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **AUDIO_SAIx**, **BSP_AUDIO_hSai**, and **SAIClockDivider**.

Referenced by **BSP_AUDIO_OUT_Init()**.

void HAL_DFSDM_ChannelMspDeInit (DFSDM_Channel_HandleTy

DeInitializes the DFSDM channel MSP.

Parameters:
hdfsdm_channel : DFSDM channel handle.

Return values:
None

Definition at line **1463** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_DFSDMx_CKOUT_CLK_DISABLE**, **AUDIO_DFSDMx_CKOUT_GPIO_PORT**, **AUDIO_DFSDMx_CKOUT_PIN**, **AUDIO_DFSDMx_CLK_DISABLE**, **AUDIO_DFSDMx_DATIN_GPIO_PORT**, and **AUDIO_DFSDMx_DATIN_PIN**.

void HAL_DFSDM_ChannelMspInit (DFSDM_Channel_HandleTypel

Initializes the DFSDM channel MSP.

Parameters:

hdfsdm_channel : DFSDM channel handle.

Return values:

None

Definition at line **1436** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_DFSDMx_CKOUT_CLK_ENABLE**, **AUDIO_DFSDMx_CKOUT_DATIN_AF**, **AUDIO_DFSDMx_CKOUT_GPIO_PORT**, **AUDIO_DFSDMx_CKOUT_PIN**, **AUDIO_DFSDMx_CLK_ENABLE**, **AUDIO_DFSDMx_DATIN_CLK_ENABLE**, **AUDIO_DFSDMx_DATIN_GPIO_PORT**, and **AUDIO_DFSDMx_DATIN_PIN**.

void HAL_DFSDM_FilterMspDeInit (DFSDM_Filter_HandleTypeDef

DeInitializes the DFSDM filter MSP.

Parameters:

hdfsdm_filter : DFSDM filter handle.

Return values:

None

Definition at line **1549** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_DFSDMx_DMAx_CLK_DISABLE**, **AUDIO_DFSDMx_DMAx_LEFT_IRQ**, **AUDIO_DFSDMx_DMAx_RIGHT_IRQ**, **AUDIO_DFSDMx_LEFT_FILTER**, **AUDIO_IN_TypeDef::hDmaDfsdmLeft**, and **AUDIO_IN_TypeDef::hDmaDfsdmRight**.

void HAL_DFSDM_FilterMspInit (DFSDM_Filter_HandleTypeDef * h

Initializes the DFSDM filter MSP.

Parameters:

hdfsdm_filter : DFSDM filter handle.

Return values:

None

Definition at line **1481** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_DFSDMx_DMAx_CLK_ENABLE**, **AUDIO_DFSDMx_DMAx_LEFT_CHANNEL**, **AUDIO_DFSDMx_DMAx_LEFT_IRQ**, **AUDIO_DFSDMx_DMAx_LEFT_REQUEST**, **AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE**, **AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE**, **AUDIO_DFSDMx_DMAx_RIGHT_CHANNEL**, **AUDIO_DFSDMx_DMAx_RIGHT_IRQ**, **AUDIO_DFSDMx_DMAx_RIGHT_REQUEST**, **AUDIO_DFSDMx_LEFT_FILTER**, **AUDIO_IN_IRQ_PREPRIO**, **AUDIO_IN_TypeDef::hDmaDfsdmLeft**, and **AUDIO_IN_TypeDef::hDmaDfsdmRight**.

void HAL_SAI_MspDeInit (SAI_HandleTypeDef * hsai)

SAI MSP De-init.

Parameters:

hsai : pointer to a SAI_HandleTypeDef structure

Return values:

None

Definition at line **1242** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_SAIx_CLK_DISABLE**, **AUDIO_SAIx_DMAx_CLK_DISABLE**, **AUDIO_SAIx_DMAx_IRQ**, **AUDIO_SAIx_FS_CLK_DISABLE**, **AUDIO_SAIx_FS_GPIO_PORT**, **AUDIO_SAIx_FS_PIN**, **AUDIO_SAIx_MCK_CLK_DISABLE**, **AUDIO_SAIx_MCK_GPIO_PORT**, **AUDIO_SAIx_MCK_PIN**, **AUDIO_SAIx_SCK_CLK_DISABLE**, **AUDIO_SAIx_SCK_GPIO_PORT**, **AUDIO_SAIx_SCK_PIN**, **AUDIO_SAIx_SD_GPIO_PORT**, **AUDIO_SAIx_SD_PIN**, and **hDmaSai**.

void HAL_SAI_MspInit (SAI_HandleTypeDef * hsai)

SAI MSP Init.

Parameters:

hsai : pointer to a SAI_HandleTypeDef structure

Return values:

None

Definition at line **1177** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_OUT_IRQ_PREPRIO**, **AUDIO_SAIx**, **AUDIO_SAIx_CLK_ENABLE**, **AUDIO_SAIx_DMAx_CHANNEL**, **AUDIO_SAIx_DMAx_CLK_ENABLE**, **AUDIO_SAIx_DMAx_IRQ**, **AUDIO_SAIx_DMAx_MEM_DATA_SIZE**,

AUDIO_SAIx_DMAx_PERIPH_DATA_SIZE,
AUDIO_SAIx_DMAx_REQUEST, **AUDIO_SAIx_FS_CLK_ENABLE**,
AUDIO_SAIx_FS_GPIO_PORT, **AUDIO_SAIx_FS_PIN**,
AUDIO_SAIx_MCK_CLK_ENABLE,
AUDIO_SAIx_MCK_GPIO_PORT, **AUDIO_SAIx_MCK_PIN**,
AUDIO_SAIx_MCK_SCK_SD_FS_AF,
AUDIO_SAIx_SCK_CLK_ENABLE,
AUDIO_SAIx_SCK_GPIO_PORT, **AUDIO_SAIx_SCK_PIN**,
AUDIO_SAIx_SD_CLK_ENABLE, **AUDIO_SAIx_SD_GPIO_PORT**,
AUDIO_SAIx_SD_PIN, and **hDmaSai**.

STM32L4R9I_EVAL BSP User Manual

Exported Constants

STM32L4R9I_EVAL AUDIO

Modules

- BSP Audio Out Option
- BSP Audio Sample Rate

Defines

#define	**AUDIO_SAIx** SAI1_Block_A
#define	**AUDIO_SAIx_CLK_ENABLE**() __HAL_RCC_SAI1_CLK_EN
#define	**AUDIO_SAIx_CLK_DISABLE**() __HAL_RCC_SAI1_CLK_DI
#define	**AUDIO_SAIx_MCK_SCK_SD_FS_AF** GPIO_AF13_SAI1
#define	**AUDIO_SAIx_MCK_CLK_ENABLE**() __HAL_RCC_GPIOB_
#define	**AUDIO_SAIx_MCK_CLK_DISABLE**() __HAL_RCC_GPIOB_
#define	**AUDIO_SAIx_MCK_PIN** GPIO_PIN_8
#define	**AUDIO_SAIx_MCK_GPIO_PORT** GPIOB
#define	**AUDIO_SAIx_FS_CLK_ENABLE**() __HAL_RCC_GPIOB_CL
#define	**AUDIO_SAIx_FS_CLK_DISABLE**() __HAL_RCC_GPIOB_C
#define	**AUDIO_SAIx_FS_PIN** GPIO_PIN_9
#define	**AUDIO_SAIx_FS_GPIO_PORT** GPIOB
#define	**AUDIO_SAIx_SCK_CLK_ENABLE**() __HAL_RCC_GPIOA_
#define	**AUDIO_SAIx_SCK_CLK_DISABLE**() __HAL_RCC_GPIOA_
#define	**AUDIO_SAIx_SCK_PIN** GPIO_PIN_8
#define	**AUDIO_SAIx_SCK_GPIO_PORT** GPIOA
#define	**AUDIO_SAIx_SD_CLK_ENABLE**() __HAL_RCC_GPIOC_CL
#define	**AUDIO_SAIx_SD_CLK_DISABLE**() __HAL_RCC_GPIOC_C
#define	**AUDIO_SAIx_SD_PIN** GPIO_PIN_1
#define	**AUDIO_SAIx_SD_GPIO_PORT** GPIOC
#define	**AUDIO_SAIx_DMAx_CLK_ENABLE**() __HAL_RCC_DMA2_
#define	**AUDIO_SAIx_DMAx_CLK_DISABLE**() __HAL_RCC_DMA2_
#define	**AUDIO_SAIx_DMAx_REQUEST** DMA_REQUEST_SAI1_A
#define	**AUDIO_SAIx_DMAx_CHANNEL** DMA2_Channel2
#define	**AUDIO_SAIx_DMAx_IRQ** DMA2_Channel2_IRQn
#define	**AUDIO_SAIx_DMAx_PERIPH_DATA_SIZE** DMA_PDATAAL
#define	**AUDIO_SAIx_DMAx_MEM_DATA_SIZE** DMA_MDATAALIGN
#define	**DMA_MAX_SZE** (uint32_t)0xFFFF
#define	**AUDIO_SAIx_DMAx_IRQHandler** DMA2_Channel2_IRQHar
#define	**AUDIO_OUT_IRQ_PREPRIO** 5 /* Select the preemption prior highest) */

```
#define AUDIO_SAIx_PLL_DISABLE()  HAL_RCCEx_DisablePLLSA
#define AUDIO_DFSDMx_LEFT_CHANNEL  DFSDM1_Channel4
#define AUDIO_DFSDMx_RIGHT_CHANNEL  DFSDM1_Channel3
#define AUDIO_DFSDMx_LEFT_FILTER  DFSDM1_Filter0
#define AUDIO_DFSDMx_RIGHT_FILTER  DFSDM1_Filter1
#define AUDIO_DFSDMx_CLK_ENABLE()  __HAL_RCC_DFSDM1_
#define AUDIO_DFSDMx_CLK_DISABLE()  __HAL_RCC_DFSDM1_
#define AUDIO_DFSDMx_CKOUT_DATIN_AF  GPIO_AF6_DFSDM1
#define AUDIO_DFSDMx_CKOUT_CLK_ENABLE()  __HAL_RCC_G
#define AUDIO_DFSDMx_CKOUT_CLK_DISABLE()  __HAL_RCC_C
#define AUDIO_DFSDMx_CKOUT_PIN  GPIO_PIN_10
#define AUDIO_DFSDMx_CKOUT_GPIO_PORT  GPIOF
#define AUDIO_DFSDMx_DATIN_CLK_ENABLE()  __HAL_RCC_GP
#define AUDIO_DFSDMx_DATIN_CLK_DISABLE()  __HAL_RCC_GI
#define AUDIO_DFSDMx_DATIN_PIN  GPIO_PIN_0
#define AUDIO_DFSDMx_DATIN_GPIO_PORT  GPIOC
#define AUDIO_DFSDMx_DMAx_CLK_ENABLE()  __HAL_RCC_DM
#define AUDIO_DFSDMx_DMAx_CLK_DISABLE()  __HAL_RCC_DM
#define AUDIO_DFSDMx_DMAx_LEFT_CHANNEL  DMA1_Channel
#define AUDIO_DFSDMx_DMAx_RIGHT_CHANNEL  DMA1_Channe
#define AUDIO_DFSDMx_DMAx_LEFT_IRQ  DMA1_Channel4_IRQn
#define AUDIO_DFSDMx_DMAx_RIGHT_IRQ  DMA1_Channel5_IRQ
#define AUDIO_DFSDMx_DMAx_LEFT_REQUEST  DMA_REQUEST
#define AUDIO_DFSDMx_DMAx_RIGHT_REQUEST  DMA_REQUES
#define AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE  DMA_PDAT
#define AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE  DMA_MDATAA
#define AUDIO_DFSDM_DMAx_LEFT_IRQHandler  DMA1_Channel
#define AUDIO_DFSDM_DMAx_RIGHT_IRQHandler  DMA1_Channe
#define AUDIO_IN_IRQ_PREPRIO  6 /* Select the preemption priority
#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  BSP_AUDIO_FREQUENCY_1
```

```
#define DEFAULT_AUDIO_IN_BIT_RESOLUTION  16
#define DEFAULT_AUDIO_IN_CHANNEL_NBR  2 /* Mono = 1, Stere
#define DEFAULT_AUDIO_IN_VOLUME  64
#define INTERNAL_BUFF_SIZE  256
#define CODEC_RESET_DELAY  5
```

Define Documentation

#define AUDIO_DFSDM_DMAx_LEFT_IRQHandler DMA1_Channel

Definition at line 181 of file stm32l4r9i_eval_audio.h.

#define AUDIO_DFSDM_DMAx_RIGHT_IRQHandler DMA1_Chann

Definition at line 182 of file stm32l4r9i_eval_audio.h.

#define AUDIO_DFSDMx_CKOUT_CLK_DISABLE ( ) __HAL_RCC

Definition at line 161 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_ChannelMspDeInit().

#define AUDIO_DFSDMx_CKOUT_CLK_ENABLE ( ) __HAL_RCC_

Definition at line 160 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_ChannelMspInit().

#define AUDIO_DFSDMx_CKOUT_DATIN_AF GPIO_AF6_DFSDM1

Definition at line 159 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_ChannelMspInit().

#define AUDIO_DFSDMx_CKOUT_GPIO_PORT GPIOF

Definition at line 163 of file stm32l4r9i_eval_audio.h.

Referenced by **HAL_DFSDM_ChannelMspDeInit()**, and **HAL_DFSDM_ChannelMspInit()**.

#define AUDIO_DFSDMx_CKOUT_PIN GPIO_PIN_10

Definition at line **162** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_DFSDM_ChannelMspDeInit()**, and **HAL_DFSDM_ChannelMspInit()**.

#define AUDIO_DFSDMx_CLK_DISABLE ( ) __HAL_RCC_DFSDM

Definition at line **158** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_DFSDM_ChannelMspDeInit()**.

#define AUDIO_DFSDMx_CLK_ENABLE ( ) __HAL_RCC_DFSDM1

Definition at line **157** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_DFSDM_ChannelMspInit()**.

#define AUDIO_DFSDMx_DATIN_CLK_DISABLE ( ) __HAL_RCC_

Definition at line **165** of file **stm32l4r9i_eval_audio.h**.

#define AUDIO_DFSDMx_DATIN_CLK_ENABLE ( ) __HAL_RCC_

Definition at line **164** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_DFSDM_ChannelMspInit()**.

#define AUDIO_DFSDMx_DATIN_GPIO_PORT GPIOC

Definition at line 167 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_ChannelMspDeInit(), and HAL_DFSDM_ChannelMspInit().

#define AUDIO_DFSDMx_DATIN_PIN GPIO_PIN_0

Definition at line 166 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_ChannelMspDeInit(), and HAL_DFSDM_ChannelMspInit().

#define AUDIO_DFSDMx_DMAx_CLK_DISABLE ( ) __HAL_RCC_I

Definition at line 171 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspDeInit().

#define AUDIO_DFSDMx_DMAx_CLK_ENABLE ( ) __HAL_RCC_D

Definition at line 170 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspInit().

#define AUDIO_DFSDMx_DMAx_LEFT_CHANNEL DMA1_Channel

Definition at line 172 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspInit().

#define AUDIO_DFSDMx_DMAx_LEFT_IRQ DMA1_Channel4_IRQ

Definition at line 174 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspDeInit(), and HAL_DFSDM_FilterMspInit().

#define AUDIO_DFSDMx_DMAx_LEFT_REQUEST DMA_REQUEST

Definition at line 176 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspInit().

#define AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE DMA_MDATAA

Definition at line 179 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspInit().

#define AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE DMA_PDAT

Definition at line 178 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspInit().

#define AUDIO_DFSDMx_DMAx_RIGHT_CHANNEL DMA1_Chann

Definition at line 173 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspInit().

#define AUDIO_DFSDMx_DMAx_RIGHT_IRQ DMA1_Channel5_IRQ

Definition at line **175** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_DFSDM_FilterMspDeInit()**, and **HAL_DFSDM_FilterMspInit()**.

#define AUDIO_DFSDMx_DMAx_RIGHT_REQUEST DMA_REQUES

Definition at line **177** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_DFSDM_FilterMspInit()**.

#define AUDIO_DFSDMx_LEFT_CHANNEL DFSDM1_Channel4

Definition at line **153** of file **stm32l4r9i_eval_audio.h**.

Referenced by **AUDIO_DFSDMx_Init()**.

#define AUDIO_DFSDMx_LEFT_FILTER DFSDM1_Filter0

Definition at line **155** of file **stm32l4r9i_eval_audio.h**.

Referenced by **AUDIO_DFSDMx_Init()**, **HAL_DFSDM_FilterMspDeInit()**, and **HAL_DFSDM_FilterMspInit()**.

#define AUDIO_DFSDMx_RIGHT_CHANNEL DFSDM1_Channel3

Definition at line **154** of file **stm32l4r9i_eval_audio.h**.

Referenced by **AUDIO_DFSDMx_Init()**.

#define AUDIO_DFSDMx_RIGHT_FILTER DFSDM1_Filter1

Definition at line 156 of file stm32l4r9i_eval_audio.h.

Referenced by AUDIO_DFSDMx_Init().

#define AUDIO_ERROR 1

Definition at line 195 of file stm32l4r9i_eval_audio.h.

Referenced by AUDIO_DFSDMx_DeInit(), AUDIO_DFSDMx_Init(), AUDIO_SAIPLLConfig(), AUDIO_SAIx_DeInit(), AUDIO_SAIx_Init(), BSP_AUDIO_IN_DeInit(), BSP_AUDIO_IN_Init(), BSP_AUDIO_IN_Pause(), BSP_AUDIO_IN_Record(), BSP_AUDIO_IN_Resume(), BSP_AUDIO_IN_SetFrequency(), BSP_AUDIO_IN_Stop(), BSP_AUDIO_OUT_ChangeBuffer(), BSP_AUDIO_OUT_DeInit(), BSP_AUDIO_OUT_Init(), BSP_AUDIO_OUT_Pause(), BSP_AUDIO_OUT_Play(), BSP_AUDIO_OUT_Resume(), BSP_AUDIO_OUT_SetFrequency(), BSP_AUDIO_OUT_SetMute(), BSP_AUDIO_OUT_SetOutputMode(), BSP_AUDIO_OUT_SetVolume(), and BSP_AUDIO_OUT_Stop().

#define AUDIO_IN_IRQ_PREPRIO 6 /* Select the preemption prior

Definition at line 185 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_DFSDM_FilterMspInit().

#define AUDIO_OK 0

Definition at line 194 of file stm32l4r9i_eval_audio.h.

Referenced by AUDIO_DFSDMx_DeInit(), AUDIO_DFSDMx_Init(), AUDIO_SAIPLLConfig(), AUDIO_SAIx_DeInit(), AUDIO_SAIx_Init(), BSP_AUDIO_IN_DeInit(),

BSP_AUDIO_IN_Init(), **BSP_AUDIO_IN_Pause()**, **BSP_AUDIO_IN_Record()**, **BSP_AUDIO_IN_Resume()**, **BSP_AUDIO_IN_SetFrequency()**, **BSP_AUDIO_IN_Stop()**, **BSP_AUDIO_OUT_ChangeBuffer()**, **BSP_AUDIO_OUT_DeInit()**, **BSP_AUDIO_OUT_Init()**, **BSP_AUDIO_OUT_Pause()**, **BSP_AUDIO_OUT_Play()**, **BSP_AUDIO_OUT_Resume()**, **BSP_AUDIO_OUT_SetFrequency()**, **BSP_AUDIO_OUT_SetMute()**, **BSP_AUDIO_OUT_SetOutputMode()**, **BSP_AUDIO_OUT_SetVolume()**, and **BSP_AUDIO_OUT_Stop()**.

#define AUDIO_OUT_IRQ_PREPRIO 5 /* Select the preemption pri

Definition at line **143** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_SAI_MspInit()**.

#define AUDIO_SAIx SAI1_Block_A

Definition at line **108** of file **stm32l4r9i_eval_audio.h**.

Referenced by **AUDIO_SAIx_Init()**, and **HAL_SAI_MspInit()**.

#define AUDIO_SAIx_CLK_DISABLE ( ) __HAL_RCC_SAI1_CLK_

Definition at line **110** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_SAI_MspDeInit()**.

#define AUDIO_SAIx_CLK_ENABLE ( ) __HAL_RCC_SAI1_CLK_E

Definition at line **109** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_SAI_MspInit()**.

#define AUDIO_SAIx_DMAx_CHANNEL DMA2_Channel2

Definition at line 134 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspInit().

#define AUDIO_SAIx_DMAx_CLK_DISABLE ( ) __HAL_RCC_DMA

Definition at line 132 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit().

#define AUDIO_SAIx_DMAx_CLK_ENABLE ( ) __HAL_RCC_DMA

Definition at line 131 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspInit().

#define AUDIO_SAIx_DMAx_IRQ DMA2_Channel2_IRQn

Definition at line 135 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit(), and HAL_SAI_MspInit().

#define AUDIO_SAIx_DMAx_IRQHandler DMA2_Channel2_IRQHa

Definition at line 140 of file stm32l4r9i_eval_audio.h.

#define AUDIO_SAIx_DMAx_MEM_DATA_SIZE DMA_MDATAALIG

Definition at line 137 of file stm32l4r9i_eval_audio.h.

Referenced by **HAL_SAI_MspInit()**.

#define AUDIO_SAIx_DMAx_PERIPH_DATA_SIZE DMA_PDATAAL

Definition at line **136** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_SAI_MspInit()**.

#define AUDIO_SAIx_DMAx_REQUEST DMA_REQUEST_SAI1_A

Definition at line **133** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_SAI_MspInit()**.

#define AUDIO_SAIx_FS_CLK_DISABLE ( ) __HAL_RCC_GPIOB_

Definition at line **118** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_SAI_MspDeInit()**.

#define AUDIO_SAIx_FS_CLK_ENABLE ( ) __HAL_RCC_GPIOB_(

Definition at line **117** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_SAI_MspInit()**.

#define AUDIO_SAIx_FS_GPIO_PORT GPIOB

Definition at line **120** of file **stm32l4r9i_eval_audio.h**.

Referenced by **HAL_SAI_MspDeInit()**, and **HAL_SAI_MspInit()**.

#define AUDIO_SAIx_FS_PIN GPIO_PIN_9

Definition at line 119 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit(), and HAL_SAI_MspInit().

#define AUDIO_SAIx_MCK_CLK_DISABLE ( ) __HAL_RCC_GPIO

Definition at line 114 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit().

#define AUDIO_SAIx_MCK_CLK_ENABLE ( ) __HAL_RCC_GPIOE

Definition at line 113 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspInit().

#define AUDIO_SAIx_MCK_GPIO_PORT GPIOB

Definition at line 116 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit(), and HAL_SAI_MspInit().

#define AUDIO_SAIx_MCK_PIN GPIO_PIN_8

Definition at line 115 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit(), and HAL_SAI_MspInit().

#define AUDIO_SAIx_MCK_SCK_SD_FS_AF GPIO_AF13_SAI1

Definition at line 111 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspInit().

#define AUDIO_SAIx_PLL_DISABLE ( ) HAL_RCCEx_DisablePLL

Definition at line 146 of file stm32l4r9i_eval_audio.h.

Referenced by AUDIO_SAIx_DeInit().

#define AUDIO_SAIx_SCK_CLK_DISABLE ( ) __HAL_RCC_GPIOA

Definition at line 122 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit().

#define AUDIO_SAIx_SCK_CLK_ENABLE ( ) __HAL_RCC_GPIOA

Definition at line 121 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspInit().

#define AUDIO_SAIx_SCK_GPIO_PORT GPIOA

Definition at line 124 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit(), and HAL_SAI_MspInit().

#define AUDIO_SAIx_SCK_PIN GPIO_PIN_8

Definition at line 123 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit(), and HAL_SAI_MspInit().

#define AUDIO_SAIx_SD_CLK_DISABLE ( ) __HAL_RCC_GPIOC_

Definition at line 126 of file stm32l4r9i_eval_audio.h.

#define AUDIO_SAIx_SD_CLK_ENABLE ( ) __HAL_RCC_GPIOC_

Definition at line 125 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspInit().

#define AUDIO_SAIx_SD_GPIO_PORT GPIOC

Definition at line 128 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit(), and HAL_SAI_MspInit().

#define AUDIO_SAIx_SD_PIN GPIO_PIN_1

Definition at line 127 of file stm32l4r9i_eval_audio.h.

Referenced by HAL_SAI_MspDeInit(), and HAL_SAI_MspInit().

#define AUDIO_TIMEOUT 2

Definition at line 196 of file stm32l4r9i_eval_audio.h.

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

Definition at line 191 of file stm32l4r9i_eval_audio.h.

#define CODEC_RESET_DELAY 5

Definition at line 212 of file stm32l4r9i_eval_audio.h.

#define DEFAULT_AUDIO_IN_BIT_RESOLUTION 16

Definition at line 200 of file stm32l4r9i_eval_audio.h.

#define DEFAULT_AUDIO_IN_CHANNEL_NBR 2 /* Mono = 1, Stere

Definition at line 201 of file stm32l4r9i_eval_audio.h.

#define DEFAULT_AUDIO_IN_FREQ BSP_AUDIO_FREQUENCY_1

Definition at line 199 of file stm32l4r9i_eval_audio.h.

#define DEFAULT_AUDIO_IN_VOLUME 64

Definition at line 202 of file stm32l4r9i_eval_audio.h.

#define DMA_MAX_SZE (uint32_t)0xFFFF

Definition at line 138 of file stm32l4r9i_eval_audio.h.

#define INTERNAL_BUFF_SIZE 256

Definition at line 205 of file stm32l4r9i_eval_audio.h.

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Defines

BUS Constants

Exported Constants

Defines

#define	**EVAL_I2Cx_SCL_PIN** GPIO_PIN_4
#define	**EVAL_I2Cx_SCL_GPIO_PORT** GPIOH
#define	**EVAL_I2Cx_SDA_PIN** GPIO_PIN_5
#define	**EVAL_I2Cx_SDA_GPIO_PORT** GPIOH
#define	**EVAL_I2Cx_SCL_SDA_AF** GPIO_AF4_I2C2
#define	**EVAL_I2Cx** I2C2
#define	**EVAL_I2Cx_CLK_ENABLE**() __HAL_RCC_I2C2_CLK_ENA
#define	**EVAL_I2Cx_CLK_DISABLE**() __HAL_RCC_I2C2_CLK_DIS
#define	**EVAL_I2Cx_SDA_GPIO_CLK_ENABLE**() __HAL_RCC_GPI
#define	**EVAL_I2Cx_SDA_GPIO_CLK_DISABLE**() __HAL_RCC_GP
#define	**EVAL_I2Cx_SCL_GPIO_CLK_ENABLE**() __HAL_RCC_GPI
#define	**EVAL_I2Cx_SCL_GPIO_CLK_DISABLE**() __HAL_RCC_GP
#define	**EVAL_I2Cx_FORCE_RESET**() __HAL_RCC_I2C2_FORCE_
#define	**EVAL_I2Cx_RELEASE_RESET**() __HAL_RCC_I2C2_RELE
#define	**EVAL_I2Cx_EV_IRQn** I2C2_EV_IRQn
#define	**EVAL_I2Cx_EV_IRQHandler** I2C2_EV_IRQHandler
#define	**EVAL_I2Cx_ER_IRQn** I2C2_ER_IRQn
#define	**EVAL_I2Cx_ER_IRQHandler** I2C2_ER_IRQHandler
#define	**EVAL_RCC_PERIPHCLK_I2C** RCC_PERIPHCLK_I2C2
#define	**EVAL_RCC_CLKSOURCE_I2C** RCC_I2C2CLKSOURCE_SY
#define	**EVAL_TS_I2Cx** I2C2
#define	**EVAL_TS_I2Cx_CLK_ENABLE**() __HAL_RCC_I2C2_CLK_E
#define	**EVAL_TS_DMAx_CLK_ENABLE**() __HAL_RCC_DMA1_CLI
#define	**EVAL_TS_I2Cx_SDA_GPIO_CLK_ENABLE**() __HAL_RCC_
#define	**EVAL_TS_I2Cx_SDA_GPIO_CLK_DISABLE**() __HAL_RCC
#define	**EVAL_TS_I2Cx_SCL_GPIO_CLK_ENABLE**() __HAL_RCC_
#define	**EVAL_TS_I2Cx_SCL_GPIO_CLK_DISABLE**() __HAL_RCC_
#define	**EVAL_TS_I2Cx_FORCE_RESET**() __HAL_RCC_I2C2_FOR
#define	**EVAL_TS_I2Cx_RELEASE_RESET**() __HAL_RCC_I2C2_RE
#define	**EVAL_TS_I2Cx_SCL_PIN** GPIO_PIN_4
#define	**EVAL_TS_I2Cx_SCL_GPIO_PORT** GPIOH

#define **EVAL_TS_I2Cx_SDA_PIN** GPIO_PIN_5

#define **EVAL_TS_I2Cx_SDA_GPIO_PORT** GPIOH

#define **EVAL_TS_I2Cx_SCL_SDA_AF** GPIO_AF4_I2C2

#define **EVAL_TS_I2Cx_EV_IRQn** I2C2_EV_IRQn

#define **EVAL_TS_I2Cx_ER_IRQn** I2C2_ER_IRQn

#define **EVAL_TS_RCC_PERIPHCLK_I2C** RCC_PERIPHCLK_I2C2

#define **EVAL_TS_RCC_CLKSOURCE_I2C** RCC_I2C2CLKSOURCE

#define **EEPROM_I2C_ADDRESS** 0xA0

#define **IO_I2C_ADDRESS** 0x84

#define **IDD_I2C_ADDRESS** **IO_I2C_ADDRESS**

#define **AUDIO_I2C_ADDRESS** 0x34

#define **LCD_RGB_I2C_ADDRESS** 0x70

#define **TS_RGB_I2C_ADDRESS** 0x70

#define **TS_DSI_I2C_ADDRESS** 0x70

#define **EVAL_I2Cx_TIMEOUT_MAX** 3000

#define **AUDIO_INT_GPIO_PORT** GPIOI /* GPIOI */

#define **AUDIO_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOI_CL

#define **AUDIO_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOI_C

#define **AUDIO_INT_PIN** GPIO_PIN_6 /* PI.06 */

#define **MFX_INT_GPIO_PORT** GPIOA /* GPIOA */

MFX interface pins.

#define **MFX_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOA_CLK

#define **MFX_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOA_CL

#define **MFX_INT_PIN** GPIO_PIN_0 /* PA.00 */

#define **MFX_INT_EXTI_IRQn** EXTI0_IRQn

#define **MFX_WAKEUP_GPIO_PORT** GPIOG /* GPIOG */

#define **MFX_WAKEUP_GPIO_CLK_ENABLE**() __HAL_RCC_GPIO

#define **MFX_WAKEUP_GPIO_CLK_DISABLE**() __HAL_RCC_GPIC

#define **MFX_WAKEUP_PIN** GPIO_PIN_9 /* PG.09 */

#define **IDD_INT_GPIO_PORT** **MFX_INT_GPIO_PORT**

#define **IDD_INT_GPIO_CLK_ENABLE**() **MFX_INT_GPIO_CLK_EN**

#define **IDD_INT_GPIO_CLK_DISABLE**() **MFX_INT_GPIO_CLK_DI**

#define **IDD_INT_PIN** **MFX_INT_PIN**

#define **IDD_INT_EXTI_IRQn** **MFX_INT_EXTI_IRQn**

#define **IDD_WAKEUP_GPIO_PORT** **MFX_WAKEUP_GPIO_PORT**

#define **IDD_WAKEUP_GPIO_CLK_ENABLE**() **MFX_WAKEUP_GPI**

#define **IDD_WAKEUP_GPIO_CLK_DISABLE**() **MFX_WAKEUP_GP**

#define **IDD_WAKEUP_PIN** **MFX_WAKEUP_PIN**

#define **IDD_AMP_CONTROL_PIN** **AGPIO_PIN_1**

Idd current measurement interface pins on MFX.

#define **LCD_DSI_TE_PIN** GPIO_PIN_6

MB1314 DSI.

#define **LCD_DSI_TE_CTRL_GPIO_PORT** GPIOG

#define **LCD_DSI_TE_CTRL_GPIO_CLK_ENABLE**() __HAL_RCC_(

#define **LCD_DSI_TE_CTRL_GPIO_CLK_DISABLE**() __HAL_RCC_

#define **LCD_DSI_BL_CTRL_PIN** GPIO_PIN_14

#define **LCD_DSI_BL_CTRL_GPIO_PORT** GPIOB

#define **LCD_DSI_BL_CTRL_GPIO_CLK_ENABLE**() __HAL_RCC_(

#define **LCD_DSI_BL_CTRL_GPIO_CLK_DISABLE**() __HAL_RCC_

#define **TS_DSI_INT_PIN** GPIO_PIN_2

#define **TS_DSI_INT_GPIO_PORT** GPIOC

#define **TS_DSI_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOC_(

#define **TS_DSI_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOC_

#define **TS_DSI_INT_EXTI_IRQn** EXTI2_IRQn

#define **LCD_RGB_DISP_PIN** **IO_PIN_11** /* MFX_GPIO11 */

MB1315 RGB.

#define **LCD_RGB_RST_PIN** **IO_PIN_12** /* MFX_GPIO12 */

#define **LCD_RGB_BL_CTRL_PIN** GPIO_PIN_5

#define **LCD_RGB_BL_CTRL_GPIO_PORT** GPIOA

#define **LCD_RGB_BL_CTRL_GPIO_CLK_ENABLE**() __HAL_RCC_

#define **LCD_RGB_BL_CTRL_GPIO_CLK_DISABLE**() __HAL_RCC

#define **TS_RGB_INT_PIN** GPIO_PIN_2

#define **TS_RGB_INT_GPIO_PORT** GPIOC

#define **TS_RGB_INT_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOC_

#define **TS_RGB_INT_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOC

#define **TS_RGB_INT_EXTI_IRQn** EXTI2_IRQn

Define Documentation

#define AUDIO_I2C_ADDRESS 0x34

Definition at line 466 of file stm32l4r9i_eval.h.

Referenced by AUDIO_CODEC_Reset(), 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_INT_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOI_

Definition at line 483 of file stm32l4r9i_eval.h.

#define AUDIO_INT_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOI_

Definition at line 482 of file stm32l4r9i_eval.h.

#define AUDIO_INT_GPIO_PORT GPIOI /* GPIOI */

Definition at line 481 of file stm32l4r9i_eval.h.

#define AUDIO_INT_PIN GPIO_PIN_6 /* PI.06 */

Definition at line 484 of file stm32l4r9i_eval.h.

#define EEPROM_I2C_ADDRESS 0xA0

Definition at line **453** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_EEPROM_Init()**.

#define EVAL_I2Cx I2C2

Definition at line **394** of file **stm32l4r9i_eval.h**.

Referenced by **I2Cx_Init()**, and **I2Cx_MspInit()**.

#define EVAL_I2Cx_CLK_DISABLE ( ) __HAL_RCC_I2C2_CLK_DI

Definition at line **396** of file **stm32l4r9i_eval.h**.

#define EVAL_I2Cx_CLK_ENABLE ( ) __HAL_RCC_I2C2_CLK_EN

Definition at line **395** of file **stm32l4r9i_eval.h**.

Referenced by **I2Cx_MspInit()**.

#define EVAL_I2Cx_ER_IRQHandler I2C2_ER_IRQHandler

Definition at line **409** of file **stm32l4r9i_eval.h**.

#define EVAL_I2Cx_ER_IRQn I2C2_ER_IRQn

Definition at line **408** of file **stm32l4r9i_eval.h**.

Referenced by **I2Cx_MspInit()**.

#define EVAL_I2Cx_EV_IRQHandler I2C2_EV_IRQHandler

Definition at line 407 of file stm32l4r9i_eval.h.

#define EVAL_I2Cx_EV_IRQn I2C2_EV_IRQn

Definition at line 406 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_FORCE_RESET ( ) __HAL_RCC_I2C2_FORCE

Definition at line 402 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_RELEASE_RESET ( ) __HAL_RCC_I2C2_REL

Definition at line 403 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_SCL_GPIO_CLK_DISABLE ( ) __HAL_RCC_G

Definition at line 400 of file stm32l4r9i_eval.h.

#define EVAL_I2Cx_SCL_GPIO_CLK_ENABLE ( ) __HAL_RCC_GI

Definition at line 399 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_SCL_GPIO_PORT GPIOH

Definition at line 388 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_SCL_PIN GPIO_PIN_4

Definition at line 387 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_SCL_SDA_AF GPIO_AF4_I2C2

Definition at line 391 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_SDA_GPIO_CLK_DISABLE ( ) __HAL_RCC_G

Definition at line 398 of file stm32l4r9i_eval.h.

#define EVAL_I2Cx_SDA_GPIO_CLK_ENABLE ( ) __HAL_RCC_G

Definition at line 397 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_SDA_GPIO_PORT GPIOH

Definition at line 390 of file stm32l4r9i_eval.h.

#define EVAL_I2Cx_SDA_PIN GPIO_PIN_5

Definition at line 389 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_I2Cx_TIMEOUT_MAX 3000

Definition at line 476 of file stm32l4r9i_eval.h.

#define EVAL_RCC_CLKSOURCE_I2C RCC_I2C2CLKSOURCE_SY

Definition at line 413 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_RCC_PERIPHCLK_I2C RCC_PERIPHCLK_I2C2

Definition at line 412 of file stm32l4r9i_eval.h.

Referenced by I2Cx_MspInit().

#define EVAL_TS_DMAx_CLK_ENABLE ( ) __HAL_RCC_DMA1_C

Definition at line 426 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx I2C2

Definition at line 424 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_CLK_ENABLE ( ) __HAL_RCC_I2C2_CLK

Definition at line 425 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_ER_IRQn I2C2_ER_IRQn

Definition at line 444 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_EV_IRQn I2C2_EV_IRQn

Definition at line 443 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_FORCE_RESET ( ) __HAL_RCC_I2C2_FO

Definition at line 432 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_RELEASE_RESET ( ) __HAL_RCC_I2C2_

Definition at line 433 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SCL_GPIO_CLK_DISABLE ( ) __HAL_RC

Definition at line 430 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SCL_GPIO_CLK_ENABLE ( ) __HAL_RCC

Definition at line 429 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SCL_GPIO_PORT GPIOH

Definition at line 437 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SCL_PIN GPIO_PIN_4

Definition at line 436 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SCL_SDA_AF GPIO_AF4_I2C2

Definition at line 440 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SDA_GPIO_CLK_DISABLE ( ) __HAL_RC

Definition at line 428 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SDA_GPIO_CLK_ENABLE ( ) __HAL_RC(

Definition at line 427 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SDA_GPIO_PORT GPIOH

Definition at line 439 of file stm32l4r9i_eval.h.

#define EVAL_TS_I2Cx_SDA_PIN GPIO_PIN_5

Definition at line 438 of file stm32l4r9i_eval.h.

#define EVAL_TS_RCC_CLKSOURCE_I2C RCC_I2C2CLKSOURCE

Definition at line 448 of file stm32l4r9i_eval.h.

#define EVAL_TS_RCC_PERIPHCLK_I2C RCC_PERIPHCLK_I2C2

Definition at line 447 of file stm32l4r9i_eval.h.

#define IDD_AMP_CONTROL_PIN AGPIO_PIN_1

Idd current measurement interface pins on MFX.

Definition at line 516 of file stm32l4r9i_eval.h.

Referenced by BSP_IDD_GetValue(), and BSP_IDD_StartMeasure().

#define IDD_I2C_ADDRESS IO_I2C_ADDRESS

Definition at line 465 of file stm32l4r9i_eval.h.

Referenced by BSP_IDD_ClearIT(), BSP_IDD_Config(), BSP_IDD_DeInit(), BSP_IDD_DisableIT(), BSP_IDD_EnableIT(), BSP_IDD_ErrorClearIT(), BSP_IDD_ErrorDisableIT(), BSP_IDD_ErrorEnableIT(), BSP_IDD_ErrorGetCode(), BSP_IDD_ErrorGetITStatus(), BSP_IDD_GetITStatus(), BSP_IDD_GetValue(), BSP_IDD_Init(), BSP_IDD_LowPower(), BSP_IDD_Reset(), BSP_IDD_StartMeasure(), and BSP_IDD_WakeUp().

#define IDD_INT_EXTI_IRQn MFX_INT_EXTI_IRQn

Definition at line 507 of file stm32l4r9i_eval.h.

#define IDD_INT_GPIO_CLK_DISABLE ( ) MFX_INT_GPIO_CLK_D

Definition at line **505** of file **stm32l4r9i_eval.h**.

#define IDD_INT_GPIO_CLK_ENABLE ( ) MFX_INT_GPIO_CLK_E

Definition at line **504** of file **stm32l4r9i_eval.h**.

#define IDD_INT_GPIO_PORT MFX_INT_GPIO_PORT

Definition at line **503** of file **stm32l4r9i_eval.h**.

#define IDD_INT_PIN MFX_INT_PIN

Definition at line **506** of file **stm32l4r9i_eval.h**.

#define IDD_WAKEUP_GPIO_CLK_DISABLE ( ) MFX_WAKEUP_G

Definition at line **510** of file **stm32l4r9i_eval.h**.

#define IDD_WAKEUP_GPIO_CLK_ENABLE ( ) MFX_WAKEUP_G

Definition at line **509** of file **stm32l4r9i_eval.h**.

#define IDD_WAKEUP_GPIO_PORT MFX_WAKEUP_GPIO_PORT

Definition at line **508** of file **stm32l4r9i_eval.h**.

#define IDD_WAKEUP_PIN MFX_WAKEUP_PIN

Definition at line **511** of file **stm32l4r9i_eval.h**.

#define IO_I2C_ADDRESS 0x84

Definition at line 464 of file stm32l4r9i_eval.h.

Referenced by BSP_IO_ConfigPin(), BSP_IO_DeInit(), BSP_IO_Init(), BSP_IO_ITClear(), BSP_IO_ITGetStatus(), BSP_IO_ReadPin(), BSP_IO_TogglePin(), and BSP_IO_WritePin().

#define LCD_DSI_BL_CTRL_GPIO_CLK_DISABLE ( ) __HAL_RCC

Definition at line 532 of file stm32l4r9i_eval.h.

#define LCD_DSI_BL_CTRL_GPIO_CLK_ENABLE ( ) __HAL_RCC

Definition at line 531 of file stm32l4r9i_eval.h.

#define LCD_DSI_BL_CTRL_GPIO_PORT GPIOB

Definition at line 530 of file stm32l4r9i_eval.h.

#define LCD_DSI_BL_CTRL_PIN GPIO_PIN_14

Definition at line 529 of file stm32l4r9i_eval.h.

#define LCD_DSI_TE_CTRL_GPIO_CLK_DISABLE ( ) __HAL_RCC

Definition at line 526 of file stm32l4r9i_eval.h.

#define LCD_DSI_TE_CTRL_GPIO_CLK_ENABLE ( ) __HAL_RCC

Definition at line **525** of file **stm32l4r9i_eval.h**.

#define LCD_DSI_TE_CTRL_GPIO_PORT GPIOG

Definition at line **524** of file **stm32l4r9i_eval.h**.

#define LCD_DSI_TE_PIN GPIO_PIN_6

MB1314 DSI.

Definition at line **523** of file **stm32l4r9i_eval.h**.

#define LCD_RGB_BL_CTRL_GPIO_CLK_DISABLE ( ) __HAL_RC

Definition at line **554** of file **stm32l4r9i_eval.h**.

#define LCD_RGB_BL_CTRL_GPIO_CLK_ENABLE ( ) __HAL_RC

Definition at line **553** of file **stm32l4r9i_eval.h**.

#define LCD_RGB_BL_CTRL_GPIO_PORT GPIOA

Definition at line **552** of file **stm32l4r9i_eval.h**.

#define LCD_RGB_BL_CTRL_PIN GPIO_PIN_5

Definition at line **551** of file **stm32l4r9i_eval.h**.

#define LCD_RGB_DISP_PIN IO_PIN_11 /* MFX_GPIO11 */

MB1315 RGB.

Definition at line **545** of file **stm32l4r9i_eval.h**.

#define LCD_RGB_I2C_ADDRESS 0x70

Definition at line **467** of file **stm32l4r9i_eval.h**.

#define LCD_RGB_RST_PIN IO_PIN_12 /* MFX_GPIO12 */

Definition at line **548** of file **stm32l4r9i_eval.h**.

#define MFX_INT_EXTI_IRQn EXTI0_IRQn

Definition at line **495** of file **stm32l4r9i_eval.h**.

Referenced by **MFX_IO_DeInit()**, and **MFX_IO_ITConfig()**.

#define MFX_INT_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOA_C

Definition at line **493** of file **stm32l4r9i_eval.h**.

#define MFX_INT_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOA_C

Definition at line **492** of file **stm32l4r9i_eval.h**.

Referenced by **MFX_IO_DeInit()**, and **MFX_IO_ITConfig()**.

#define MFX_INT_GPIO_PORT GPIOA /* GPIOA */

MFX interface pins.

Definition at line 491 of file stm32l4r9i_eval.h.

Referenced by MFX_IO_DeInit(), and MFX_IO_ITConfig().

#define MFX_INT_PIN GPIO_PIN_0 /* PA.00 */

Definition at line 494 of file stm32l4r9i_eval.h.

Referenced by MFX_IO_DeInit(), and MFX_IO_ITConfig().

#define MFX_WAKEUP_GPIO_CLK_DISABLE ( ) __HAL_RCC_GP

Definition at line 499 of file stm32l4r9i_eval.h.

#define MFX_WAKEUP_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPI

Definition at line 498 of file stm32l4r9i_eval.h.

Referenced by MFX_IO_DeInit(), and MFX_IO_EnableWakeupPin().

#define MFX_WAKEUP_GPIO_PORT GPIOG /* GPIOG */

Definition at line 497 of file stm32l4r9i_eval.h.

Referenced by MFX_IO_DeInit(), MFX_IO_EnableWakeupPin(), and MFX_IO_Wakeup().

#define MFX_WAKEUP_PIN GPIO_PIN_9 /* PG.09 */

Definition at line 500 of file stm32l4r9i_eval.h.

Referenced by **MFX_IO_DeInit()**, **MFX_IO_EnableWakeupPin()**, and **MFX_IO_Wakeup()**.

#define TS_DSI_I2C_ADDRESS 0x70

Definition at line **469** of file **stm32l4r9i_eval.h**.

#define TS_DSI_INT_EXTI_IRQn EXTI2_IRQn

Definition at line **539** of file **stm32l4r9i_eval.h**.

#define TS_DSI_INT_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOC

Definition at line **538** of file **stm32l4r9i_eval.h**.

#define TS_DSI_INT_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOC

Definition at line **537** of file **stm32l4r9i_eval.h**.

#define TS_DSI_INT_GPIO_PORT GPIOC

Definition at line **536** of file **stm32l4r9i_eval.h**.

#define TS_DSI_INT_PIN GPIO_PIN_2

Definition at line **535** of file **stm32l4r9i_eval.h**.

#define TS_RGB_I2C_ADDRESS 0x70

Definition at line **468** of file **stm32l4r9i_eval.h**.

#define TS_RGB_INT_EXTI_IRQn EXTI2_IRQn

Definition at line **561** of file **stm32l4r9i_eval.h**.

#define TS_RGB_INT_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIC

Definition at line **560** of file **stm32l4r9i_eval.h**.

#define TS_RGB_INT_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIO

Definition at line **559** of file **stm32l4r9i_eval.h**.

#define TS_RGB_INT_GPIO_PORT GPIOC

Definition at line **558** of file **stm32l4r9i_eval.h**.

#define TS_RGB_INT_PIN GPIO_PIN_2

Definition at line **557** of file **stm32l4r9i_eval.h**.

STM32L4R9I_EVAL BSP User Manual

Functions

Bus Operations Functions

STM32L4R9I_EVAL Common

Functions

static void	**I2Cx_MspInit** (I2C_HandleTypeDef *hi2c) Eval I2Cx MSP Initialization.
static void	**I2Cx_Init** (uint32_t user) Eval I2Cx Bus initialization.
static void	**I2Cx_MspDeInit** (I2C_HandleTypeDef *hi2c) Eval I2Cx MSP Deinitialization.
static void	**I2Cx_DeInit** (uint32_t user) Eval I2Cx Bus deinitialization.
static HAL_StatusTypeDef	**I2Cx_WriteBuffer** (uint16_t Addr, uint16_t Reg, uint16_t RegSize, uint8_t *pBuffer, uint16_t Length) Write a value in a register of the device through BUS.
static void	**I2Cx_WriteData** (uint16_t Addr, uint16_t Reg, uint16_t RegSize, uint8_t Value) Write a value in a register of the device through BUS.
static void	**I2Cx_WriteMultiple** (uint16_t Addr, uint16_t Reg, uint16_t MemAddress, uint8_t *Buffer, uint16_t Length) Write multiple data value in a register of the device through BUS.
static uint8_t	**I2Cx_ReadData** (uint16_t Addr, uint16_t Reg, uint16_t RegSize) Read a register of the device through BUS.
static uint8_t	**I2Cx_ReadMultiple** (uint16_t Addr, uint16_t Reg, uint16_t MemAddress, uint8_t *Buffer, uint16_t Length) Read multiple data from a register of the device through BUS.

static HAL_StatusTypeDef	**I2Cx_ReadBuffer** (uint16_t Addr, uint16_t Reg, uint16_t RegSize, uint8_t *pBuffer, uint16_t Length) Read multiple data on the BUS.
static HAL_StatusTypeDef	**I2Cx_IsDeviceReady** (uint16_t DevAddress, uint32_t Trials) Checks if target device is ready for communication.
static void	**I2Cx_Error** (void) Eval I2Cx error treatment function.
void	**MFX_IO_Init** (void) Initialize MFX_IO low level.
void	**MFX_IO_ITConfig** (void) Configure MFX_IO low level Interrupt.
void	**MFX_IO_Write** (uint16_t Addr, uint8_t Reg, uint8_t Value) MFX_IO writes single data.
uint8_t	**MFX_IO_Read** (uint16_t Addr, uint8_t Reg) MFX_IO reads single data.
uint16_t	**MFX_IO_ReadMultiple** (uint16_t Addr, uint8_t Reg, uint8_t *Buffer, uint16_t Length) MFX_IO reads multiple data.
void	**MFX_IO_Delay** (uint32_t Delay) MFX_IO delay.
void	**MFX_IO_Wakeup** (void) MFX_IO wakeup.
void	**MFX_IO_EnableWakeupPin** (void) MFX_IO enable wakeup pin.
void	**MFX_IO_DeInit** (void) DeInitializes MFX_IO low level.
void	**TS_IO_Init** (void) Initializes Touchscreen low level.

void	**TS_IO_Write** (uint8_t Addr, uint8_t Reg, uint8_t Value) Writes a single data.
uint8_t	**TS_IO_Read** (uint8_t Addr, uint8_t Reg) Reads a single data.
uint16_t	**TS_IO_ReadMultiple** (uint8_t Addr, uint8_t Reg, uint8_t *Buffer, uint16_t Length) Reads multiple data with I2C communication channel from TouchScreen.
void	**TS_IO_Delay** (uint32_t Delay) TS delay.
void	**AUDIO_IO_Init** (void) Initialize Audio low level.
void	**AUDIO_IO_DeInit** (void) Deinitialize Audio low level.
void	**AUDIO_IO_Write** (uint8_t Addr, uint16_t Reg, uint16_t Value) Writes a single data.
uint16_t	**AUDIO_IO_Read** (uint8_t Addr, uint16_t Reg) Reads a single data.
void	**AUDIO_IO_Delay** (uint32_t Delay) AUDIO Codec delay.
void	**EEPROM_IO_Init** (void) Initialize peripherals used by the I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_WriteData** (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize) Write data to I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_ReadData** (uint16_t DevAddress, uint16_t MemAddress,

	uint8_t *pBuffer, uint32_t BufferSize) Read data from I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_IsDeviceReady** (uint16_t DevAddress, uint32_t Trials) Checks if target device is ready for communication.

Function Documentation

void AUDIO_IO_DeInit (void)

Deinitialize Audio low level.

Return values:
None

Definition at line 1308 of file stm32l4r9i_eval.c.

References BSP_I2C_AUDIO_USER, and I2Cx_DeInit().

void AUDIO_IO_Delay (uint32_t Delay)

AUDIO Codec delay.

Parameters:
Delay,: Delay in ms

Return values:
None

Definition at line 1355 of file stm32l4r9i_eval.c.

void AUDIO_IO_Init (void)

Initialize Audio low level.

Return values:
None

Definition at line 1299 of file stm32l4r9i_eval.c.

References **BSP_I2C_AUDIO_USER**, and **I2Cx_Init()**.

uint16_t AUDIO_IO_Read (uint8_t Addr,
uint16_t Reg
)

Reads a single data.

Parameters:

Addr,: I2C address
Reg,: Reg address

Return values:

Data to be read

Definition at line **1336** of file **stm32l4r9i_eval.c**.

References **I2Cx_ReadBuffer()**.

void AUDIO_IO_Write (uint8_t Addr,
uint16_t Reg,
uint16_t Value
)

Writes a single data.

Parameters:

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

Return values:

None

Definition at line **1320** of file **stm32l4r9i_eval.c**.

References **I2Cx_WriteBuffer()**.

void EEPROM_IO_Init (void)

Initialize peripherals used by the I2C EEPROM driver.

Return values:
None

Definition at line **1365** of file **stm32l4r9i_eval.c**.

References **BSP_I2C_EEPROM_USER**, and **I2Cx_Init()**.

Referenced by **BSP_EEPROM_Init()**.

HAL_StatusTypeDef EEPROM_IO_IsDeviceReady (uint16_t DevAd
uint32_t Trials
)

Checks if target device is ready for communication.

Note:
This function is used with Memory devices

Parameters:
DevAddress,: Target device address
Trials,: Number of trials

Return values:
HAL status

Definition at line **1403** of file **stm32l4r9i_eval.c**.

References **I2Cx_IsDeviceReady()**.

Referenced by **BSP_EEPROM_Init()**, and **BSP_EEPROM_WaitEepromStandbyState()**.

HAL_StatusTypeDef EEPROM_IO_ReadData (
uint16_t DevAddress,
uint16_t MemAddress,
uint8_t * pBuffer,
uint32_t BufferSize
)

Read data from I2C EEPROM driver.

Parameters:

DevAddress,:	Target device address
MemAddress,:	Internal memory address
pBuffer,:	Pointer to data buffer
BufferSize,:	Amount of data to be read

Return values:

HAL status

Definition at line **1391** of file **stm32l4r9i_eval.c**.

References **I2Cx_ReadBuffer()**.

Referenced by **BSP_EEPROM_ReadBuffer()**.

HAL_StatusTypeDef EEPROM_IO_WriteData (
uint16_t DevAddress,
uint16_t MemAddress,
uint8_t * pBuffer,
uint32_t BufferSize
)

Write data to I2C EEPROM driver.

Parameters:

DevAddress,:	Target device address
MemAddress,:	Internal memory address
pBuffer,:	Pointer to data buffer
BufferSize,:	Amount of data to be sent

Return values:

HAL status

Definition at line **1378** of file **stm32l4r9i_eval.c**.

References **I2Cx_WriteBuffer()**.

Referenced by **BSP_EEPROM_WritePage()**.

static void I2Cx_DeInit (uint32_t user) [static]

Eval I2Cx Bus deinitialization.

Return values:

None

Definition at line **820** of file **stm32l4r9i_eval.c**.

References **BSP_I2C_NO_USER**, and **I2Cx_MspDeInit()**.

Referenced by **AUDIO_IO_DeInit()**, **I2Cx_Error()**, and **MFX_IO_DeInit()**.

static void I2Cx_Error (void) [static]

Eval I2Cx error treatment function.

Return values:
None

Definition at line 1072 of file stm32l4r9i_eval.c.

References BSP_ErrorHandler(), BSP_I2C_ALL_USERS, I2Cx_DeInit(), I2Cx_Init(), and v_bspI2cUsers.

Referenced by I2Cx_ReadBuffer(), I2Cx_ReadData(), I2Cx_ReadMultiple(), I2Cx_WriteBuffer(), I2Cx_WriteData(), and I2Cx_WriteMultiple().

static void I2Cx_Init (uint32_t user) [static]

Eval I2Cx Bus initialization.

Return values:
None

Definition at line 779 of file stm32l4r9i_eval.c.

References EVAL_I2Cx, and I2Cx_MspInit().

Referenced by AUDIO_IO_Init(), EEPROM_IO_Init(), I2Cx_Error(), MFX_IO_Init(), and TS_IO_Init().

static HAL_StatusTypeDef I2Cx_IsDeviceReady (uint16_t DevAddr, uint32_t Trials) [static]

Checks if target device is ready for communication.

Note:
This function is used with Memory devices

Parameters:

DevAddress,: Target device address
Trials,: Number of trials

Return values:
HAL status

Definition at line **1048** of file **stm32l4r9i_eval.c**.

Referenced by **EEPROM_IO_IsDeviceReady()**.

static void I2Cx_MspDeInit (I2C_HandleTypeDef * hi2c) [static]

Eval I2Cx MSP Deinitialization.

Parameters:
hi2c,: I2C handle

Return values:
None

Definition at line **812** of file **stm32l4r9i_eval.c**.

Referenced by **I2Cx_DeInit()**.

static void I2Cx_MspInit (I2C_HandleTypeDef * hi2c) [static]

Eval I2Cx MSP Initialization.

Parameters:
hi2c,: I2C handle

Return values:
None

Definition at line **717** of file **stm32l4r9i_eval.c**.

References **EVAL_I2Cx**, **EVAL_I2Cx_CLK_ENABLE**, **EVAL_I2Cx_ER_IRQn**, **EVAL_I2Cx_EV_IRQn**, **EVAL_I2Cx_FORCE_RESET**, **EVAL_I2Cx_RELEASE_RESET**, **EVAL_I2Cx_SCL_GPIO_CLK_ENABLE**, **EVAL_I2Cx_SCL_GPIO_PORT**, **EVAL_I2Cx_SCL_PIN**, **EVAL_I2Cx_SCL_SDA_AF**, **EVAL_I2Cx_SDA_GPIO_CLK_ENABLE**, **EVAL_I2Cx_SDA_PIN**, **EVAL_RCC_CLKSOURCE_I2C**, and **EVAL_RCC_PERIPHCLK_I2C**.

Referenced by **I2Cx_Init()**.

static HAL_StatusTypeDef I2Cx_ReadBuffer (uint16_t Addr,
uint16_t Reg,
uint16_t RegSize,
uint8_t * pBuffer,
uint16_t Length
) [static]

Read multiple data on the BUS.

Parameters:

Addr,: I2C Address
Reg,: Reg Address
RegSize : The target register size (can be 8BIT or 16BIT)
pBuffer,: pointer to read data buffer
Length,: length of the data

Return values:

0 if no problems to read multiple data

Definition at line **1016** of file **stm32l4r9i_eval.c**.

References **I2Cx_Error()**.

Referenced by **AUDIO_IO_Read()**, **EEPROM_IO_ReadData()**, and

MFX_IO_ReadMultiple().

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

Read a register of the device through BUS.

Parameters:

Addr,:	Device address on BUS
Reg,:	The target register address to read
RegSize,:	The target register size (can be 8BIT or 16BIT)

Return values:

read	register value

Definition at line 944 of file stm32l4r9i_eval.c.

References I2Cx_Error().

Referenced by MFX_IO_Read().

```
static uint8_t I2Cx_ReadMultiple ( uint16_t  Addr,
                                   uint16_t  Reg,
                                   uint16_t  MemAddress,
                                   uint8_t *  Buffer,
                                   uint16_t  Length
                                 )           [static]
```

Read multiple data from a register of the device through BUS.

Parameters:

Addr,:	Device address on BUS

Reg,: The target register address to read
MemAddress,: Memory address
Buffer,: Pointer to data buffer
Length,: Length of the data

Return values:
Number of read data

Definition at line **980** of file **stm32l4r9i_eval.c**.

References **I2Cx_Error()**.

Referenced by **TS_IO_Read()**, and **TS_IO_ReadMultiple()**.

static HAL_StatusTypeDef I2Cx_WriteBuffer (uint16_t Addr,
uint16_t Reg,
uint16_t RegSize,
uint8_t * pBuffer,
uint16_t Length
) [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
RegSize,: The target register size (can be 8BIT or 16BIT)
pBuffer,: The target register value to be written
Length,: buffer size to be written

Return values:
None

Definition at line **847** of file **stm32l4r9i_eval.c**.

References **I2Cx_Error()**.

Referenced by **AUDIO_IO_Write()**, and **EEPROM_IO_WriteData()**.

static void I2Cx_WriteData (uint16_t Addr,
uint16_t Reg,
uint16_t RegSize,
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
RegSize,:	The target register size (can be 8BIT or 16BIT)
Value,:	The target register value to be written

Return values:

None

Definition at line **880** of file **stm32l4r9i_eval.c**.

References **I2Cx_Error()**.

Referenced by **MFX_IO_Write()**.

static void I2Cx_WriteMultiple (uint16_t Addr,
uint16_t Reg,
uint16_t MemAddress,
uint8_t * Buffer,
uint16_t Length
) [static]

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

Parameters:

Addr,:	Device address on BUS Bus.
Reg,:	The target register address to write
MemAddress,:	Memory address
Buffer,:	The target register value to be written
Length,:	buffer size to be written

Return values:

None

Definition at line **913** of file **stm32l4r9i_eval.c**.

References **I2Cx_Error()**.

Referenced by **TS_IO_Write()**.

void MFX_IO_DeInit (void)

DeInitializes MFX_IO low level.

Return values:

None

Definition at line **1208** of file **stm32l4r9i_eval.c**.

References **BSP_I2C_MFX_USER**, **I2Cx_DeInit()**, **MFX_INT_EXTI_IRQn**, **MFX_INT_GPIO_CLK_ENABLE**, **MFX_INT_GPIO_PORT**, **MFX_INT_PIN**, **MFX_WAKEUP_GPIO_CLK_ENABLE**, **MFX_WAKEUP_GPIO_PORT**, and **MFX_WAKEUP_PIN**.

void MFX_IO_Delay (uint32_t Delay)

MFX_IO delay.

Parameters:

Delay,: Delay in ms

Return values:

None

Definition at line **1164** of file **stm32l4r9i_eval.c**.

void MFX_IO_EnableWakeupPin (void)

MFX_IO enable wakeup pin.

Return values:

None

Definition at line **1189** of file **stm32l4r9i_eval.c**.

References **MFX_WAKEUP_GPIO_CLK_ENABLE**, **MFX_WAKEUP_GPIO_PORT**, and **MFX_WAKEUP_PIN**.

void MFX_IO_Init (void)

Initialize MFX_IO low level.

Return values:

None

Definition at line **1096** of file **stm32l4r9i_eval.c**.

References **BSP_I2C_MFX_USER**, and **I2Cx_Init()**.

Referenced by **BSP_IO_Init()**.

void MFX_IO_ITConfig (void)

Configure MFX_IO low level Interrupt.

Return values:

None

Definition at line **1105** of file **stm32l4r9i_eval.c**.

References **MFX_INT_EXTI_IRQn**, **MFX_INT_GPIO_CLK_ENABLE**, **MFX_INT_GPIO_PORT**, and **MFX_INT_PIN**.

Referenced by **BSP_JOY_Init()**.

uint8_t MFX_IO_Read (uint16_t Addr,
uint8_t Reg
)

MFX_IO reads single data.

Parameters:

Addr,: I2C address

Reg,: Reg address

Return values:

Read data

Definition at line **1142** of file **stm32l4r9i_eval.c**.

References **I2Cx_ReadData()**.

uint16_t MFX_IO_ReadMultiple (uint16_t Addr,
uint8_t Reg,
uint8_t * Buffer,

uint16_t Length
)

MFX_IO reads multiple data.

Parameters:

Addr,:	I2C address
Reg,:	Reg address
Buffer,:	Pointer to data buffer
Length,:	Length of the data

Definition at line **1154** of file **stm32l4r9i_eval.c**.

References **I2Cx_ReadBuffer()**.

void MFX_IO_Wakeup (void)

MFX_IO wakeup.

Return values:

None

Definition at line **1173** of file **stm32l4r9i_eval.c**.

References **MFX_WAKEUP_GPIO_PORT**, and **MFX_WAKEUP_PIN**.

void MFX_IO_Write (uint16_t Addr,
uint8_t Reg,
uint8_t Value
)

MFX_IO writes single data.

Parameters:

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

Return values:
None

Definition at line **1131** of file **stm32l4r9i_eval.c**.

References **I2Cx_WriteData()**.

void TS_IO_Delay (uint32_t Delay)

TS delay.

Parameters:
Delay,: Delay in ms

Return values:
None

Definition at line **1287** of file **stm32l4r9i_eval.c**.

void TS_IO_Init (void)

Initializes Touchscreen low level.

Return values:
None

Definition at line **1236** of file **stm32l4r9i_eval.c**.

References **BSP_I2C_TS_USER**, and **I2Cx_Init()**.

```
uint8_t TS_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 **1259** of file **stm32l4r9i_eval.c**.

References **I2Cx_ReadMultiple()**.

```
uint16_t TS_IO_ReadMultiple ( uint8_t    Addr,
                              uint8_t    Reg,
                              uint8_t *  Buffer,
                              uint16_t   Length
                            )
```

Reads multiple data with I2C communication channel from TouchScreen.

Parameters:

Addr,: I2C address

Reg,: Register address

Buffer,: Pointer to data buffer

Length,: Length of the data

Return values:

Number of read data

Definition at line **1277** of file **stm32l4r9i_eval.c**.

References **I2Cx_ReadMultiple()**.

```
void TS_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

Return values:

None

Definition at line **1248** of file **stm32l4r9i_eval.c**.

References **I2Cx_WriteMultiple()**.

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL NOR

Defines

#define	**NOR_STATUS_OK** 0x00 NOR status structure definition.
#define	**NOR_STATUS_ERROR** 0x01
#define	**NOR_DEVICE_ADDR** ((uint32_t)0x68000000)
#define	**NOR_MEMORY_WIDTH** FMC_NORSRAM_MEM_BUS_WID
#define	**NOR_BURSTACCESS** FMC_BURST_ACCESS_MODE_DIS
#define	**NOR_WRITEBURST** FMC_WRITE_BURST_DISABLE
#define	**BLOCKERASE_TIMEOUT** 2000U /* NOR block erase timeou */
#define	**CHIPERASE_TIMEOUT** 400000U /* NOR chip erase timeout s */
#define	**PROGRAM_TIMEOUT** 13U /* NOR program timeout : 64 byte 200 us = 12.8 ms */
#define	**NOR_READY_BUSY_PIN** GPIO_PIN_6
#define	**NOR_READY_BUSY_GPIO** GPIOD
#define	**NOR_READY_STATE** GPIO_PIN_SET
#define	**NOR_BUSY_STATE** GPIO_PIN_RESET

Define Documentation

#define BLOCKERASE_TIMEOUT 2000U /* NOR block erase timeo

Definition at line 81 of file stm32l4r9i_eval_nor.h.

Referenced by BSP_NOR_Erase_Block().

#define CHIPERASE_TIMEOUT 400000U /* NOR chip erase timeou

Definition at line 82 of file stm32l4r9i_eval_nor.h.

Referenced by BSP_NOR_Erase_Chip().

#define NOR_BURSTACCESS FMC_BURST_ACCESS_MODE_DIS

Definition at line 76 of file stm32l4r9i_eval_nor.h.

#define NOR_BUSY_STATE GPIO_PIN_RESET

Definition at line 89 of file stm32l4r9i_eval_nor.h.

Referenced by HAL_NOR_MspWait().

#define NOR_DEVICE_ADDR ((uint32_t)0x68000000)

Definition at line 72 of file stm32l4r9i_eval_nor.h.

Referenced by BSP_NOR_Erase_Block(), BSP_NOR_Erase_Chip(), BSP_NOR_ProgramData(), BSP_NOR_ReadData(), and BSP_NOR_WriteData().

#define NOR_MEMORY_WIDTH FMC_NORSRAM_MEM_BUS_WID

Definition at line 74 of file stm32l4r9i_eval_nor.h.

#define NOR_READY_BUSY_GPIO GPIOD

Definition at line 87 of file stm32l4r9i_eval_nor.h.

Referenced by HAL_NOR_MspWait(), and NOR_MspInit().

#define NOR_READY_BUSY_PIN GPIO_PIN_6

Definition at line 86 of file stm32l4r9i_eval_nor.h.

Referenced by HAL_NOR_MspWait(), and NOR_MspInit().

#define NOR_READY_STATE GPIO_PIN_SET

Definition at line 88 of file stm32l4r9i_eval_nor.h.

Referenced by HAL_NOR_MspWait().

#define NOR_STATUS_ERROR 0x01

Definition at line 70 of file stm32l4r9i_eval_nor.h.

Referenced by BSP_NOR_Erase_Block(), BSP_NOR_Erase_Chip(), BSP_NOR_Init(), BSP_NOR_ProgramData(), BSP_NOR_Read_ID(), BSP_NOR_ReadData(), and BSP_NOR_WriteData().

#define NOR_STATUS_OK 0x00

NOR status structure definition.

Definition at line **69** of file **stm32l4r9i_eval_nor.h**.

Referenced by **BSP_NOR_Erase_Block()**, **BSP_NOR_Erase_Chip()**, **BSP_NOR_Init()**, **BSP_NOR_ProgramData()**, **BSP_NOR_Read_ID()**, **BSP_NOR_ReadData()**, and **BSP_NOR_WriteData()**.

#define NOR_WRITEBURST FMC_WRITE_BURST_DISABLE

Definition at line **78** of file **stm32l4r9i_eval_nor.h**.

#define PROGRAM_TIMEOUT 13U /* NOR program timeout : 64 by

Definition at line **83** of file **stm32l4r9i_eval_nor.h**.

Referenced by **BSP_NOR_ProgramData()**, and **BSP_NOR_WriteData()**.

STM32L4R9I_EVAL BSP User Manual

BSP Audio Sample Rate

Exported Constants

Defines

#define	**BSP_AUDIO_FREQUENCY_96K**	SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_48K**	SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_44K**	SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_32K**	SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_22K**	SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_16K**	SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_11K**	SAI_AUDIO_FREQUENCY
#define	**BSP_AUDIO_FREQUENCY_8K**	SAI_AUDIO_FREQUENCY_

Define Documentation

#define BSP_AUDIO_FREQUENCY_11K SAI_AUDIO_FREQUENCY

Definition at line 99 of file stm32l4r9i_eval_audio.h.

#define BSP_AUDIO_FREQUENCY_16K SAI_AUDIO_FREQUENCY

Definition at line 98 of file stm32l4r9i_eval_audio.h.

#define BSP_AUDIO_FREQUENCY_22K SAI_AUDIO_FREQUENCY

Definition at line 97 of file stm32l4r9i_eval_audio.h.

#define BSP_AUDIO_FREQUENCY_32K SAI_AUDIO_FREQUENCY

Definition at line 96 of file stm32l4r9i_eval_audio.h.

#define BSP_AUDIO_FREQUENCY_44K SAI_AUDIO_FREQUENCY

Definition at line 95 of file stm32l4r9i_eval_audio.h.

#define BSP_AUDIO_FREQUENCY_48K SAI_AUDIO_FREQUENCY

Definition at line 94 of file stm32l4r9i_eval_audio.h.

#define BSP_AUDIO_FREQUENCY_8K SAI_AUDIO_FREQUENCY_

Definition at line 100 of file stm32l4r9i_eval_audio.h.

#define BSP_AUDIO_FREQUENCY_96K SAI_AUDIO_FREQUENCY

Definition at line 93 of file stm32l4r9i_eval_audio.h.

STM32L4R9I_EVAL BSP User Manual

Exported Variables

STM32L4R9I_EVAL AUDIO

Variables

SAI_HandleTypeDef	**BSP_AUDIO_hSai**
DFSDM_Filter_HandleTypeDef	**BSP_AUDIO_hDfsdmLeftFilter**
DFSDM_Filter_HandleTypeDef	**BSP_AUDIO_hDfsdmRightFilter**
SAI_HandleTypeDef	**BSP_AUDIO_hSai**
DFSDM_Filter_HandleTypeDef	**BSP_AUDIO_hDfsdmLeftFilter**
DFSDM_Filter_HandleTypeDef	**BSP_AUDIO_hDfsdmRightFilter**

Variable Documentation

DFSDM_Filter_HandleTypeDef BSP_AUDIO_hDfsdmLeftFilter

Definition at line **265** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_DeInit()**, **AUDIO_DFSDMx_Init()**, **BSP_AUDIO_IN_Pause()**, **BSP_AUDIO_IN_Record()**, **BSP_AUDIO_IN_Resume()**, **BSP_AUDIO_IN_Stop()**, **HAL_DFSDM_FilterRegConvCpltCallback()**, and **HAL_DFSDM_FilterRegConvHalfCpltCallback()**.

DFSDM_Filter_HandleTypeDef BSP_AUDIO_hDfsdmLeftFilter

Definition at line **265** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_DeInit()**, **AUDIO_DFSDMx_Init()**, **BSP_AUDIO_IN_Pause()**, **BSP_AUDIO_IN_Record()**, **BSP_AUDIO_IN_Resume()**, **BSP_AUDIO_IN_Stop()**, **HAL_DFSDM_FilterRegConvCpltCallback()**, and **HAL_DFSDM_FilterRegConvHalfCpltCallback()**.

DFSDM_Filter_HandleTypeDef BSP_AUDIO_hDfsdmRightFilter

Definition at line **266** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_DeInit()**, **AUDIO_DFSDMx_Init()**, **BSP_AUDIO_IN_Pause()**, **BSP_AUDIO_IN_Record()**, **BSP_AUDIO_IN_Resume()**, and **BSP_AUDIO_IN_Stop()**.

DFSDM_Filter_HandleTypeDef BSP_AUDIO_hDfsdmRightFilter

Definition at line **266** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_DeInit()**, **AUDIO_DFSDMx_Init()**, **BSP_AUDIO_IN_Pause()**, **BSP_AUDIO_IN_Record()**, **BSP_AUDIO_IN_Resume()**, and **BSP_AUDIO_IN_Stop()**.

SAI_HandleTypeDef BSP_AUDIO_hSai

Definition at line **262** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_SAIx_DeInit()**, **AUDIO_SAIx_Init()**, **BSP_AUDIO_OUT_ChangeAudioConfig()**, **BSP_AUDIO_OUT_ChangeBuffer()**, **BSP_AUDIO_OUT_Pause()**, **BSP_AUDIO_OUT_Play()**, **BSP_AUDIO_OUT_Resume()**, **BSP_AUDIO_OUT_SetFrequency()**, and **BSP_AUDIO_OUT_Stop()**.

SAI_HandleTypeDef BSP_AUDIO_hSai

Definition at line **262** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_SAIx_DeInit()**, **AUDIO_SAIx_Init()**, **BSP_AUDIO_OUT_ChangeAudioConfig()**, **BSP_AUDIO_OUT_ChangeBuffer()**, **BSP_AUDIO_OUT_Pause()**, **BSP_AUDIO_OUT_Play()**, **BSP_AUDIO_OUT_Resume()**, **BSP_AUDIO_OUT_SetFrequency()**, and **BSP_AUDIO_OUT_Stop()**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL 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_DeInit** (void) De-Initializes audio codec related peripherals.
uint8_t	**BSP_AUDIO_OUT_Play** (uint16_t *pData, uint32_t Size) Starts playing audio stream from a data buffer for a determined size.
uint8_t	**BSP_AUDIO_OUT_ChangeBuffer** (uint16_t *pData, uint16_t Size) Sends n-Bytes on the SAI interface.
uint8_t	**BSP_AUDIO_OUT_Pause** (void) This function Pauses the audio file stream.
uint8_t	**BSP_AUDIO_OUT_Resume** (void) This function Resumes the audio file stream.
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).
uint8_t	**BSP_AUDIO_OUT_SetFrequency** (uint32_t AudioFreq) Update the audio frequency.
void	**BSP_AUDIO_OUT_ChangeAudioConfig** (uint32_t AudioOutOption) Changes the Audio Out Configuration.
void	**BSP_AUDIO_OUT_RegisterCallbacks** (**Audio_CallbackTypeDef** ErrorCallback,

	Audio_CallbackTypeDef HalfTransferCallback, **Audio_CallbackTypeDef** TransferCompleteCallback) register user callback functions
void	**HAL_SAI_TxCpltCallback** (SAI_HandleTypeDef *hsai) Tx Transfer completed callbacks.
void	**HAL_SAI_TxHalfCpltCallback** (SAI_HandleTypeDef *hsai) Tx Half Transfer completed callbacks.
void	**HAL_SAI_ErrorCallback** (SAI_HandleTypeDef *hsai) SAI error callbacks.
uint8_t	**BSP_AUDIO_IN_Init** (uint32_t AudioFreq, uint32_t BitRes, uint32_t ChnlNbr) Initializes wave recording.
uint8_t	**BSP_AUDIO_IN_DeInit** (void) De-Initializes microphone related peripherals.
uint8_t	**BSP_AUDIO_IN_Record** (uint16_t *pbuf, uint32_t size) Starts audio recording.
uint8_t	**BSP_AUDIO_IN_SetFrequency** (uint32_t AudioFreq) Updates the audio frequency.
void	**HAL_DFSDM_FilterRegConvCpltCallback** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter) Regular conversion complete callback.
void	**HAL_DFSDM_FilterRegConvHalfCpltCallback** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter) Half regular conversion complete callback.
void	**HAL_DFSDM_FilterErrorCallback** (DFSDM_Filter_HandleTypeDef *hdfsdm_filter) Error callback.
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.
void	**BSP_AUDIO_IN_RegisterCallbacks**

(**Audio_CallbackTypeDef** ErrorCallback,
Audio_CallbackTypeDef HalfTransferCallback,
Audio_CallbackTypeDef TransferCompleteCallback)

register user callback functions

Function Documentation

uint8_t BSP_AUDIO_IN_DeInit (void)

De-Initializes microphone related peripherals.

Return values:

BSP AUDIO status

Definition at line **773** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_DFSDMx_DeInit()**, **AUDIO_ERROR**, and **AUDIO_OK**.

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.

BitRes,: Audio bit resolution to be configured..

ChnlNbr,: Number of channel to be configured.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **724** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_DFSDMx_Init()**, **AUDIO_ERROR**, **AUDIO_OK**, **AUDIO_SAIPLLConfig()**, **AUDIO_IN_TypeDef::BitResolution**,

AUDIO_IN_TypeDef::CbError,
AUDIO_IN_TypeDef::CbHalfTransfer,
AUDIO_IN_TypeDef::CbTransferComplete,
AUDIO_IN_TypeDef::ChannelNbr, **DmaLeftRecBuffCplt**,
DmaLeftRecHalfBuffCplt, **DmaRightRecBuffCplt**,
DmaRightRecHalfBuffCplt, **AUDIO_OUT_TypeDef::Frequency**,
AUDIO_IN_TypeDef::Frequency,
AUDIO_IN_TypeDef::InputDevice, and
AUDIO_OUT_TypeDef::OutputDevice.

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 **993** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**,
BSP_AUDIO_hDfsdmLeftFilter, and
BSP_AUDIO_hDfsdmRightFilter.

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

Note:

The Right channel is start at first with synchro on start of Left channel

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **794** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **BSP_AUDIO_hDfsdmLeftFilter**, **BSP_AUDIO_hDfsdmRightFilter**, **AUDIO_IN_TypeDef::ChannelNbr**, **AUDIO_IN_TypeDef::LeftRecBuff**, **AUDIO_IN_TypeDef::pRecBuf**, **AUDIO_IN_TypeDef::RecSize**, and **AUDIO_IN_TypeDef::RightRecBuff**.

void BSP_AUDIO_IN_RegisterCallbacks (Audio_CallbackTypeDef Audio_CallbackTypeDef Audio_CallbackTypeDef)

register user callback functions

Parameters:

ErrorCallback,:	pointer to the error callback function
HalfTransferCallback,:	pointer to the half transfer callback function
TransferCompleteCallback,:	pointer to the transfer complete callback function

Return values:

None

Definition at line **1044** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_IN_TypeDef::CbError**, **AUDIO_IN_TypeDef::CbHalfTransfer**, and **AUDIO_IN_TypeDef::CbTransferComplete**.

uint8_t BSP_AUDIO_IN_Resume (void)

Resumes the audio file stream.

Note:
The Right channel is start a first with synchro on start Left channel

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **1015** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **BSP_AUDIO_hDfsdmLeftFilter**, **BSP_AUDIO_hDfsdmRightFilter**, **AUDIO_IN_TypeDef::ChannelNbr**, **AUDIO_IN_TypeDef::LeftRecBuff**, **AUDIO_IN_TypeDef::RecSize**, and **AUDIO_IN_TypeDef::RightRecBuff**.

uint8_t BSP_AUDIO_IN_SetFrequency (uint32_t AudioFreq)

Updates the audio frequency.

Parameters:

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

Note:
This API should be called after the **BSP_AUDIO_IN_Init()** to adjust the audio frequency.

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **847** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_DFSDMx_DeInit()**, **AUDIO_DFSDMx_Init()**, **AUDIO_ERROR**, **AUDIO_OK**, and **AUDIO_SAIPLLConfig()**.

uint8_t BSP_AUDIO_IN_Stop (void)

Stops audio recording.

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **960** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **BSP_AUDIO_hDfsdmLeftFilter**, **BSP_AUDIO_hDfsdmRightFilter**, **AUDIO_IN_TypeDef::LeftRecBuff**, and **AUDIO_IN_TypeDef::RightRecBuff**.

void BSP_AUDIO_OUT_ChangeAudioConfig (uint32_t AudioOutO

Changes the Audio Out Configuration.

Parameters:

AudioOutOption,:	specifies the audio out new configuration This parameter can be any value of **BSP Audio Out Option**

Note:

This API should be called after the **BSP_AUDIO_OUT_Init()** to

adjust the audio out configuration.

Return values:

None

Definition at line **598** of file **stm32l4r9i_eval_audio.c**.

References **BSP_AUDIO_hSai**, **BSP_AUDIO_OUT_CIRCULARMODE**, and **BSP_AUDIO_OUT_STEREOMODE**.

uint8_t BSP_AUDIO_OUT_ChangeBuffer (uint16_t * pData, uint16_t Size)

Sends n-Bytes on the SAI interface.

Parameters:

pData,: pointer on data address

Size,: number of data to be written

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **412** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, and **BSP_AUDIO_hSai**.

uint8_t BSP_AUDIO_OUT_DeInit (void)

De-Initializes audio codec related peripherals.

Return values:

BSP AUDIO status

Definition at line **366** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **AUDIO_SAIx_DeInit()**, and **AUDIO_OUT_TypeDef::AudioDrv**.

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, or OUTPUT_DEVICE_BOTH.
Volume,:	Initial volume level (from 0 (Mute) to 100 (Max))
AudioFreq,:	Audio frequency used to play the audio stream.

Return values:

BSP AUDIO status

Note:
The SAI PLL input clock must be configure in the user application. The SAI PLL configuration done within this function assumes that the SAI PLL input clock runs at 4 MHz.

Definition at line **301** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_CODEC_Reset()**, **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIO_SAIPLLConfig()**, **AUDIO_SAIx_Init()**, **AUDIO_OUT_TypeDef::AudioDrv**, **BSP_AUDIO_OUT_SetVolume()**, **AUDIO_OUT_TypeDef::CbError**, **AUDIO_OUT_TypeDef::CbHalfTransfer**,

AUDIO_OUT_TypeDef::CbTransferComplete,
AUDIO_OUT_TypeDef::Frequency,
AUDIO_IN_TypeDef::Frequency,
AUDIO_IN_TypeDef::InputDevice, and
AUDIO_OUT_TypeDef::OutputDevice.

uint8_t BSP_AUDIO_OUT_Pause (void)

This function Pauses the audio file stream.

In case of using DMA, the DMA Pause feature is used.

> **Note:**
> 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 **432** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIO_OUT_TypeDef::AudioDrv**, and **BSP_AUDIO_hSai**.

uint8_t BSP_AUDIO_OUT_Play (uint16_t * pData, uint32_t Size)

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

Parameters:

pData,:	pointer on PCM samples buffer

Size,: Number of audio data HALF WORD.

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **389** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIO_OUT_TypeDef::AudioDrv**, **BSP_AUDIO_hSai**, and **DMA_MAX**.

void BSP_AUDIO_OUT_RegisterCallbacks (Audio_CallbackTypeDe
Audio_CallbackTypeDe
Audio_CallbackTypeDe
)

register user callback functions

Parameters:

ErrorCallback,:	pointer to the error callback function
HalfTransferCallback,:	pointer to the half transfer callback function
TransferCompleteCallback,:	pointer to the transfer complete callback function

Return values:

None

Definition at line **658** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_OUT_TypeDef::CbError**, **AUDIO_OUT_TypeDef::CbHalfTransfer**, and **AUDIO_OUT_TypeDef::CbTransferComplete**.

uint8_t BSP_AUDIO_OUT_Resume (void)

This function Resumes the audio file stream.

Note:
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 **457** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIO_OUT_TypeDef::AudioDrv**, and **BSP_AUDIO_hSai**.

uint8_t 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.
The SAI PLL configuration done within this function assumes that the SAI PLL input clock runs at 4 MHz.

Return values:

AUDIO_OK	if correct communication, else wrong communication

Definition at line **567** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_OK**, **AUDIO_SAIPLLConfig()**, **BSP_AUDIO_hSai**, **AUDIO_OUT_TypeDef::Frequency**, and **SAIClockDivider**.

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 **529** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **AUDIO_OUT_TypeDef::AudioDrv**.

uint8_t BSP_AUDIO_OUT_SetOutputMode (uint8_t Output)

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

Parameters:

Output,: The audio output target:
OUTPUT_DEVICE_SPEAKER,
OUTPUT_DEVICE_HEADPHONE or
OUTPUT_DEVICE_BOTH

Return values:

AUDIO_OK if correct communication, else wrong communication

Definition at line **547** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **AUDIO_OUT_TypeDef::AudioDrv**.

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 **512** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, and **AUDIO_OUT_TypeDef::AudioDrv**.

Referenced by **BSP_AUDIO_OUT_Init()**.

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_SW: for software power off (by

writing registers). Then no need to reconfigure the Codec after power on.
- 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 **483** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_ERROR**, **AUDIO_I2C_ADDRESS**, **AUDIO_OK**, **AUDIO_OUT_TypeDef::AudioDrv**, and **BSP_AUDIO_hSai**.

void HAL_DFSDM_FilterErrorCallback (DFSDM_Filter_HandleTypeI

Error callback.

Parameters:

hdfsdm_filter : DFSDM filter handle.

Return values:

None

Definition at line **947** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_IN_TypeDef::CbError**.

void HAL_DFSDM_FilterRegConvCpltCallback (DFSDM_Filter_Han

Regular conversion complete callback.

Note:

In interrupt mode, user has to read conversion value in this

function using HAL_DFSDM_FilterGetRegularValue.

Parameters:

hdfsdm_filter : DFSDM filter handle.

Return values:

None

Definition at line **877** of file **stm32l4r9i_eval_audio.c**.

References **BSP_AUDIO_hDfsdmLeftFilter**, **AUDIO_IN_TypeDef::CbTransferComplete**, **AUDIO_IN_TypeDef::ChannelNbr**, **DmaLeftRecBuffCplt**, **DmaRightRecBuffCplt**, **AUDIO_IN_TypeDef::LeftRecBuff**, **AUDIO_IN_TypeDef::pRecBuf**, **AUDIO_IN_TypeDef::RecSize**, **AUDIO_IN_TypeDef::RightRecBuff**, and **SaturaLH**.

void HAL_DFSDM_FilterRegConvHalfCpltCallback (DFSDM_Filter_

Half regular conversion complete callback.

Parameters:

hdfsdm_filter : DFSDM filter handle.

Return values:

None

Definition at line **912** of file **stm32l4r9i_eval_audio.c**.

References **BSP_AUDIO_hDfsdmLeftFilter**, **AUDIO_IN_TypeDef::CbHalfTransfer**, **AUDIO_IN_TypeDef::ChannelNbr**, **DmaLeftRecHalfBuffCplt**, **DmaRightRecHalfBuffCplt**, **AUDIO_IN_TypeDef::LeftRecBuff**, **AUDIO_IN_TypeDef::pRecBuf**, **AUDIO_IN_TypeDef::RecSize**, **AUDIO_IN_TypeDef::RightRecBuff**, and **SaturaLH**.

void HAL_SAI_ErrorCallback (SAI_HandleTypeDef * hsai)

SAI error callbacks.

Parameters:

hsai,: SAI handle

Return values:

None

Definition at line **700** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_OUT_TypeDef::CbError**.

void HAL_SAI_TxCpltCallback (SAI_HandleTypeDef * hsai)

Tx Transfer completed callbacks.

Parameters:

hsai,: SAI handle

Return values:

None

Definition at line **672** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_OUT_TypeDef::CbTransferComplete**.

void HAL_SAI_TxHalfCpltCallback (SAI_HandleTypeDef * hsai)

Tx Half Transfer completed callbacks.

Parameters:

hsai,: SAI handle

Return values:

None

Definition at line **686** of file **stm32l4r9i_eval_audio.c**.

References **AUDIO_OUT_TypeDef::CbHalfTransfer**.

STM32L4R9I_EVAL BSP User Manual

Defines

BSP Audio Out Option

Exported Constants

Defines

#define	**BSP_AUDIO_OUT_CIRCULARMODE** ((uint32_t)0x0000000 /* BUFFER CIRCULAR MODE */
#define	**BSP_AUDIO_OUT_NORMALMODE** ((uint32_t)0x00000002) /* BUFFER NORMAL MODE */
#define	**BSP_AUDIO_OUT_STEREOMODE** ((uint32_t)0x00000004) STEREO MODE */
#define	**BSP_AUDIO_OUT_MONOMODE** ((uint32_t)0x00000008) /* MONO MODE */

Define Documentation

#define BSP_AUDIO_OUT_CIRCULARMODE ((uint32_t)0x0000000

Definition at line **82** of file **stm32l4r9i_eval_audio.h**.

Referenced by **BSP_AUDIO_OUT_ChangeAudioConfig()**.

#define BSP_AUDIO_OUT_MONOMODE ((uint32_t)0x00000008) /*

Definition at line **85** of file **stm32l4r9i_eval_audio.h**.

#define BSP_AUDIO_OUT_NORMALMODE ((uint32_t)0x00000002)

Definition at line **83** of file **stm32l4r9i_eval_audio.h**.

#define BSP_AUDIO_OUT_STEREOMODE ((uint32_t)0x00000004)

Definition at line **84** of file **stm32l4r9i_eval_audio.h**.

Referenced by **BSP_AUDIO_OUT_ChangeAudioConfig()**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL Common

Functions

__weak void	**BSP_ErrorHandler** (void) Error Handler.
uint32_t	**BSP_GetVersion** (void) This method returns the STM32L4R9I EVAL BSP Driver revision.
void	**BSP_LED_Init** (**Led_TypeDef** Led) Initialize LED GPIO.
void	**BSP_LED_DeInit** (**Led_TypeDef** Led) DeInitialize LED GPIO.
void	**BSP_LED_On** (**Led_TypeDef** Led) Turn selected LED On.
void	**BSP_LED_Off** (**Led_TypeDef** Led) Turn selected LED Off.
void	**BSP_LED_Toggle** (**Led_TypeDef** Led) Toggle the selected LED.
void	**BSP_PB_Init** (**Button_TypeDef** Button, **ButtonMode_TypeDef** ButtonMode) Initialize push button GPIO and EXTI Line.
void	**BSP_PB_DeInit** (**Button_TypeDef** Button) DeInitialize push button.
uint32_t	**BSP_PB_GetState** (**Button_TypeDef** Button) Return the selected button state.
uint8_t	**BSP_JOY_Init** (**JOYMode_TypeDef** JoyMode) Initialize all buttons of the joystick in GPIO or EXTI modes.
void	**BSP_JOY_DeInit** (void) DeInitialize all buttons of the joystick.
JOYState_TypeDef	**BSP_JOY_GetState** (void) Return the current joystick status.
void	**BSP_COM_Init** (**COM_TypeDef** COM, UART_HandleTypeDef *huart)

	Initialize COM port.
void	**BSP_COM_DeInit** (**COM_TypeDef** COM, UART_HandleTypeDef *huart) DeInitialize COM port.
uint32_t	**BSP_POTENTIOMETER_Init** (void) Init the potentioneter RV2 measurement thanks to ADC.
uint32_t	**BSP_POTENTIOMETER_DeInit** (void) DeInitialize the potentioneter RV2 measurement thanks to ADC.
uint32_t	**BSP_POTENTIOMETER_GetLevel** (void) Get the potentioneter RV2 measurement thanks to ADC.

Function Documentation

void BSP_COM_DeInit (COM_TypeDef COM,
UART_HandleTypeDef * huart
)

DeInitialize COM port.

Parameters:

COM,: Specifies the COM port to be deinitialized. This parameter can be one of following parameters:
- COM1

huart,: pointer to a UART_HandleTypeDef structure that contains the configuration information for the specified UART peripheral.

Return values:

None

Definition at line **584** of file **stm32l4r9i_eval.c**.

void BSP_COM_Init (COM_TypeDef COM,
UART_HandleTypeDef * huart
)

Initialize COM port.

Parameters:

COM,: Specifies the COM port to be configured. This parameter can be one of following parameters:
- COM1

huart,: pointer to a UART_HandleTypeDef structure that contains the configuration information for the specified UART peripheral.

Return values:
None

Definition at line 542 of file stm32l4r9i_eval.c.

References COMx_CLK_ENABLE, COMx_RX_GPIO_CLK_ENABLE, and COMx_TX_GPIO_CLK_ENABLE.

void BSP_ErrorHandler (void)

Error Handler.

Note:
Defined as a weak function to be overwritten by the application.

Return values:
None

Definition at line 260 of file stm32l4r9i_eval.c.

Referenced by BSP_IDD_Init(), BSP_JOY_Init(), and I2Cx_Error().

uint32_t BSP_GetVersion (void)

This method returns the STM32L4R9I EVAL BSP Driver revision.

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

Definition at line 269 of file stm32l4r9i_eval.c.

References __STM32L4R9I_EVAL_BSP_VERSION.

void BSP_JOY_DeInit (void)

DeInitialize all buttons of the joystick.

Return values:

None

Definition at line **482** of file **stm32l4r9i_eval.c**.

References **BSP_IO_ConfigPin()**, and **JOY_ALL_PINS**.

JOYState_TypeDef BSP_JOY_GetState (void)

Return the current joystick status.

Return values:

Code of the joystick key pressed This code can be one of the following values:

- JOY_NONE
- JOY_SEL
- JOY_DOWN
- JOY_LEFT
- JOY_RIGHT
- JOY_UP

Definition at line **498** of file **stm32l4r9i_eval.c**.

References **BSP_IO_ReadPin()**, **DOWN_JOY_PIN**, **JOY_ALL_PINS**, **JOY_DOWN**, **JOY_LEFT**, **JOY_NONE**, **JOY_RIGHT**, **JOY_SEL**, **JOY_UP**, **LEFT_JOY_PIN**, **RIGHT_JOY_PIN**, **SEL_JOY_PIN**, and **UP_JOY_PIN**.

uint8_t BSP_JOY_Init (JOYMode_TypeDef JoyMode)

Initialize all buttons of the joystick in GPIO or EXTI modes.

Parameters:

JoyMode,: Joystick mode. This parameter can be one of the following values:

- JOY_MODE_GPIO: Joystick pins will be used as simple IOs
- JOY_MODE_EXTI: Joystick pins will be connected to EXTI line with interrupt generation capability

Return values:

HAL_OK,: if all initializations are OK. Other value if error.

Definition at line **454** of file **stm32l4r9i_eval.c**.

References **BSP_ErrorHandler()**, **BSP_IO_ConfigPin()**, **BSP_IO_Init()**, **IO_ERROR**, **JOY_ALL_PINS**, **JOY_MODE_EXTI**, and **MFX_IO_ITConfig()**.

void BSP_LED_DeInit (Led_TypeDef Led)

DeInitialize LED GPIO.

Parameters:

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

- LED1
- LED2
- LED3
- LED4

Return values:

None

Definition at line **313** of file **stm32l4r9i_eval.c**.

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

LEDx_GPIO_CLK_ENABLE.

void BSP_LED_Init (Led_TypeDef Led)

Initialize LED GPIO.

Parameters:

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

- LED1
- LED2
- LED3
- LED4

Return values:

None

Definition at line **284** of file **stm32l4r9i_eval.c**.

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

void BSP_LED_Off (Led_TypeDef Led)

Turn selected LED Off.

Parameters:

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

- LED1
- LED2
- LED3
- LED4

Return values:

None

Definition at line **346** of file **stm32l4r9i_eval.c**.

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

void BSP_LED_On (Led_TypeDef Led)

Turn selected LED On.

Parameters:

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

- LED1
- LED2
- LED3
- LED4

Return values:

None

Definition at line **331** of file **stm32l4r9i_eval.c**.

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

void BSP_LED_Toggle (Led_TypeDef Led)

Toggle the selected LED.

Parameters:

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

- LED1
- LED2
- LED3

- LED4

Return values:

None

Definition at line **361** of file **stm32l4r9i_eval.c**.

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

void BSP_PB_DeInit (Button_TypeDef Button)

DeInitialize push button.

Parameters:

Button,: Button to be deinitialized. This parameter can be one of the following values:
- BUTTON_WAKEUP: Wakeup/Tamper Push Button

Return values:

None

Definition at line **419** of file **stm32l4r9i_eval.c**.

References **BUTTON_PORT**, **BUTTON_WAKEUP**, and **BUTTONx_GPIO_CLK_ENABLE**.

uint32_t BSP_PB_GetState (Button_TypeDef Button)

Return the selected button state.

Parameters:

Button,: Button to be checked. This parameter can be one of the following values:
- BUTTON_WAKEUP: Wakeup/Tamper Push

Button

Return values:

Button state

Definition at line **440** of file **stm32l4r9i_eval.c**.

References **BUTTON_PORT**.

void BSP_PB_Init (**Button_TypeDef** **Button,**
ButtonMode_TypeDef **ButtonMode**
)

Initialize push button GPIO and EXTI Line.

Parameters:

Button,: Button to be configured. This parameter can be one of the following values:
- BUTTON_WAKEUP: Wakeup/Tamper Push Button

ButtonMode,: Button mode requested. This parameter can be one of the following values:
- BUTTON_MODE_GPIO: Button will be used as simple IO
- BUTTON_MODE_EXTI: Button will be connected to EXTI line with interrupt generation capability

Return values:

None

Definition at line **378** of file **stm32l4r9i_eval.c**.

References **BUTTON_MODE_GPIO**, **BUTTON_PORT**, **BUTTON_WAKEUP**, and **BUTTONx_GPIO_CLK_ENABLE**.

uint32_t BSP_POTENTIOMETER_DeInit (void)

DeInitialize the potentioneter RV2 measurement thanks to ADC.

Limitations : the potentiometer are exclusive with MFX, Audio codex, OctoSPIP1, Debugging connector and MC operation. And they are mutually exclusive. Please deinit all these components before usage.

Return values:

BSP_POTENTIOMETER_OK if no error

Definition at line **668** of file **stm32l4r9i_eval.c**.

References **BSP_POTENTIOMETER_ERROR**, and **BSP_POTENTIOMETER_OK**.

uint32_t BSP_POTENTIOMETER_GetLevel (void)

Get the potentioneter RV2 measurement thanks to ADC.

Limitations : the potentiometer are exclusive with MFX, Audio codex, OctoSPIP1, Debugging connector and MC operation. And they are mutually exclusive. Please deinit all these components before usage.

Return values:

value if no error else 0xFFFFFFFF

Definition at line **682** of file **stm32l4r9i_eval.c**.

References **ADCx_POLL_TIMEOUT**, and **BSP_POTENTIOMETER_ERROR**.

uint32_t BSP_POTENTIOMETER_Init (void)

Init the potentioneter RV2 measurement thanks to ADC.

Limitations : the potentiometer are exclusive with MFX, Audio codex, OctoSPIP1, Debugging connector and MC operation. And they are mutually exclusive. Please deinit all these components before usage.

Return values:

BSP_POTENTIOMETER_OK if no error

Definition at line **601** of file **stm32l4r9i_eval.c**.

References **ADCx**, **ADCx_CHANNEL**, **ADCx_CHANNEL_GPIO_CLK_ENABLE**, **ADCx_CHANNEL_GPIO_PORT**, **ADCx_CHANNEL_PIN**, **ADCx_CLK_ENABLE**, **BSP_POTENTIOMETER_ERROR**, and **BSP_POTENTIOMETER_OK**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL DSI LCD |
STM32L4R9I_EVAL LCD

Functions

uint8_t	**BSP_DSI_LCD_DeInit** (void) De-initialize the DSI LCD.
uint32_t	**BSP_DSI_LCD_GetXSize** (void) Gets the LCD X size.
uint32_t	**BSP_DSI_LCD_GetYSize** (void) Gets the LCD Y size.
uint8_t	**BSP_DSI_LCD_SelectLayer** (uint32_t LayerIndex) Selects the LCD Layer.
uint8_t	**BSP_DSI_LCD_SetLayerVisible** (uint32_t LayerIndex, FunctionalState State) Sets an LCD Layer visible.
uint8_t	**BSP_DSI_LCD_SetTransparency** (uint32_t LayerIndex, uint8_t Transparency) Configures the transparency.
uint8_t	**BSP_DSI_LCD_SetColorKeying** (uint32_t LayerIndex, uint32_t RGBValue) Configures and sets the color keying.
uint8_t	**BSP_DSI_LCD_ResetColorKeying** (uint32_t LayerIndex) Disables the color keying.
void	**BSP_DSI_LCD_SetTextColor** (uint32_t Color) Sets the LCD text color.
uint32_t	**BSP_DSI_LCD_GetTextColor** (void) Gets the LCD text color.
void	**BSP_DSI_LCD_SetBackColor** (uint32_t Color) Sets the LCD background color.
uint32_t	**BSP_DSI_LCD_GetBackColor** (void) Gets the LCD background color.
void	**BSP_DSI_LCD_SetFont** (sFONT *fonts) Sets the LCD text font.
sFONT *	**BSP_DSI_LCD_GetFont** (void)

	Gets the LCD text font.
uint32_t	**BSP_DSI_LCD_ReadPixel** (uint16_t Xpos, uint16_t Ypos) Reads an LCD pixel.
void	**BSP_DSI_LCD_Clear** (uint32_t Color) Clears the whole currently active layer of LTDC.
void	**BSP_DSI_LCD_ClearStringLine** (uint32_t Line) Clears the selected line in currently active layer.
void	**BSP_DSI_LCD_DisplayChar** (uint16_t Xpos, uint16_t Ypos, uint8_t Ascii) Displays one character in currently active layer.
void	**BSP_DSI_LCD_DisplayStringAt** (uint16_t Xpos, uint16_t Ypos, uint8_t *Text, **Text_AlignModeTypdef** Mode) Displays characters in currently active layer.
void	**BSP_DSI_LCD_DisplayStringAtLine** (uint16_t Line, uint8_t *ptr) Displays string on specified line of the LCD.
void	**BSP_DSI_LCD_DrawHLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length) Draws an horizontal line in currently active layer.
void	**BSP_DSI_LCD_DrawVLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length) Draws a vertical line in currently active layer.
void	**BSP_DSI_LCD_DrawLine** (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2) Draws an uni-line (between two points) in currently active layer.
void	**BSP_DSI_LCD_DrawRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height) Draws a rectangle in currently active layer.
void	**BSP_DSI_LCD_DrawCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius) Draws a circle in currently active layer.

void	**BSP_DSI_LCD_DrawPolygon** (**pPoint** Points, uint16_t PointCount) Draws an poly-line (between many points) in currently active layer.
void	**BSP_DSI_LCD_DrawEllipse** (int Xpos, int Ypos, int XRadius, int YRadius) Draws an ellipse on LCD in currently active layer.
void	**BSP_DSI_LCD_DrawBitmap** (uint32_t Xpos, uint32_t Ypos, uint8_t *pbmp) Draws a bitmap picture loaded in the internal Flash (32 bpp) in currently active layer.
void	**BSP_DSI_LCD_FillRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height) Draws a full rectangle in currently active layer.
void	**BSP_DSI_LCD_FillCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius) Draws a full circle in currently active layer.
void	**BSP_DSI_LCD_FillPolygon** (**pPoint** Points, uint16_t PointCount) Draws a full poly-line (between many points) in currently active layer.
void	**BSP_DSI_LCD_FillEllipse** (int Xpos, int Ypos, int XRadius, int YRadius) Draws a full ellipse in currently active layer.
void	**BSP_DSI_LCD_DrawPixel** (uint16_t Xpos, uint16_t Ypos, uint32_t RGB_Code) Draws a pixel on LCD.
void	**BSP_DSI_LCD_DisplayOn** (void) Switch back on the display if was switched off by previous call of **BSP_DSI_LCD_DisplayOff()**.
void	**BSP_DSI_LCD_DisplayOff** (void) Switch Off the display.
void	**BSP_DSI_LCD_Refresh** (void) Refresh the display.

uint8_t	**BSP_DSI_LCD_IsFrameBufferAvailable** (void) Check if frame buffer is available.
void	**BSP_DSI_LCD_SetBrightness** (uint8_t BrightnessValue) Set the brightness value.
__weak void	**BSP_DSI_LCD_DMA2D_IRQHandler** (void) Handles DMA2D interrupt request.
__weak void	**BSP_DSI_LCD_DSI_IRQHandler** (void) Handles DSI interrupt request.
__weak void	**BSP_DSI_LCD_LTDC_IRQHandler** (void) Handles LTDC interrupt request.
__weak void	**BSP_DSI_LCD_LTDC_ER_IRQHandler** (void) This function handles LTDC Error interrupt Handler.
__weak void	**BSP_DSI_LCD_MspDeInit** (void) De-Initializes the BSP LCD Msp.
__weak void	**BSP_DSI_LCD_MspInit** (void) Initialize the BSP LCD Msp.
uint8_t	**BSP_DSI_LCD_Init** (void)

Function Documentation

void BSP_DSI_LCD_Clear (uint32_t Color)

Clears the whole currently active layer of LTDC.

Parameters:

Color,: Color of the background (in ARGB8888 format)

Definition at line **857** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LL_FillBuffer()**.

void BSP_DSI_LCD_ClearStringLine (uint32_t Line)

Clears the selected line in currently active layer.

Parameters:

Line,: Line to be cleared

Definition at line **868** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, **LCD_DrawPropTypeDef::BackColor**, **BSP_DSI_LCD_FillRect()**, **BSP_DSI_LCD_SetTextColor()**, **LCD_DrawPropTypeDef::pFont**, and **LCD_DrawPropTypeDef::TextColor**.

uint8_t BSP_DSI_LCD_DeInit (void)

De-initialize the DSI LCD.

Return values:

LCD state

Definition at line **581** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **BSP_DSI_LCD_MspDeInit()**, **bsp_lcd_initialized**, **LCD_ERROR**, and **LCD_OK**.

void BSP_DSI_LCD_DisplayChar (uint16_t Xpos,
uint16_t Ypos,
uint8_t Ascii
)

Displays one character in currently active layer.

Parameters:

- **Xpos,:** Start column address
- **Ypos,:** Line where to display the character shape.
- **Ascii,:** Character ascii code This parameter must be a number between Min_Data = 0x20 and Max_Data = 0x7E

Definition at line **887** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, **DrawChar()**, and **LCD_DrawPropTypeDef::pFont**.

Referenced by **BSP_DSI_LCD_DisplayStringAt()**.

void BSP_DSI_LCD_DisplayOff (void)

Switch Off the display.

Definition at line **1455** of file **stm32l4r9i_eval_dsi_lcd.c**.

void BSP_DSI_LCD_DisplayOn (void)

Switch back on the display if was switched off by previous call of **BSP_DSI_LCD_DisplayOff()**.

Definition at line **1442** of file **stm32l4r9i_eval_dsi_lcd.c**.

void BSP_DSI_LCD_DisplayStringAt (
uint16_t Xpos,
uint16_t Ypos,
uint8_t * Text,
Text_AlignModeTypdef Mode
)

Displays characters in currently active layer.

Parameters:

- **Xpos,:** X position (in pixel)
- **Ypos,:** Y position (in pixel)
- **Text,:** Pointer to string to display on LCD
- **Mode,:** Display mode This parameter can be one of the following values:
  - CENTER_MODE
  - RIGHT_MODE
  - LEFT_MODE

Definition at line **903** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, **BSP_DSI_LCD_DisplayChar()**, **CENTER_MODE**, **LEFT_MODE**, **LCD_DrawPropTypeDef::pFont**, and **RIGHT_MODE**.

Referenced by **BSP_DSI_LCD_DisplayStringAtLine()**.

void BSP_DSI_LCD_DisplayStringAtLine (
uint16_t Line,
uint8_t * ptr

)

Displays string on specified line of the LCD.

Parameters:

Line,: Line where to display the character shape

ptr,: Pointer to string to display on LCD

Definition at line **965** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **BSP_DSI_LCD_DisplayStringAt()**, **BSP_DSI_LCD_GetFont()**, and **CENTER_MODE**.

void BSP_DSI_LCD_DMA2D_IRQHandler (void)

Handles DMA2D interrupt request.

Note:
Application can surcharge if needed this function implementation.

Definition at line **1510** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **hdma2d_eval**.

void BSP_DSI_LCD_DrawBitmap (uint32_t Xpos,
uint32_t Ypos,
uint8_t * pbmp
)

Draws a bitmap picture loaded in the internal Flash (32 bpp) in currently active layer.

Parameters:

Xpos,: Bmp X position in the LCD

Ypos,: Bmp Y position in the LCD
pbmp,: Pointer to Bmp picture address in the internal Flash

Definition at line **1207** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LL_ConvertLineToARGB8888()**.

void BSP_DSI_LCD_DrawCircle (uint16_t Xpos,
uint16_t Ypos,
uint16_t Radius
)

Draws a circle in currently active layer.

Parameters:
Xpos,: X position
Ypos,: Y position
Radius,: Circle radius

Definition at line **1102** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **BSP_DSI_LCD_DrawPixel()**.

Referenced by **BSP_DSI_LCD_FillCircle()**.

void BSP_DSI_LCD_DrawEllipse (int Xpos,
int Ypos,
int XRadius,
int YRadius
)

Draws an ellipse on LCD in currently active layer.

Parameters:

Xpos,: X position
Ypos,: Y position
XRadius,: Ellipse X radius
YRadius,: Ellipse Y radius

Definition at line **1175** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **BSP_DSI_LCD_DrawPixel()**.

void BSP_DSI_LCD_DrawHLine (uint16_t Xpos,
uint16_t Ypos,
uint16_t Length
)

Draws an horizontal line in currently active layer.

Parameters:
Xpos,: X position
Ypos,: Y position
Length,: Line length

Definition at line **976** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LL_FillBuffer()**.

Referenced by **BSP_DSI_LCD_DrawRect()**, **BSP_DSI_LCD_FillCircle()**, and **BSP_DSI_LCD_FillEllipse()**.

void BSP_DSI_LCD_DrawLine (uint16_t x1,
uint16_t y1,
uint16_t x2,
uint16_t y2
)

Draws an uni-line (between two points) in currently active layer.

Parameters:

- **x1,:** **Point** 1 X position
- **y1,:** **Point** 1 Y position
- **x2,:** **Point** 2 X position
- **y2,:** **Point** 2 Y position

Definition at line **1011** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ABS**, **ActiveLayer**, and **BSP_DSI_LCD_DrawPixel()**.

Referenced by **BSP_DSI_LCD_DrawPolygon()**, and **FillTriangle()**.

void BSP_DSI_LCD_DrawPixel (uint16_t Xpos,
uint16_t Ypos,
uint32_t RGB_Code
)

Draws a pixel on LCD.

Parameters:

- **Xpos,:** X position
- **Ypos,:** Y position
- **RGB_Code,:** Pixel color in ARGB mode (8-8-8-8)

Definition at line **1433** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**.

Referenced by **BSP_DSI_LCD_DrawCircle()**, **BSP_DSI_LCD_DrawEllipse()**, **BSP_DSI_LCD_DrawLine()**, and **DrawChar()**.

void BSP_DSI_LCD_DrawPolygon (pPoint Points,

uint16_t PointCount
)

Draws an poly-line (between many points) in currently active layer.

Parameters:

Points,:	Pointer to the points array
PointCount,:	Number of points

Definition at line **1148** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **BSP_DSI_LCD_DrawLine()**, **Point::X**, and **Point::Y**.

void BSP_DSI_LCD_DrawRect (**uint16_t Xpos,**
uint16_t Ypos,
uint16_t Width,
uint16_t Height
)

Draws a rectangle in currently active layer.

Parameters:

Xpos,:	X position
Ypos,:	Y position
Width,:	Rectangle width
Height,:	Rectangle height

Definition at line **1085** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **BSP_DSI_LCD_DrawHLine()**, and **BSP_DSI_LCD_DrawVLine()**.

void BSP_DSI_LCD_DrawVLine (**uint16_t Xpos,**
uint16_t Ypos,

uint16_t Length
)

Draws a vertical line in currently active layer.

Parameters:

Xpos,:	X position
Ypos,:	Y position
Length,:	Line length

Definition at line **993** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LL_FillBuffer()**.

Referenced by **BSP_DSI_LCD_DrawRect()**.

void BSP_DSI_LCD_DSI_IRQHandler (void)

Handles DSI interrupt request.

Note:
Application can surcharge if needed this function implementation.

Definition at line **1519** of file **stm32l4r9i_eval_dsi_lcd.c**.

void BSP_DSI_LCD_FillCircle (uint16_t Xpos,
uint16_t Ypos,
uint16_t Radius
)

Draws a full circle in currently active layer.

Parameters:

Xpos,: X position
Ypos,: Y position
Radius,: Circle radius

Definition at line **1287** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, **BSP_DSI_LCD_DrawCircle()**, **BSP_DSI_LCD_DrawHLine()**, and **BSP_DSI_LCD_SetTextColor()**.

void BSP_DSI_LCD_FillEllipse (int Xpos,
int Ypos,
int XRadius,
int YRadius
)

Draws a full ellipse in currently active layer.

Parameters:
Xpos,: X position
Ypos,: Y position
XRadius,: Ellipse X radius
YRadius,: Ellipse Y radius

Definition at line **1401** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **BSP_DSI_LCD_DrawHLine()**.

void BSP_DSI_LCD_FillPolygon (pPoint Points,
uint16_t PointCount
)

Draws a full poly-line (between many points) in currently active layer.

Parameters:

Points,: Pointer to the points array
PointCount,: Number of points

Definition at line **1334** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **FillTriangle()**, **POLY_X**, **POLY_Y**, **Point::X**, and **Point::Y**.

void BSP_DSI_LCD_FillRect (uint16_t Xpos,
uint16_t Ypos,
uint16_t Width,
uint16_t Height
)

Draws a full rectangle in currently active layer.

Parameters:

Xpos,: X position
Ypos,: Y position
Width,: Rectangle width
Height,: Rectangle height

Definition at line **1267** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, **BSP_DSI_LCD_SetTextColor()**, and **LL_FillBuffer()**.

Referenced by **BSP_DSI_LCD_ClearStringLine()**.

uint32_t BSP_DSI_LCD_GetBackColor (void)

Gets the LCD background color.

Return values:

Used background color

Definition at line **812** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LCD_DrawPropTypeDef::BackColor**.

sFONT * BSP_DSI_LCD_GetFont (void)

Gets the LCD text font.

Return values:

Used layer font

Definition at line **830** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LCD_DrawPropTypeDef::pFont**.

Referenced by **BSP_DSI_LCD_DisplayStringAtLine()**.

uint32_t BSP_DSI_LCD_GetTextColor (void)

Gets the LCD text color.

Return values:

Used text color.

Definition at line **794** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LCD_DrawPropTypeDef::TextColor**.

uint32_t BSP_DSI_LCD_GetXSize (void)

Gets the LCD X size.

Return values:

Used LCD X size

Definition at line **644** of file **stm32l4r9i_eval_dsi_lcd.c**.

uint32_t BSP_DSI_LCD_GetYSize (void)

Gets the LCD Y size.

Return values:

Used LCD Y size

Definition at line **653** of file **stm32l4r9i_eval_dsi_lcd.c**.

uint8_t BSP_DSI_LCD_Init (void)

uint8_t BSP_DSI_LCD_IsFrameBufferAvailable (void)

Check if frame buffer is available.

Return values:

LCD_OK if frame buffer is available else LCD_ERROR (frame buffer busy)

Definition at line **1481** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **LCD_ERROR**, and **LCD_OK**.

void BSP_DSI_LCD_LTDC_ER_IRQHandler (void)

This function handles LTDC Error interrupt Handler.

Note:

Application can surcharge if needed this function implementation.

Definition at line **1539** of file **stm32l4r9i_eval_dsi_lcd.c**.

void BSP_DSI_LCD_LTDC_IRQHandler (void)

Handles LTDC interrupt request.

> **Note:**
> Application can surcharge if needed this function implementation.

Definition at line **1529** of file **stm32l4r9i_eval_dsi_lcd.c**.

void BSP_DSI_LCD_MspDeInit (void)

De-Initializes the BSP LCD Msp.

> **Note:**
> Application can surcharge if needed this function implementation.

Definition at line **1548** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **bsp_lcd_hse_to_disable**.

Referenced by **BSP_DSI_LCD_DeInit()**.

void BSP_DSI_LCD_MspInit (void)

Initialize the BSP LCD Msp.

> **Note:**
> Application can surcharge if needed this function implementation.

Definition at line **1591** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **bsp_lcd_hse_to_disable**.

uint32_t BSP_DSI_LCD_ReadPixel (uint16_t Xpos,
uint16_t Ypos
)

Reads an LCD pixel.

Parameters:

Xpos,: X position

Ypos,: Y position

Return values:

ARGB8888 pixel color

Definition at line **841** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**.

void BSP_DSI_LCD_Refresh (void)

Refresh the display.

Definition at line **1468** of file **stm32l4r9i_eval_dsi_lcd.c**.

uint8_t BSP_DSI_LCD_ResetColorKeying (uint32_t LayerIndex)

Disables the color keying.

Parameters:

LayerIndex,: Layer foreground (1) or background (0)

Note:

: Only backgroung layer can be used.

Return values:

LCD state

Definition at line **765** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **LCD_ERROR**, **LCD_OK**, and **LTDC_ACTIVE_LAYER_BACKGROUND**.

uint8_t BSP_DSI_LCD_SelectLayer (uint32_t LayerIndex)

Selects the LCD Layer.

Parameters:

LayerIndex,: Layer foreground (1) or background (0)

Note:

: Only backgroung layer can be used.

Return values:

LCD state

Definition at line **664** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, **LCD_ERROR**, **LCD_OK**, and **LTDC_ACTIVE_LAYER_BACKGROUND**.

void BSP_DSI_LCD_SetBackColor (uint32_t Color)

Sets the LCD background color.

Parameters:

Color,: Layer background color code ARGB8888

Definition at line **803** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LCD_DrawPropTypeDef::BackColor**.

void BSP_DSI_LCD_SetBrightness (uint8_t BrightnessValue)

Set the brightness value.

Parameters:

BrightnessValue,: [0% Min (black), 100% Max]

Definition at line **1494** of file **stm32l4r9i_eval_dsi_lcd.c**.

uint8_t BSP_DSI_LCD_SetColorKeying (uint32_t LayerIndex, uint32_t RGBValue)

Configures and sets the color keying.

Parameters:

LayerIndex,: Layer foreground (1) or background (0)

RGBValue,: Color reference

Note:

: Only backgroung layer can be used.

Return values:

LCD state

Definition at line **742** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **LCD_ERROR**, **LCD_OK**, and **LTDC_ACTIVE_LAYER_BACKGROUND**.

void BSP_DSI_LCD_SetFont (sFONT * fonts)

Sets the LCD text font.

Parameters:

fonts,: Layer font to be used

Definition at line **821** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LCD_DrawPropTypeDef::pFont**.

uint8_t BSP_DSI_LCD_SetLayerVisible (uint32_t LayerInde, FunctionalState State)

Sets an LCD Layer visible.

Parameters:

LayerIndex,: Visible Layer

State,: New state of the specified layer This parameter can be one of the following values:

- ENABLE
- DISABLE

Note:

: Only backgroung layer can be used.

Return values:

LCD state

Definition at line **689** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **LCD_ERROR**, **LCD_OK**, and **LTDC_ACTIVE_LAYER_BACKGROUND**.

void BSP_DSI_LCD_SetTextColor (uint32_t Color)

Sets the LCD text color.

Parameters:

Color,: Text color code ARGB8888

Definition at line **785** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, and **LCD_DrawPropTypeDef::TextColor**.

Referenced by **BSP_DSI_LCD_ClearStringLine()**, **BSP_DSI_LCD_FillCircle()**, and **BSP_DSI_LCD_FillRect()**.

uint8_t BSP_DSI_LCD_SetTransparency (uint32_t LayerIndex, uint8_t Transparency)

Configures the transparency.

Parameters:

LayerIndex,:	Layer foreground or background.
Transparency,:	Transparency This parameter must be a number between Min_Data = 0x00 and Max_Data = 0xFF

Note:

: Only backgroung layer can be used.

Return values:

LCD state

Definition at line **720** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **LCD_ERROR**, **LCD_OK**, and **LTDC_ACTIVE_LAYER_BACKGROUND**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL DSI TS

Functions

uint8_t	**BSP_DSI_TS_Init** (uint16_t ts_SizeX, uint16_t ts_SizeY)
uint8_t	**BSP_DSI_TS_DeInit** (void)
uint8_t	**BSP_DSI_TS_GetState** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_DSI_TS_GestureConfig** (FunctionalState State)
uint8_t	**BSP_DSI_TS_Get_GestureId** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_DSI_TS_ResetTouchData** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_DSI_TS_ITConfig** (void)
uint8_t	**BSP_DSI_TS_ITGetStatus** (void)
void	**BSP_DSI_TS_ITClear** (void)

Function Documentation

uint8_t BSP_DSI_TS_DeInit (void)

uint8_t BSP_DSI_TS_GestureConfig (FunctionalState State)

uint8_t BSP_DSI_TS_Get_GestureId (TS_StateTypeDef * TS_State

uint8_t BSP_DSI_TS_GetState (TS_StateTypeDef * TS_State)

uint8_t BSP_DSI_TS_Init (uint16_t ts_SizeX,
uint16_t ts_SizeY
)

void BSP_DSI_TS_ITClear (void)

uint8_t BSP_DSI_TS_ITConfig (void)

uint8_t BSP_DSI_TS_ITGetStatus (void)

uint8_t BSP_DSI_TS_ResetTouchData (TS_StateTypeDef * TS_Stat

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL EEPROM

Functions

uint32_t	**BSP_EEPROM_Init** (void) Initializes peripherals used by the I2C EEPROM driver.
uint8_t	**BSP_EEPROM_DeInit** (void) DeInitializes the EEPROM.
uint32_t	**BSP_EEPROM_ReadBuffer** (uint8_t *pBuffer, uint16_t ReadAddr, uint16_t *NumByteToRead) Reads a block of data from the EEPROM.
uint32_t	**BSP_EEPROM_WritePage** (uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite) Writes more than one byte to the EEPROM with a single WRITE cycle.
uint32_t	**BSP_EEPROM_WriteBuffer** (uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite) Writes buffer of data to the I2C EEPROM.
uint32_t	**BSP_EEPROM_WaitEepromStandbyState** (void) Wait for EEPROM Standby state.
__weak void	**BSP_EEPROM_TIMEOUT_UserCallback** (void) Basic management of the timeout situation.
void	**EEPROM_IO_Init** (void) Initialize peripherals used by the I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_WriteData** (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize) Write data to I2C EEPROM driver.
HAL_StatusTypeDef	**EEPROM_IO_ReadData** (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize) Read data from I2C EEPROM driver.

HAL_StatusTypeDef	**EEPROM_IO_IsDeviceReady** (uint16_t DevAddress, uint32_t Trials) Checks if target device is ready for communication.

Function Documentation

uint8_t BSP_EEPROM_DeInit (void)

DeInitializes the EEPROM.

Return values:

EEPROM state

Definition at line **137** of file **stm32l4r9i_eval_eeprom.c**.

References **EEPROM_OK**.

uint32_t BSP_EEPROM_Init (void)

Initializes peripherals used by the I2C EEPROM driver.

Return values:

EEPROM_OK (0) if operation is correctly performed, else return value different from EEPROM_OK (0)

Definition at line **119** of file **stm32l4r9i_eval_eeprom.c**.

References **EEPROM_FAIL**, **EEPROM_I2C_ADDRESS**, **EEPROM_IO_Init()**, **EEPROM_IO_IsDeviceReady()**, **EEPROM_MAX_TRIALS**, **EEPROM_OK**, and **EEPROMAddress**.

uint32_t BSP_EEPROM_ReadBuffer (uint8_t * pBuffer, uint16_t ReadAddr, uint16_t * NumByteToRead)

Reads a block of data from the EEPROM.

Parameters:

pBuffer,:	pointer to the buffer that receives the data read from the EEPROM.
ReadAddr,:	EEPROM's internal address to start reading from.
NumByteToRead,:	pointer to the variable holding number of bytes to be read from the EEPROM.

Note:

The variable pointed by NumByteToRead is reset to 0 when all the data are read from the EEPROM. Application should monitor this variable in order know when the transfer is complete.

Return values:

EEPROM_OK	(0) if operation is correctly performed, else return value different from EEPROM_OK (0) or the timeout user callback.

Definition at line **158** of file **stm32l4r9i_eval_eeprom.c**.

References **BSP_EEPROM_TIMEOUT_UserCallback()**, **EEPROM_FAIL**, **EEPROM_IO_ReadData()**, **EEPROM_OK**, **EEPROMAddress**, and **EEPROMDataRead**.

void BSP_EEPROM_TIMEOUT_UserCallback (void)

Basic management of the timeout situation.

Return values:

None

Definition at line **413** of file **stm32l4r9i_eval_eeprom.c**.

Referenced by **BSP_EEPROM_ReadBuffer()**, **BSP_EEPROM_WaitEepromStandbyState()**, and **BSP_EEPROM_WritePage()**.

uint32_t BSP_EEPROM_WaitEepromStandbyState (void)

Wait for EEPROM Standby state.

Note:
This function allows to wait and check that EEPROM has finished the last operation. It is mostly used after Write operation: after receiving the buffer to be written, the EEPROM may need additional time to actually perform the write operation. During this time, it doesn't answer to I2C packets addressed to it. Once the write operation is complete the EEPROM responds to its address.

Return values:

EEPROM_OK	(0) if operation is correctly performed, else return value different from EEPROM_OK (0) or the timeout user callback.

Definition at line **397** of file **stm32l4r9i_eval_eeprom.c**.

References **BSP_EEPROM_TIMEOUT_UserCallback()**, **EEPROM_IO_IsDeviceReady()**, **EEPROM_MAX_TRIALS**, **EEPROM_OK**, **EEPROM_TIMEOUT**, and **EEPROMAddress**.

Referenced by **BSP_EEPROM_WritePage()**.

uint32_t BSP_EEPROM_WriteBuffer (uint8_t * pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite)

Writes buffer of data to the I2C EEPROM.

Parameters:

pBuffer,: pointer to the buffer containing the data to be written to the EEPROM.

WriteAddr,: EEPROM's internal address to write to.

NumByteToWrite,: number of bytes to write to the EEPROM.

Return values:

EEPROM_OK (0) if operation is correctly performed, else return value different from EEPROM_OK (0) or the timeout user callback.

Definition at line **240** of file **stm32l4r9i_eval_eeprom.c**.

References **BSP_EEPROM_WritePage()**, **EEPROM_OK**, and **EEPROM_PAGESIZE**.

uint32_t BSP_EEPROM_WritePage (uint8_t * pBuffer,
uint16_t WriteAddr,
uint8_t * NumByteToWrite
)

Writes more than one byte to the EEPROM with a single WRITE cycle.

Note:

The number of bytes (combined to write start address) must not cross the EEPROM page boundary. This function can only write into the boundaries of an EEPROM page. This function doesn't check on boundaries condition (in this driver the function **BSP_EEPROM_WriteBuffer()** which calls **BSP_EEPROM_WritePage()** is responsible of checking on Page boundaries).

Parameters:

pBuffer,: pointer to the buffer containing the data to be written to the EEPROM.

WriteAddr,:	EEPROM's internal address to write to.
NumByteToWrite,:	pointer to the variable holding number of bytes to be written into the EEPROM.

Note:

The variable pointed by NumByteToWrite is reset to 0 when all the data are written to the EEPROM. Application should monitor this variable in order know when the transfer is complete.
This function just configure the communication and enable the DMA channel to transfer data. Meanwhile, the user application may perform other tasks in parallel.

Return values:

EEPROM_OK	(0) if operation is correctly performed, else return value different from EEPROM_OK (0) or the timeout user callback.

Definition at line **205** of file **stm32l4r9i_eval_eeprom.c**.

References **BSP_EEPROM_TIMEOUT_UserCallback()**, **BSP_EEPROM_WaitEepromStandbyState()**, **EEPROM_FAIL**, **EEPROM_IO_WriteData()**, **EEPROM_OK**, **EEPROMAddress**, and **EEPROMDataWrite**.

Referenced by **BSP_EEPROM_WriteBuffer()**.

void EEPROM_IO_Init (void)

Initialize peripherals used by the I2C EEPROM driver.

Return values:

None

Definition at line **1365** of file **stm32l4r9i_eval.c**.

References **BSP_I2C_EEPROM_USER**, and **I2Cx_Init()**.

Referenced by **BSP_EEPROM_Init()**.

HAL_StatusTypeDef EEPROM_IO_IsDeviceReady (uint16_t DevAd
uint32_t Trials
)

Checks if target device is ready for communication.

Note:
This function is used with Memory devices

Parameters:

DevAddress,:	Target device address
Trials,:	Number of trials

Return values:

HAL	status

Definition at line **1403** of file **stm32l4r9i_eval.c**.

References **I2Cx_IsDeviceReady()**.

Referenced by **BSP_EEPROM_Init()**, and **BSP_EEPROM_WaitEepromStandbyState()**.

HAL_StatusTypeDef EEPROM_IO_ReadData (uint16_t DevAddres
uint16_t MemAddres
uint8_t * pBuffer,
uint32_t BufferSize
)

Read data from I2C EEPROM driver.

Parameters:

DevAddress,: Target device address
MemAddress,: Internal memory address
pBuffer,: Pointer to data buffer
BufferSize,: Amount of data to be read

Return values:
HAL status

Definition at line **1391** of file **stm32l4r9i_eval.c**.

References **I2Cx_ReadBuffer()**.

Referenced by **BSP_EEPROM_ReadBuffer()**.

HAL_StatusTypeDef EEPROM_IO_WriteData (uint16_t DevAddress
uint16_t MemAddres
uint8_t * pBuffer,
uint32_t BufferSize
)

Write data to I2C EEPROM driver.

Parameters:
DevAddress,: Target device address
MemAddress,: Internal memory address
pBuffer,: Pointer to data buffer
BufferSize,: Amount of data to be sent

Return values:
HAL status

Definition at line **1378** of file **stm32l4r9i_eval.c**.

References **I2Cx_WriteBuffer()**.

Referenced by **BSP_EEPROM_WritePage()**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL IDD

Functions

uint8_t	**BSP_IDD_Init** (void) Configures IDD measurement component.
void	**BSP_IDD_DeInit** (void) Unconfigures IDD measurement component.
void	**BSP_IDD_Reset** (void) Reset Idd measurement component.
void	**BSP_IDD_LowPower** (void) Turn Idd measurement component in low power (standby/sleep) mode.
void	**BSP_IDD_StartMeasure** (void) Start Measurement campaign.
void	**BSP_IDD_Config** (IDD_ConfigTypeDef IddConfig) Configure Idd component.
void	**BSP_IDD_GetValue** (uint32_t *IddValue) Get Idd current value.
void	**BSP_IDD_EnableIT** (void) Enable Idd interrupt that warn end of measurement.
void	**BSP_IDD_ClearIT** (void) Clear Idd interrupt that warn end of measurement.
uint8_t	**BSP_IDD_GetITStatus** (void) Get Idd interrupt status.
void	**BSP_IDD_DisableIT** (void) Disable Idd interrupt that warn end of measurement.
uint8_t	**BSP_IDD_ErrorGetCode** (void) Get Error Code .
void	**BSP_IDD_ErrorEnableIT** (void) Enable error interrupt that warn end of measurement.
void	**BSP_IDD_ErrorClearIT** (void) Clear Error interrupt that warn end of measurement.
uint8_t	**BSP_IDD_ErrorGetITStatus** (void)

	Get Error interrupt status.
void	**BSP_IDD_ErrorDisableIT** (void) Disable Error interrupt.
void	**BSP_IDD_WakeUp** (void) Wake up Idd measurement component.

Function Documentation

void BSP_IDD_ClearIT (void)

Clear Idd interrupt that warn end of measurement.

Return values:
None

Definition at line **234** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

void BSP_IDD_Config (IDD_ConfigTypeDef IddConfig)

Configure Idd component.

Parameters:
IddConfig,: structure of idd parameters

Return values:
None

Definition at line **194** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

Referenced by **BSP_IDD_Init()**.

void BSP_IDD_DeInit (void)

Unconfigures IDD measurement component.

Return values:

IDD_OK if no problem during deinitialization

Definition at line **140** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

void BSP_IDD_DisableIT (void)

Disable Idd interrupt that warn end of measurement.

Return values:

None

Definition at line **262** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

void BSP_IDD_EnableIT (void)

Enable Idd interrupt that warn end of measurement.

Return values:

None

Definition at line **222** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

void BSP_IDD_ErrorClearIT (void)

Clear Error interrupt that warn end of measurement.

Return values:

None

Definition at line **317** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

void BSP_IDD_ErrorDisableIT (void)

Disable Error interrupt.

Return values:
None

Definition at line **345** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

void BSP_IDD_ErrorEnableIT (void)

Enable error interrupt that warn end of measurement.

Return values:
None

Definition at line **305** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

uint8_t BSP_IDD_ErrorGetCode (void)

Get Error Code .

Return values:
Error code or error status

Definition at line **274** of file **stm32l4r9i_eval_idd.c**.

References **IDD_ERROR**, **IDD_I2C_ADDRESS**, and **IddDrv**.

uint8_t BSP_IDD_ErrorGetITStatus (void)

Get Error interrupt status.

Return values:
Status

Definition at line **329** of file **stm32l4r9i_eval_idd.c**.

References **IDD_I2C_ADDRESS**, and **IddDrv**.

uint8_t BSP_IDD_GetITStatus (void)

Get Idd interrupt status.

Return values:
status

Definition at line **246** of file **stm32l4r9i_eval_idd.c**.

References **IDD_ERROR**, **IDD_I2C_ADDRESS**, and **IddDrv**.

void BSP_IDD_GetValue (uint32_t * IddValue)

Get Idd current value.

Parameters:
IddValue,: Pointer on u32 to store Idd. Value unit is 10 nA.

Return values:
None

Definition at line 207 of file stm32l4r9i_eval_idd.c.

References BSP_IO_ConfigPin(), IDD_AMP_CONTROL_PIN, IDD_I2C_ADDRESS, and IddDrv.

uint8_t BSP_IDD_Init (void)

Configures IDD measurement component.

Return values:

IDD_OK if no problem during initialization

Definition at line 73 of file stm32l4r9i_eval_idd.c.

References BSP_ErrorHandler(), BSP_IDD_Config(), BSP_IO_Init(), EVAL_IDD_AMPLI_GAIN, EVAL_IDD_SHUNT0_STABDELAY, EVAL_IDD_SHUNT0_VALUE, EVAL_IDD_SHUNT1_STABDELAY, EVAL_IDD_SHUNT1_VALUE, EVAL_IDD_SHUNT2_STABDELAY, EVAL_IDD_SHUNT2_VALUE, EVAL_IDD_SHUNT4_STABDELAY, EVAL_IDD_SHUNT4_VALUE, EVAL_IDD_VDD_MIN, IDD_ERROR, IDD_I2C_ADDRESS, IDD_OK, IddDrv, and IO_ERROR.

void BSP_IDD_LowPower (void)

Turn Idd measurement component in low power (standby/sleep) mode.

Return values:

None

Definition at line 164 of file stm32l4r9i_eval_idd.c.

References IDD_I2C_ADDRESS, and IddDrv.

void BSP_IDD_Reset (void)

Reset Idd measurement component.

Return values:
None

Definition at line 152 of file stm32l4r9i_eval_idd.c.

References IDD_I2C_ADDRESS, and IddDrv.

void BSP_IDD_StartMeasure (void)

Start Measurement campaign.

Return values:
None

Definition at line 176 of file stm32l4r9i_eval_idd.c.

References BSP_IO_ConfigPin(), BSP_IO_WritePin(), IDD_AMP_CONTROL_PIN, IDD_I2C_ADDRESS, and IddDrv.

void BSP_IDD_WakeUp (void)

Wake up Idd measurement component.

Return values:
None

Definition at line 357 of file stm32l4r9i_eval_idd.c.

References IDD_I2C_ADDRESS, and IddDrv.

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL IO

Functions

uint8_t	**BSP_IO_Init** (void) Initialize and configure the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).
uint8_t	**BSP_IO_DeInit** (void) DeInitialize the IO to allow Mfx Initialization to be executed again.
uint32_t	**BSP_IO_ITGetStatus** (uint32_t IO_Pin) Get the selected pins IT status.
void	**BSP_IO_ITClear** (uint32_t IO_Pin) Clear the selected IO IT pending bit.
uint8_t	**BSP_IO_ConfigPin** (uint32_t IO_Pin, IO_ModeTypedef IO_Mode) Configure the IO pin(s) according to IO mode structure value.
void	**BSP_IO_WritePin** (uint32_t IO_Pin, uint8_t PinState) Set the selected IO pin(s) state.
uint32_t	**BSP_IO_ReadPin** (uint32_t IO_Pin) Get the selected IO pin(s) current state.
void	**BSP_IO_TogglePin** (uint32_t IO_Pin) Toggle the selected IO pin(s) state.

Function Documentation

uint8_t BSP_IO_ConfigPin (uint32_t IO_Pin, IO_ModeTypedef IO_Mode)

Configure the IO pin(s) according to IO mode structure value.

Parameters:

IO_Pin,: Output IO pin(s) to be set or reset. This parameter can be any combination of the IO pin(s).

IO_Mode,: IO pin mode to configure This parameter can be one of the following values:

- IO_MODE_INPUT
- IO_MODE_OUTPUT
- IO_MODE_IT_RISING_EDGE
- IO_MODE_IT_FALLING_EDGE
- IO_MODE_IT_LOW_LEVEL
- IO_MODE_IT_HIGH_LEVEL

Return values:

IO_OK,: if all initializations are OK. Other value if error.

Definition at line **218** of file **stm32l4r9i_eval_io.c**.

References **io_driver**, **IO_I2C_ADDRESS**, and **IO_OK**.

Referenced by **BSP_IDD_GetValue()**, **BSP_IDD_StartMeasure()**, **BSP_JOY_DeInit()**, **BSP_JOY_Init()**, **BSP_SD_Init()**, **BSP_SD_IsDetected()**, **BSP_SD_ITConfig()**, and **LCD_Reset()**.

uint8_t BSP_IO_DeInit (void)

DeInitialize the IO to allow Mfx Initialization to be executed again.

Note:
BSP_IO_Init() has no effect if the io_driver is already initialized
BSP_IO_DeInit() allows to erase the pointer such to allow init to be effective

Return values:
IO_OK

Definition at line 166 of file stm32l4r9i_eval_io.c.

References io_driver, IO_I2C_ADDRESS, and IO_OK.

uint8_t BSP_IO_Init (void)

Initialize and configure the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).

Note:
BSP_IO_Init() is using HAL_Delay() function to ensure that stmpe811 IO Expander is correctly reset. HAL_Delay() function provides accurate delay (in milliseconds) based on variable incremented in SysTick ISR. This implies that if BSP_IO_Init() is called from a peripheral ISR process, then the SysTick interrupt must have higher priority (numerically lower) than the peripheral interrupt. Otherwise the caller ISR process will be blocked.

Return values:
IO_OK,: if all initializations are OK. Other value if error.

Definition at line 116 of file stm32l4r9i_eval_io.c.

References io_driver, IO_ERROR, IO_I2C_ADDRESS, IO_OK, IO_PIN_ALL, and MFX_IO_Init().

Referenced by BSP_IDD_Init(), BSP_JOY_Init(), BSP_SD_Init(), and LCD_Reset().

void BSP_IO_ITClear (uint32_t IO_Pin)

Clear the selected IO IT pending bit.

Parameters:

IO_Pin,: Selected pin(s) to clear the status. This parameter can be any combination of the IO pins.

Return values:

None

Definition at line **198** of file **stm32l4r9i_eval_io.c**.

References **io_driver**, and **IO_I2C_ADDRESS**.

uint32_t BSP_IO_ITGetStatus (uint32_t IO_Pin)

Get the selected pins IT status.

Parameters:

IO_Pin,: Selected pin(s) to check the status. This parameter can be any combination of the IO pins.

Return values:

Status of the checked IO pin(s).

Definition at line **186** of file **stm32l4r9i_eval_io.c**.

References **io_driver**, and **IO_I2C_ADDRESS**.

uint32_t BSP_IO_ReadPin (uint32_t IO_Pin)

Get the selected IO pin(s) current state.

Parameters:

IO_Pin,: Selected pin(s) to read. This parameter can be any combination of the IO pin(s).

Return values:

The current pins state

Definition at line **245** of file **stm32l4r9i_eval_io.c**.

References **io_driver**, and **IO_I2C_ADDRESS**.

Referenced by **BSP_JOY_GetState()**, **BSP_SD_IsDetected()**, and **BSP_SD_ITConfig()**.

void BSP_IO_TogglePin (uint32_t IO_Pin)

Toggle the selected IO pin(s) state.

Parameters:

IO_Pin,: Selected IO pin(s) to toggle. This parameter can be any combination of the IO pin(s).

Return values:

None

Definition at line **256** of file **stm32l4r9i_eval_io.c**.

References **io_driver**, and **IO_I2C_ADDRESS**.

void BSP_IO_WritePin (uint32_t IO_Pin,
uint8_t PinState
)

Set the selected IO pin(s) state.

Parameters:

IO_Pin,: Selected IO pin(s) to write. This parameter can be any combination of the IO pin(s).

PinState,: New pin state to write

Return values:

None

Definition at line **233** of file **stm32l4r9i_eval_io.c**.

References **io_driver**, and **IO_I2C_ADDRESS**.

Referenced by **BSP_IDD_StartMeasure()**, and **LCD_Reset()**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL NOR

Functions

uint8_t	**BSP_NOR_Init** (void) Initializes the NOR device.
uint8_t	**BSP_NOR_ReadData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Reads an amount of data from the NOR device.
void	**BSP_NOR_ReturnToReadMode** (void) Returns the NOR memory to read mode.
uint8_t	**BSP_NOR_WriteData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Writes an amount of data to the NOR device.
uint8_t	**BSP_NOR_ProgramData** (uint32_t uwStartAddress, uint16_t *pData, uint32_t uwDataSize) Programs an amount of data to the NOR device.
uint8_t	**BSP_NOR_Erase_Block** (uint32_t BlockAddress) Erases the specified block of the NOR device.
uint8_t	**BSP_NOR_Erase_Chip** (void) Erases the entire NOR chip.
uint8_t	**BSP_NOR_Read_ID** (NOR_IDTypeDef *pNOR_ID) Reads NOR flash IDs.

Function Documentation

uint8_t BSP_NOR_Erase_Block (uint32_t BlockAddress)

Erases the specified block of the NOR device.

Parameters:

BlockAddress,: Block address to erase

Return values:

NOR memory status

Definition at line **283** of file **stm32l4r9i_eval_nor.c**.

References **BLOCKERASE_TIMEOUT**, **NOR_DEVICE_ADDR**, **NOR_STATUS_ERROR**, **NOR_STATUS_OK**, and **norHandle**.

uint8_t BSP_NOR_Erase_Chip (void)

Erases the entire NOR chip.

Return values:

NOR memory status

Definition at line **303** of file **stm32l4r9i_eval_nor.c**.

References **CHIPERASE_TIMEOUT**, **NOR_DEVICE_ADDR**, **NOR_STATUS_ERROR**, **NOR_STATUS_OK**, and **norHandle**.

uint8_t BSP_NOR_Init (void)

Initializes the NOR device.

Return values:

NOR memory status

Definition at line **131** of file **stm32l4r9i_eval_nor.c**.

References **NOR_ADDR_SETUP_TIME**, **NOR_DATA_HOLD_TIME**, **NOR_DATA_SETUP_TIME**, **NOR_MspInit()**, **NOR_STATUS_ERROR**, **NOR_STATUS_OK**, **NOR_TURN_AROUND_TIME**, **norHandle**, and **Timing**.

uint8_t BSP_NOR_ProgramData (uint32_t uwStartAddress, uint16_t * pData, uint32_t uwDataSize)

Programs an amount of data to the NOR device.

Parameters:

uwStartAddress,: Write start address
pData,: Pointer to data to be written
uwDataSize,: Size of data to write

Return values:

NOR memory status

Definition at line **242** of file **stm32l4r9i_eval_nor.c**.

References **NOR_DEVICE_ADDR**, **NOR_MAX_BUFFER_SIZE**, **NOR_STATUS_ERROR**, **NOR_STATUS_OK**, **norHandle**, and **PROGRAM_TIMEOUT**.

uint8_t BSP_NOR_Read_ID (NOR_IDTypeDef * pNOR_ID)

Reads NOR flash IDs.

Parameters:

pNOR_ID : Pointer to NOR ID structure

Return values:

NOR memory status

Definition at line **324** of file **stm32l4r9i_eval_nor.c**.

References **NOR_STATUS_ERROR**, **NOR_STATUS_OK**, and **norHandle**.

uint8_t BSP_NOR_ReadData (uint32_t uwStartAddress, uint16_t * pData, uint32_t uwDataSize)

Reads an amount of data from the NOR device.

Parameters:

uwStartAddress,:	Read start address
pData,:	Pointer to data to be read
uwDataSize,:	Size of data to read

Return values:

NOR memory status

Definition at line **183** of file **stm32l4r9i_eval_nor.c**.

References **NOR_DEVICE_ADDR**, **NOR_STATUS_ERROR**, **NOR_STATUS_OK**, and **norHandle**.

void BSP_NOR_ReturnToReadMode (void)

Returns the NOR memory to read mode.

Return values:

None

Definition at line **199** of file **stm32l4r9i_eval_nor.c**.

References **norHandle**.

uint8_t BSP_NOR_WriteData (uint32_t uwStartAddress,
uint16_t * pData,
uint32_t uwDataSize
)

Writes an amount of data to the NOR device.

Parameters:

uwStartAddress,:	Write start address
pData,:	Pointer to data to be written
uwDataSize,:	Size of data to write

Return values:

NOR memory status

Definition at line **211** of file **stm32l4r9i_eval_nor.c**.

References **NOR_DEVICE_ADDR**, **NOR_STATUS_ERROR**, **NOR_STATUS_OK**, **norHandle**, and **PROGRAM_TIMEOUT**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL OSPI NOR

Functions

uint8_t	**BSP_OSPI_NOR_Init** (void) Initializes the OSPI interface.
uint8_t	**BSP_OSPI_NOR_DeInit** (void) De-Initializes the OSPI interface.
uint8_t	**BSP_OSPI_NOR_Read** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Write** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Block** (uint32_t BlockAddress) Erases the specified block of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Sector** (uint32_t Sector) Erases the specified sector of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_Erase_Chip** (void) Erases the entire OSPI memory.
uint8_t	**BSP_OSPI_NOR_GetStatus** (void) Reads current status of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_GetInfo** (**OSPI_NOR_Info** *pInfo) Return the configuration of the OSPI memory.
uint8_t	**BSP_OSPI_NOR_EnableMemoryMappedMode** (void) Configure the OSPI in memory-mapped mode.
uint8_t	**BSP_OSPI_NOR_SuspendErase** (void) This function suspends an ongoing erase command.
uint8_t	**BSP_OSPI_NOR_ResumeErase** (void) This function resumes a paused erase command.
uint8_t	**BSP_OSPI_NOR_EnterDeepPowerDown** (void) This function enter the OSPI memory in deep power down mode.
uint8_t	**BSP_OSPI_NOR_LeaveDeepPowerDown** (void) This function leave the OSPI memory from deep power down

mode.

Function Documentation

uint8_t BSP_OSPI_NOR_DeInit (void)

De-Initializes the OSPI interface.

Return values:
OSPI memory status

Definition at line **189** of file **stm32l4r9i_eval_ospi_nor.c**.

References **OSPI_NOR_ERROR**, **OSPI_NOR_MspDeInit()**, **OSPI_NOR_OCTAL_DISABLE**, **OSPI_NOR_OctalMode()**, **OSPI_NOR_OK**, and **OSPINORHandle**.

uint8_t BSP_OSPI_NOR_EnableMemoryMappedMode (void)

Configure the OSPI in memory-mapped mode.

Return values:
OSPI memory status

Definition at line **626** of file **stm32l4r9i_eval_ospi_nor.c**.

References **DUMMY_CYCLES_READ_OCTAL**, **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, **OSPI_NOR_WriteEnable()**, and **OSPINORHandle**.

uint8_t BSP_OSPI_NOR_EnterDeepPowerDown (void)

This function enter the OSPI memory in deep power down mode.

Return values:
OSPI memory status

Definition at line 795 of file stm32l4r9i_eval_ospi_nor.c.

References OSPI_NOR_ERROR, OSPI_NOR_OK, and OSPINORHandle.

uint8_t BSP_OSPI_NOR_Erase_Block (uint32_t BlockAddress)

Erases the specified block of the OSPI memory.

Parameters:

BlockAddress,: Block address to erase

Return values:

OSPI memory status

Definition at line 368 of file stm32l4r9i_eval_ospi_nor.c.

References OSPI_NOR_AutoPollingMemReady(), OSPI_NOR_ERROR, OSPI_NOR_OK, OSPI_NOR_WriteEnable(), and OSPINORHandle.

uint8_t BSP_OSPI_NOR_Erase_Chip (void)

Erases the entire OSPI memory.

Return values:

OSPI memory status

Definition at line 477 of file stm32l4r9i_eval_ospi_nor.c.

References OSPI_NOR_AutoPollingMemReady(), OSPI_NOR_ERROR, OSPI_NOR_OK, OSPI_NOR_WriteEnable(), and OSPINORHandle.

uint8_t BSP_OSPI_NOR_Erase_Sector (uint32_t Sector)

Erases the specified sector of the OSPI memory.

Parameters:

Sector,: Sector address to erase (0 to 255)

Return values:

OSPI memory status

Note:

This function is non blocking meaning that sector erase operation is started but not completed when the function returns. Application has to call BSP_QSPI_GetStatus() to know when the device is available again (i.e. erase operation completed).

Definition at line **426** of file **stm32l4r9i_eval_ospi_nor.c**.

References **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, **OSPI_NOR_WriteEnable()**, and **OSPINORHandle**.

uint8_t BSP_OSPI_NOR_GetInfo (OSPI_NOR_Info * pInfo)

Return the configuration of the OSPI memory.

Parameters:

pInfo,: pointer on the configuration structure

Return values:

OSPI memory status

Definition at line **610** of file **stm32l4r9i_eval_ospi_nor.c**.

References **OSPI_NOR_Info::EraseSectorSize**, **OSPI_NOR_Info::EraseSectorsNumber**, **OSPI_NOR_Info::FlashSize**, **OSPI_NOR_OK**, **OSPI_NOR_Info::ProgPageSize**, and

OSPI_NOR_Info::ProgPagesNumber.

uint8_t BSP_OSPI_NOR_GetStatus (void)

Reads current status of the OSPI memory.

Return values:

OSPI memory status

Definition at line **525** of file **stm32l4r9i_eval_ospi_nor.c**.

References **DUMMY_CYCLES_READ_OCTAL**, **OSPI_NOR_BUSY**, **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, **OSPI_NOR_SUSPENDED**, and **OSPINORHandle**.

Referenced by **BSP_OSPI_NOR_ResumeErase()**, and **BSP_OSPI_NOR_SuspendErase()**.

uint8_t BSP_OSPI_NOR_Init (void)

Initializes the OSPI interface.

Return values:

OSPI memory status

Definition at line **133** of file **stm32l4r9i_eval_ospi_nor.c**.

References **OSPI_NOR_ERROR**, **OSPI_NOR_MspInit()**, **OSPI_NOR_NOT_SUPPORTED**, **OSPI_NOR_OCTAL_ENABLE**, **OSPI_NOR_OctalMode()**, **OSPI_NOR_OK**, **OSPI_NOR_ResetMemory()**, and **OSPINORHandle**.

uint8_t BSP_OSPI_NOR_LeaveDeepPowerDown (void)

This function leave the OSPI memory from deep power down mode.

Return values:

OSPI memory status

Definition at line **833** of file **stm32l4r9i_eval_ospi_nor.c**.

References **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, and **OSPINORHandle**.

uint8_t BSP_OSPI_NOR_Read (uint8_t * pData,
uint32_t ReadAddr,
uint32_t Size
)

Reads an amount of data from the OSPI memory.

Parameters:

pData,:	Pointer to data to be read
ReadAddr,:	Read start address
Size,:	Size of data to read

Return values:

OSPI memory status

Definition at line **227** of file **stm32l4r9i_eval_ospi_nor.c**.

References **DUMMY_CYCLES_READ_OCTAL**, **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, and **OSPINORHandle**.

uint8_t BSP_OSPI_NOR_ResumeErase (void)

This function resumes a paused erase command.

Return values:

OSPI memory status

Definition at line **742** of file **stm32l4r9i_eval_ospi_nor.c**.

References **BSP_OSPI_NOR_GetStatus()**, **OSPI_NOR_BUSY**, **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, **OSPI_NOR_SUSPENDED**, and **OSPINORHandle**.

uint8_t BSP_OSPI_NOR_SuspendErase (void)

This function suspends an ongoing erase command.

Return values:

OSPI memory status

Definition at line **693** of file **stm32l4r9i_eval_ospi_nor.c**.

References **BSP_OSPI_NOR_GetStatus()**, **OSPI_NOR_BUSY**, **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, **OSPI_NOR_SUSPENDED**, and **OSPINORHandle**.

uint8_t BSP_OSPI_NOR_Write (uint8_t * pData,
uint32_t WriteAddr,
uint32_t Size
)

Writes an amount of data to the OSPI memory.

Parameters:

pData,: Pointer to data to be written

WriteAddr,: Write start address

Size,: Size of data to write

Return values:

OSPI memory status

Definition at line **281** of file **stm32l4r9i_eval_ospi_nor.c**.

References **OSPI_NOR_AutoPollingMemReady()**, **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, **OSPI_NOR_WriteEnable()**, and **OSPINORHandle**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL OSPI RAM

Functions

uint8_t	**BSP_OSPI_RAM_Init** (void) Initializes the OSPI interface.
uint8_t	**BSP_OSPI_RAM_DeInit** (void) De-Initializes the OSPI interface.
uint8_t	**BSP_OSPI_RAM_Read** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the HyperRAM memory.
uint8_t	**BSP_OSPI_RAM_Read_DMA** (uint8_t *pData, uint32_t ReadAddr, uint32_t Size) Reads an amount of data from the HyperRAM memory in DMA mode.
uint8_t	**BSP_OSPI_RAM_Write** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the HyperRAM memory.
uint8_t	**BSP_OSPI_RAM_Write_DMA** (uint8_t *pData, uint32_t WriteAddr, uint32_t Size) Writes an amount of data to the HyperRAM memory in DMA mode.
uint8_t	**BSP_OSPI_RAM_EnableMemoryMappedMode** (void) Configure the OSPI in memory-mapped mode.
uint8_t	**BSP_OSPI_RAM_EnterDeepPowerDown** (void) This function enter the HyperRAM memory in deep power down mode.
uint8_t	**BSP_OSPI_RAM_LeaveDeepPowerDown** (void) This function leave the HyperRAM memory from deep power down mode.
void	**BSP_OSPI_RAM_DMA_IRQHandler** (void) Handles OctoSPI HyperRAM DMA transfer interrupt request.
void	**BSP_OSPI_RAM_IRQHandler** (void) Handles OctoSPI HyperRAM interrupt request.

Function Documentation

uint8_t BSP_OSPI_RAM_DeInit (void)

De-Initializes the OSPI interface.

Return values:
HyperRAM memory status

Definition at line 179 of file stm32l4r9i_eval_ospi_ram.c.

References OSPI_RAM_ERROR, OSPI_RAM_MspDeInit(), OSPI_RAM_OK, and OspiRamHandle.

void BSP_OSPI_RAM_DMA_IRQHandler (void)

Handles OctoSPI HyperRAM DMA transfer interrupt request.

Return values:
None

Definition at line 462 of file stm32l4r9i_eval_ospi_ram.c.

References OspiRamHandle.

uint8_t BSP_OSPI_RAM_EnableMemoryMappedMode (void)

Configure the OSPI in memory-mapped mode.

Return values:
HyperRAM memory status

Definition at line 331 of file stm32l4r9i_eval_ospi_ram.c.

References **OSPI_RAM_ERROR**, **OSPI_RAM_OK**, and **OspiRamHandle**.

uint8_t BSP_OSPI_RAM_EnterDeepPowerDown (void)

This function enter the HyperRAM memory in deep power down mode.

Return values:

HyperRAM memory status

Definition at line **363** of file **stm32l4r9i_eval_ospi_ram.c**.

References **OSPI_RAM_ERROR**, **OSPI_RAM_OK**, and **OspiRamHandle**.

uint8_t BSP_OSPI_RAM_Init (void)

Initializes the OSPI interface.

Return values:

HyperRAM memory status

Definition at line **127** of file **stm32l4r9i_eval_ospi_ram.c**.

References **OSPI_RAM_ERROR**, **OSPI_RAM_MspInit()**, **OSPI_RAM_OK**, and **OspiRamHandle**.

void BSP_OSPI_RAM_IRQHandler (void)

Handles OctoSPI HyperRAM interrupt request.

Return values:

None

Definition at line 471 of file stm32l4r9i_eval_ospi_ram.c.

References OspiRamHandle.

uint8_t BSP_OSPI_RAM_LeaveDeepPowerDown (void)

This function leave the HyperRAM memory from deep power down mode.

Return values:

HyperRAM memory status

Definition at line 428 of file stm32l4r9i_eval_ospi_ram.c.

References OSPI_RAM_ERROR, OSPI_RAM_OK, and OspiRamHandle.

uint8_t BSP_OSPI_RAM_Read (uint8_t * pData, uint32_t ReadAddr, uint32_t Size)

Reads an amount of data from the HyperRAM memory.

Parameters:

pData,:	Pointer to data to be read
ReadAddr,:	Read start address
Size,:	Size of data to read

Return values:

HyperRAM memory status

Definition at line 202 of file stm32l4r9i_eval_ospi_ram.c.

References OSPI_RAM_ERROR, OSPI_RAM_OK, and

OspiRamHandle.

```
uint8_t BSP_OSPI_RAM_Read_DMA ( uint8_t *  pData,
                                uint32_t  ReadAddr,
                                uint32_t  Size
                              )
```

Reads an amount of data from the HyperRAM memory in DMA mode.

Parameters:

pData,:	Pointer to data to be read
ReadAddr,:	Read start address
Size,:	Size of data to read

Return values:

HyperRAM	memory status

Definition at line **235** of file **stm32l4r9i_eval_ospi_ram.c**.

References **OSPI_RAM_ERROR**, **OSPI_RAM_OK**, and **OspiRamHandle**.

```
uint8_t BSP_OSPI_RAM_Write ( uint8_t *  pData,
                             uint32_t  WriteAddr,
                             uint32_t  Size
                           )
```

Writes an amount of data to the HyperRAM memory.

Parameters:

pData,:	Pointer to data to be written
WriteAddr,:	Write start address
Size,:	Size of data to write

Return values:

HyperRAM memory status

Definition at line **268** of file **stm32l4r9i_eval_ospi_ram.c**.

References **OSPI_RAM_ERROR**, **OSPI_RAM_OK**, and **OspiRamHandle**.

uint8_t BSP_OSPI_RAM_Write_DMA (uint8_t * pData,
uint32_t WriteAddr,
uint32_t Size
)

Writes an amount of data to the HyperRAM memory in DMA mode.

Parameters:

pData,:	Pointer to data to be written
WriteAddr,:	Write start address
Size,:	Size of data to write

Return values:

HyperRAM memory status

Definition at line **301** of file **stm32l4r9i_eval_ospi_ram.c**.

References **OSPI_RAM_ERROR**, **OSPI_RAM_OK**, and **OspiRamHandle**.

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL RGB LCD

Functions

uint8_t	**BSP_RGB_LCD_Init** (void)
uint8_t	**BSP_RGB_LCD_DeInit** (void)
uint32_t	**BSP_RGB_LCD_GetXSize** (void)
uint32_t	**BSP_RGB_LCD_GetYSize** (void)
void	**BSP_RGB_LCD_SetXSize** (uint32_t imageWidthPixels)
void	**BSP_RGB_LCD_SetYSize** (uint32_t imageHeightPixels)
void	**BSP_RGB_LCD_LayerDefaultInit** (uint16_t LayerIndex, uint32_t FrameBuffer)
void	**BSP_RGB_LCD_LayerRgb565Init** (uint16_t LayerIndex, uint32_t FB_Address)
void	**BSP_RGB_LCD_SetTransparency** (uint32_t LayerIndex, uint8_t Transparency)
void	**BSP_RGB_LCD_SetTransparency_NoReload** (uint32_t LayerIndex, uint8_t Transparency)
void	**BSP_RGB_LCD_SetLayerAddress** (uint32_t LayerIndex, uint32_t Address)
void	**BSP_RGB_LCD_SetLayerAddress_NoReload** (uint32_t LayerIndex, uint32_t Address)
void	**BSP_RGB_LCD_SetColorKeying** (uint32_t LayerIndex, uint32_t RGBValue)
void	**BSP_RGB_LCD_SetColorKeying_NoReload** (uint32_t LayerIndex, uint32_t RGBValue)
void	**BSP_RGB_LCD_ResetColorKeying** (uint32_t LayerIndex)
void	**BSP_RGB_LCD_ResetColorKeying_NoReload** (uint32_t LayerIndex)
void	**BSP_RGB_LCD_SetLayerWindow** (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
void	**BSP_RGB_LCD_SetLayerWindow_NoReload** (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
void	**BSP_RGB_LCD_SelectLayer** (uint32_t LayerIndex)

void	**BSP_RGB_LCD_SetLayerVisible** (uint32_t LayerIndex, FunctionalState State)
void	**BSP_RGB_LCD_SetLayerVisible_NoReload** (uint32_t LayerIndex, FunctionalState State)
void	**BSP_RGB_LCD_Reload** (uint32_t ReloadType)
void	**BSP_RGB_LCD_SetTextColor** (uint32_t Color)
uint32_t	**BSP_RGB_LCD_GetTextColor** (void)
void	**BSP_RGB_LCD_SetBackColor** (uint32_t Color)
uint32_t	**BSP_RGB_LCD_GetBackColor** (void)
void	**BSP_RGB_LCD_SetFont** (sFONT *fonts)
sFONT *	**BSP_RGB_LCD_GetFont** (void)
uint32_t	**BSP_RGB_LCD_ReadPixel** (uint16_t Xpos, uint16_t Ypos)
void	**BSP_RGB_LCD_DrawPixel** (uint16_t Xpos, uint16_t Ypos, uint32_t pixel)
void	**BSP_RGB_LCD_Clear** (uint32_t Color)
void	**BSP_RGB_LCD_ClearStringLine** (uint32_t Line)
void	**BSP_RGB_LCD_DisplayStringAtLine** (uint16_t Line, uint8_t *ptr)
void	**BSP_RGB_LCD_DisplayStringAt** (uint16_t Xpos, uint16_t Ypos, uint8_t *Text, **Text_AlignModeTypdef** Mode)
void	**BSP_RGB_LCD_DisplayChar** (uint16_t Xpos, uint16_t Ypos, uint8_t Ascii)
void	**BSP_RGB_LCD_DrawHLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length)
void	**BSP_RGB_LCD_DrawVLine** (uint16_t Xpos, uint16_t Ypos, uint16_t Length)
void	**BSP_RGB_LCD_DrawLine** (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2)
void	**BSP_RGB_LCD_DrawRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
void	**BSP_RGB_LCD_DrawCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)
void	**BSP_RGB_LCD_DrawPolygon** (**pPoint** Points, uint16_t PointCount)
void	**BSP_RGB_LCD_DrawEllipse** (int Xpos, int Ypos, int

	XRadius, int YRadius)
void	**BSP_RGB_LCD_DrawBitmap** (uint32_t Xpos, uint32_t Ypos, uint8_t *pbmp)
void	**BSP_RGB_LCD_FillRect** (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
void	**BSP_RGB_LCD_FillCircle** (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)
void	**BSP_RGB_LCD_FillPolygon** (**pPoint** Points, uint16_t PointCount)
void	**BSP_RGB_LCD_FillEllipse** (int Xpos, int Ypos, int XRadius, int YRadius)
void	**BSP_RGB_LCD_DisplayOff** (void)
void	**BSP_RGB_LCD_DisplayOn** (void)
void	**BSP_RGB_LCD_MspInit** (LTDC_HandleTypeDef *hltdc, void *Params)
void	**BSP_RGB_LCD_MspDeInit** (LTDC_HandleTypeDef *hltdc, void *Params)
void	**BSP_RGB_LCD_ClockConfig** (LTDC_HandleTypeDef *hltdc, void *Params)
void	**BSP_RGB_LCD_LTDC_IRQHandler** (void)
void	**BSP_RGB_LCD_LTDC_ER_IRQHandler** (void)
void	**BSP_RGB_LCD_DMA2D_IRQHandler** (void)

Function Documentation

void BSP_RGB_LCD_Clear (uint32_t Color)

void BSP_RGB_LCD_ClearStringLine (uint32_t Line)

void BSP_RGB_LCD_ClockConfig (LTDC_HandleTypeDef * hltdc,
void * Param
)

uint8_t BSP_RGB_LCD_DeInit (void)

void BSP_RGB_LCD_DisplayChar (uint16_t Xpos,
uint16_t Ypos,
uint8_t Ascii
)

void BSP_RGB_LCD_DisplayOff (void)

void BSP_RGB_LCD_DisplayOn (void)

void BSP_RGB_LCD_DisplayStringAt (uint16_t Xpo
uint16_t Ypo
uint8_t * Tex
Text_AlignModeTypdef Mo
)

void BSP_RGB_LCD_DisplayStringAtLine (uint16_t Line,
uint8_t * ptr
)

```
void BSP_RGB_LCD_DMA2D_IRQHandler ( void   )
```

```
void BSP_RGB_LCD_DrawBitmap ( uint32_t  Xpos,
                              uint32_t  Ypos,
                              uint8_t *  pbmp
                            )
```

```
void BSP_RGB_LCD_DrawCircle ( uint16_t  Xpos,
                              uint16_t  Ypos,
                              uint16_t  Radius
                            )
```

```
void BSP_RGB_LCD_DrawEllipse ( int  Xpos,
                               int  Ypos,
                               int  XRadius,
                               int  YRadius
                             )
```

```
void BSP_RGB_LCD_DrawHLine ( uint16_t  Xpos,
                             uint16_t  Ypos,
                             uint16_t  Length
                           )
```

```
void BSP_RGB_LCD_DrawLine ( uint16_t  x1,
                            uint16_t  y1,
                            uint16_t  x2,
                            uint16_t  y2
                          )
```

```
void BSP_RGB_LCD_DrawPixel ( uint16_t  Xpos,
                             uint16_t  Ypos,
```

```
                                    uint32_t  pixel
                                  )
```

```
void BSP_RGB_LCD_DrawPolygon (pPoint    Points,
                              uint16_t  PointCount
                             )
```

```
void BSP_RGB_LCD_DrawRect (uint16_t  Xpos,
                           uint16_t  Ypos,
                           uint16_t  Width,
                           uint16_t  Height
                          )
```

```
void BSP_RGB_LCD_DrawVLine (uint16_t  Xpos,
                            uint16_t  Ypos,
                            uint16_t  Length
                           )
```

```
void BSP_RGB_LCD_FillCircle (uint16_t  Xpos,
                             uint16_t  Ypos,
                             uint16_t  Radius
                            )
```

```
void BSP_RGB_LCD_FillEllipse (int  Xpos,
                              int  Ypos,
                              int  XRadius,
                              int  YRadius
                             )
```

```
void BSP_RGB_LCD_FillPolygon (pPoint    Points,
                              uint16_t  PointCount
```

)

void BSP_RGB_LCD_FillRect (uint16_t Xpos,
uint16_t Ypos,
uint16_t Width,
uint16_t Height
)

uint32_t BSP_RGB_LCD_GetBackColor (void)

sFONT* BSP_RGB_LCD_GetFont (void)

uint32_t BSP_RGB_LCD_GetTextColor (void)

uint32_t BSP_RGB_LCD_GetXSize (void)

uint32_t BSP_RGB_LCD_GetYSize (void)

uint8_t BSP_RGB_LCD_Init (void)

void BSP_RGB_LCD_LayerDefaultInit (uint16_t LayerIndex,
uint32_t FrameBuffer
)

void BSP_RGB_LCD_LayerRgb565Init (uint16_t LayerIndex,
uint32_t FB_Address
)

void BSP_RGB_LCD_LTDC_ER_IRQHandler (void)

void BSP_RGB_LCD_LTDC_IRQHandler (void)

void BSP_RGB_LCD_MspDeInit (LTDC_HandleTypeDef * hltdc,
void * Params
)

void BSP_RGB_LCD_MspInit (LTDC_HandleTypeDef * hltdc,
void * Params
)

uint32_t BSP_RGB_LCD_ReadPixel (uint16_t Xpos,
uint16_t Ypos
)

void BSP_RGB_LCD_Reload (uint32_t ReloadType)

void BSP_RGB_LCD_ResetColorKeying (uint32_t LayerIndex)

void BSP_RGB_LCD_ResetColorKeying_NoReload (uint32_t Laye

void BSP_RGB_LCD_SelectLayer (uint32_t LayerIndex)

void BSP_RGB_LCD_SetBackColor (uint32_t Color)

void BSP_RGB_LCD_SetColorKeying (uint32_t LayerIndex,
uint32_t RGBValue
)

void BSP_RGB_LCD_SetColorKeying_NoReload (uint32_t LayerIn
uint32_t RGBVa
)

void BSP_RGB_LCD_SetFont (sFONT * fonts)

void BSP_RGB_LCD_SetLayerAddress (uint32_t LayerIndex,
uint32_t Address
)

void BSP_RGB_LCD_SetLayerAddress_NoReload (uint32_t Layer
uint32_t Addre
)

void BSP_RGB_LCD_SetLayerVisible (uint32_t LayerIndex
FunctionalState State
)

void BSP_RGB_LCD_SetLayerVisible_NoReload (uint32_t
FunctionalState
)

void BSP_RGB_LCD_SetLayerWindow (uint16_t LayerIndex,
uint16_t Xpos,
uint16_t Ypos,
uint16_t Width,
uint16_t Height
)

void BSP_RGB_LCD_SetLayerWindow_NoReload (uint16_t LayerI
uint16_t Xpos,
uint16_t Ypos,
uint16_t Width,
uint16_t Height
)

void BSP_RGB_LCD_SetTextColor (uint32_t Color)

void BSP_RGB_LCD_SetTransparency (uint32_t LayerIndex,
uint8_t Transparency
)

void BSP_RGB_LCD_SetTransparency_NoReload (uint32_t LayerI
uint8_t Transp
)

void BSP_RGB_LCD_SetXSize (uint32_t imageWidthPixels)

void BSP_RGB_LCD_SetYSize (uint32_t imageHeightPixels)

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL_RGB_TS_Exported_Functio

Functions

uint8_t	**BSP_RGB_TS_Init** (uint16_t ts_SizeX, uint16_t ts_SizeY)
uint8_t	**BSP_RGB_TS_DeInit** (void)
uint8_t	**BSP_RGB_TS_GetState** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_RGB_TS_Get_GestureId** (**TS_StateTypeDef** *TS_State)
uint8_t	**BSP_RGB_TS_ITConfig** (void)
uint8_t	**BSP_RGB_TS_ITGetStatus** (void)
void	**BSP_RGB_TS_ITClear** (void)
uint8_t	**BSP_RGB_TS_ResetTouchData** (**TS_StateTypeDef** *TS_State)

Function Documentation

uint8_t BSP_RGB_TS_DeInit (void)

uint8_t BSP_RGB_TS_Get_GestureId (TS_StateTypeDef * TS_Stat

uint8_t BSP_RGB_TS_GetState (TS_StateTypeDef * TS_State)

uint8_t BSP_RGB_TS_Init (uint16_t ts_SizeX,
uint16_t ts_SizeY
)

void BSP_RGB_TS_ITClear (void)

uint8_t BSP_RGB_TS_ITConfig (void)

uint8_t BSP_RGB_TS_ITGetStatus (void)

uint8_t BSP_RGB_TS_ResetTouchData (TS_StateTypeDef * TS_St

STM32L4R9I_EVAL BSP User Manual

Exported Functions

STM32L4R9I_EVAL SD

Functions

uint8_t	**BSP_SD_Init** (void) Initializes the SD card device.
uint8_t	**BSP_SD_DeInit** (void) DeInitializes the SD card device.
uint8_t	**BSP_SD_ITConfig** (void) Configures Interrupt mode for SD detection pin.
uint8_t	**BSP_SD_IsDetected** (void) Detects if SD card is correctly plugged in the memory slot or not.
uint8_t	**BSP_SD_ReadBlocks** (uint32_t *pData, uint32_t ReadAddr, uint32_t NumOfBlocks, uint32_t Timeout) Reads block(s) from a specified address in an SD card, in polling mode.
uint8_t	**BSP_SD_WriteBlocks** (uint32_t *pData, uint32_t WriteAddr, uint32_t NumOfBlocks, uint32_t Timeout) Writes block(s) to a specified address in an SD card, in polling mode.
uint8_t	**BSP_SD_ReadBlocks_DMA** (uint32_t *pData, uint32_t ReadAddr, uint32_t NumOfBlocks) Reads block(s) from a specified address in an SD card, in DMA mode.
uint8_t	**BSP_SD_WriteBlocks_DMA** (uint32_t *pData, uint32_t WriteAddr, uint32_t NumOfBlocks) Writes block(s) to a specified address in an SD card, in DMA mode.
uint8_t	**BSP_SD_Erase** (uint32_t StartAddr, uint32_t EndAddr) Erases the specified memory area of the given SD card.
uint8_t	**BSP_SD_GetCardState** (void) Gets the current SD card data status.
void	**BSP_SD_GetCardInfo** (**BSP_SD_CardInfo** *CardInfo)

	Get SD information about specific SD card.
__weak void	**BSP_SD_MspInit** (SD_HandleTypeDef *hsd, void *Params) Initializes the SD MSP.
__weak void	**BSP_SD_MspDeInit** (SD_HandleTypeDef *hsd, void *Params) De-Initializes the SD MSP.
__weak void	**BSP_SD_AbortCallback** (void) BSP SD Abort callback.
__weak void	**BSP_SD_WriteCpltCallback** (void) BSP Tx Transfer completed callback.
__weak void	**BSP_SD_ReadCpltCallback** (void) BSP Rx Transfer completed callback.
void	**HAL_SD_AbortCallback** (SD_HandleTypeDef *hsd) SD Abort callback.
void	**HAL_SD_TxCpltCallback** (SD_HandleTypeDef *hsd) Tx Transfer completed callback.
void	**HAL_SD_RxCpltCallback** (SD_HandleTypeDef *hsd) Rx Transfer completed callback.

Function Documentation

void BSP_SD_AbortCallback (void)

BSP SD Abort callback.

Return values:
None

Definition at line 465 of file stm32l4r9i_eval_sd.c.

Referenced by HAL_SD_AbortCallback().

uint8_t BSP_SD_DeInit (void)

DeInitializes the SD card device.

Return values:
SD status

Definition at line 156 of file stm32l4r9i_eval_sd.c.

References BSP_SD_MspDeInit(), hsd_eval, MSD_ERROR, MSD_OK, and UseExtiModeDetection.

uint8_t BSP_SD_Erase (uint32_t StartAddr, uint32_t EndAddr)

Erases the specified memory area of the given SD card.

Parameters:
StartAddr,: Start byte address

EndAddr,: End byte address

Return values:

SD status

Definition at line **332** of file **stm32l4r9i_eval_sd.c**.

References **hsd_eval**, **MSD_ERROR**, and **MSD_OK**.

void BSP_SD_GetCardInfo (BSP_SD_CardInfo * CardInfo)

Get SD information about specific SD card.

Parameters:

CardInfo,: Pointer to HAL_SD_CardInfoTypedef structure

Return values:

None

Definition at line **365** of file **stm32l4r9i_eval_sd.c**.

References **hsd_eval**.

uint8_t BSP_SD_GetCardState (void)

Gets the current SD card data status.

Return values:

Data transfer state. This value can be one of the following values:

- SD_TRANSFER_OK: No data transfer is acting
- SD_TRANSFER_BUSY: Data transfer is acting

Definition at line **355** of file **stm32l4r9i_eval_sd.c**.

References **hsd_eval**, **SD_TRANSFER_BUSY**, and **SD_TRANSFER_OK**.

uint8_t BSP_SD_Init (void)

Initializes the SD card device.

Return values:

SD status

Definition at line **115** of file **stm32l4r9i_eval_sd.c**.

References **BSP_IO_ConfigPin()**, **BSP_IO_Init()**, **BSP_SD_IsDetected()**, **BSP_SD_MspInit()**, **hsd_eval**, **MSD_ERROR**, **MSD_ERROR_SD_NOT_PRESENT**, **MSD_OK**, **SD_DETECT_PIN**, and **SD_PRESENT**.

uint8_t BSP_SD_IsDetected (void)

Detects if SD card is correctly plugged in the memory slot or not.

Return values:

Returns if SD is detected or not

Definition at line **204** of file **stm32l4r9i_eval_sd.c**.

References **BSP_IO_ConfigPin()**, **BSP_IO_ReadPin()**, **SD_DETECT_PIN**, **SD_NOT_PRESENT**, **SD_PRESENT**, and **UseExtiModeDetection**.

Referenced by **BSP_SD_Init()**.

uint8_t BSP_SD_ITConfig (void)

Configures Interrupt mode for SD detection pin.

Return values:

IO_OK,: if all initializations are OK. Other value if error.

Definition at line **181** of file **stm32l4r9i_eval_sd.c**.

References **BSP_IO_ConfigPin()**, **BSP_IO_ReadPin()**, **SD_DETECT_PIN**, and **UseExtiModeDetection**.

void BSP_SD_MspDeInit (SD_HandleTypeDef * hsd,
void * Params
)

De-Initializes the SD MSP.

Return values:

None

Definition at line **432** of file **stm32l4r9i_eval_sd.c**.

References **SDMMCx_IRQn**.

Referenced by **BSP_SD_DeInit()**.

void BSP_SD_MspInit (SD_HandleTypeDef * hsd,
void * Params
)

Initializes the SD MSP.

Return values:

None

Definition at line **375** of file **stm32l4r9i_eval_sd.c**.

References **SDMMCx_IRQn**.

Referenced by **BSP_SD_Init()**.

uint8_t BSP_SD_ReadBlocks (uint32_t * pData,
uint32_t ReadAddr,
uint32_t NumOfBlocks,
uint32_t Timeout
)

Reads block(s) from a specified address in an SD card, in polling mode.

Parameters:

pData,:	Pointer to the buffer that will contain the data to transmit
ReadAddr,:	Address from where data is to be read
NumOfBlocks,:	Number of SD blocks to read
Timeout,:	Timeout for read operation

Return values:

SD status

Definition at line **238** of file **stm32l4r9i_eval_sd.c**.

References **hsd_eval**, **MSD_ERROR**, and **MSD_OK**.

uint8_t BSP_SD_ReadBlocks_DMA (uint32_t * pData,
uint32_t ReadAddr,
uint32_t NumOfBlocks
)

Reads block(s) from a specified address in an SD card, in DMA mode.

Parameters:

pData,:	Pointer to the buffer that will contain the data to transmit
ReadAddr,:	Address from where data is to be read
NumOfBlocks,:	Number of SD blocks to read

Return values:

SD status

Definition at line **285** of file **stm32l4r9i_eval_sd.c**.

References **hsd_eval**, **MSD_ERROR**, and **MSD_OK**.

void BSP_SD_ReadCpltCallback (void)

BSP Rx Transfer completed callback.

Return values:

None

Definition at line **483** of file **stm32l4r9i_eval_sd.c**.

Referenced by **HAL_SD_RxCpltCallback()**.

uint8_t BSP_SD_WriteBlocks (uint32_t * pData,
uint32_t WriteAddr,
uint32_t NumOfBlocks,
uint32_t Timeout
)

Writes block(s) to a specified address in an SD card, in polling mode.

Parameters:

pData,:	Pointer to the buffer that will contain the data to transmit
WriteAddr,:	Address from where data is to be written
NumOfBlocks,:	Number of SD blocks to write
Timeout,:	Timeout for write operation

Return values:

SD status

Definition at line **262** of file **stm32l4r9i_eval_sd.c**.

References **hsd_eval**, **MSD_ERROR**, and **MSD_OK**.

uint8_t BSP_SD_WriteBlocks_DMA (uint32_t * pData,
uint32_t WriteAddr,
uint32_t NumOfBlocks
)

Writes block(s) to a specified address in an SD card, in DMA mode.

Parameters:

pData,:	Pointer to the buffer that will contain the data to transmit
WriteAddr,:	Address from where data is to be written
NumOfBlocks,:	Number of SD blocks to write

Return values:

SD status

Definition at line **309** of file **stm32l4r9i_eval_sd.c**.

References **hsd_eval**, **MSD_ERROR**, and **MSD_OK**.

void BSP_SD_WriteCpltCallback (void)

BSP Tx Transfer completed callback.

Return values:
None

Definition at line **474** of file **stm32l4r9i_eval_sd.c**.

Referenced by **HAL_SD_TxCpltCallback()**.

void HAL_SD_AbortCallback (SD_HandleTypeDef * hsd)

SD Abort callback.

Parameters:
hsd,: SD handle

Return values:
None

Definition at line **493** of file **stm32l4r9i_eval_sd.c**.

References **BSP_SD_AbortCallback()**.

void HAL_SD_RxCpltCallback (SD_HandleTypeDef * hsd)

Rx Transfer completed callback.

Parameters:
hsd,: SD handle

Return values:
None

Definition at line **513** of file **stm32l4r9i_eval_sd.c**.

References **BSP_SD_ReadCpltCallback()**.

void HAL_SD_TxCpltCallback (SD_HandleTypeDef * hsd)

Tx Transfer completed callback.

Parameters:

hsd,: SD handle

Return values:

None

Definition at line **503** of file **stm32l4r9i_eval_sd.c**.

References **BSP_SD_WriteCpltCallback()**.

STM32L4R9I_EVAL BSP User Manual

Exported Types

STM32L4R9I_EVAL SD

Defines

#define **BSP_SD_CardInfo** HAL_SD_CardInfoTypeDef
SD Card information structure.

Define Documentation

#define BSP_SD_CardInfo HAL_SD_CardInfoTypeDef

SD Card information structure.

Definition at line **69** of file **stm32l4r9i_eval_sd.h**.

STM32L4R9I_EVAL BSP User Manual

Exported Types

STM32L4R9I_EVAL Common

Enumerations

enum	**Led_TypeDef** {
	LED1 = 0, **LED_GREEN** = LED1, **LED2** = 1, **LED_ORANGE** = LED2,
	LED3 = 2, **LED_RED** = LED3, **LED4** = 3, **LED_BLUE** = LED4
	}
	LED Types Definition. More...

enum	**Button_TypeDef** {
	BUTTON_WAKEUP = 0, **BUTTON_SEL** = 1, **BUTTON_LEFT** = 2, **BUTTON_RIGHT** = 3,
	BUTTON_DOWN = 4, **BUTTON_UP** = 5
	}
	BUTTON Types Definition. More...

enum	**ButtonMode_TypeDef** { **BUTTON_MODE_GPIO** = 0, **BUTTON_MODE_EXTI** = 1 }

enum	**JOYState_TypeDef** {
	JOY_SEL = 0, **JOY_LEFT** = 1, **JOY_RIGHT** = 2, **JOY_DOWN** = 3,
	JOY_UP = 4, **JOY_NONE** = 5
	}
	JOYSTICK Types Definition. More...

enum	**JOYMode_TypeDef** { **JOY_MODE_GPIO** = 0, **JOY_MODE_EXTI** = 1 }
enum	**COM_TypeDef** { **COM1** = 0 } COM Types Definition. More...

Enumeration Type Documentation

enum Button_TypeDef

BUTTON Types Definition.

Enumerator:

BUTTON_WAKEUP
BUTTON_SEL
BUTTON_LEFT
BUTTON_RIGHT
BUTTON_DOWN
BUTTON_UP

Definition at line **95** of file **stm32l4r9i_eval.h**.

enum ButtonMode_TypeDef

Enumerator:

BUTTON_MODE_GPIO
BUTTON_MODE_EXTI

Definition at line **106** of file **stm32l4r9i_eval.h**.

enum COM_TypeDef

COM Types Definition.

Enumerator:

COM1

Definition at line **137** of file **stm32l4r9i_eval.h**.

enum JOYMode_TypeDef

Enumerator:

JOY_MODE_GPIO

JOY_MODE_EXTI

Definition at line **127** of file **stm32l4r9i_eval.h**.

enum JOYState_TypeDef

JOYSTICK Types Definition.

Enumerator:

JOY_SEL

JOY_LEFT

JOY_RIGHT

JOY_DOWN

JOY_UP

JOY_NONE

Definition at line **116** of file **stm32l4r9i_eval.h**.

enum Led_TypeDef

LED Types Definition.

Enumerator:

LED1

LED_GREEN

LED2

LED_ORANGE

LED3

LED_RED
LED4
LED_BLUE

Definition at line **76** of file **stm32l4r9i_eval.h**.

STM32L4R9I_EVAL BSP User Manual

Exported Variables

STM32L4R9I_EVAL Common

Variables

GPIO_TypeDef *	**LED_PORT** [**LEDn**] LED variables.
const uint16_t	**LED_PIN** [**LEDn**]
GPIO_TypeDef *	**BUTTON_PORT** [**BUTTONn**] BUTTON variables.
const uint16_t	**BUTTON_PIN** [**BUTTONn**]
const uint8_t	**BUTTON_IRQn** [**BUTTONn**]
const uint16_t	**JOY_PIN** [**JOYn**] JOYSTICK variables.
const uint8_t	**JOY_IRQn** [**JOYn**]
USART_TypeDef *	**COM_USART** [**COMn**] = {**EVAL_COM1**} COM variables.
GPIO_TypeDef *	**COM_TX_PORT** [**COMn**] = {**EVAL_COM1_TX_GPIO_PORT**}
GPIO_TypeDef *	**COM_RX_PORT** [**COMn**] = {**EVAL_COM1_RX_GPIO_PORT**}
const uint16_t	**COM_TX_PIN** [**COMn**] = {**EVAL_COM1_TX_PIN**}
const uint16_t	**COM_RX_PIN** [**COMn**] = {**EVAL_COM1_RX_PIN**}
const uint16_t	**COM_TX_AF** [**COMn**] = {**EVAL_COM1_TX_AF**}
const uint16_t	**COM_RX_AF** [**COMn**] = {**EVAL_COM1_RX_AF**}
uint32_t	**hi2c_evalTimeout** = **EVAL_I2Cx_TIMEOUT_MAX** BUS variables.
I2C_HandleTypeDef	**hi2c_eval**
static uint32_t	**v_bspI2cUsers** = **BSP_I2C_NO_USER**
static ADC_HandleTypeDef	**hEvalADC** = {0}

Variable Documentation

const uint8_t BUTTON_IRQn[BUTTONn]

Initial value:

```
          {WAKEUP_BUTTON_EXTI_IRQn,

SEL_JOY_EXTI_IRQn,

LEFT_JOY_EXTI_IRQn,

RIGHT_JOY_EXTI_IRQn,

DOWN_JOY_EXTI_IRQn,

UP_JOY_EXTI_IRQn}
```

Definition at line **132** of file **stm32l4r9i_eval.c**.

const uint16_t BUTTON_PIN[BUTTONn]

Initial value:

```
           {WAKEUP_BUTTON_PIN,

SEL_JOY_PIN,

LEFT_JOY_PIN,

RIGHT_JOY_PIN,

DOWN_JOY_PIN,

UP_JOY_PIN}
```

Definition at line **125** of file **stm32l4r9i_eval.c**.

GPIO_TypeDef* BUTTON_PORT[BUTTONn]

Initial value:

```
          {WAKEUP_BUTTON_GPIO_PORT,

(GPIO_TypeDef *)0xFFFF,

(GPIO_TypeDef *)0xFFFF,

(GPIO_TypeDef *)0xFFFF,

(GPIO_TypeDef *)0xFFFF,

(GPIO_TypeDef *)0xFFFF}
```

BUTTON variables.

Definition at line **117** of file **stm32l4r9i_eval.c**.

Referenced by **BSP_PB_DeInit()**, **BSP_PB_GetState()**, and **BSP_PB_Init()**.

const uint16_t COM_RX_AF[COMn] = {EVAL_COM1_RX_AF}

Definition at line **170** of file **stm32l4r9i_eval.c**.

const uint16_t COM_RX_PIN[COMn] = {EVAL_COM1_RX_PIN}

Definition at line **166** of file **stm32l4r9i_eval.c**.

GPIO_TypeDef* COM_RX_PORT[COMn] = {EVAL_COM1_RX_GPIO_

Definition at line **162** of file **stm32l4r9i_eval.c**.

const uint16_t COM_TX_AF[COMn] = {EVAL_COM1_TX_AF}

Definition at line **168** of file **stm32l4r9i_eval.c**.

const uint16_t COM_TX_PIN[COMn] = {EVAL_COM1_TX_PIN}

Definition at line **164** of file **stm32l4r9i_eval.c**.

GPIO_TypeDef* COM_TX_PORT[COMn] = {EVAL_COM1_TX_GPIO_

Definition at line **160** of file **stm32l4r9i_eval.c**.

USART_TypeDef* COM_USART[COMn] = {EVAL_COM1}

COM variables.

Definition at line **158** of file **stm32l4r9i_eval.c**.

ADC_HandleTypeDef hEvalADC = {0} `[static]`

Definition at line **186** of file **stm32l4r9i_eval.c**.

I2C_HandleTypeDef hi2c_eval

Definition at line **178** of file **stm32l4r9i_eval.c**.

uint32_t hi2c_evalTimeout = EVAL_I2Cx_TIMEOUT_MAX

BUS variables.

Definition at line **177** of file **stm32l4r9i_eval.c**.

const uint8_t JOY_IRQn[JOYn]

Initial value:

```
 {SEL_JOY_EXTI_IRQn,
                                    LEFT_JOY_EXTI_IRQn
,
                                    RIGHT_JOY_EXTI_IR
Qn,
                                    DOWN_JOY_EXTI_IRQn
,
                                    UP_JOY_EXTI_IRQn}
```

Definition at line **149** of file **stm32l4r9i_eval.c**.

const uint16_t JOY_PIN[JOYn]

Initial value:

```
 {SEL_JOY_PIN,
                                    LEFT_JOY_PIN,
                                    RIGHT_JOY_PIN,
                                    DOWN_JOY_PIN,
                                    UP_JOY_PIN}
```

JOYSTICK variables.

Definition at line **143** of file **stm32l4r9i_eval.c**.

const uint16_t LED_PIN[LEDn]

Initial value:

```
                   {LED1_PIN,

LED2_PIN,
```

```
LED3_PIN,

LED4_PIN}
```

Definition at line **109** of file **stm32l4r9i_eval.c**.

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

GPIO_TypeDef* LED_PORT[LEDn]

Initial value:

```
                {LED1_GPIO_PORT,

LED2_GPIO_PORT,

LED3_GPIO_PORT,

LED4_GPIO_PORT}
```

LED variables.

Definition at line **104** of file **stm32l4r9i_eval.c**.

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

uint32_t v_bspI2cUsers = BSP_I2C_NO_USER [static]

Definition at line **179** of file **stm32l4r9i_eval.c**.

Referenced by **I2Cx_Error()**.

User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Defines

BUTTON Constants

Exported Constants

Defines

#define	**JOYn** 5
#define	**BUTTONn** 1 + **JOYn**
#define	**WAKEUP_BUTTON_PIN** GPIO_PIN_13 Wakeup/Tamper push-button.
#define	**WAKEUP_BUTTON_GPIO_PORT** GPIOC
#define	**WAKEUP_BUTTON_GPIO_CLK_ENABLE**() __HAL_RCC_G
#define	**WAKEUP_BUTTON_GPIO_CLK_DISABLE**() __HAL_RCC_(
#define	**WAKEUP_BUTTON_EXTI_IRQn** EXTI15_10_IRQn
#define	**BUTTONx_GPIO_CLK_ENABLE**(__BUTTON__) do { if((__B **BUTTON_WAKEUP**) { **WAKEUP_BUTTON_GPIO_CLK_ENA**
#define	**BUTTONx_GPIO_CLK_DISABLE**(__BUTTON__) do { if((__E **BUTTON_WAKEUP**) { **WAKEUP_BUTTON_GPIO_CLK_DISA**
#define	**SEL_JOY_PIN** **IO_PIN_0** Joystick Sel push-button.
#define	**SEL_JOY_EXTI_IRQn** EXTI0_IRQn
#define	**DOWN_JOY_PIN** **IO_PIN_1** Joystick Down push-button.
#define	**DOWN_JOY_EXTI_IRQn** EXTI0_IRQn
#define	**LEFT_JOY_PIN** **IO_PIN_2** Joystick Left push-button.
#define	**LEFT_JOY_EXTI_IRQn** EXTI0_IRQn
#define	**RIGHT_JOY_PIN** **IO_PIN_3** Joystick Right push-button.
#define	**RIGHT_JOY_EXTI_IRQn** EXTI0_IRQn
#define	**UP_JOY_PIN** **IO_PIN_4** Joystick Up push-button.
#define	**UP_JOY_EXTI_IRQn** EXTI0_IRQn
#define	**JOY_ALL_PINS** (**RIGHT_JOY_PIN** | **LEFT_JOY_PIN** | **UP_J** **DOWN_JOY_PIN** | **SEL_JOY_PIN**)
#define	**JOY_NONE_PIN** **JOY_ALL_PINS**

Define Documentation

#define BUTTONn 1 + JOYn

Definition at line 189 of file stm32l4r9i_eval.h.

#define BUTTONx_GPIO_CLK_DISABLE (__BUTTON__) do { if((

Definition at line 202 of file stm32l4r9i_eval.h.

#define BUTTONx_GPIO_CLK_ENABLE (__BUTTON__) do { if((

Definition at line 200 of file stm32l4r9i_eval.h.

Referenced by BSP_PB_DeInit(), and BSP_PB_Init().

#define DOWN_JOY_EXTI_IRQn EXTI0_IRQn

Definition at line 225 of file stm32l4r9i_eval.h.

#define DOWN_JOY_PIN IO_PIN_1

Joystick Down push-button.

Definition at line 222 of file stm32l4r9i_eval.h.

Referenced by BSP_JOY_GetState().

#define JOY_ALL_PINS (RIGHT_JOY_PIN | LEFT_JOY_PIN | UP_J

Definition at line 264 of file stm32l4r9i_eval.h.

Referenced by **BSP_JOY_DeInit()**, **BSP_JOY_GetState()**, and **BSP_JOY_Init()**.

#define JOY_NONE_PIN JOY_ALL_PINS

Definition at line **265** of file **stm32l4r9i_eval.h**.

#define JOYn 5

Definition at line **188** of file **stm32l4r9i_eval.h**.

#define LEFT_JOY_EXTI_IRQn EXTI0_IRQn

Definition at line **237** of file **stm32l4r9i_eval.h**.

#define LEFT_JOY_PIN IO_PIN_2

Joystick Left push-button.

Definition at line **234** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_JOY_GetState()**.

#define RIGHT_JOY_EXTI_IRQn EXTI0_IRQn

Definition at line **249** of file **stm32l4r9i_eval.h**.

#define RIGHT_JOY_PIN IO_PIN_3

Joystick Right push-button.

Definition at line **246** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_JOY_GetState()**.

#define SEL_JOY_EXTI_IRQn EXTI0_IRQn

Definition at line **213** of file **stm32l4r9i_eval.h**.

#define SEL_JOY_PIN IO_PIN_0

Joystick Sel push-button.

Definition at line **212** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_JOY_GetState()**.

#define UP_JOY_EXTI_IRQn EXTI0_IRQn

Definition at line **261** of file **stm32l4r9i_eval.h**.

#define UP_JOY_PIN IO_PIN_4

Joystick Up push-button.

Definition at line **258** of file **stm32l4r9i_eval.h**.

Referenced by **BSP_JOY_GetState()**.

#define WAKEUP_BUTTON_EXTI_IRQn EXTI15_10_IRQn

Definition at line **198** of file **stm32l4r9i_eval.h**.

#define WAKEUP_BUTTON_GPIO_CLK_DISABLE ( ) __HAL_RCC

Definition at line 197 of file stm32l4r9i_eval.h.

#define WAKEUP_BUTTON_GPIO_CLK_ENABLE ( ) __HAL_RCC_

Definition at line 196 of file stm32l4r9i_eval.h.

#define WAKEUP_BUTTON_GPIO_PORT GPIOC

Definition at line 195 of file stm32l4r9i_eval.h.

#define WAKEUP_BUTTON_PIN GPIO_PIN_13

Wakeup/Tamper push-button.

Definition at line 194 of file stm32l4r9i_eval.h.

STM32L4R9I_EVAL BSP User Manual

Defines

COM Constants

Exported Constants

Defines

#define	**COMn** 1
#define	**EVAL_COM1** LPUART1 Definition for COM port1, connected to USART1.
#define	**EVAL_COM1_CLK_ENABLE**() __HAL_RCC_LPUART1_CLK
#define	**EVAL_COM1_CLK_DISABLE**() __HAL_RCC_LPUART1_CL
#define	**EVAL_COM1_TX_PIN** GPIO_PIN_7
#define	**EVAL_COM1_TX_GPIO_PORT** GPIOG
#define	**EVAL_COM1_TX_GPIO_CLK_ENABLE**() __HAL_RCC_GPI
#define	**EVAL_COM1_TX_GPIO_CLK_DISABLE**() __HAL_RCC_GP
#define	**EVAL_COM1_TX_AF** GPIO_AF8_LPUART1
#define	**EVAL_COM1_RX_PIN** GPIO_PIN_8
#define	**EVAL_COM1_RX_GPIO_PORT** GPIOG
#define	**EVAL_COM1_RX_GPIO_CLK_ENABLE**() __HAL_RCC_GPI
#define	**EVAL_COM1_RX_GPIO_CLK_DISABLE**() __HAL_RCC_GP
#define	**EVAL_COM1_RX_AF** GPIO_AF8_LPUART1
#define	**EVAL_COM1_IRQn** LPUART1_IRQn
#define	**COMx_CLK_ENABLE**(__INDEX__) do { if((__INDEX__) == C **EVAL_COM1_CLK_ENABLE**(); } } while(0)
#define	**COMx_CLK_DISABLE**(__INDEX__) do { if((__INDEX__) == **EVAL_COM1_CLK_DISABLE**(); } } while(0)
#define	**COMx_TX_GPIO_CLK_ENABLE**(__INDEX__) do { if((__IND **EVAL_COM1_TX_GPIO_CLK_ENABLE**(); } } while(0)
#define	**COMx_TX_GPIO_CLK_DISABLE**(__INDEX__) do { if((__IND **EVAL_COM1_TX_GPIO_CLK_DISABLE**(); } } while(0)
#define	**COMx_RX_GPIO_CLK_ENABLE**(__INDEX__) do { if((__IND **EVAL_COM1_RX_GPIO_CLK_ENABLE**(); } } while(0)
#define	**COMx_RX_GPIO_CLK_DISABLE**(__INDEX__) do { if((__IND **EVAL_COM1_RX_GPIO_CLK_DISABLE**(); } } while(0)

Define Documentation

#define COMn 1

Definition at line 274 of file stm32l4r9i_eval.h.

#define COMx_CLK_DISABLE (__INDEX__) do { if((__INDEX__)

Definition at line 298 of file stm32l4r9i_eval.h.

#define COMx_CLK_ENABLE (__INDEX__) do { if((__INDEX__) =

Definition at line 297 of file stm32l4r9i_eval.h.

Referenced by BSP_COM_Init().

#define COMx_RX_GPIO_CLK_DISABLE (__INDEX__) do { if((__

Definition at line 304 of file stm32l4r9i_eval.h.

#define COMx_RX_GPIO_CLK_ENABLE (__INDEX__) do { if((__

Definition at line 303 of file stm32l4r9i_eval.h.

Referenced by BSP_COM_Init().

#define COMx_TX_GPIO_CLK_DISABLE (__INDEX__) do { if((__

Definition at line 301 of file stm32l4r9i_eval.h.

#define COMx_TX_GPIO_CLK_ENABLE (__INDEX__) do { if((__

Definition at line 300 of file stm32l4r9i_eval.h.

Referenced by BSP_COM_Init().

#define EVAL_COM1 LPUART1

Definition for COM port1, connected to USART1.

Definition at line 279 of file stm32l4r9i_eval.h.

#define EVAL_COM1_CLK_DISABLE ( ) __HAL_RCC_LPUART1_C

Definition at line 281 of file stm32l4r9i_eval.h.

#define EVAL_COM1_CLK_ENABLE ( ) __HAL_RCC_LPUART1_C

Definition at line 280 of file stm32l4r9i_eval.h.

#define EVAL_COM1_IRQn LPUART1_IRQn

Definition at line 295 of file stm32l4r9i_eval.h.

#define EVAL_COM1_RX_AF GPIO_AF8_LPUART1

Definition at line 293 of file stm32l4r9i_eval.h.

#define EVAL_COM1_RX_GPIO_CLK_DISABLE ( ) __HAL_RCC_C

Definition at line 292 of file stm32l4r9i_eval.h.

#define EVAL_COM1_RX_GPIO_CLK_ENABLE ( ) __HAL_RCC_G

Definition at line 291 of file stm32l4r9i_eval.h.

#define EVAL_COM1_RX_GPIO_PORT GPIOG

Definition at line 290 of file stm32l4r9i_eval.h.

#define EVAL_COM1_RX_PIN GPIO_PIN_8

Definition at line 289 of file stm32l4r9i_eval.h.

#define EVAL_COM1_TX_AF GPIO_AF8_LPUART1

Definition at line 287 of file stm32l4r9i_eval.h.

#define EVAL_COM1_TX_GPIO_CLK_DISABLE ( ) __HAL_RCC_G

Definition at line 286 of file stm32l4r9i_eval.h.

#define EVAL_COM1_TX_GPIO_CLK_ENABLE ( ) __HAL_RCC_G

Definition at line 285 of file stm32l4r9i_eval.h.

#define EVAL_COM1_TX_GPIO_PORT GPIOG

Definition at line 284 of file stm32l4r9i_eval.h.

#define EVAL_COM1_TX_PIN GPIO_PIN_7

Definition at line **283** of file **stm32l4r9i_eval.h**.

STM32L4R9I_EVAL BSP User Manual

Private Macros

STM32L4R9I_EVAL AUDIO

Defines

#define	**SAIClockDivider**(__FREQUENCY__)
#define	**DFSDMOverSampling**(__FREQUENCY__)
#define	**DFSDMClockDivider**(__FREQUENCY__)
#define	**DFSDMFilterOrder**(__FREQUENCY__)
#define	**DFSDMRightBitShift**(__FREQUENCY__)
#define	**SaturaLH**(N, L, H) (((N)<(L))?(L):(((N)>(H))?(H):(N)))

Define Documentation

#define DFSDMClockDivider (__FREQUENCY__)

Value:

```
(__FREQUENCY__ == AUDIO_FREQUENCY_8K)  ? 24 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_11K) ?
4 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_16K) ?
24 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_22K) ?
4 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_32K) ?
24 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_44K) ?
4  \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_48K) ?
32 : 32  \
```

Definition at line **201** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_Init()**.

#define DFSDMFilterOrder (__FREQUENCY__)

Value:

```
(__FREQUENCY__ == AUDIO_FREQUENCY_8K)  ? DFSDM_FI
LTER_SINC3_ORDER \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_11K) ?
DFSDM_FILTER_SINC3_ORDER \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_16K) ?
DFSDM_FILTER_SINC3_ORDER \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_22K) ?
DFSDM_FILTER_SINC3_ORDER \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_32K) ?
DFSDM_FILTER_SINC4_ORDER \
```

```
      : (__FREQUENCY__ == AUDIO_FREQUENCY_44K) ?
DFSDM_FILTER_SINC4_ORDER  \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_48K) ?
DFSDM_FILTER_SINC4_ORDER : DFSDM_FILTER_SINC5_ORD
ER  \
```

Definition at line **210** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_Init()**.

#define DFSDMOverSampling (__FREQUENCY__)

Value:

```
(__FREQUENCY__ == AUDIO_FREQUENCY_8K)  ? 256 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_11K) ?
256 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_16K) ?
128 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_22K) ?
128 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_32K) ?
64 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_44K) ?
64  \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_48K) ?
32 : 16  \
```

Definition at line **192** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_Init()**.

#define DFSDMRightBitShift (__FREQUENCY__)

Value:

```
(__FREQUENCY__ == AUDIO_FREQUENCY_8K)  ? 5 \
      : (__FREQUENCY__ == AUDIO_FREQUENCY_11K) ?
```

```
5 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_16K) ?
2 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_22K) ?
2 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_32K) ?
5 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_44K) ?
5  \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_48K) ?
1 : 2  \
```

Definition at line **219** of file **stm32l4r9i_eval_audio.c**.

Referenced by **AUDIO_DFSDMx_Init()**.

#define SAIClockDivider (__FREQUENCY__)

Value:

```
(__FREQUENCY__ == AUDIO_FREQUENCY_8K)  ? 24 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_11K) ?
4 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_16K) ?
12 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_22K) ?
2 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_32K) ?
6 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_44K) ?
1 \
       : (__FREQUENCY__ == AUDIO_FREQUENCY_48K) ?
4 : 2  \
```

Definition at line **182** of file **stm32l4r9i_eval_audio.c**.

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

```
#define SaturaLH ( N,
                   L,
                   H
                 )    (((N)<(L))?(L):(((N)>(H))?(H):(N)))
```

Definition at line **229** of file **stm32l4r9i_eval_audio.c**.

Referenced by **HAL_DFSDM_FilterRegConvCpltCallback()**, and **HAL_DFSDM_FilterRegConvHalfCpltCallback()**.

STM32L4R9I_EVAL BSP User Manual

Exported Macros

STM32L4R9I_EVAL 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 **232** of file **stm32l4r9i_eval_audio.h**.

Referenced by **BSP_AUDIO_OUT_Play()**.

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL AUDIO

Variables

static **AUDIO_OUT_TypeDef**	**hAudioOut** = {0}
static **AUDIO_IN_TypeDef**	**hAudioIn** = {0}
static DMA_HandleTypeDef	**hDmaSai**
static uint32_t	**DmaLeftRecHalfBuffCplt**
static uint32_t	**DmaLeftRecBuffCplt**
static uint32_t	**DmaRightRecHalfBuffCplt**
static uint32_t	**DmaRightRecBuffCplt**

Variable Documentation

uint32_t DmaLeftRecBuffCplt [static]

Definition at line 249 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Init(), and HAL_DFSDM_FilterRegConvCpltCallback().

uint32_t DmaLeftRecHalfBuffCplt [static]

Definition at line 248 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Init(), and HAL_DFSDM_FilterRegConvHalfCpltCallback().

uint32_t DmaRightRecBuffCplt [static]

Definition at line 251 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Init(), and HAL_DFSDM_FilterRegConvCpltCallback().

uint32_t DmaRightRecHalfBuffCplt [static]

Definition at line 250 of file stm32l4r9i_eval_audio.c.

Referenced by BSP_AUDIO_IN_Init(), and HAL_DFSDM_FilterRegConvHalfCpltCallback().

AUDIO_IN_TypeDef hAudioIn = {0} [static]

Definition at line **243** of file **stm32l4r9i_eval_audio.c**.

AUDIO_OUT_TypeDef hAudioOut = {0} [static]

Definition at line **240** of file **stm32l4r9i_eval_audio.c**.

DMA_HandleTypeDef hDmaSai [static]

Definition at line **246** of file **stm32l4r9i_eval_audio.c**.

Referenced by **HAL_SAI_MspDeInit()**, and **HAL_SAI_MspInit()**.

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL OSPI RAM

Variables

OSPI_HandleTypeDef	**OspiRamHandle**
DMA_HandleTypeDef	**DmaOspiRamHandle**

Variable Documentation

DMA_HandleTypeDef DmaOspiRamHandle

Definition at line **98** of file **stm32l4r9i_eval_ospi_ram.c**.

Referenced by **OSPI_RAM_MspDeInit()**, and **OSPI_RAM_MspInit()**.

OSPI_HandleTypeDef OspiRamHandle

Definition at line **97** of file **stm32l4r9i_eval_ospi_ram.c**.

Referenced by **BSP_OSPI_RAM_DeInit()**, **BSP_OSPI_RAM_DMA_IRQHandler()**, **BSP_OSPI_RAM_EnableMemoryMappedMode()**, **BSP_OSPI_RAM_EnterDeepPowerDown()**, **BSP_OSPI_RAM_Init()**, **BSP_OSPI_RAM_IRQHandler()**, **BSP_OSPI_RAM_LeaveDeepPowerDown()**, **BSP_OSPI_RAM_Read()**, **BSP_OSPI_RAM_Read_DMA()**, **BSP_OSPI_RAM_Write()**, **BSP_OSPI_RAM_Write_DMA()**, and **OSPI_RAM_MspInit()**.

STM32L4R9I_EVAL BSP User Manual

Private Functions

STM32L4R9I_EVAL DSI LCD

Functions

void	**HAL_DSI_EndOfRefreshCallback** (DSI_HandleTypeDef *hdsi) End of Refresh DSI callback.
static void	**LCD_Reset** (void) LCD Reset Hw reset the LCD DSI activating its XRES signal (active low for some time) and desactivating it later.
static void	**DrawChar** (uint16_t Xpos, uint16_t Ypos, const uint8_t *c) Draws a character on LCD.
static void	**FillTriangle** (uint16_t x1, uint16_t x2, uint16_t x3, uint16_t y1, uint16_t y2, uint16_t y3) Fills a triangle (between 3 points).
static void	**LL_FillBuffer** (uint32_t LayerIndex, void *pDst, uint32_t xSize, uint32_t ySize, uint32_t OffLine, uint32_t ColorIndex) Fills a buffer.
static void	**LL_ConvertLineToARGB8888** (void *pSrc, void *pDst, uint32_t xSize, uint32_t ColorMode) Converts a line to an ARGB8888 pixel format.

Function Documentation

```
static void DrawChar ( uint16_t          Xpos,
                       uint16_t          Ypos,
                       const uint8_t *   c
                     )                   [static]
```

Draws a character on LCD.

Parameters:

- **Xpos,:** Line where to display the character shape
- **Ypos,:** Start column address
- **c,:** Pointer to the character data

Definition at line **1726** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ActiveLayer**, **BSP_DSI_LCD_DrawPixel()**, and **LCD_DrawPropTypeDef::pFont**.

Referenced by **BSP_DSI_LCD_DisplayChar()**.

```
static void FillTriangle ( uint16_t  x1,
                           uint16_t  x2,
                           uint16_t  x3,
                           uint16_t  y1,
                           uint16_t  y2,
                           uint16_t  y3
                         )           [static]
```

Fills a triangle (between 3 points).

Parameters:

- **x1,:** **Point** 1 X position

y1,: **Point** 1 Y position
x2,: **Point** 2 X position
y2,: **Point** 2 Y position
x3,: **Point** 3 X position
y3,: **Point** 3 Y position

Definition at line **1784** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **ABS**, and **BSP_DSI_LCD_DrawLine()**.

Referenced by **BSP_DSI_LCD_FillPolygon()**.

void HAL_DSI_EndOfRefreshCallback (DSI_HandleTypeDef * hdsi

End of Refresh DSI callback.

Parameters:

hdsi,: pointer to a DSI_HandleTypeDef structure that contains the configuration information for the DSI.

Return values:

None

Definition at line **1685** of file **stm32l4r9i_eval_dsi_lcd.c**.

static void LCD_Reset (void) [static]

LCD Reset Hw reset the LCD DSI activating its XRES signal (active low for some time) and desactivating it later.

Definition at line **1699** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **BSP_IO_ConfigPin()**, **BSP_IO_Init()**, **BSP_IO_WritePin()**, and **IO_PIN_9**.

```
static void LL_ConvertLineToARGB8888 ( void *    pSrc,
                                       void *    pDst,
                                       uint32_t  xSize,
                                       uint32_t  ColorMode
                                     )           [static]
```

Converts a line to an ARGB8888 pixel format.

Parameters:

pSrc,:	Pointer to source buffer
pDst,:	Output color
xSize,:	Buffer width
ColorMode,:	Input color mode

Definition at line **1893** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **hdma2d_eval**.

Referenced by **BSP_DSI_LCD_DrawBitmap()**.

```
static void LL_FillBuffer ( uint32_t  LayerIndex,
                            void *    pDst,
                            uint32_t  xSize,
                            uint32_t  ySize,
                            uint32_t  OffLine,
                            uint32_t  ColorIndex
                          )           [static]
```

Fills a buffer.

Parameters:

LayerIndex,:	Layer index
pDst,:	Pointer to destination buffer

xSize,:	Buffer width
ySize,:	Buffer height
OffLine,:	Offset
ColorIndex,:	Color index

Definition at line **1861** of file **stm32l4r9i_eval_dsi_lcd.c**.

References **hdma2d_eval**.

Referenced by **BSP_DSI_LCD_Clear()**, **BSP_DSI_LCD_DrawHLine()**, **BSP_DSI_LCD_DrawVLine()**, and **BSP_DSI_LCD_FillRect()**.

STM32L4R9I_EVAL BSP User Manual

Private Constants

STM32L4R9I_EVAL OSPI NOR

Defines

#define	**OSPI_NOR_OCTAL_DISABLE** 0x0
#define	**OSPI_NOR_OCTAL_ENABLE** 0x1
#define	**DUMMY_CYCLES_READ_OCTAL** MX25LM51245G_DUMM
#define	**DUMMY_CYCLES_CR_CFG** MX25LM51245G_CR2_DC_66

Define Documentation

#define DUMMY_CYCLES_CR_CFG MX25LM51245G_CR2_DC_66

Definition at line 88 of file stm32l4r9i_eval_ospi_nor.c.

Referenced by OSPI_NOR_OctalMode().

#define DUMMY_CYCLES_READ_OCTAL MX25LM51245G_DUMM

Definition at line 87 of file stm32l4r9i_eval_ospi_nor.c.

Referenced by BSP_OSPI_NOR_EnableMemoryMappedMode(), BSP_OSPI_NOR_GetStatus(), BSP_OSPI_NOR_Read(), OSPI_NOR_AutoPollingMemReady(), OSPI_NOR_OctalMode(), and OSPI_NOR_WriteEnable().

#define OSPI_NOR_OCTAL_DISABLE 0x0

Definition at line 84 of file stm32l4r9i_eval_ospi_nor.c.

Referenced by BSP_OSPI_NOR_DeInit().

#define OSPI_NOR_OCTAL_ENABLE 0x1

Definition at line 85 of file stm32l4r9i_eval_ospi_nor.c.

Referenced by BSP_OSPI_NOR_Init(), and OSPI_NOR_OctalMode().

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL EEPROM

Defines

#define	**EEPROM_PAGESIZE** ((uint8_t)4)
#define	**EEPROM_MAX_SIZE** ((uint16_t)0x2000) /* 64Kbit */
#define	**EEPROM_MAX_TRIALS** ((uint32_t)3000)
#define	**EEPROM_OK** ((uint32_t)0)
#define	**EEPROM_FAIL** ((uint32_t)1)
#define	**EEPROM_TIMEOUT** ((uint32_t)2)

Define Documentation

#define EEPROM_FAIL ((uint32_t)1)

Definition at line 73 of file stm32l4r9i_eval_eeprom.h.

Referenced by BSP_EEPROM_Init(), BSP_EEPROM_ReadBuffer(), and BSP_EEPROM_WritePage().

#define EEPROM_MAX_SIZE ((uint16_t)0x2000) /* 64Kbit */

Definition at line 66 of file stm32l4r9i_eval_eeprom.h.

#define EEPROM_MAX_TRIALS ((uint32_t)3000)

Definition at line 70 of file stm32l4r9i_eval_eeprom.h.

Referenced by BSP_EEPROM_Init(), and BSP_EEPROM_WaitEepromStandbyState().

#define EEPROM_OK ((uint32_t)0)

Definition at line 72 of file stm32l4r9i_eval_eeprom.h.

Referenced by BSP_EEPROM_DeInit(), BSP_EEPROM_Init(), BSP_EEPROM_ReadBuffer(), BSP_EEPROM_WaitEepromStandbyState(), BSP_EEPROM_WriteBuffer(), and BSP_EEPROM_WritePage().

#define EEPROM_PAGESIZE ((uint8_t)4)

Definition at line 65 of file stm32l4r9i_eval_eeprom.h.

Referenced by **BSP_EEPROM_WriteBuffer()**.

#define EEPROM_TIMEOUT ((uint32_t)2)

Definition at line **74** of file **stm32l4r9i_eval_eeprom.h**.

Referenced by **BSP_EEPROM_WaitEepromStandbyState()**.

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL EEPROM

Variables

__IO uint16_t	**EEPROMAddress** = 0
__IO uint16_t	**EEPROMDataRead**
__IO uint8_t	**EEPROMDataWrite**

Variable Documentation

__IO uint16_t EEPROMAddress = 0

Definition at line 103 of file stm32l4r9i_eval_eeprom.c.

Referenced by BSP_EEPROM_Init(), BSP_EEPROM_ReadBuffer(), BSP_EEPROM_WaitEepromStandbyState(), and BSP_EEPROM_WritePage().

__IO uint16_t EEPROMDataRead

Definition at line 104 of file stm32l4r9i_eval_eeprom.c.

Referenced by BSP_EEPROM_ReadBuffer().

__IO uint8_t EEPROMDataWrite

Definition at line 105 of file stm32l4r9i_eval_eeprom.c.

Referenced by BSP_EEPROM_WritePage().

STM32L4R9I_EVAL BSP User Manual

Exported Defines

STM32L4R9I_EVAL IDD

Defines

#define	**EVAL_IDD_SHUNT0_VALUE** ((uint16_t) 1000) Shunt values on evaluation board in milli ohms.
#define	**EVAL_IDD_SHUNT1_VALUE** ((uint16_t) 24)
#define	**EVAL_IDD_SHUNT2_VALUE** ((uint16_t) 620)
#define	**EVAL_IDD_SHUNT4_VALUE** ((uint16_t) 10000)
#define	**EVAL_IDD_SHUNT0_STABDELAY** ((uint8_t) 149) Shunt stabilization delay on evaluation board in milli ohms.
#define	**EVAL_IDD_SHUNT1_STABDELAY** ((uint8_t) 149)
#define	**EVAL_IDD_SHUNT2_STABDELAY** ((uint8_t) 149)
#define	**EVAL_IDD_SHUNT4_STABDELAY** ((uint8_t) 255)
#define	**EVAL_IDD_AMPLI_GAIN** ((uint16_t) 4967) IDD Ampli Gain on evaluation board.
#define	**EVAL_IDD_VDD_MIN** ((uint16_t) 1710) IDD Vdd Min on evaluation board.

Enumerations

enum	**IDD_StatusTypeDef** { **IDD_OK** = 0, **IDD_TIMEOUT** = 1, **IDD_ZERO_VALUE** = 2, **IDD_ERROR** = 0xFF } IDD Status values. More...

Define Documentation

#define EVAL_IDD_AMPLI_GAIN ((uint16_t) 4967)

IDD Ampli Gain on evaluation board.

value is gain * 100

Definition at line **95** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_SHUNT0_STABDELAY ((uint8_t) 149)

Shunt stabilization delay on evaluation board in milli ohms.

value in millisec

Definition at line **87** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_SHUNT0_VALUE ((uint16_t) 1000)

Shunt values on evaluation board in milli ohms.

value in milliohm

Definition at line **79** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_SHUNT1_STABDELAY ((uint8_t) 149)

value in millisec

Definition at line **88** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_SHUNT1_VALUE ((uint16_t) 24)

value in ohm

Definition at line **80** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_SHUNT2_STABDELAY ((uint8_t) 149)

value in millisec

Definition at line **89** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_SHUNT2_VALUE ((uint16_t) 620)

value in ohm

Definition at line **81** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_SHUNT4_STABDELAY ((uint8_t) 255)

value in millisec

Definition at line **90** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_SHUNT4_VALUE ((uint16_t) 10000)

value in ohm

Definition at line **82** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

#define EVAL_IDD_VDD_MIN ((uint16_t) 1710)

IDD Vdd Min on evaluation board.

value in millivolt

Definition at line **100** of file **stm32l4r9i_eval_idd.h**.

Referenced by **BSP_IDD_Init()**.

Enumeration Type Documentation

enum IDD_StatusTypeDef

IDD Status values.

Enumerator:

IDD_OK

IDD_TIMEOUT

IDD_ZERO_VALUE

IDD_ERROR

Definition at line **67** of file **stm32l4r9i_eval_idd.h**.

STM32L4R9I_EVAL BSP User Manual

Private Functions

STM32L4R9I_EVAL NOR

Functions

static void	**NOR_MspInit** (void) Initializes the NOR MSP.
void	**HAL_NOR_MspWait** (NOR_HandleTypeDef *hnor, uint32_t Timeout) NOR BSP Wait for Ready/Busy signal.

Function Documentation

void HAL_NOR_MspWait (NOR_HandleTypeDef * hnor,
uint32_t Timeout
)

NOR BSP Wait for Ready/Busy signal.

Parameters:

hnor,: Pointer to NOR handle

Timeout,: Timeout duration

Return values:

None

Definition at line **427** of file **stm32l4r9i_eval_nor.c**.

References **NOR_BUSY_STATE**, **NOR_READY_BUSY_GPIO**, **NOR_READY_BUSY_PIN**, and **NOR_READY_STATE**.

static void NOR_MspInit (void) [static]

Initializes the NOR MSP.

Return values:

None

Definition at line **348** of file **stm32l4r9i_eval_nor.c**.

References **NOR_READY_BUSY_GPIO**, and **NOR_READY_BUSY_PIN**.

Referenced by **BSP_NOR_Init()**.

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Exported Variables

STM32L4R9I_EVAL DSI LCD

Variables

DMA2D_HandleTypeDef	**hdma2d_eval**
DMA2D_HandleTypeDef	**hdma2d_eval**

Variable Documentation

DMA2D_HandleTypeDef hdma2d_eval

Definition at line **126** of file **stm32l4r9i_eval_dsi_lcd.c**.

Referenced by **BSP_DSI_LCD_DMA2D_IRQHandler()**, **LL_ConvertLineToARGB8888()**, and **LL_FillBuffer()**.

DMA2D_HandleTypeDef hdma2d_eval

Definition at line **126** of file **stm32l4r9i_eval_dsi_lcd.c**.

Referenced by **BSP_DSI_LCD_DMA2D_IRQHandler()**, **LL_ConvertLineToARGB8888()**, and **LL_FillBuffer()**.

STM32L4R9I_EVAL BSP User Manual

Exported Variables

STM32L4R9I_EVAL RGB LCD

Variables

LTDC_HandleTypeDef	**hltdc_eval**
uint32_t	**PhysFrameBuffer** []

Variable Documentation

LTDC_HandleTypeDef hltdc_eval

uint32_t PhysFrameBuffer[]

STM32L4R9I_EVAL BSP User Manual

Exported Variables

STM32L4R9I_EVAL SD

Variables

SD_HandleTypeDef	**hsd_eval**

Variable Documentation

SD_HandleTypeDef hsd_eval

Definition at line **97** of file **stm32l4r9i_eval_sd.c**.

Referenced by **BSP_SD_DeInit()**, **BSP_SD_Erase()**, **BSP_SD_GetCardInfo()**, **BSP_SD_GetCardState()**, **BSP_SD_Init()**, **BSP_SD_ReadBlocks()**, **BSP_SD_ReadBlocks_DMA()**, **BSP_SD_WriteBlocks()**, and **BSP_SD_WriteBlocks_DMA()**.

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL SD

Variables

SD_HandleTypeDef	**hsd_eval**
static uint8_t	**UseExtiModeDetection** = 0

Variable Documentation

SD_HandleTypeDef hsd_eval

Definition at line 97 of file stm32l4r9i_eval_sd.c.

Referenced by BSP_SD_DeInit(), BSP_SD_Erase(), BSP_SD_GetCardInfo(), BSP_SD_GetCardState(), BSP_SD_Init(), BSP_SD_ReadBlocks(), BSP_SD_ReadBlocks_DMA(), BSP_SD_WriteBlocks(), and BSP_SD_WriteBlocks_DMA().

uint8_t UseExtiModeDetection = 0 [static]

Definition at line 98 of file stm32l4r9i_eval_sd.c.

Referenced by BSP_SD_DeInit(), BSP_SD_IsDetected(), and BSP_SD_ITConfig().

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL IDD

Variables

static IDD_DrvTypeDef * **IddDrv** = NULL

Variable Documentation

IDD_DrvTypeDef* IddDrv = NULL [static]

Definition at line 59 of file stm32l4r9i_eval_idd.c.

Referenced by BSP_IDD_ClearIT(), BSP_IDD_Config(), BSP_IDD_DeInit(), BSP_IDD_DisableIT(), BSP_IDD_EnableIT(), BSP_IDD_ErrorClearIT(), BSP_IDD_ErrorDisableIT(), BSP_IDD_ErrorEnableIT(), BSP_IDD_ErrorGetCode(), BSP_IDD_ErrorGetITStatus(), BSP_IDD_GetITStatus(), BSP_IDD_GetValue(), BSP_IDD_Init(), BSP_IDD_LowPower(), BSP_IDD_Reset(), BSP_IDD_StartMeasure(), and BSP_IDD_WakeUp().

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL IO

Variables

static IO_DrvTypeDef *	**io_driver**

Variable Documentation

IO_DrvTypeDef* io_driver [static]

Definition at line **92** of file **stm32l4r9i_eval_io.c**.

Referenced by **BSP_IO_ConfigPin()**, **BSP_IO_DeInit()**, **BSP_IO_Init()**, **BSP_IO_ITClear()**, **BSP_IO_ITGetStatus()**, **BSP_IO_ReadPin()**, **BSP_IO_TogglePin()**, and **BSP_IO_WritePin()**.

STM32L4R9I_EVAL BSP User Manual

Exported Types

STM32L4R9I_EVAL IO

Enumerations

enum	**IO_StatusTypeDef** { **IO_OK** = 0x00, **IO_ERROR** = 0x01, **IO_TIMEOUT** = 0x02 }

Enumeration Type Documentation

enum IO_StatusTypeDef

Enumerator:

IO_OK

IO_ERROR

IO_TIMEOUT

Definition at line **66** of file **stm32l4r9i_eval_io.h**.

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL LCD

Defines

#define	**LCD_LayerCfgTypeDef** LTDC_LayerCfgTypeDef
#define	**LTDC_MAX_LAYER_NUMBER** ((uint32_t) 2)
#define	**LTDC_ACTIVE_LAYER_BACKGROUND** LTDC_LAYER_1
#define	**LTDC_ACTIVE_LAYER_FOREGROUND** LTDC_LAYER_2
#define	**LTDC_NB_OF_LAYERS** ((uint32_t) 2)
#define	**LTDC_DEFAULT_ACTIVE_LAYER** **LTDC_ACTIVE_LAYER_**
#define	**LCD_OK** 0x00
#define	**LCD_ERROR** 0x01
#define	**LCD_TIMEOUT** 0x02
#define	**LCD_COLOR_BLUE** ((uint32_t) 0xFF0000FF)
#define	**LCD_COLOR_GREEN** ((uint32_t) 0xFF00FF00)
#define	**LCD_COLOR_RED** ((uint32_t) 0xFFFF0000)
#define	**LCD_COLOR_CYAN** ((uint32_t) 0xFF00FFFF)
#define	**LCD_COLOR_MAGENTA** ((uint32_t) 0xFFFF00FF)
#define	**LCD_COLOR_YELLOW** ((uint32_t) 0xFFFFFF00)
#define	**LCD_COLOR_LIGHTBLUE** ((uint32_t) 0xFF8080FF)
#define	**LCD_COLOR_LIGHTGREEN** ((uint32_t) 0xFF80FF80)
#define	**LCD_COLOR_LIGHTRED** ((uint32_t) 0xFFFF8080)
#define	**LCD_COLOR_LIGHTCYAN** ((uint32_t) 0xFF80FFFF)
#define	**LCD_COLOR_LIGHTMAGENTA** ((uint32_t) 0xFFFF80FF)
#define	**LCD_COLOR_LIGHTYELLOW** ((uint32_t) 0xFFFFFF80)
#define	**LCD_COLOR_DARKBLUE** ((uint32_t) 0xFF000080)
#define	**LCD_COLOR_DARKGREEN** ((uint32_t) 0xFF008000)
#define	**LCD_COLOR_DARKRED** ((uint32_t) 0xFF800000)
#define	**LCD_COLOR_DARKCYAN** ((uint32_t) 0xFF008080)
#define	**LCD_COLOR_DARKMAGENTA** ((uint32_t) 0xFF800080)
#define	**LCD_COLOR_DARKYELLOW** ((uint32_t) 0xFF808000)
#define	**LCD_COLOR_WHITE** ((uint32_t) 0xFFFFFFFF)
#define	**LCD_COLOR_LIGHTGRAY** ((uint32_t) 0xFFD3D3D3)
#define	**LCD_COLOR_GRAY** ((uint32_t) 0xFF808080)
#define	**LCD_COLOR_DARKGRAY** ((uint32_t) 0xFF404040)

```
#define LCD_COLOR_BLACK ((uint32_t) 0xFF000000)
#define LCD_COLOR_BROWN ((uint32_t) 0xFFA52A2A)
#define LCD_COLOR_ORANGE ((uint32_t) 0xFFFFA500)
#define LCD_COLOR_TRANSPARENT ((uint32_t) 0xFF000000)
#define LCD_DEFAULT_FONT Font24
```

Define Documentation

#define LCD_COLOR_BLACK ((uint32_t) 0xFF000000)

Definition at line 131 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_BLUE ((uint32_t) 0xFF0000FF)

Definition at line 87 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_BROWN ((uint32_t) 0xFFA52A2A)

Definition at line 133 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_CYAN ((uint32_t) 0xFF00FFFF)

Definition at line 93 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_DARKBLUE ((uint32_t) 0xFF000080)

Definition at line 111 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_DARKCYAN ((uint32_t) 0xFF008080)

Definition at line 117 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_DARKGRAY ((uint32_t) 0xFF404040)

Definition at line 129 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_DARKGREEN ((uint32_t) 0xFF008000)

Definition at line 113 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_DARKMAGENTA ((uint32_t) 0xFF800080)

Definition at line 119 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_DARKRED ((uint32_t) 0xFF800000)

Definition at line 115 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_DARKYELLOW ((uint32_t) 0xFF808000)

Definition at line 121 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_GRAY ((uint32_t) 0xFF808080)

Definition at line 127 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_GREEN ((uint32_t) 0xFF00FF00)

Definition at line 89 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_LIGHTBLUE ((uint32_t) 0xFF8080FF)

Definition at line 99 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_LIGHTCYAN ((uint32_t) 0xFF80FFFF)

Definition at line 105 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_LIGHTGRAY ((uint32_t) 0xFFD3D3D3)

Definition at line 125 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_LIGHTGREEN ((uint32_t) 0xFF80FF80)

Definition at line 101 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_LIGHTMAGENTA ((uint32_t) 0xFFFF80FF)

Definition at line 107 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_LIGHTRED ((uint32_t) 0xFFFF8080)

Definition at line 103 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_LIGHTYELLOW ((uint32_t) 0xFFFFFF80)

Definition at line 109 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_MAGENTA ((uint32_t) 0xFFFF00FF)

Definition at line 95 of file stm32l4r9i_eval_lcd.h.

#define LCD_COLOR_ORANGE ((uint32_t) 0xFFFFA500)

Definition at line **135** of file **stm32l4r9i_eval_lcd.h**.

#define LCD_COLOR_RED ((uint32_t) 0xFFFF0000)

Definition at line **91** of file **stm32l4r9i_eval_lcd.h**.

#define LCD_COLOR_TRANSPARENT ((uint32_t) 0xFF000000)

Definition at line **137** of file **stm32l4r9i_eval_lcd.h**.

#define LCD_COLOR_WHITE ((uint32_t) 0xFFFFFFFF)

Definition at line **123** of file **stm32l4r9i_eval_lcd.h**.

#define LCD_COLOR_YELLOW ((uint32_t) 0xFFFFFF00)

Definition at line **97** of file **stm32l4r9i_eval_lcd.h**.

#define LCD_DEFAULT_FONT Font24

Definition at line **140** of file **stm32l4r9i_eval_lcd.h**.

#define LCD_ERROR 0x01

Definition at line **82** of file **stm32l4r9i_eval_lcd.h**.

Referenced by **BSP_DSI_LCD_DeInit()**, **BSP_DSI_LCD_IsFrameBufferAvailable()**, **BSP_DSI_LCD_ResetColorKeying()**, **BSP_DSI_LCD_SelectLayer()**, **BSP_DSI_LCD_SetColorKeying()**, **BSP_DSI_LCD_SetLayerVisible()**, and

BSP_DSI_LCD_SetTransparency().

#define LCD_LayerCfgTypeDef LTDC_LayerCfgTypeDef

Definition at line 63 of file stm32l4r9i_eval_lcd.h.

#define LCD_OK 0x00

Definition at line 81 of file stm32l4r9i_eval_lcd.h.

Referenced by BSP_DSI_LCD_DeInit(), BSP_DSI_LCD_IsFrameBufferAvailable(), BSP_DSI_LCD_ResetColorKeying(), BSP_DSI_LCD_SelectLayer(), BSP_DSI_LCD_SetColorKeying(), BSP_DSI_LCD_SetLayerVisible(), and BSP_DSI_LCD_SetTransparency().

#define LCD_TIMEOUT 0x02

Definition at line 83 of file stm32l4r9i_eval_lcd.h.

#define LTDC_ACTIVE_LAYER_BACKGROUND LTDC_LAYER_1

Definition at line 69 of file stm32l4r9i_eval_lcd.h.

Referenced by BSP_DSI_LCD_ResetColorKeying(), BSP_DSI_LCD_SelectLayer(), BSP_DSI_LCD_SetColorKeying(), BSP_DSI_LCD_SetLayerVisible(), and BSP_DSI_LCD_SetTransparency().

#define LTDC_ACTIVE_LAYER_FOREGROUND LTDC_LAYER_2

Definition at line **72** of file **stm32l4r9i_eval_lcd.h**.

#define LTDC_DEFAULT_ACTIVE_LAYER LTDC_ACTIVE_LAYER_

Definition at line **78** of file **stm32l4r9i_eval_lcd.h**.

#define LTDC_MAX_LAYER_NUMBER ((uint32_t) 2)

Definition at line **66** of file **stm32l4r9i_eval_lcd.h**.

#define LTDC_NB_OF_LAYERS ((uint32_t) 2)

Definition at line **75** of file **stm32l4r9i_eval_lcd.h**.

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL RGB LCD

Defines

#define	**LCD_FB_START_ADDRESS** (uint32_t)**PhysFrameBuffer** /* SRAM used for LCD Frame buffer */ LCD FB_StartAddress.
#define	**LCD_RELOAD_IMMEDIATE** ((uint32_t)LTDC_SRCR_IMR) LCD Reload Types.
#define	**LCD_RELOAD_VERTICAL_BLANKING** ((uint32_t)LTDC_SR

Define Documentation

#define LCD_FB_START_ADDRESS (uint32_t)PhysFrameBuffer /*

LCD FB_StartAddress.

Definition at line 74 of file stm32l4r9i_eval_rgb_lcd.h.

#define LCD_RELOAD_IMMEDIATE ((uint32_t)LTDC_SRCR_IMR)

LCD Reload Types.

Definition at line 79 of file stm32l4r9i_eval_rgb_lcd.h.

#define LCD_RELOAD_VERTICAL_BLANKING ((uint32_t)LTDC_SI

Definition at line 80 of file stm32l4r9i_eval_rgb_lcd.h.

STM32L4R9I_EVAL BSP User Manual

LED Constants

Exported Constants

Defines

#define	**LEDn** 4
#define	**LED1_PIN** GPIO_PIN_15
#define	**LED1_GPIO_PORT** GPIOB
#define	**LED1_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOB_CLK_E
#define	**LED1_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOB_CLK_D
#define	**LED2_PIN** GPIO_PIN_13
#define	**LED2_GPIO_PORT** GPIOH
#define	**LED2_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOH_CLK_E
#define	**LED2_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOH_CLK_D
#define	**LED3_PIN** GPIO_PIN_14
#define	**LED3_GPIO_PORT** GPIOH
#define	**LED3_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOH_CLK_E
#define	**LED3_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOH_CLK_D
#define	**LED4_PIN** GPIO_PIN_15
#define	**LED4_GPIO_PORT** GPIOH
#define	**LED4_GPIO_CLK_ENABLE**() __HAL_RCC_GPIOH_CLK_E
#define	**LED4_GPIO_CLK_DISABLE**() __HAL_RCC_GPIOH_CLK_D
#define	**LEDx_GPIO_CLK_ENABLE**(__LED__)
#define	**LEDx_GPIO_CLK_DISABLE**(__LED__)

Define Documentation

#define LED1_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOB_CLK

Definition at line 159 of file stm32l4r9i_eval.h.

#define LED1_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOB_CLK_

Definition at line 158 of file stm32l4r9i_eval.h.

#define LED1_GPIO_PORT GPIOB

Definition at line 157 of file stm32l4r9i_eval.h.

#define LED1_PIN GPIO_PIN_15

Definition at line 156 of file stm32l4r9i_eval.h.

#define LED2_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOH_CLK

Definition at line 164 of file stm32l4r9i_eval.h.

#define LED2_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOH_CLK_

Definition at line 163 of file stm32l4r9i_eval.h.

#define LED2_GPIO_PORT GPIOH

Definition at line 162 of file stm32l4r9i_eval.h.

#define LED2_PIN GPIO_PIN_13

Definition at line 161 of file stm32l4r9i_eval.h.

#define LED3_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOH_CLK

Definition at line 169 of file stm32l4r9i_eval.h.

#define LED3_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOH_CLK_

Definition at line 168 of file stm32l4r9i_eval.h.

#define LED3_GPIO_PORT GPIOH

Definition at line 167 of file stm32l4r9i_eval.h.

#define LED3_PIN GPIO_PIN_14

Definition at line 166 of file stm32l4r9i_eval.h.

#define LED4_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOH_CLK

Definition at line 174 of file stm32l4r9i_eval.h.

#define LED4_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOH_CLK_

Definition at line 173 of file stm32l4r9i_eval.h.

#define LED4_GPIO_PORT GPIOH

Definition at line 172 of file stm32l4r9i_eval.h.

#define LED4_PIN GPIO_PIN_15

Definition at line 171 of file stm32l4r9i_eval.h.

#define LEDn 4

Definition at line 154 of file stm32l4r9i_eval.h.

#define LEDx_GPIO_CLK_DISABLE (__LED__)

Value:

```
do { if((__LED__) == LED1) { __HAL_RCC_GPIOB_CLK_
DISABLE(); } else \
                                                __H
AL_RCC_GPIOH_CLK_DISABLE(); } while(0)
```

Definition at line 179 of file stm32l4r9i_eval.h.

#define LEDx_GPIO_CLK_ENABLE (__LED__)

Value:

```
do { if((__LED__) == LED1) { __HAL_RCC_GPIOB_CLK_
ENABLE(); } else \
                                                __H
AL_RCC_GPIOH_CLK_ENABLE(); } while(0)
```

Definition at line 176 of file stm32l4r9i_eval.h.

Referenced by BSP_LED_DeInit(), and BSP_LED_Init().

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL SD

Defines

#define	**MSD_OK** ((uint8_t)0x00) SD status structure definition.
#define	**MSD_ERROR** ((uint8_t)0x01)
#define	**MSD_ERROR_SD_NOT_PRESENT** ((uint8_t)0x02)
#define	**SD_TRANSFER_OK** ((uint8_t)0x00) SD transfer state definition.
#define	**SD_TRANSFER_BUSY** ((uint8_t)0x01)
#define	**SD_TRANSFER_ERROR** ((uint8_t)0x02)
#define	**SD_DATATIMEOUT** ((uint32_t)100000000)
#define	**SD_PRESENT** ((uint8_t)0x01)
#define	**SD_NOT_PRESENT** ((uint8_t)0x00)
#define	**SDMMCx_IRQHandler** SDMMC1_IRQHandler
#define	**SDMMCx_IRQn** SDMMC1_IRQn

Define Documentation

#define MSD_ERROR ((uint8_t)0x01)

Definition at line 83 of file stm32l4r9i_eval_sd.h.

Referenced by BSP_SD_DeInit(), BSP_SD_Erase(), BSP_SD_Init(), BSP_SD_ReadBlocks(), BSP_SD_ReadBlocks_DMA(), BSP_SD_WriteBlocks(), and BSP_SD_WriteBlocks_DMA().

#define MSD_ERROR_SD_NOT_PRESENT ((uint8_t)0x02)

Definition at line 84 of file stm32l4r9i_eval_sd.h.

Referenced by BSP_SD_Init().

#define MSD_OK ((uint8_t)0x00)

SD status structure definition.

Definition at line 82 of file stm32l4r9i_eval_sd.h.

Referenced by BSP_SD_DeInit(), BSP_SD_Erase(), BSP_SD_Init(), BSP_SD_ReadBlocks(), BSP_SD_ReadBlocks_DMA(), BSP_SD_WriteBlocks(), and BSP_SD_WriteBlocks_DMA().

#define SD_DATATIMEOUT ((uint32_t)100000000)

Definition at line 93 of file stm32l4r9i_eval_sd.h.

#define SD_NOT_PRESENT ((uint8_t)0x00)

Definition at line **96** of file **stm32l4r9i_eval_sd.h**.

Referenced by **BSP_SD_IsDetected()**.

#define SD_PRESENT ((uint8_t)0x01)

Definition at line **95** of file **stm32l4r9i_eval_sd.h**.

Referenced by **BSP_SD_Init()**, and **BSP_SD_IsDetected()**.

#define SD_TRANSFER_BUSY ((uint8_t)0x01)

Definition at line **90** of file **stm32l4r9i_eval_sd.h**.

Referenced by **BSP_SD_GetCardState()**.

#define SD_TRANSFER_ERROR ((uint8_t)0x02)

Definition at line **91** of file **stm32l4r9i_eval_sd.h**.

#define SD_TRANSFER_OK ((uint8_t)0x00)

SD transfer state definition.

Definition at line **89** of file **stm32l4r9i_eval_sd.h**.

Referenced by **BSP_SD_GetCardState()**.

#define SDMMCx_IRQHandler SDMMC1_IRQHandler

Definition at line **99** of file **stm32l4r9i_eval_sd.h**.

#define SDMMCx_IRQn SDMMC1_IRQn

Definition at line 100 of file stm32l4r9i_eval_sd.h.

Referenced by BSP_SD_MspDeInit(), and BSP_SD_MspInit().

STM32L4R9I_EVAL BSP User Manual

Private Constants

STM32L4R9I_EVAL NOR

Defines

#define	**NOR_ADDR_SETUP_TIME** 9 /* 70ns with a clock at 120 MHz (period of 8.33 ns) */
#define	**NOR_DATA_SETUP_TIME** 5 /* 45ns with a clock at 120 MHz (period of 8.33 ns) */
#define	**NOR_DATA_HOLD_TIME** 0
#define	**NOR_TURN_AROUND_TIME** 4 /* 30ns with a clock at 120 MHz (perido of 8.33 ns) */
#define	**NOR_MAX_BUFFER_SIZE** 32 /* 16-bits bus so 32 words */

Define Documentation

#define NOR_ADDR_SETUP_TIME 9 /* 70ns with a clock at 120 MH

Definition at line 89 of file stm32l4r9i_eval_nor.c.

Referenced by BSP_NOR_Init().

#define NOR_DATA_HOLD_TIME 0

Definition at line 91 of file stm32l4r9i_eval_nor.c.

Referenced by BSP_NOR_Init().

#define NOR_DATA_SETUP_TIME 5 /* 45ns with a clock at 120 MH

Definition at line 90 of file stm32l4r9i_eval_nor.c.

Referenced by BSP_NOR_Init().

#define NOR_MAX_BUFFER_SIZE 32 /* 16-bits bus so 32 words */

Definition at line 94 of file stm32l4r9i_eval_nor.c.

Referenced by BSP_NOR_ProgramData().

#define NOR_TURN_AROUND_TIME 4 /* 30ns with a clock at 120 I

Definition at line 92 of file stm32l4r9i_eval_nor.c.

Referenced by BSP_NOR_Init().

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL NOR

Variables

static NOR_HandleTypeDef	**norHandle**
static FMC_NORSRAM_TimingTypeDef	**Timing**

Variable Documentation

NOR_HandleTypeDef norHandle [static]

Definition at line 105 of file stm32l4r9i_eval_nor.c.

Referenced by BSP_NOR_Erase_Block(), BSP_NOR_Erase_Chip(), BSP_NOR_Init(), BSP_NOR_ProgramData(), BSP_NOR_Read_ID(), BSP_NOR_ReadData(), BSP_NOR_ReturnToReadMode(), and BSP_NOR_WriteData().

FMC_NORSRAM_TimingTypeDef Timing [static]

Definition at line 106 of file stm32l4r9i_eval_nor.c.

Referenced by BSP_NOR_Init().

STM32L4R9I_EVAL BSP User Manual

Private Functions

STM32L4R9I_EVAL OSPI NOR

Functions

static void	**OSPI_NOR_MspInit** (void) Initializes the OSPI MSP.
static void	**OSPI_NOR_MspDeInit** (void) De-Initializes the OSPI MSP.
static uint8_t	**OSPI_NOR_ResetMemory** (OSPI_HandleTypeDef *hospi) This function reset the OSPI memory.
static uint8_t	**OSPI_NOR_WriteEnable** (OSPI_HandleTypeDef *hospi) This function send a Write Enable and wait it is effective.
static uint8_t	**OSPI_NOR_AutoPollingMemReady** (OSPI_HandleTypeDef *hospi, uint32_t Timeout) This function read the SR of the memory and wait the EOP.
static uint8_t	**OSPI_NOR_OctalMode** (OSPI_HandleTypeDef *hospi, uint8_t Operation) This function enables/disables the octal mode of the memory.

Function Documentation

static uint8_t OSPI_NOR_AutoPollingMemReady (OSPI_HandleTyp
uint32_t
)

This function read the SR of the memory and wait the EOP.

Parameters:

hospi,: OSPI handle

Timeout,: Timeout for auto-polling

Return values:

None

Definition at line **1121** of file **stm32l4r9i_eval_ospi_nor.c**.

References **DUMMY_CYCLES_READ_OCTAL**, **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, and **OSPINORHandle**.

Referenced by **BSP_OSPI_NOR_Erase_Block()**, **BSP_OSPI_NOR_Erase_Chip()**, **BSP_OSPI_NOR_Write()**, and **OSPI_NOR_OctalMode()**.

static void OSPI_NOR_MspDeInit (void) [static]

De-Initializes the OSPI MSP.

Return values:

None

Definition at line **941** of file **stm32l4r9i_eval_ospi_nor.c**.

Referenced by **BSP_OSPI_NOR_DeInit()**.

static void OSPI_NOR_MspInit (void) [static]

Initializes the OSPI MSP.

Return values:

None

Definition at line **879** of file **stm32l4r9i_eval_ospi_nor.c**.

Referenced by **BSP_OSPI_NOR_Init()**.

static uint8_t OSPI_NOR_OctalMode (OSPI_HandleTypeDef * hosp
uint8_t Oper
) [stat

This function enables/disables the octal mode of the memory.

Parameters:

hospi,: OSPI handle

Operation,: Octal operation mode enable/disable

Return values:

None

Definition at line **1184** of file **stm32l4r9i_eval_ospi_nor.c**.

References **DUMMY_CYCLES_CR_CFG**, **DUMMY_CYCLES_READ_OCTAL**, **OSPI_NOR_AutoPollingMemReady()**, **OSPI_NOR_ERROR**, **OSPI_NOR_OCTAL_ENABLE**, **OSPI_NOR_OK**, **OSPI_NOR_WriteEnable()**, and **OSPINORHandle**.

Referenced by **BSP_OSPI_NOR_DeInit()**, and **BSP_OSPI_NOR_Init()**.

static uint8_t OSPI_NOR_ResetMemory (OSPI_HandleTypeDef * h

This function reset the OSPI memory.

Parameters:

hospi,: OSPI handle

Return values:

None

Definition at line **979** of file **stm32l4r9i_eval_ospi_nor.c**.

References **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, and **OSPINORHandle**.

Referenced by **BSP_OSPI_NOR_Init()**.

static uint8_t OSPI_NOR_WriteEnable (OSPI_HandleTypeDef * hos

This function send a Write Enable and wait it is effective.

Parameters:

hospi,: OSPI handle

Return values:

None

Definition at line **1041** of file **stm32l4r9i_eval_ospi_nor.c**.

References **DUMMY_CYCLES_READ_OCTAL**, **OSPI_NOR_ERROR**, **OSPI_NOR_OK**, and **OSPINORHandle**.

Referenced by **BSP_OSPI_NOR_EnableMemoryMappedMode()**, **BSP_OSPI_NOR_Erase_Block()**, **BSP_OSPI_NOR_Erase_Chip()**, **BSP_OSPI_NOR_Erase_Sector()**, **BSP_OSPI_NOR_Write()**, and **OSPI_NOR_OctalMode()**.

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL OSPI NOR

Defines

#define	**OSPI_NOR_OK** ((uint8_t)0x00)
#define	**OSPI_NOR_ERROR** ((uint8_t)0x01)
#define	**OSPI_NOR_BUSY** ((uint8_t)0x02)
#define	**OSPI_NOR_NOT_SUPPORTED** ((uint8_t)0x04)
#define	**OSPI_NOR_SUSPENDED** ((uint8_t)0x08)

Define Documentation

#define OSPI_NOR_BUSY ((uint8_t)0x02)

Definition at line 68 of file stm32l4r9i_eval_ospi_nor.h.

Referenced by BSP_OSPI_NOR_GetStatus(), BSP_OSPI_NOR_ResumeErase(), and BSP_OSPI_NOR_SuspendErase().

#define OSPI_NOR_ERROR ((uint8_t)0x01)

Definition at line 67 of file stm32l4r9i_eval_ospi_nor.h.

Referenced by BSP_OSPI_NOR_DeInit(), BSP_OSPI_NOR_EnableMemoryMappedMode(), BSP_OSPI_NOR_EnterDeepPowerDown(), BSP_OSPI_NOR_Erase_Block(), BSP_OSPI_NOR_Erase_Chip(), BSP_OSPI_NOR_Erase_Sector(), BSP_OSPI_NOR_GetStatus(), BSP_OSPI_NOR_Init(), BSP_OSPI_NOR_LeaveDeepPowerDown(), BSP_OSPI_NOR_Read(), BSP_OSPI_NOR_ResumeErase(), BSP_OSPI_NOR_SuspendErase(), BSP_OSPI_NOR_Write(), OSPI_NOR_AutoPollingMemReady(), OSPI_NOR_OctalMode(), OSPI_NOR_ResetMemory(), and OSPI_NOR_WriteEnable().

#define OSPI_NOR_NOT_SUPPORTED ((uint8_t)0x04)

Definition at line 69 of file stm32l4r9i_eval_ospi_nor.h.

Referenced by BSP_OSPI_NOR_Init().

#define OSPI_NOR_OK ((uint8_t)0x00)

Definition at line **66** of file **stm32l4r9i_eval_ospi_nor.h**.

Referenced by **BSP_OSPI_NOR_DeInit()**, **BSP_OSPI_NOR_EnableMemoryMappedMode()**, **BSP_OSPI_NOR_EnterDeepPowerDown()**, **BSP_OSPI_NOR_Erase_Block()**, **BSP_OSPI_NOR_Erase_Chip()**, **BSP_OSPI_NOR_Erase_Sector()**, **BSP_OSPI_NOR_GetInfo()**, **BSP_OSPI_NOR_GetStatus()**, **BSP_OSPI_NOR_Init()**, **BSP_OSPI_NOR_LeaveDeepPowerDown()**, **BSP_OSPI_NOR_Read()**, **BSP_OSPI_NOR_ResumeErase()**, **BSP_OSPI_NOR_SuspendErase()**, **BSP_OSPI_NOR_Write()**, **OSPI_NOR_AutoPollingMemReady()**, **OSPI_NOR_OctalMode()**, **OSPI_NOR_ResetMemory()**, and **OSPI_NOR_WriteEnable()**.

#define OSPI_NOR_SUSPENDED ((uint8_t)0x08)

Definition at line **70** of file **stm32l4r9i_eval_ospi_nor.h**.

Referenced by **BSP_OSPI_NOR_GetStatus()**, **BSP_OSPI_NOR_ResumeErase()**, and **BSP_OSPI_NOR_SuspendErase()**.

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL OSPI RAM

Defines

#define **OSPI_RAM_OK** ((uint8_t)0x00)
#define **OSPI_RAM_ERROR** ((uint8_t)0x01)

Define Documentation

#define OSPI_RAM_ERROR ((uint8_t)0x01)

Definition at line 67 of file stm32l4r9i_eval_ospi_ram.h.

Referenced by BSP_OSPI_RAM_DeInit(), BSP_OSPI_RAM_EnableMemoryMappedMode(), BSP_OSPI_RAM_EnterDeepPowerDown(), BSP_OSPI_RAM_Init(), BSP_OSPI_RAM_LeaveDeepPowerDown(), BSP_OSPI_RAM_Read(), BSP_OSPI_RAM_Read_DMA(), BSP_OSPI_RAM_Write(), and BSP_OSPI_RAM_Write_DMA().

#define OSPI_RAM_OK ((uint8_t)0x00)

Definition at line 66 of file stm32l4r9i_eval_ospi_ram.h.

Referenced by BSP_OSPI_RAM_DeInit(), BSP_OSPI_RAM_EnableMemoryMappedMode(), BSP_OSPI_RAM_EnterDeepPowerDown(), BSP_OSPI_RAM_Init(), BSP_OSPI_RAM_LeaveDeepPowerDown(), BSP_OSPI_RAM_Read(), BSP_OSPI_RAM_Read_DMA(), BSP_OSPI_RAM_Write(), and BSP_OSPI_RAM_Write_DMA().

STM32L4R9I_EVAL BSP User Manual

Private Constants

STM32L4R9I_EVAL OSPI RAM

Defines

#define	**OSPI_RAM_QUAD_DISABLE** 0x0
#define	**OSPI_RAM_QUAD_ENABLE** 0x1
#define	**OSPI_RAM_HIGH_PERF_DISABLE** 0x0
#define	**OSPI_RAM_HIGH_PERF_ENABLE** 0x1

Define Documentation

#define OSPI_RAM_HIGH_PERF_DISABLE 0x0

Definition at line **87** of file **stm32l4r9i_eval_ospi_ram.c**.

#define OSPI_RAM_HIGH_PERF_ENABLE 0x1

Definition at line **88** of file **stm32l4r9i_eval_ospi_ram.c**.

#define OSPI_RAM_QUAD_DISABLE 0x0

Definition at line **84** of file **stm32l4r9i_eval_ospi_ram.c**.

#define OSPI_RAM_QUAD_ENABLE 0x1

Definition at line **85** of file **stm32l4r9i_eval_ospi_ram.c**.

STM32L4R9I_EVAL BSP User Manual

Private Functions

STM32L4R9I_EVAL OSPI RAM

Functions

static void	**OSPI_RAM_MspInit** (void) Initializes the OSPI MSP.
static void	**OSPI_RAM_MspDeInit** (void) De-Initializes the OSPI MSP.

Function Documentation

static void OSPI_RAM_MspDeInit (void) [static]

De-Initializes the OSPI MSP.

Return values:

None

Definition at line **554** of file **stm32l4r9i_eval_ospi_ram.c**.

References **DmaOspiRamHandle**.

Referenced by **BSP_OSPI_RAM_DeInit()**.

static void OSPI_RAM_MspInit (void) [static]

Initializes the OSPI MSP.

Return values:

None

Definition at line **487** of file **stm32l4r9i_eval_ospi_ram.c**.

References **DmaOspiRamHandle**, and **OspiRamHandle**.

Referenced by **BSP_OSPI_RAM_Init()**.

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL OSPI NOR

Variables

OSPI_HandleTypeDef	**OSPINORHandle**

Variable Documentation

OSPI_HandleTypeDef OSPINORHandle

Definition at line **101** of file **stm32l4r9i_eval_ospi_nor.c**.

Referenced by **BSP_OSPI_NOR_DeInit()**, **BSP_OSPI_NOR_EnableMemoryMappedMode()**, **BSP_OSPI_NOR_EnterDeepPowerDown()**, **BSP_OSPI_NOR_Erase_Block()**, **BSP_OSPI_NOR_Erase_Chip()**, **BSP_OSPI_NOR_Erase_Sector()**, **BSP_OSPI_NOR_GetStatus()**, **BSP_OSPI_NOR_Init()**, **BSP_OSPI_NOR_LeaveDeepPowerDown()**, **BSP_OSPI_NOR_Read()**, **BSP_OSPI_NOR_ResumeErase()**, **BSP_OSPI_NOR_SuspendErase()**, **BSP_OSPI_NOR_Write()**, **OSPI_NOR_AutoPollingMemReady()**, **OSPI_NOR_OctalMode()**, **OSPI_NOR_ResetMemory()**, and **OSPI_NOR_WriteEnable()**.

STM32L4R9I_EVAL BSP User Manual

Private Constants

STM32L4R9I_EVAL SRAM

Defines

#define	**SRAM_ADDR_SETUP_TIME** 2 /* 10ns with a clock at 120 MHz (period of 8.33 ns) */
#define	**SRAM_DATA_SETUP_TIME** 1 /* 6ns with a clock at 120 MHz (period of 8.33 ns) */
#define	**SRAM_DATA_HOLD_TIME** 0
#define	**SRAM_TURN_AROUND_TIME** 1 /* 4ns with a clock at 120 MHz (period of 8.33 ns) */

Define Documentation

#define SRAM_ADDR_SETUP_TIME 2 /* 10ns with a clock at 120 M

Definition at line **103** of file **stm32l4r9i_eval_sram.c**.

Referenced by **BSP_SRAM_Init()**.

#define SRAM_DATA_HOLD_TIME 0

Definition at line **105** of file **stm32l4r9i_eval_sram.c**.

Referenced by **BSP_SRAM_Init()**.

#define SRAM_DATA_SETUP_TIME 1 /* 6ns with a clock at 120 MH

Definition at line **104** of file **stm32l4r9i_eval_sram.c**.

Referenced by **BSP_SRAM_Init()**.

#define SRAM_TURN_AROUND_TIME 1 /* 4ns with a clock at 120

Definition at line **106** of file **stm32l4r9i_eval_sram.c**.

Referenced by **BSP_SRAM_Init()**.

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL SRAM

Defines

#define	**SRAM_OK** 0x00 SRAM status structure definition.
#define	**SRAM_ERROR** 0x01
#define	**SRAM_DEVICE_ADDR** ((uint32_t)0x60000000)
#define	**SRAM_DEVICE_SIZE** ((uint32_t)0x200000) /* SRAM device 1024 words of 16-bits */
#define	**SRAM_MEMORY_WIDTH** FMC_NORSRAM_MEM_BUS_WI
#define	**SRAM_BURSTACCESS** FMC_BURST_ACCESS_MODE_DI
#define	**SRAM_WRITEBURST** FMC_WRITE_BURST_DISABLE
#define	**SRAM_DMAx_CLK_ENABLE** __HAL_RCC_DMA1_CLK_EN
#define	**SRAM_DMAx_CHANNEL** DMA1_Channel1
#define	**SRAM_DMAx_IRQn** DMA1_Channel1_IRQn
#define	**SRAM_DMAx_IRQHandler** DMA1_Channel1_IRQHandler

Define Documentation

#define SRAM_BURSTACCESS FMC_BURST_ACCESS_MODE_DI

Definition at line 78 of file stm32l4r9i_eval_sram.h.

#define SRAM_DEVICE_ADDR ((uint32_t)0x60000000)

Definition at line 73 of file stm32l4r9i_eval_sram.h.

#define SRAM_DEVICE_SIZE ((uint32_t)0x200000) /* SRAM device

Definition at line 74 of file stm32l4r9i_eval_sram.h.

#define SRAM_DMAx_CHANNEL DMA1_Channel1

Definition at line 84 of file stm32l4r9i_eval_sram.h.

Referenced by SRAM_MspInit().

#define SRAM_DMAx_CLK_ENABLE __HAL_RCC_DMA1_CLK_EN

Definition at line 83 of file stm32l4r9i_eval_sram.h.

Referenced by SRAM_MspInit().

#define SRAM_DMAx_IRQHandler DMA1_Channel1_IRQHandler

Definition at line 86 of file stm32l4r9i_eval_sram.h.

#define SRAM_DMAx_IRQn DMA1_Channel1_IRQn

Definition at line 85 of file stm32l4r9i_eval_sram.h.

Referenced by SRAM_MspInit().

#define SRAM_ERROR 0x01

Definition at line 71 of file stm32l4r9i_eval_sram.h.

Referenced by BSP_SRAM_Init(), BSP_SRAM_ReadData(), BSP_SRAM_ReadData_DMA(), BSP_SRAM_WriteData(), and BSP_SRAM_WriteData_DMA().

#define SRAM_MEMORY_WIDTH FMC_NORSRAM_MEM_BUS_WI

Definition at line 76 of file stm32l4r9i_eval_sram.h.

#define SRAM_OK 0x00

SRAM status structure definition.

Definition at line 70 of file stm32l4r9i_eval_sram.h.

Referenced by BSP_SRAM_Init(), BSP_SRAM_ReadData(), BSP_SRAM_ReadData_DMA(), BSP_SRAM_WriteData(), and BSP_SRAM_WriteData_DMA().

#define SRAM_WRITEBURST FMC_WRITE_BURST_DISABLE

Definition at line 80 of file stm32l4r9i_eval_sram.h.

Generated on Thu Oct 12 2017 10:54:00 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

Private Functions

STM32L4R9I_EVAL SRAM

Functions

static void	**SRAM_MspInit** (void) Initializes SRAM MSP.

Function Documentation

static void SRAM_MspInit (void) [static]

Initializes SRAM MSP.

Return values:

None

Definition at line **267** of file **stm32l4r9i_eval_sram.c**.

References **SRAM_DMAx_CHANNEL**, **SRAM_DMAx_CLK_ENABLE**, **SRAM_DMAx_IRQn**, and **sramHandle**.

Referenced by **BSP_SRAM_Init()**.

STM32L4R9I_EVAL BSP User Manual

Private Variables

STM32L4R9I_EVAL SRAM

Variables

static SRAM_HandleTypeDef	**sramHandle**
static FMC_NORSRAM_TimingTypeDef	**Timing**

Variable Documentation

SRAM_HandleTypeDef sramHandle [static]

Definition at line 89 of file stm32l4r9i_eval_sram.c.

Referenced by BSP_SRAM_DMA_IRQHandler(), BSP_SRAM_Init(), BSP_SRAM_ReadData(), BSP_SRAM_ReadData_DMA(), BSP_SRAM_WriteData(), BSP_SRAM_WriteData_DMA(), and SRAM_MspInit().

FMC_NORSRAM_TimingTypeDef Timing [static]

Definition at line 90 of file stm32l4r9i_eval_sram.c.

Referenced by BSP_SRAM_Init().

STM32L4R9I_EVAL BSP User Manual

Imported Variables

STM32L4R9I_EVAL RGB TS

Variables

char *	**ts_event_string_tab** [**TOUCH_EVENT_NB_MAX**] Table for touchscreen event information display on LCD : table indexed on enum **TS_TouchEventTypeDef** information.
char *	**ts_gesture_id_string_tab** [**GEST_ID_NB_MAX**] Table for touchscreen gesture Id information display on LCD : table indexed on enum **TS_GestureIdTypeDef** information.

Variable Documentation

char* ts_event_string_tab[TOUCH_EVENT_NB_MAX]

Table for touchscreen event information display on LCD : table indexed on enum **TS_TouchEventTypeDef** information.

char* ts_gesture_id_string_tab[GEST_ID_NB_MAX]

Table for touchscreen gesture Id information display on LCD : table indexed on enum **TS_GestureIdTypeDef** information.

STM32L4R9I_EVAL BSP User Manual

Defines

Exported Constants

STM32L4R9I_EVAL TS

Defines

#define	**TS_NO_IRQ_PENDING** ((uint8_t) 0)
#define	**TS_IRQ_PENDING** ((uint8_t) 1)
#define	**TS_SWAP_NONE** ((uint8_t) 0x01)
#define	**TS_SWAP_X** ((uint8_t) 0x02)
#define	**TS_SWAP_Y** ((uint8_t) 0x04)
#define	**TS_SWAP_XY** ((uint8_t) 0x08)
#define	**TS_ORIENTATION_PORTRAIT** ((uint8_t) 0)
#define	**TS_ORIENTATION_LANDSCAPE** ((uint8_t) 1)

Define Documentation

#define TS_IRQ_PENDING ((uint8_t) 1)

Definition at line 89 of file stm32l4r9i_eval_ts.h.

#define TS_NO_IRQ_PENDING ((uint8_t) 0)

Definition at line 88 of file stm32l4r9i_eval_ts.h.

#define TS_ORIENTATION_LANDSCAPE ((uint8_t) 1)

Definition at line 97 of file stm32l4r9i_eval_ts.h.

#define TS_ORIENTATION_PORTRAIT ((uint8_t) 0)

Definition at line 96 of file stm32l4r9i_eval_ts.h.

#define TS_SWAP_NONE ((uint8_t) 0x01)

Definition at line 91 of file stm32l4r9i_eval_ts.h.

#define TS_SWAP_X ((uint8_t) 0x02)

Definition at line 92 of file stm32l4r9i_eval_ts.h.

#define TS_SWAP_XY ((uint8_t) 0x08)

Definition at line 94 of file stm32l4r9i_eval_ts.h.

#define TS_SWAP_Y ((uint8_t) 0x04)

Definition at line 93 of file stm32l4r9i_eval_ts.h.

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains definitions for STM32L4R9I_EVAL's LEDs,
00006   *          push-buttons and COM ports hardware resources.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_H
00039 #define __STM32L4R9I_EVAL_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /**
00046   * @brief  Define for STM32L4R9I_EVAL board
00047   */
00048
00049 /* Includes ------------------------------
-----------------------------------*/
00050 #include "stm32l4xx_hal.h"
00051 #include "stm32l4r9i_eval_io.h"
00052 #if defined(BSP_USE_CMSIS_OS)
00053 #include "cmsis_os.h"
00054 #endif
00055
00056 /** @addtogroup BSP
00057   * @{
00058   */
00059
00060 /** @addtogroup STM32L4R9I_EVAL
00061   * @{
00062   */
00063
00064
00065 /** @addtogroup STM32L4R9I_EVAL_Common
00066   * @{
00067   */
00068
```

```
00069 /** @defgroup STM32L4R9I_EVAL_Exported_Types
 Exported Types
00070   * @{
00071   */
00072
00073 /**
00074  * @brief LED Types Definition
00075  */
00076 typedef enum
00077 {
00078   LED1 = 0,
00079   LED_GREEN  = LED1,
00080
00081   LED2 = 1,
00082   LED_ORANGE = LED2,
00083
00084   LED3 = 2,
00085   LED_RED    = LED3,
00086
00087   LED4 = 3,
00088   LED_BLUE   = LED4
00089
00090 }Led_TypeDef;
00091
00092 /**
00093  * @brief BUTTON Types Definition
00094  */
00095 typedef enum
00096 {
00097   BUTTON_WAKEUP = 0,
00098   BUTTON_SEL    = 1,
00099   BUTTON_LEFT   = 2,
00100   BUTTON_RIGHT = 3,
00101   BUTTON_DOWN   = 4,
00102   BUTTON_UP     = 5
00103
00104 }Button_TypeDef;
```

```
00105
00106 typedef enum
00107 {
00108   BUTTON_MODE_GPIO = 0,
00109   BUTTON_MODE_EXTI = 1
00110
00111 }ButtonMode_TypeDef;
00112
00113 /**
00114  * @brief JOYSTICK Types Definition
00115  */
00116 typedef enum
00117 {
00118   JOY_SEL   = 0,
00119   JOY_LEFT  = 1,
00120   JOY_RIGHT = 2,
00121   JOY_DOWN  = 3,
00122   JOY_UP    = 4,
00123   JOY_NONE  = 5
00124
00125 }JOYState_TypeDef;
00126
00127 typedef enum
00128 {
00129   JOY_MODE_GPIO = 0,
00130   JOY_MODE_EXTI = 1
00131
00132 }JOYMode_TypeDef;
00133
00134 /**
00135  * @brief COM Types Definition
00136  */
00137 typedef enum
00138 {
00139   COM1 = 0
00140
00141 }COM_TypeDef;
```

```
00142
00143 /**
00144   * @}
00145   */
00146
00147 /** @defgroup STM32L4R9I_EVAL_Exported_Const
ants Exported Constants
00148   * @{
00149   */
00150
00151 /** @defgroup STM32L4R9I_EVAL_LED LED Consta
nts
00152   * @{
00153   */
00154 #define LEDn                                4
00155
00156 #define LED1_PIN                            GPI
O_PIN_15
00157 #define LED1_GPIO_PORT                      GPI
OB
00158 #define LED1_GPIO_CLK_ENABLE()              __H
AL_RCC_GPIOB_CLK_ENABLE()
00159 #define LED1_GPIO_CLK_DISABLE()             __H
AL_RCC_GPIOB_CLK_DISABLE()
00160
00161 #define LED2_PIN                            GPI
O_PIN_13
00162 #define LED2_GPIO_PORT                      GPI
OH
00163 #define LED2_GPIO_CLK_ENABLE()              __H
AL_RCC_GPIOH_CLK_ENABLE()
00164 #define LED2_GPIO_CLK_DISABLE()             __H
AL_RCC_GPIOH_CLK_DISABLE()
00165
00166 #define LED3_PIN                            GPI
O_PIN_14
00167 #define LED3_GPIO_PORT                      GPI
```

```
OH
00168 #define LED3_GPIO_CLK_ENABLE()            __H
AL_RCC_GPIOH_CLK_ENABLE()
00169 #define LED3_GPIO_CLK_DISABLE()           __H
AL_RCC_GPIOH_CLK_DISABLE()
00170
00171 #define LED4_PIN                          GPI
O_PIN_15
00172 #define LED4_GPIO_PORT                    GPI
OH
00173 #define LED4_GPIO_CLK_ENABLE()            __H
AL_RCC_GPIOH_CLK_ENABLE()
00174 #define LED4_GPIO_CLK_DISABLE()           __H
AL_RCC_GPIOH_CLK_DISABLE()
00175
00176 #define LEDx_GPIO_CLK_ENABLE(__LED__)    do
{ if((__LED__) == LED1) { __HAL_RCC_GPIOB_CLK_ENAB
LE(); } else \
00177
  __HAL_RCC_GPIOH_CLK_ENABLE(); } while(0)
00178
00179 #define LEDx_GPIO_CLK_DISABLE(__LED__)   do
{ if((__LED__) == LED1) { __HAL_RCC_GPIOB_CLK_DISA
BLE(); } else \
00180
  __HAL_RCC_GPIOH_CLK_DISABLE(); } while(0)
00181 /**
00182   * @}
00183   */
00184
00185 /** @defgroup STM32L4R9I_EVAL_BUTTON  BUTTON
 Constants
00186   * @{
00187   */
00188 #define JOYn                              5
00189 #define BUTTONn                           1 +
 JOYn
```

```
00190
00191 /**
00192  * @brief Wakeup/Tamper push-button
00193  */
00194 #define WAKEUP_BUTTON_PIN                GPI
O_PIN_13
00195 #define WAKEUP_BUTTON_GPIO_PORT          GPI
OC
00196 #define WAKEUP_BUTTON_GPIO_CLK_ENABLE()  __H
AL_RCC_GPIOC_CLK_ENABLE()
00197 #define WAKEUP_BUTTON_GPIO_CLK_DISABLE() __H
AL_RCC_GPIOC_CLK_DISABLE()
00198 #define WAKEUP_BUTTON_EXTI_IRQn          EXT
I15_10_IRQn
00199
00200 #define BUTTONx_GPIO_CLK_ENABLE(__BUTTON__)
    do { if((__BUTTON__) == BUTTON_WAKEUP) { WAKEU
P_BUTTON_GPIO_CLK_ENABLE(); } } while(0)
00201
00202 #define BUTTONx_GPIO_CLK_DISABLE(__BUTTON__)
    do { if((__BUTTON__) == BUTTON_WAKEUP) { WAKEU
P_BUTTON_GPIO_CLK_DISABLE(); } } while(0)
00203
00204
00205 /*****************************************
***************/
00206 /* The joystick buttons are accessed thru th
e IO Expander */
00207 /*****************************************
***************/
00208
00209 /**
00210  * @brief Joystick Sel push-button
00211  */
00212 #define SEL_JOY_PIN                      IO_
PIN_0
00213 #define SEL_JOY_EXTI_IRQn                EXT
```

```
I0_IRQn
00214
00215 /**
00216  * @brief Joystick Down push-button
00217  */
00218 #if defined(USE_STM32L4R9I_EVAL_REVA)
00219 /* Hide Joystick 180° rotation misplacement
*/
00220 #define DOWN_JOY_PIN                        IO_
PIN_4
00221 #else
00222 #define DOWN_JOY_PIN                        IO_
PIN_1
00223 #endif
00224
00225 #define DOWN_JOY_EXTI_IRQn                  EXT
I0_IRQn
00226
00227 /**
00228  * @brief Joystick Left push-button
00229  */
00230 #if defined(USE_STM32L4R9I_EVAL_REVA)
00231 /* Hide Joystick 180° rotation misplacement
*/
00232 #define LEFT_JOY_PIN                        IO_
PIN_3
00233 #else
00234 #define LEFT_JOY_PIN                        IO_
PIN_2
00235 #endif
00236
00237 #define LEFT_JOY_EXTI_IRQn                  EXT
I0_IRQn
00238
00239 /**
00240  * @brief Joystick Right push-button
00241  */
```

```
00242 #if defined(USE_STM32L4R9I_EVAL_REVA)
00243 /* Hide Joystick 180° rotation misplacement
*/
00244 #define RIGHT_JOY_PIN                       IO_
PIN_2
00245 #else
00246 #define RIGHT_JOY_PIN                       IO_
PIN_3
00247 #endif
00248
00249 #define RIGHT_JOY_EXTI_IRQn                 EXT
I0_IRQn
00250
00251 /**
00252  * @brief Joystick Up push-button
00253  */
00254 #if defined(USE_STM32L4R9I_EVAL_REVA)
00255 /* Hide Joystick 180° rotation misplacement
*/
00256 #define UP_JOY_PIN                          IO_
PIN_1
00257 #else
00258 #define UP_JOY_PIN                          IO_
PIN_4
00259 #endif
00260
00261 #define UP_JOY_EXTI_IRQn                    EXT
I0_IRQn
00262
00263
00264 #define JOY_ALL_PINS                        (RI
GHT_JOY_PIN | LEFT_JOY_PIN | UP_JOY_PIN | DOWN_JOY
_PIN | SEL_JOY_PIN)
00265 #define JOY_NONE_PIN                        JOY
_ALL_PINS
00266
00267 /**
```

```
00268   * @}
00269   */
00270
00271 /** @defgroup STM32L4R9I_EVAL_COM  COM Const
ants
00272   * @{
00273   */
00274 #define COMn
    1
00275
00276 /**
00277  * @brief Definition for COM port1, connecte
d to USART1
00278  */
00279 #define EVAL_COM1
    LPUART1
00280 #define EVAL_COM1_CLK_ENABLE()
    __HAL_RCC_LPUART1_CLK_ENABLE()
00281 #define EVAL_COM1_CLK_DISABLE()
    __HAL_RCC_LPUART1_CLK_DISABLE()
00282
00283 #define EVAL_COM1_TX_PIN
    GPIO_PIN_7
00284 #define EVAL_COM1_TX_GPIO_PORT
    GPIOG
00285 #define EVAL_COM1_TX_GPIO_CLK_ENABLE()
    __HAL_RCC_GPIOG_CLK_ENABLE()
00286 #define EVAL_COM1_TX_GPIO_CLK_DISABLE()
    __HAL_RCC_GPIOG_CLK_DISABLE()
00287 #define EVAL_COM1_TX_AF
    GPIO_AF8_LPUART1
00288
00289 #define EVAL_COM1_RX_PIN
    GPIO_PIN_8
00290 #define EVAL_COM1_RX_GPIO_PORT
    GPIOG
00291 #define EVAL_COM1_RX_GPIO_CLK_ENABLE()
```

```
    __HAL_RCC_GPIOG_CLK_ENABLE()
00292 #define EVAL_COM1_RX_GPIO_CLK_DISABLE()
    __HAL_RCC_GPIOG_CLK_DISABLE()
00293 #define EVAL_COM1_RX_AF
    GPIO_AF8_LPUART1
00294
00295 #define EVAL_COM1_IRQn
    LPUART1_IRQn
00296
00297 #define COMx_CLK_ENABLE(__INDEX__)
    do { if((__INDEX__) == COM1) { EVAL_COM1_CLK_E
NABLE(); } } while(0)
00298 #define COMx_CLK_DISABLE(__INDEX__)
    do { if((__INDEX__) == COM1) { EVAL_COM1_CLK_D
ISABLE(); } } while(0)
00299
00300 #define COMx_TX_GPIO_CLK_ENABLE(__INDEX__)
    do { if((__INDEX__) == COM1) { EVAL_COM1_TX_GP
IO_CLK_ENABLE(); } } while(0)
00301 #define COMx_TX_GPIO_CLK_DISABLE(__INDEX__)
    do { if((__INDEX__) == COM1) { EVAL_COM1_TX_GP
IO_CLK_DISABLE(); } } while(0)
00302
00303 #define COMx_RX_GPIO_CLK_ENABLE(__INDEX__)
    do { if((__INDEX__) == COM1) { EVAL_COM1_RX_GP
IO_CLK_ENABLE(); } } while(0)
00304 #define COMx_RX_GPIO_CLK_DISABLE(__INDEX__)
    do { if((__INDEX__) == COM1) { EVAL_COM1_RX_GP
IO_CLK_DISABLE(); } } while(0)
00305
00306 /**
00307   * @}
00308   */
00309
00310 #ifdef HAL_ADC_MODULE_ENABLED
00311
00312 /** @defgroup STM32L4R9I_EVAL_POTENTIOMETER
```

```
 POTENTIOMETER Constants
00313   * @{
00314   */
00315 /**
00316  * @brief  Potentiometer Status values
00317  */
00318 #define BSP_POTENTIOMETER_OK       0x00
00319 #define BSP_POTENTIOMETER_ERROR    0xFF
00320
00321 /**
00322  * @brief Definition for Potentiometer, conn
ected to ADC1
00323  */
00324 #define ADCx                             ADC1
00325 #define ADCx_CLK_ENABLE()                __HA
L_RCC_ADC_CLK_ENABLE()
00326 #define ADCx_CHANNEL_GPIO_CLK_ENABLE()  __HA
L_RCC_GPIOA_CLK_ENABLE()
00327
00328 #define ADCx_FORCE_RESET()               __HA
L_RCC_ADC_FORCE_RESET()
00329 #define ADCx_RELEASE_RESET()             __HA
L_RCC_ADC_RELEASE_RESET()
00330
00331 /* Definition for ADCx Channel Pin */
00332 #define ADCx_CHANNEL_PIN                 GPIO
_PIN_0
00333 #define ADCx_CHANNEL_GPIO_PORT           GPIOA

00334
00335 /* Definition for ADCx's Channel */
00336 #define ADCx_CHANNEL                     ADC_
CHANNEL_5
00337 #define SAMPLINGTIME                     ADC_
SAMPLETIME_3CYCLES
00338 #define ADCx_POLL_TIMEOUT                10
00339
```

```
00340 #endif /* HAL_ADC_MODULE_ENABLED */
00341
00342 /**
00343  * @brief Definition for Potentiometer conne
cted to OPAMP
00344  */
00345 /* Definition for OPAMPx Pins */
00346 #define OPAMPx_INP_PIN                  GPIO
_PIN_0
00347 #define OPAMPx_INP_GPIO_PORT            GPIOA

00348 #define OPAMPx_INM_PIN                  GPIO
_PIN_1
00349 #define OPAMPx_INM_GPIO_PORT            GPIOA

00350 #define OPAMPx_VOUT_PIN                 GPIO
_PIN_3
00351 #define OPAMPx_VOUT_GPIO_PORT           GPIOA

00352
00353 #define OPAMPx                          OPAMP

00354 #define OPAMPx_CLK_ENABLE()             __HA
L_RCC_OPAMP_CLK_ENABLE()
00355 #define OPAMPx_CLK_DISABLE()            __HA
L_RCC_OPAMP_CLK_DISABLE()
00356 #define OPAMPx_GPIO_CLK_ENABLE()        __HA
L_RCC_GPIOA_CLK_ENABLE()
00357 #define OPAMPx_GPIO_CLK_DISABLE()       __HA
L_RCC_GPIOA_CLK_DISABLE()
00358
00359 #define OPAMPx_FORCE_RESET()            __HA
L_RCC_OPAMP_FORCE_RESET()
00360 #define OPAMPx_RELEASE_RESET()          __HA
L_RCC_OPAMP_RELEASE_RESET()
00361
00362 /**
```

```
  * @brief Eval Pins definition connected to MFX
  */

#define DSI_RST_PIN                    IO_PIN_9
#define EXT_RST_PIN                    IO_PIN_8
#define LCD_DISP_PIN                   IO_PIN_11
#define LCD_RST_PIN                    IO_PIN_12
#define OTG_FS_POWER_SWITCH_PIN        IO_PIN_6
#define OTG_FS_OVER_CURRENT_PIN        IO_PIN_7
#define SD_DETECT_PIN                  IO_PIN_5
#define SWIRE_PIN                      IO_PIN_13

/**
  * @}
  */

/** @defgroup STM32L4R9I_EVAL_BUS  BUS Constants
  * @{
  */
/* Exported constant IO --------------------------------------------------------*/
/*###################### I2Cx ###############################*/
/* User can use this section to tailor I2Cx instance used and associated
   resources */
/* Definition for I2Cx Pins */
```

```
00387 #define EVAL_I2Cx_SCL_PIN
    GPIO_PIN_4
00388 #define EVAL_I2Cx_SCL_GPIO_PORT
    GPIOH
00389 #define EVAL_I2Cx_SDA_PIN
    GPIO_PIN_5
00390 #define EVAL_I2Cx_SDA_GPIO_PORT
    GPIOH
00391 #define EVAL_I2Cx_SCL_SDA_AF
    GPIO_AF4_I2C2
00392
00393 /* Definition for I2Cx clock resources */
00394 #define EVAL_I2Cx
    I2C2
00395 #define EVAL_I2Cx_CLK_ENABLE()
    __HAL_RCC_I2C2_CLK_ENABLE()
00396 #define EVAL_I2Cx_CLK_DISABLE()
    __HAL_RCC_I2C2_CLK_DISABLE()
00397 #define EVAL_I2Cx_SDA_GPIO_CLK_ENABLE()
    __HAL_RCC_GPIOH_CLK_ENABLE()
00398 #define EVAL_I2Cx_SDA_GPIO_CLK_DISABLE()
    __HAL_RCC_GPIOH_CLK_DISABLE()
00399 #define EVAL_I2Cx_SCL_GPIO_CLK_ENABLE()
    __HAL_RCC_GPIOH_CLK_ENABLE()
00400 #define EVAL_I2Cx_SCL_GPIO_CLK_DISABLE()
    __HAL_RCC_GPIOH_CLK_DISABLE()
00401
00402 #define EVAL_I2Cx_FORCE_RESET()
    __HAL_RCC_I2C2_FORCE_RESET()
00403 #define EVAL_I2Cx_RELEASE_RESET()
    __HAL_RCC_I2C2_RELEASE_RESET()
00404
00405 /* Definition for I2Cx's NVIC */
00406 #define EVAL_I2Cx_EV_IRQn
    I2C2_EV_IRQn
00407 #define EVAL_I2Cx_EV_IRQHandler
    I2C2_EV_IRQHandler
```

```
00408 #define EVAL_I2Cx_ER_IRQn
    I2C2_ER_IRQn
00409 #define EVAL_I2Cx_ER_IRQHandler
    I2C2_ER_IRQHandler
00410
00411 /* I2C clock setting */
00412 #define EVAL_RCC_PERIPHCLK_I2C
    RCC_PERIPHCLK_I2C2
00413 #define EVAL_RCC_CLKSOURCE_I2C
    RCC_I2C2CLKSOURCE_SYSCLK
00414
00415 /* I2C TIMING Register define when I2C clock
 source is SYSCLK */
00416 /* I2C TIMING is calculated in case of the I
2C Clock source is the SYSCLK = 120 MHz */
00417 /* Set 0xC080242F value to reach 100 KHz spe
ed (Rise time = 640ns, Fall time = 20ns) */
00418 #ifndef EVAL_I2Cx_TIMING
00419  #define EVAL_I2Cx_TIMING
    0xC080242F
00420 #endif /* EVAL_I2Cx_TIMING */
00421
00422
00423 /* Definition for TS (Capacitive Touch Panel
) I2Cx resources */
00424 #define EVAL_TS_I2Cx
    I2C2
00425 #define EVAL_TS_I2Cx_CLK_ENABLE()
    __HAL_RCC_I2C2_CLK_ENABLE()
00426 #define EVAL_TS_DMAx_CLK_ENABLE()
    __HAL_RCC_DMA1_CLK_ENABLE()
00427 #define EVAL_TS_I2Cx_SDA_GPIO_CLK_ENABLE()
    __HAL_RCC_GPIOH_CLK_ENABLE()
00428 #define EVAL_TS_I2Cx_SDA_GPIO_CLK_DISABLE()
    __HAL_RCC_GPIOH_CLK_DISABLE()
00429 #define EVAL_TS_I2Cx_SCL_GPIO_CLK_ENABLE()
    __HAL_RCC_GPIOH_CLK_ENABLE()
```

```
00430 #define EVAL_TS_I2Cx_SCL_GPIO_CLK_DISABLE()
    __HAL_RCC_GPIOH_CLK_DISABLE()
00431
00432 #define EVAL_TS_I2Cx_FORCE_RESET()
    __HAL_RCC_I2C2_FORCE_RESET()
00433 #define EVAL_TS_I2Cx_RELEASE_RESET()
    __HAL_RCC_I2C2_RELEASE_RESET()
00434
00435 /* Definition for I2Cx Pins */
00436 #define EVAL_TS_I2Cx_SCL_PIN
    GPIO_PIN_4
00437 #define EVAL_TS_I2Cx_SCL_GPIO_PORT
    GPIOH
00438 #define EVAL_TS_I2Cx_SDA_PIN
    GPIO_PIN_5
00439 #define EVAL_TS_I2Cx_SDA_GPIO_PORT
    GPIOH
00440 #define EVAL_TS_I2Cx_SCL_SDA_AF
    GPIO_AF4_I2C2
00441
00442 /* I2C interrupt requests */
00443 #define EVAL_TS_I2Cx_EV_IRQn
    I2C2_EV_IRQn
00444 #define EVAL_TS_I2Cx_ER_IRQn
    I2C2_ER_IRQn
00445
00446 /* I2C clock setting */
00447 #define EVAL_TS_RCC_PERIPHCLK_I2C
    RCC_PERIPHCLK_I2C2
00448 #define EVAL_TS_RCC_CLKSOURCE_I2C
    RCC_I2C2CLKSOURCE_SYSCLK
00449
00450
00451 /* Exported constantIO ----------------------
----------------------------------*/
00452
00453 #define EEPROM_I2C_ADDRESS
```

```
    0xA0
00454
00455 /* I2C clock speed configuration (in Hz)
00456    WARNING:
00457    Make sure that this define is not already
 declared in other files (ie.
00458    stm324xg_eval.h file). It can be used in
parallel by other modules. */
00459 #ifndef BSP_I2C_SPEED
00460  #define BSP_I2C_SPEED
    100000
00461 #endif /* BSP_I2C_SPEED */
00462
00463
00464 #define IO_I2C_ADDRESS
    0x84
00465 #define IDD_I2C_ADDRESS
    IO_I2C_ADDRESS
00466 #define AUDIO_I2C_ADDRESS
    0x34
00467 #define LCD_RGB_I2C_ADDRESS
    0x70
00468 #define TS_RGB_I2C_ADDRESS
    0x70
00469 #define TS_DSI_I2C_ADDRESS
    0x70
00470
00471 /* Maximum Timeout values for flags waiting
loops. These timeouts are not based
00472    on accurate values, they just guarantee t
hat the application will not remain
00473    stuck if the I2C communication is corrupt
ed.
00474    You may modify these timeout values depen
ding on CPU frequency and application
00475    conditions (interrupts routines ...). */
00476 #define EVAL_I2Cx_TIMEOUT_MAX
```

```
    3000
00477
00478
00479 /*#################### Audio Codec ########
##################*/
00480 /* Audio INT Pin definition */
00481 #define AUDIO_INT_GPIO_PORT              GP
IOI                       /* GPIOI */
00482 #define AUDIO_INT_GPIO_CLK_ENABLE()      __
HAL_RCC_GPIOI_CLK_ENABLE()
00483 #define AUDIO_INT_GPIO_CLK_DISABLE()     __
HAL_RCC_GPIOI_CLK_DISABLE()
00484 #define AUDIO_INT_PIN                    GP
IO_PIN_6                  /* PI.06 */
00485
00486
00487 /*#################### MFX ################
##################*/
00488 /**
00489   * @brief  MFX interface pins
00490   */
00491 #define MFX_INT_GPIO_PORT                GP
IOA                       /* GPIOA */
00492 #define MFX_INT_GPIO_CLK_ENABLE()        __
HAL_RCC_GPIOA_CLK_ENABLE()
00493 #define MFX_INT_GPIO_CLK_DISABLE()       __
HAL_RCC_GPIOA_CLK_DISABLE()
00494 #define MFX_INT_PIN                      GP
IO_PIN_0                  /* PA.00 */
00495 #define MFX_INT_EXTI_IRQn                EX
TI0_IRQn
00496
00497 #define MFX_WAKEUP_GPIO_PORT             GP
IOG                       /* GPIOG */
00498 #define MFX_WAKEUP_GPIO_CLK_ENABLE()     __
HAL_RCC_GPIOG_CLK_ENABLE()
00499 #define MFX_WAKEUP_GPIO_CLK_DISABLE()    __
```

```
HAL_RCC_GPIOG_CLK_DISABLE()
00500 #define MFX_WAKEUP_PIN                          GP
IO_PIN_9                  /* PG.09 */
00501
00502 /* Legacy  */
00503 #define IDD_INT_GPIO_PORT                       MF
X_INT_GPIO_PORT
00504 #define IDD_INT_GPIO_CLK_ENABLE()               MF
X_INT_GPIO_CLK_ENABLE()
00505 #define IDD_INT_GPIO_CLK_DISABLE()              MF
X_INT_GPIO_CLK_DISABLE()
00506 #define IDD_INT_PIN                             MF
X_INT_PIN
00507 #define IDD_INT_EXTI_IRQn                       MF
X_INT_EXTI_IRQn
00508 #define IDD_WAKEUP_GPIO_PORT                    MF
X_WAKEUP_GPIO_PORT
00509 #define IDD_WAKEUP_GPIO_CLK_ENABLE()            MF
X_WAKEUP_GPIO_CLK_ENABLE()
00510 #define IDD_WAKEUP_GPIO_CLK_DISABLE()           MF
X_WAKEUP_GPIO_CLK_DISABLE()
00511 #define IDD_WAKEUP_PIN                          MF
X_WAKEUP_PIN
00512
00513 /**
00514   * @brief  Idd current measurement interfac
e pins on MFX
00515   */
00516 #define IDD_AMP_CONTROL_PIN                     AG
PIO_PIN_1
00517
00518 /*####################### LCD ################
##########*/
00519
00520 /** MB1314 DSI **/
00521
00522 /* DSI TE pin */
```

```
00523 #define LCD_DSI_TE_PIN
 GPIO_PIN_6
00524 #define LCD_DSI_TE_CTRL_GPIO_PORT
 GPIOG
00525 #define LCD_DSI_TE_CTRL_GPIO_CLK_ENABLE()
 __HAL_RCC_GPIOG_CLK_ENABLE()
00526 #define LCD_DSI_TE_CTRL_GPIO_CLK_DISABLE()
 __HAL_RCC_GPIOG_CLK_DISABLE()
00527
00528 /* Backlight control pin */
00529 #define LCD_DSI_BL_CTRL_PIN
 GPIO_PIN_14
00530 #define LCD_DSI_BL_CTRL_GPIO_PORT
 GPIOB
00531 #define LCD_DSI_BL_CTRL_GPIO_CLK_ENABLE()
 __HAL_RCC_GPIOB_CLK_ENABLE()
00532 #define LCD_DSI_BL_CTRL_GPIO_CLK_DISABLE()
 __HAL_RCC_GPIOB_CLK_DISABLE()
00533
00534 /* DSI TS INT pin */
00535 #define TS_DSI_INT_PIN
 GPIO_PIN_2
00536 #define TS_DSI_INT_GPIO_PORT
 GPIOC
00537 #define TS_DSI_INT_GPIO_CLK_ENABLE()
 __HAL_RCC_GPIOC_CLK_ENABLE()
00538 #define TS_DSI_INT_GPIO_CLK_DISABLE()
 __HAL_RCC_GPIOC_CLK_DISABLE()
00539 #define TS_DSI_INT_EXTI_IRQn
 EXTI2_IRQn
00540
00541
00542 /** MB1315 RGB **/
00543
00544 /* Display enable pin */
00545 #define LCD_RGB_DISP_PIN
 IO_PIN_11   /* MFX_GPIO11 */
```

```
00546
00547 /* Display enable pin */
00548 #define LCD_RGB_RST_PIN
 IO_PIN_12   /* MFX_GPIO12 */
00549
00550 /* Backlight control pin */
00551 #define LCD_RGB_BL_CTRL_PIN
 GPIO_PIN_5
00552 #define LCD_RGB_BL_CTRL_GPIO_PORT
 GPIOA
00553 #define LCD_RGB_BL_CTRL_GPIO_CLK_ENABLE()
 __HAL_RCC_GPIOA_CLK_ENABLE()
00554 #define LCD_RGB_BL_CTRL_GPIO_CLK_DISABLE()
 __HAL_RCC_GPIOA_CLK_DISABLE()
00555
00556 /* RGB TS INT pin */
00557 #define TS_RGB_INT_PIN
 GPIO_PIN_2
00558 #define TS_RGB_INT_GPIO_PORT
 GPIOC
00559 #define TS_RGB_INT_GPIO_CLK_ENABLE()
 __HAL_RCC_GPIOC_CLK_ENABLE()
00560 #define TS_RGB_INT_GPIO_CLK_DISABLE()
 __HAL_RCC_GPIOC_CLK_DISABLE()
00561 #define TS_RGB_INT_EXTI_IRQn
 EXTI2_IRQn
00562
00563 /**
00564   * @}
00565   */
00566
00567 /**
00568   * @}
00569   */
00570
00571 /** @defgroup STM32L4R9I_EVAL_Exported_Funct
ions Exported Functions
```

```
00572   * @{
00573   */
00574 uint32_t         BSP_GetVersion(void);
00575 void             BSP_LED_Init(Led_TypeDef Le
d);
00576 void             BSP_LED_DeInit(Led_TypeDef
Led);
00577 void             BSP_LED_On(Led_TypeDef Led)
;
00578 void             BSP_LED_Off(Led_TypeDef Led
);
00579 void             BSP_LED_Toggle(Led_TypeDef
Led);
00580 void             BSP_PB_Init(Button_TypeDef
Button, ButtonMode_TypeDef ButtonMode);
00581 void             BSP_PB_DeInit(Button_TypeDef
 Button);
00582 uint32_t         BSP_PB_GetState(Button_Type
Def Button);
00583 #if defined(HAL_UART_MODULE_ENABLED)
00584 void             BSP_COM_Init(COM_TypeDef CO
M, UART_HandleTypeDef* huart);
00585 void             BSP_COM_DeInit(COM_TypeDef
COM, UART_HandleTypeDef *huart);
00586 #endif /* HAL_UART_MODULE_ENABLED */
00587 #ifdef HAL_ADC_MODULE_ENABLED
00588 uint32_t         BSP_POTENTIOMETER_Init(void
);
00589 uint32_t         BSP_POTENTIOMETER_DeInit(vo
id);
00590 uint32_t         BSP_POTENTIOMETER_GetLevel(
void);
00591 #endif /* HAL_ADC_MODULE_ENABLED */
00592 uint8_t          BSP_JOY_Init(JOYMode_TypeDef
 JoyMode);
00593 void             BSP_JOY_DeInit(void);
00594 JOYState_TypeDef BSP_JOY_GetState(void);
```

```
00595
00596 /* These __weak functions can be surcharged
by application code for specific application needs
 */
00597 void              BSP_ErrorHandler(void);
00598
00599 /**
00600   * @}
00601   */
00602
00603 /**
00604   * @}
00605   */
00606
00607 /**
00608   * @}
00609   */
00610
00611 /**
00612   * @}
00613   */
00614
00615 #ifdef __cplusplus
00616 }
00617 #endif
00618
00619 #endif /* __STM32L4R9I_EVAL_H */
00620
00621 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_io.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_io.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_io.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_IO_H
00039 #define __STM32L4R9I_EVAL_IO_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-----------------------------------*/
00046 #include "stm32l4r9i_eval.h"
00047 #include "../Components/mfxstm32l152/mfxstm3
2l152.h"
00048
00049 /** @addtogroup BSP
00050   * @{
00051   */
00052
00053 /** @addtogroup STM32L4R9I_EVAL
00054   * @{
00055   */
00056
00057 /** @addtogroup STM32L4R9I_EVAL_IO
00058   * @{
00059   */
00060
00061 /* Exported types ------------------------
-----------------------------------*/
00062
00063 /** @defgroup STM32L4R9I_EVAL_IO_Exported_Ty
pes Exported Types
00064   * @{
00065   */
```

```
00066 typedef enum
00067 {
00068   IO_OK       = 0x00,
00069   IO_ERROR    = 0x01,
00070   IO_TIMEOUT  = 0x02
00071 }IO_StatusTypeDef;
00072 /**
00073   * @}
00074   */
00075
00076 /** @defgroup STM32L4R9I_EVAL_IO_Exported_Co
nstants Exported Constants
00077   * @{
00078   */
00079 #define IO_PIN_0                    ((uint32_t
)0x00000001)
00080 #define IO_PIN_1                    ((uint32_t
)0x00000002)
00081 #define IO_PIN_2                    ((uint32_t
)0x00000004)
00082 #define IO_PIN_3                    ((uint32_t
)0x00000008)
00083 #define IO_PIN_4                    ((uint32_t
)0x00000010)
00084 #define IO_PIN_5                    ((uint32_t
)0x00000020)
00085 #define IO_PIN_6                    ((uint32_t
)0x00000040)
00086 #define IO_PIN_7                    ((uint32_t
)0x00000080)
00087 #define IO_PIN_8                    ((uint32_t
)0x00000100)
00088 #define IO_PIN_9                    ((uint32_t
)0x00000200)
00089 #define IO_PIN_10                   ((uint32_t
)0x00000400)
00090 #define IO_PIN_11                   ((uint32_t
```

```
)0x00000800)
00091 #define IO_PIN_12                   ((uint32_t
)0x00001000)
00092 #define IO_PIN_13                   ((uint32_t
)0x00002000)
00093 #define IO_PIN_14                   ((uint32_t
)0x00004000)
00094 #define IO_PIN_15                   ((uint32_t
)0x00008000)
00095 #define AGPIO_PIN_0                 ((uint32_t
)0x00010000)
00096 #define AGPIO_PIN_1                 ((uint32_t
)0x00020000)
00097 #define IO_PIN_ALL                  ((uint32_t
)0x0003FFFF)
00098 /**
00099   * @}
00100   */
00101
00102 /* Exported functions ---------------------
-----------------------------------*/
00103
00104 /** @defgroup STM32L4R9I_EVAL_IO_Exported_Fu
nctions Exported Functions
00105   * @{
00106   */
00107
00108 uint8_t  BSP_IO_Init(void);
00109 uint8_t  BSP_IO_DeInit(void);
00110 uint32_t BSP_IO_ITGetStatus(uint32_t IO_Pin)
;
00111 void     BSP_IO_ITClear(uint32_t IO_Pin);
00112 uint8_t  BSP_IO_ConfigPin(uint32_t IO_Pin, I
O_ModeTypedef IO_Mode);
00113 void     BSP_IO_WritePin(uint32_t IO_Pin, ui
nt8_t PinState);
00114 uint32_t BSP_IO_ReadPin(uint32_t IO_Pin);
```

```
00115 void     BSP_IO_TogglePin(uint32_t IO_Pin);
00116
00117 /**
00118   * @}
00119   */
00120
00121 /**
00122   * @}
00123   */
00124
00125 /**
00126   * @}
00127   */
00128
00129 /**
00130   * @}
00131   */
00132
00133 #ifdef __cplusplus
00134 }
00135 #endif
00136
00137 #endif /* __STM32L4R9I_EVAL_IO_H */
00138
00139 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval.c
00004   * @author  MCD Application Team
00005   * @brief   This file provides a set of firmware functions to manage Leds,
00006   *          push-button and COM ports for STM32L4R9I_EVAL
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Includes ------------------------------
-----------------------------------*/
00038 #include "stm32l4r9i_eval.h"
00039 #include "stm32l4r9i_eval_io.h"
00040
00041 /** @addtogroup BSP
00042   * @{
00043   */
00044
00045 /** @defgroup STM32L4R9I_EVAL STM32L4R9I_EVAL

00046   * @{
00047   */
00048
00049 /** @defgroup STM32L4R9I_EVAL_Common STM32L4
R9I_EVAL Common
00050   * @{
00051   */
00052
00053 /** @defgroup STM32L4R9I_EVAL_Private_TypesD
efinitions Private Types Definitions
00054   * @brief This file provides firmware funct
ions to manage Leds, push-buttons,
00055   *        Joystick, and COM ports available
 on STM32L4R9I-EVAL evaluation board
00056   *        from STMicroelectronics.
00057   * @{
00058   */
00059
00060 /**
00061   * @}
00062   */
00063
00064 /** @defgroup STM32L4R9I_EVAL_Exported_Defin
```

```
es Exported Defines
00065   * @{
00066   */
00067
00068 /**
00069  * @brief STM32L4R9I EVAL BSP Driver version
 number
00070    */
00071 #define __STM32L4R9I_EVAL_BSP_VERSION_MAIN
 (0x01) /*!< [31:24] main version */
00072 #define __STM32L4R9I_EVAL_BSP_VERSION_SUB1
 (0x00) /*!< [23:16] sub1 version */
00073 #define __STM32L4R9I_EVAL_BSP_VERSION_SUB2
 (0x00) /*!< [15:8]  sub2 version */
00074 #define __STM32L4R9I_EVAL_BSP_VERSION_RC
 (0x00) /*!< [7:0]  release candidate */
00075 #define __STM32L4R9I_EVAL_BSP_VERSION
     ((__STM32L4R9I_EVAL_BSP_VERSION_MAIN << 24)\
00076
     |(__STM32L4R9I_EVAL_BSP_VERSION_SUB1 << 16)\
00077
     |(__STM32L4R9I_EVAL_BSP_VERSION_SUB2 << 8 )\
00078
     |(__STM32L4R9I_EVAL_BSP_VERSION_RC))
00079
00080 #ifdef HAL_I2C_MODULE_ENABLED
00081 /**
00082   * @brief BSP I2C users
00083   */
00084 #define BSP_I2C_NO_USER     0x00000000U
00085 #define BSP_I2C_AUDIO_USER  0x00000001U
00086 #define BSP_I2C_TS_USER     0x00000002U
00087 #define BSP_I2C_MFX_USER    0x00000004U
00088 #define BSP_I2C_EEPROM_USER 0x00000008U
00089 #define BSP_I2C_ALL_USERS   0x0000000FU
00090 #endif
00091
```

```
00092 /**
00093   * @}
00094   */
00095
00096
00097 /** @defgroup STM32L4R9I_EVAL_Exported_Variables Exported Variables
00098   * @{
00099   */
00100
00101 /**
00102  * @brief LED variables
00103  */
00104 GPIO_TypeDef*   LED_PORT[LEDn] =
    {LED1_GPIO_PORT,
00105
     LED2_GPIO_PORT,
00106
     LED3_GPIO_PORT,
00107
     LED4_GPIO_PORT};
00108
00109 const uint16_t  LED_PIN[LEDn] =
    {LED1_PIN,
00110
     LED2_PIN,
00111
     LED3_PIN,
00112
     LED4_PIN};
00113
00114 /**
00115  * @brief BUTTON variables
00116  */
00117 GPIO_TypeDef*   BUTTON_PORT[BUTTONn] =
    {WAKEUP_BUTTON_GPIO_PORT,
00118
```

```
     (GPIO_TypeDef *)0xFFFF,
00119
     (GPIO_TypeDef *)0xFFFF,
00120
     (GPIO_TypeDef *)0xFFFF,
00121
     (GPIO_TypeDef *)0xFFFF,
00122
     (GPIO_TypeDef *)0xFFFF};
00123 /* 0xFFFF to mark as invalid port (Joystick
buttons accessed thru IOExpander) */
00124
00125 const uint16_t  BUTTON_PIN[BUTTONn] =
    {WAKEUP_BUTTON_PIN,
00126
     SEL_JOY_PIN,
00127
     LEFT_JOY_PIN,
00128
     RIGHT_JOY_PIN,
00129
     DOWN_JOY_PIN,
00130
     UP_JOY_PIN};
00131
00132 const uint8_t   BUTTON_IRQn[BUTTONn] =
    {WAKEUP_BUTTON_EXTI_IRQn,
00133
     SEL_JOY_EXTI_IRQn,
00134
     LEFT_JOY_EXTI_IRQn,
00135
     RIGHT_JOY_EXTI_IRQn,
00136
     DOWN_JOY_EXTI_IRQn,
00137
     UP_JOY_EXTI_IRQn};
```

```
00138
00139 /**
00140  * @brief JOYSTICK variables
00141  */
00142
00143 const uint16_t JOY_PIN[JOYn] = {SEL_JOY_PIN,
00144                                 LEFT_JOY_PIN
,
00145                                 RIGHT_JOY_PIN
,
00146                                 DOWN_JOY_PIN
,
00147                                 UP_JOY_PIN};
00148
00149 const uint8_t JOY_IRQn[JOYn] = {SEL_JOY_EXTI
_IRQn,
00150                                 LEFT_JOY_EXT
I_IRQn,
00151                                 RIGHT_JOY_EX
TI_IRQn,
00152                                 DOWN_JOY_EXT
I_IRQn,
00153                                 UP_JOY_EXTI_
IRQn};
00154
00155 /**
00156  * @brief COM variables
00157  */
00158 USART_TypeDef*  COM_USART[COMn]   =
    {EVAL_COM1};
00159
00160 GPIO_TypeDef*   COM_TX_PORT[COMn] =
    {EVAL_COM1_TX_GPIO_PORT};
00161
00162 GPIO_TypeDef*   COM_RX_PORT[COMn] =
    {EVAL_COM1_RX_GPIO_PORT};
00163
```

```
00164 const uint16_t  COM_TX_PIN[COMn]  =
    {EVAL_COM1_TX_PIN};
00165
00166 const uint16_t  COM_RX_PIN[COMn]  =
    {EVAL_COM1_RX_PIN};
00167
00168 const uint16_t COM_TX_AF[COMn]    =
    {EVAL_COM1_TX_AF};
00169
00170 const uint16_t COM_RX_AF[COMn]    =
    {EVAL_COM1_RX_AF};
00171
00172 /**
00173  * @brief BUS variables
00174  */
00175
00176 #ifdef HAL_I2C_MODULE_ENABLED
00177 uint32_t hi2c_evalTimeout = EVAL_I2Cx_TIMEOU
T_MAX;   /*<! Value of Timeout when I2C communicat
ion fails */
00178 I2C_HandleTypeDef hi2c_eval;
00179 static uint32_t v_bspI2cUsers = BSP_I2C_NO_U
SER;
00180 #if defined(BSP_USE_CMSIS_OS)
00181 static osSemaphoreId BspI2cSemaphore;
00182 #endif
00183 #endif /* HAL_I2C_MODULE_ENABLED */
00184
00185 #ifdef HAL_ADC_MODULE_ENABLED
00186 static ADC_HandleTypeDef hEvalADC = {0};
00187 #endif
00188
00189 /**
00190   * @}
00191   */
00192
00193 /** @defgroup STM32L4R9I_EVAL_Private_Functi
```

```
onPrototypes Private Functions
00194   * @{
00195   */
00196 /* I2Cx bus function */
00197 #ifdef HAL_I2C_MODULE_ENABLED
00198 /* Link function I2C for EEPROM, Audio Codec
 peripheral and IO expanders*/
00199 static void                I2Cx_Init(uint32_t
 user);
00200 static void                I2Cx_MspInit(I2C_H
andleTypeDef *hi2c);
00201 static void                I2Cx_DeInit(uint32
_t user);
00202 static void                I2Cx_MspDeInit(I2C
_HandleTypeDef *hi2c);
00203 static HAL_StatusTypeDef  I2Cx_WriteBuffer(u
int16_t Addr, uint16_t Reg, uint16_t RegSize, uint
8_t *pBuffer, uint16_t Length);
00204 static HAL_StatusTypeDef  I2Cx_ReadBuffer(ui
nt16_t Addr, uint16_t Reg, uint16_t RegSize, uint8
_t *pBuffer, uint16_t Length);
00205 static HAL_StatusTypeDef  I2Cx_IsDeviceReady
(uint16_t DevAddress, uint32_t Trials);
00206 static void                I2Cx_Error (void);
00207
00208 /* Link function I2C for IO expanders */
00209 static void                I2Cx_WriteData(uin
t16_t Addr, uint16_t Reg, uint16_t RegSize, uint8_
t Value);
00210 static void                I2Cx_WriteMultiple
(uint16_t Addr, uint16_t Reg, uint16_t MemAddress,
 uint8_t *Buffer, uint16_t Length);
00211 static uint8_t             I2Cx_ReadData(uint
16_t Addr, uint16_t Reg, uint16_t RegSize);
00212 static uint8_t             I2Cx_ReadMultiple(
uint16_t Addr, uint16_t Reg, uint16_t MemAddress,
uint8_t *Buffer, uint16_t Length);
```

```
00213
00214 /* Link function for IOExpander IO functions
 */
00215 void                    MFX_IO_Init(void);
00216 void                    MFX_IO_DeInit(void
);
00217 void                    MFX_IO_Wakeup(void
);
00218 void                    MFX_IO_EnableWakeu
pPin(void);
00219 void                    MFX_IO_ITConfig(vo
id);
00220 void                    MFX_IO_Delay(uint3
2_t Delay);
00221 void                    MFX_IO_Write(uint1
6_t Addr, uint8_t Reg, uint8_t Value);
00222 uint8_t                 MFX_IO_Read(uint16
_t Addr, uint8_t Reg);
00223 uint16_t                MFX_IO_ReadMultiple
(uint16_t Addr, uint8_t Reg, uint8_t *Buffer, uint
16_t Length);
00224
00225 /* Link function for TOUCHSCREEN IO function
s */
00226 void                    TS_IO_Init(void);
00227 void                    TS_IO_Write(uint8_
t Addr, uint8_t Reg, uint8_t Value);
00228 uint8_t                 TS_IO_Read(uint8_t
 Addr, uint8_t Reg);
00229 uint16_t                TS_IO_ReadMultiple
(uint8_t Addr, uint8_t Reg, uint8_t *Buffer, uint1
6_t Length);
00230 void                    TS_IO_Delay(uint32
_t Delay);
00231
00232 /* Link function for EEPROM peripheral over
I2C */
```

```
00233 void                    EEPROM_IO_Init(void
);
00234 HAL_StatusTypeDef       EEPROM_IO_WriteData
(uint16_t DevAddress, uint16_t MemAddress, uint8_t
* pBuffer, uint32_t BufferSize);
00235 HAL_StatusTypeDef       EEPROM_IO_ReadData
(uint16_t DevAddress, uint16_t MemAddress, uint8_t
* pBuffer, uint32_t BufferSize);
00236 HAL_StatusTypeDef       EEPROM_IO_IsDevice
Ready(uint16_t DevAddress, uint32_t Trials);
00237
00238 /* Link function for Audio Codec peripheral
*/
00239 void                    AUDIO_IO_Init(void
);
00240 void                    AUDIO_IO_DeInit(vo
id);
00241 void                    AUDIO_IO_Write(uin
t8_t Addr, uint16_t Reg, uint16_t Value);
00242 uint16_t                AUDIO_IO_Read(uint
8_t Addr, uint16_t Reg);
00243 void                    AUDIO_IO_Delay(uin
t32_t delay);
00244
00245 #endif/* HAL_I2C_MODULE_ENABLED */
00246
00247 /**
00248   * @}
00249   */
00250
00251 /** @defgroup STM32L4R9I_EVAL_Exported_Funct
ions Exported Functions
00252   * @{
00253   */
00254
00255 /**
00256   * @brief  Error Handler
```

```
00257   * @note   Defined as a weak function to be
 overwritten by the application.
00258   * @retval None
00259   */
00260 __weak void BSP_ErrorHandler(void)
00261 {
00262   while(1);
00263 }
00264
00265 /**
00266   * @brief  This method returns the STM32L4R
9I EVAL BSP Driver revision
00267   * @retval version : 0xXYZR (8bits for each
 decimal, R for RC)
00268   */
00269 uint32_t BSP_GetVersion(void)
00270 {
00271   return __STM32L4R9I_EVAL_BSP_VERSION;
00272 }
00273
00274 /**
00275   * @brief  Initialize LED GPIO.
00276   * @param  Led: Specifies the Led to be con
figured.
00277   *   This parameter can be one of following
 parameters:
00278   *     @arg LED1
00279   *     @arg LED2
00280   *     @arg LED3
00281   *     @arg LED4
00282   * @retval None
00283   */
00284 void BSP_LED_Init(Led_TypeDef Led)
00285 {
00286   GPIO_InitTypeDef  gpioinitstruct = {0};
00287
00288   /* Enable the GPIO_LED clock */
```

```
00289   LEDx_GPIO_CLK_ENABLE(Led);
00290
00291   /* Configure the GPIO_LED pin */
00292   gpioinitstruct.Pin    = LED_PIN[Led];
00293   gpioinitstruct.Mode   = GPIO_MODE_OUTPUT_P
P;
00294   gpioinitstruct.Pull   = GPIO_PULLUP;
00295   gpioinitstruct.Speed  = GPIO_SPEED_FREQ_VE
RY_HIGH;
00296
00297   HAL_GPIO_Init(LED_PORT[Led], &gpioinitstru
ct);
00298
00299   /* By default, turn off LED */
00300   HAL_GPIO_WritePin(LED_PORT[Led], LED_PIN[L
ed], GPIO_PIN_SET);
00301 }
00302
00303 /**
00304   * @brief  DeInitialize LED GPIO.
00305   * @param  Led: Specifies the Led to be con
figured.
00306   *   This parameter can be one of following
 parameters:
00307   *     @arg LED1
00308   *     @arg LED2
00309   *     @arg LED3
00310   *     @arg LED4
00311   * @retval None
00312   */
00313 void BSP_LED_DeInit(Led_TypeDef Led)
00314 {
00315   /* Enable the GPIO_LED clock */
00316   LEDx_GPIO_CLK_ENABLE(Led);
00317
00318   HAL_GPIO_DeInit(LED_PORT[Led], LED_PIN[Led
]);
```

```
00319 }
00320
00321 /**
00322   * @brief  Turn selected LED On.
00323   * @param  Led: Specifies the Led to be set
 on.
00324   *   This parameter can be one of following
 parameters:
00325   *     @arg LED1
00326   *     @arg LED2
00327   *     @arg LED3
00328   *     @arg LED4
00329   * @retval None
00330   */
00331 void BSP_LED_On(Led_TypeDef Led)
00332 {
00333   HAL_GPIO_WritePin(LED_PORT[Led], LED_PIN[L
ed], GPIO_PIN_RESET);
00334 }
00335
00336 /**
00337   * @brief  Turn selected LED Off.
00338   * @param  Led: Specifies the Led to be set
 off.
00339   *   This parameter can be one of following
 parameters:
00340   *     @arg LED1
00341   *     @arg LED2
00342   *     @arg LED3
00343   *     @arg LED4
00344   * @retval None
00345   */
00346 void BSP_LED_Off(Led_TypeDef Led)
00347 {
00348   HAL_GPIO_WritePin(LED_PORT[Led], LED_PIN[L
ed], GPIO_PIN_SET);
00349 }
```

```
00350
00351 /**
00352   * @brief  Toggle the selected LED.
00353   * @param  Led: Specifies the Led to be tog
gled.
00354   *   This parameter can be one of following
 parameters:
00355   *     @arg LED1
00356   *     @arg LED2
00357   *     @arg LED3
00358   *     @arg LED4
00359   * @retval None
00360   */
00361 void BSP_LED_Toggle(Led_TypeDef Led)
00362 {
00363   HAL_GPIO_TogglePin(LED_PORT[Led], LED_PIN[
Led]);
00364 }
00365
00366 /**
00367   * @brief  Initialize push button GPIO and
EXTI Line.
00368   * @param  Button: Button to be configured.
00369   *   This parameter can be one of the follo
wing values:
00370   *     @arg BUTTON_WAKEUP: Wakeup/Tamper Pu
sh Button
00371   * @param  ButtonMode: Button mode requeste
d.
00372   *   This parameter can be one of the follo
wing values:
00373   *     @arg BUTTON_MODE_GPIO: Button will b
e used as simple IO
00374   *     @arg BUTTON_MODE_EXTI: Button will b
e connected to EXTI line
00375   *                            with interrup
t generation capability
```

```
00376   * @retval None
00377   */
00378 void BSP_PB_Init(Button_TypeDef Button, Butt
onMode_TypeDef ButtonMode)
00379 {
00380   GPIO_InitTypeDef gpioinitstruct = {0};
00381
00382   if (Button == BUTTON_WAKEUP)
00383   {
00384     /* Enable the corresponding Push Button
clock */
00385     BUTTONx_GPIO_CLK_ENABLE(Button);
00386
00387     /* Configure Push Button pin as input */
00388     gpioinitstruct.Pin = BUTTON_PIN[Button];
00389     gpioinitstruct.Pull = GPIO_PULLDOWN;
00390     gpioinitstruct.Speed = GPIO_SPEED_FREQ_V
ERY_HIGH;
00391
00392     if (ButtonMode == BUTTON_MODE_GPIO)
00393     {
00394       /* Configure Button pin as input */
00395       gpioinitstruct.Mode = GPIO_MODE_INPUT;
00396       HAL_GPIO_Init(BUTTON_PORT[Button], &gp
ioinitstruct);
00397     }
00398     else
00399     {
00400       /* Configure Tamper/Wake-up Push Butto
n pin as input with External interrupt, rising edg
e */
00401       gpioinitstruct.Mode = GPIO_MODE_IT_RIS
ING;
00402
00403       HAL_GPIO_Init(BUTTON_PORT[Button], &gp
ioinitstruct);
00404
```

```
00405       /* Enable and set Button EXTI Interrup
t to the lowest priority */
00406       HAL_NVIC_SetPriority((IRQn_Type)(BUTTO
N_IRQn[Button]), 0x0F, 0);
00407       HAL_NVIC_EnableIRQ((IRQn_Type)(BUTTON_
IRQn[Button]));
00408     }
00409   }
00410 }
00411
00412 /**
00413   * @brief  DeInitialize push button.
00414   * @param  Button: Button to be deinitializ
ed.
00415   *   This parameter can be one of the follo
wing values:
00416   *     @arg BUTTON_WAKEUP: Wakeup/Tamper Pu
sh Button
00417   * @retval None
00418   */
00419 void BSP_PB_DeInit(Button_TypeDef Button)
00420 {
00421   if (Button == BUTTON_WAKEUP)
00422   {
00423     /* Enable the corresponding Push Button
clock */
00424     BUTTONx_GPIO_CLK_ENABLE(Button);
00425
00426     HAL_GPIO_DeInit(BUTTON_PORT[Button], BUT
TON_PIN[Button]);
00427
00428     /* Make sure Button EXTI Interrupt is di
sabled */
00429     HAL_NVIC_DisableIRQ((IRQn_Type)(BUTTON_I
RQn[Button]));
00430   }
00431 }
```

```
00432
00433 /**
00434   * @brief  Return the selected button state.

00435   * @param  Button: Button to be checked.
00436   *   This parameter can be one of the follo
wing values:
00437   *     @arg BUTTON_WAKEUP: Wakeup/Tamper Pu
sh Button
00438   * @retval Button state
00439   */
00440 uint32_t BSP_PB_GetState(Button_TypeDef Butt
on)
00441 {
00442   return HAL_GPIO_ReadPin(BUTTON_PORT[Button
], BUTTON_PIN[Button]);
00443 }
00444
00445 /**
00446   * @brief  Initialize all buttons of the jo
ystick in GPIO or EXTI modes.
00447   * @param  JoyMode: Joystick mode.
00448   *    This parameter can be one of the foll
owing values:
00449   *     @arg  JOY_MODE_GPIO: Joystick pins w
ill be used as simple IOs
00450   *     @arg  JOY_MODE_EXTI: Joystick pins w
ill be connected to EXTI line
00451   *                                  with int
errupt generation capability
00452   * @retval HAL_OK: if all initializations a
re OK. Other value if error.
00453   */
00454 uint8_t BSP_JOY_Init(JOYMode_TypeDef JoyMode
)
00455 {
00456   if (BSP_IO_Init() == IO_ERROR)
```

```
00457   {
00458     BSP_ErrorHandler();
00459   }
00460
00461   /* Configure joystick pins in IT mode */
00462   if(JoyMode == JOY_MODE_EXTI)
00463   {
00464     BSP_IO_ConfigPin(JOY_ALL_PINS, IO_MODE_I
T_LOW_LEVEL_PU);
00465
00466     /* Configure IO Expander interrupt */
00467     MFX_IO_ITConfig();
00468   }
00469   else
00470   {
00471     /* GPIO mode configuration */
00472     BSP_IO_ConfigPin(JOY_ALL_PINS,IO_MODE_IN
PUT_PU);
00473   }
00474
00475   return HAL_OK;
00476 }
00477
00478 /**
00479   * @brief  DeInitialize all buttons of the
joystick.
00480   * @retval None
00481   */
00482 void BSP_JOY_DeInit(void)
00483 {
00484   BSP_IO_ConfigPin(JOY_ALL_PINS, IO_MODE_OFF
);
00485 }
00486
00487 /**
00488   * @brief  Return the current joystick stat
us.
```

```
00489   * @retval Code of the joystick key pressed
00490   *          This code can be one of the fol
lowing values:
00491   *            @arg  JOY_NONE
00492   *            @arg  JOY_SEL
00493   *            @arg  JOY_DOWN
00494   *            @arg  JOY_LEFT
00495   *            @arg  JOY_RIGHT
00496   *            @arg  JOY_UP
00497   */
00498 JOYState_TypeDef BSP_JOY_GetState(void)
00499 {
00500   uint32_t tmp = 0;
00501
00502   /* Read the status  of all the joystick pi
ns */
00503   tmp = BSP_IO_ReadPin(JOY_ALL_PINS);
00504
00505   /* Check the pressed joystick button */
00506   if(!(tmp & SEL_JOY_PIN))
00507   {
00508     return(JOYState_TypeDef) JOY_SEL;
00509   }
00510   else if(!(tmp & DOWN_JOY_PIN))
00511   {
00512     return(JOYState_TypeDef) JOY_DOWN;
00513   }
00514   else if(!(tmp & LEFT_JOY_PIN))
00515   {
00516     return(JOYState_TypeDef) JOY_LEFT;
00517   }
00518   else if(!(tmp & RIGHT_JOY_PIN))
00519   {
00520     return(JOYState_TypeDef) JOY_RIGHT;
00521   }
00522   else if(!(tmp & UP_JOY_PIN))
00523   {
```

```
00524     return(JOYState_TypeDef) JOY_UP;
00525   }
00526   else
00527   {
00528     return(JOYState_TypeDef) JOY_NONE;
00529   }
00530 }
00531
00532 #ifdef HAL_UART_MODULE_ENABLED
00533 /**
00534   * @brief  Initialize COM port.
00535   * @param  COM: Specifies the COM port to b
e configured.
00536   *   This parameter can be one of following
 parameters:
00537   *     @arg COM1
00538   * @param  huart: pointer to a UART_HandleT
ypeDef structure that
00539   *   contains the configuration information
 for the specified UART peripheral.
00540   * @retval None
00541   */
00542 void BSP_COM_Init(COM_TypeDef COM, UART_Hand
leTypeDef* huart)
00543 {
00544   GPIO_InitTypeDef gpioinitstruct = {0};
00545
00546   /* Enable GPIO clock */
00547   COMx_TX_GPIO_CLK_ENABLE(COM);
00548   COMx_RX_GPIO_CLK_ENABLE(COM);
00549
00550   /* Enable USART clock */
00551   COMx_CLK_ENABLE(COM);
00552
00553   /* IOSV bit MUST be set to access GPIO por
t G[2:15] */
00554   __HAL_RCC_PWR_CLK_ENABLE();
```

```
00555   HAL_PWREx_EnableVddIO2();
00556
00557   /* Configure USART Tx as alternate functio
n push-pull */
00558   gpioinitstruct.Pin = COM_TX_PIN[COM];
00559   gpioinitstruct.Mode = GPIO_MODE_AF_PP;
00560   gpioinitstruct.Speed = GPIO_SPEED_FREQ_VER
Y_HIGH;
00561   gpioinitstruct.Pull = GPIO_PULLUP;
00562   gpioinitstruct.Alternate = COM_TX_AF[COM];
00563   HAL_GPIO_Init(COM_TX_PORT[COM], &gpioinits
truct);
00564
00565   /* Configure USART Rx as alternate functio
n push-pull */
00566   gpioinitstruct.Pin = COM_RX_PIN[COM];
00567   gpioinitstruct.Alternate = COM_RX_AF[COM];
00568   HAL_GPIO_Init(COM_RX_PORT[COM], &gpioinits
truct);
00569
00570   /* USART configuration */
00571   huart->Instance = COM_USART[COM];
00572   HAL_UART_Init(huart);
00573 }
00574
00575 /**
00576   * @brief  DeInitialize COM port.
00577   * @param  COM: Specifies the COM port to b
e deinitialized.
00578   *   This parameter can be one of following
 parameters:
00579   *     @arg COM1
00580   * @param  huart: pointer to a UART_HandleT
ypeDef structure that
00581   *   contains the configuration information
 for the specified UART peripheral.
00582   * @retval None
```

```
00583   */
00584 void BSP_COM_DeInit(COM_TypeDef COM, UART_Ha
ndleTypeDef* huart)
00585 {
00586   /* USART configuration */
00587   huart->Instance = COM_USART[COM];
00588   HAL_UART_DeInit(huart);
00589 }
00590 #endif /* HAL_UART_MODULE_ENABLED */
00591
00592
00593 #ifdef HAL_ADC_MODULE_ENABLED
00594 /**
00595   * @brief  Init the potentioneter RV2 measu
rement thanks to ADC.
00596   * Limitations : the potentiometer are excl
usive with MFX, Audio codex,
00597   * OctoSPIP1, Debugging connector and MC op
eration. And they are mutually exclusive.
00598   * Please deinit all these components befor
e usage.
00599   * @retval BSP_POTENTIOMETER_OK if no error
00600   */
00601 uint32_t BSP_POTENTIOMETER_Init(void)
00602 {
00603   GPIO_InitTypeDef          GPIO_InitStruct;
00604   ADC_ChannelConfTypeDef    sConfig;
00605
00606   /* ADC clock enable */
00607   ADCx_CLK_ENABLE();
00608   /* ADC an GPIO Periph clock enable */
00609   ADCx_CHANNEL_GPIO_CLK_ENABLE();
00610   /* ADC Periph interface clock configuratio
n */
00611   __HAL_RCC_ADC_CONFIG(RCC_ADCCLKSOURCE_SYSC
LK);
00612
```

```
00613   /* ADC Channel GPIO pin configuration */
00614   GPIO_InitStruct.Pin = ADCx_CHANNEL_PIN;
00615   GPIO_InitStruct.Mode = GPIO_MODE_ANALOG;
00616   GPIO_InitStruct.Pull = GPIO_NOPULL;
00617
00618   HAL_GPIO_Init(ADCx_CHANNEL_GPIO_PORT, &GPI
O_InitStruct);
00619
00620   /*##-1- Configure the ADC peripheral #####
###################################*/
00621   hEvalADC.Instance          = ADCx;
00622
00623   if (HAL_ADC_DeInit(&hEvalADC)) return BSP_
POTENTIOMETER_ERROR;
00624
00625   hEvalADC.Init.ClockPrescaler        = ADC_
CLOCK_SYNC_PCLK_DIV2;      /* Synchronous clock mo
de, input ADC clock not divided */
00626   hEvalADC.Init.Resolution            = ADC_
RESOLUTION_12B;            /* 12-bit resolution fo
r converted data */
00627   hEvalADC.Init.DataAlign             = ADC_
DATAALIGN_RIGHT;           /* Right-alignment for
converted data */
00628   hEvalADC.Init.ScanConvMode          = DISA
BLE;                       /* Sequencer disabled (
ADC conversion on only 1 channel: channel set on r
ank 1) */
00629   hEvalADC.Init.EOCSelection          = ADC_
EOC_SINGLE_CONV;           /* EOC flag picked-up t
o indicate conversion end */
00630   hEvalADC.Init.LowPowerAutoWait      = DISA
BLE;                       /* Auto-delayed convers
ion feature disabled */
00631   hEvalADC.Init.ContinuousConvMode    = DISA
BLE;                       /* Continuous mode disa
bled to have only 1 conversion at each conversion
```

```
trig */
00632   hEvalADC.Init.NbrOfConversion       = 1;
                            /* Parameter discarded
because sequencer is disabled */
00633   hEvalADC.Init.DiscontinuousConvMode = DISA
BLE;                        /* Parameter discarded
because sequencer is disabled */
00634   hEvalADC.Init.NbrOfDiscConversion   = 1;
                            /* Parameter discarded
because sequencer is disabled */
00635   hEvalADC.Init.ExternalTrigConv      = ADC_
SOFTWARE_START;             /* Software start to tr
ig the 1st conversion manually, without external e
vent */
00636   hEvalADC.Init.ExternalTrigConvEdge  = ADC_
EXTERNALTRIGCONVEDGE_NONE; /* Parameter discarded
because software trigger chosen */
00637   hEvalADC.Init.DMAContinuousRequests = DISA
BLE;                        /* DMA one-shot mode se
lected (not applied to this example) */
00638   hEvalADC.Init.Overrun               = ADC_
OVR_DATA_OVERWRITTEN;       /* DR register is overw
ritten with the last conversion result in case of
overrun */
00639   hEvalADC.Init.OversamplingMode      = DISA
BLE;                        /* No oversampling */
00640
00641   if (HAL_ADC_Init(&hEvalADC) != HAL_OK) ret
urn BSP_POTENTIOMETER_ERROR;
00642
00643   /*##-2- Configure ADC regular channel ####
##################################*/
00644   sConfig.Channel      = ADCx_CHANNEL;
          /* Sampled channel number */
00645   sConfig.Rank         = ADC_REGULAR_RANK_1;
          /* Rank of sampled channel number ADCx_C
HANNEL */
```

```
00646   sConfig.SamplingTime = ADC_SAMPLETIME_6CYC
LES_5;    /* Sampling time (number of clock cycles
 unit) */
00647   sConfig.SingleDiff   = ADC_SINGLE_ENDED;
          /* Single-ended input channel */
00648   sConfig.OffsetNumber = ADC_OFFSET_NONE;
          /* No offset subtraction */
00649   sConfig.Offset = 0;
          /* Parameter discarded because offset co
rrection is disabled */
00650
00651
00652   if (HAL_ADC_ConfigChannel(&hEvalADC, &sCon
fig) != HAL_OK) return BSP_POTENTIOMETER_ERROR;
00653
00654   /* Run the ADC calibration in single-ended
 mode */
00655   if (HAL_ADCEx_Calibration_Start(&hEvalADC,
 ADC_SINGLE_ENDED)) return BSP_POTENTIOMETER_ERROR
;
00656
00657
00658   return BSP_POTENTIOMETER_OK;
00659 }
00660
00661 /**
00662   * @brief  DeInitialize the potentioneter R
V2 measurement thanks to ADC.
00663   * Limitations : the potentiometer are excl
usive with MFX, Audio codex,
00664   * OctoSPIP1, Debugging connector and MC op
eration. And they are mutually exclusive.
00665   * Please deinit all these components befor
e usage.
00666   * @retval BSP_POTENTIOMETER_OK if no error
00667   */
00668 uint32_t BSP_POTENTIOMETER_DeInit(void)
```

```
00669 {
00670   if (HAL_ADC_DeInit(&hEvalADC) != HAL_OK) r
eturn BSP_POTENTIOMETER_ERROR;
00671
00672   return BSP_POTENTIOMETER_OK;
00673 }
00674
00675 /**
00676   * @brief  Get the potentioneter RV2 measur
ement thanks to ADC.
00677   * Limitations : the potentiometer are excl
usive with MFX, Audio codex,
00678   * OctoSPIP1, Debugging connector and MC op
eration. And they are mutually exclusive.
00679   * Please deinit all these components befor
e usage.
00680   * @retval value if no error else 0xFFFFFFFF

00681   */
00682 uint32_t BSP_POTENTIOMETER_GetLevel(void)
00683 {
00684   /*##-3- Start the conversion process #####
##################################*/
00685   if (HAL_ADC_Start(&hEvalADC) != HAL_OK) re
turn BSP_POTENTIOMETER_ERROR;
00686
00687   if(HAL_ADC_PollForConversion(&hEvalADC, AD
Cx_POLL_TIMEOUT)== HAL_OK)
00688   {
00689     return (HAL_ADC_GetValue(&hEvalADC));
00690   }
00691   else
00692   {
00693     return  0xFFFFFFFF;
00694   }
00695 }
00696 #endif /* HAL_ADC_MODULE_ENABLED */
```

```
/**
  * @}
  */

/** @defgroup STM32L4R9I_EVAL_BusOperations_Functions Bus Operations Functions
  * @{
  */

/*****************************************************************************
                                BUS OPERATIONS
*****************************************************************************/
#ifdef HAL_I2C_MODULE_ENABLED
/****************************** I2C Routines*******************************/

/**
  * @brief Eval I2Cx MSP Initialization
  * @param hi2c: I2C handle
  * @retval None
  */
static void I2Cx_MspInit(I2C_HandleTypeDef * hi2c)
{
  GPIO_InitTypeDef  gpioinitstruct = {0};
  RCC_PeriphCLKInitTypeDef  RCC_PeriphCLKInitStruct;

  /* IOSV bit MUST be set to access GPIO port G[2:15] */
  __HAL_RCC_PWR_CLK_ENABLE();
  HAL_PWREx_EnableVddIO2();

  if (hi2c->Instance == EVAL_I2Cx)
```

```
00727   {
00728     /*##-1- Configure the Eval I2C clock sou
rce. The clock is derived from the SYSCLK #*/
00729     if(EVAL_RCC_PERIPHCLK_I2C == RCC_PERIPHC
LK_I2C2)
00730     {
00731       RCC_PeriphCLKInitStruct.PeriphClockSel
ection = RCC_PERIPHCLK_I2C2;
00732       RCC_PeriphCLKInitStruct.I2c2ClockSelec
tion   = EVAL_RCC_CLKSOURCE_I2C;
00733     }
00734     else
00735     {
00736       RCC_PeriphCLKInitStruct.PeriphClockSel
ection = RCC_PERIPHCLK_I2C1;
00737       RCC_PeriphCLKInitStruct.I2c1ClockSelec
tion   = EVAL_RCC_CLKSOURCE_I2C;
00738     }
00739     HAL_RCCEx_PeriphCLKConfig(&RCC_PeriphCLK
InitStruct);
00740
00741     /*##-2- Configure the GPIOs ############
####################################*/
00742
00743     /* Enable GPIO clock */
00744     EVAL_I2Cx_SDA_GPIO_CLK_ENABLE();
00745     EVAL_I2Cx_SCL_GPIO_CLK_ENABLE();
00746
00747     /* Configure I2C Rx/Tx as alternate func
tion  */
00748     gpioinitstruct.Pin       = EVAL_I2Cx_SCL
_PIN | EVAL_I2Cx_SDA_PIN;
00749     gpioinitstruct.Mode      = GPIO_MODE_AF_
OD;
00750     gpioinitstruct.Pull      = GPIO_PULLUP;
00751     gpioinitstruct.Speed     = GPIO_SPEED_FR
EQ_MEDIUM; // GPIO_SPEED_FREQ_VERY_HIGH;
```

```
00752     gpioinitstruct.Alternate = EVAL_I2Cx_SCL
_SDA_AF;
00753     HAL_GPIO_Init(EVAL_I2Cx_SCL_GPIO_PORT, &
gpioinitstruct);
00754
00755     /*##-3- Configure the Eval I2Cx peripher
al ######################################*/
00756     /* Enable Eval_I2Cx clock */
00757     EVAL_I2Cx_CLK_ENABLE();
00758
00759     /* Force the I2C Peripheral Clock Reset
*/
00760     EVAL_I2Cx_FORCE_RESET();
00761
00762     /* Release the I2C Peripheral Clock Rese
t */
00763     EVAL_I2Cx_RELEASE_RESET();
00764
00765     /* Enable and set Eval I2Cx Interrupt to
 the highest priority */
00766     HAL_NVIC_SetPriority(EVAL_I2Cx_EV_IRQn,
0x00, 0);
00767     HAL_NVIC_EnableIRQ(EVAL_I2Cx_EV_IRQn);
00768
00769     /* Enable and set Eval I2Cx Interrupt to
 the highest priority */
00770     HAL_NVIC_SetPriority(EVAL_I2Cx_ER_IRQn,
0x00, 0);
00771     HAL_NVIC_EnableIRQ(EVAL_I2Cx_ER_IRQn);
00772   }
00773 }
00774
00775 /**
00776   * @brief Eval I2Cx Bus initialization
00777   * @retval None
00778   */
00779 static void I2Cx_Init(uint32_t user)
```

```
00780 {
00781   if(HAL_I2C_GetState(&hi2c_eval) == HAL_I2C
_STATE_RESET)
00782   {
00783 #if defined(BSP_USE_CMSIS_OS)
00784     /* Create semaphore to prevent multiple
I2C access */
00785     osSemaphoreDef(BSP_I2C_SEM);
00786     BspI2cSemaphore = osSemaphoreCreate(osSe
maphore(BSP_I2C_SEM), 1);
00787 #endif
00788
00789     hi2c_eval.Instance              = EVAL_I
2Cx;
00790     hi2c_eval.Init.Timing           = EVAL_I
2Cx_TIMING;
00791     hi2c_eval.Init.OwnAddress1      = 0;
00792     hi2c_eval.Init.AddressingMode   = I2C_AD
DRESSINGMODE_7BIT;
00793     hi2c_eval.Init.DualAddressMode  = I2C_DU
ALADDRESS_DISABLE;
00794     hi2c_eval.Init.OwnAddress2      = 0;
00795     hi2c_eval.Init.GeneralCallMode  = I2C_GE
NERALCALL_DISABLE;
00796     hi2c_eval.Init.NoStretchMode    = I2C_NO
STRETCH_DISABLE;
00797
00798     HAL_I2C_DeInit(&hi2c_eval);
00799     /* Init the I2C */
00800     I2Cx_MspInit(&hi2c_eval);
00801     HAL_I2C_Init(&hi2c_eval);
00802   }
00803   /* Update BSP I2C users list */
00804   v_bspI2cUsers |= user;
00805 }
00806
00807 /**
```

```
00808   * @brief Eval I2Cx MSP Deinitialization
00809   * @param hi2c: I2C handle
00810   * @retval None
00811   */
00812 static void I2Cx_MspDeInit(I2C_HandleTypeDef
 *hi2c)
00813 {
00814 }
00815
00816 /**
00817   * @brief Eval I2Cx Bus deinitialization
00818   * @retval None
00819   */
00820 static void I2Cx_DeInit(uint32_t user)
00821 {
00822   /* Update BSP I2C users list */
00823   v_bspI2cUsers &= ~(user);
00824
00825   if((HAL_I2C_GetState(&hi2c_eval) != HAL_I2
C_STATE_RESET) && (v_bspI2cUsers == BSP_I2C_NO_USER
))
00826   {
00827     /* Deinit the I2C */
00828     HAL_I2C_DeInit(&hi2c_eval);
00829     I2Cx_MspDeInit(&hi2c_eval);
00830
00831 #if defined(BSP_USE_CMSIS_OS)
00832     /* Delete semaphore to prevent multiple
I2C access */
00833     osSemaphoreDelete(BspI2cSemaphore);
00834 #endif
00835   }
00836 }
00837
00838 /**
00839   * @brief  Write a value in a register of t
he device through BUS.
```

```
00840   * @param  Addr: Device address on BUS Bus.
00841   * @param  Reg: The target register address
 to write
00842   * @param  RegSize: The target register siz
e (can be 8BIT or 16BIT)
00843   * @param  pBuffer: The target register val
ue to be written
00844   * @param  Length: buffer size to be written

00845   * @retval None
00846   */
00847 static HAL_StatusTypeDef I2Cx_WriteBuffer(ui
nt16_t Addr, uint16_t Reg, uint16_t RegSize, uint8
_t *pBuffer, uint16_t Length)
00848 {
00849   HAL_StatusTypeDef status = HAL_OK;
00850
00851 #if defined(BSP_USE_CMSIS_OS)
00852   /* Get semaphore to prevent multiple I2C a
ccess */
00853   osSemaphoreWait(BspI2cSemaphore, osWaitFor
ever);
00854 #endif
00855
00856   status = HAL_I2C_Mem_Write(&hi2c_eval, Add
r, (uint16_t)Reg, RegSize, pBuffer, Length, hi2c_e
valTimeout);
00857
00858 #if defined(BSP_USE_CMSIS_OS)
00859   /* Release semaphore to prevent multiple I
2C access */
00860   osSemaphoreRelease(BspI2cSemaphore);
00861 #endif
00862
00863 /* Check the communication status */
00864   if(status != HAL_OK)
00865   {
```

```
00866     /* Re-Initiaize the BUS */
00867     I2Cx_Error();
00868   }
00869   return status;
00870 }
00871
00872 /**
00873   * @brief  Write a value in a register of t
he device through BUS.
00874   * @param  Addr: Device address on BUS Bus.
00875   * @param  Reg: The target register address
 to write
00876   * @param  RegSize: The target register siz
e (can be 8BIT or 16BIT)
00877   * @param  Value: The target register value
 to be written
00878   * @retval None
00879   */
00880 static void I2Cx_WriteData(uint16_t Addr, ui
nt16_t Reg, uint16_t RegSize, uint8_t Value)
00881 {
00882   HAL_StatusTypeDef status = HAL_OK;
00883
00884 #if defined(BSP_USE_CMSIS_OS)
00885   /* Get semaphore to prevent multiple I2C a
ccess */
00886   osSemaphoreWait(BspI2cSemaphore, osWaitFor
ever);
00887 #endif
00888
00889   status = HAL_I2C_Mem_Write(&hi2c_eval, Add
r, (uint16_t)Reg, RegSize, &Value, 1, hi2c_evalTim
eout);
00890
00891 #if defined(BSP_USE_CMSIS_OS)
00892   /* Release semaphore to prevent multiple I
2C access */
```

```
00893   osSemaphoreRelease(BspI2cSemaphore);
00894 #endif
00895
00896   /* Check the communication status */
00897   if(status != HAL_OK)
00898   {
00899     /* Re-Initiaize the BUS */
00900     I2Cx_Error();
00901   }
00902 }
00903
00904 /**
00905   * @brief  Write multiple data value in a r
egister of the device through BUS.
00906   * @param  Addr: Device address on BUS Bus.
00907   * @param  Reg: The target register address
 to write
00908   * @param  MemAddress: Memory address
00909   * @param  Buffer: The target register valu
e to be written
00910   * @param  Length: buffer size to be written

00911   * @retval None
00912   */
00913 static void I2Cx_WriteMultiple(uint16_t Addr
, uint16_t Reg, uint16_t MemAddress, uint8_t *Buff
er, uint16_t Length)
00914   {
00915   HAL_StatusTypeDef status = HAL_OK;
00916
00917 #if defined(BSP_USE_CMSIS_OS)
00918   /* Get semaphore to prevent multiple I2C a
ccess */
00919   osSemaphoreWait(BspI2cSemaphore, osWaitFor
ever);
00920 #endif
00921
```

```
00922   status = HAL_I2C_Mem_Write(&hi2c_eval, Add
r, (uint16_t)Reg, MemAddress, Buffer, Length, hi2c
_evalTimeout);
00923
00924 #if defined(BSP_USE_CMSIS_OS)
00925   /* Release semaphore to prevent multiple I
2C access */
00926   osSemaphoreRelease(BspI2cSemaphore);
00927 #endif
00928
00929   /* Check the communication status */
00930   if(status != HAL_OK)
00931   {
00932     /* Re-Initiaize the BUS */
00933     I2Cx_Error();
00934   }
00935 }
00936
00937 /**
00938   * @brief  Read a register of the device th
rough BUS
00939   * @param  Addr: Device address on BUS
00940   * @param  Reg: The target register address
 to read
00941   * @param  RegSize: The target register siz
e (can be 8BIT or 16BIT)
00942   * @retval read register value
00943   */
00944 static uint8_t I2Cx_ReadData(uint16_t Addr,
uint16_t Reg, uint16_t RegSize)
00945 {
00946   HAL_StatusTypeDef status = HAL_OK;
00947   uint8_t value = 0x0;
00948
00949 #if defined(BSP_USE_CMSIS_OS)
00950   /* Get semaphore to prevent multiple I2C a
ccess */
```

```
00951   osSemaphoreWait(BspI2cSemaphore, osWaitFor
ever);
00952 #endif
00953
00954   status = HAL_I2C_Mem_Read(&hi2c_eval, Addr
, Reg, RegSize, &value, 1, hi2c_evalTimeout);
00955
00956 #if defined(BSP_USE_CMSIS_OS)
00957   /* Release semaphore to prevent multiple I
2C access */
00958   osSemaphoreRelease(BspI2cSemaphore);
00959 #endif
00960
00961   /* Check the communication status */
00962   if(status != HAL_OK)
00963   {
00964     /* Execute user timeout callback */
00965     I2Cx_Error();
00966
00967   }
00968   return value;
00969 }
00970
00971 /**
00972   * @brief  Read multiple data from a regist
er of the device through BUS
00973   * @param  Addr: Device address on BUS
00974   * @param  Reg: The target register address
 to read
00975   * @param  MemAddress: Memory address
00976   * @param  Buffer: Pointer to data buffer
00977   * @param  Length: Length of the data
00978   * @retval Number of read data
00979   */
00980 static uint8_t I2Cx_ReadMultiple(uint16_t Ad
dr, uint16_t Reg, uint16_t MemAddress, uint8_t *Bu
ffer, uint16_t Length)
```

```
00981 {
00982   HAL_StatusTypeDef status = HAL_OK;
00983   uint8_t value = 0x0;
00984 
00985 #if defined(BSP_USE_CMSIS_OS)
00986   /* Get semaphore to prevent multiple I2C a
ccess */
00987   osSemaphoreWait(BspI2cSemaphore, osWaitFor
ever);
00988 #endif
00989 
00990   status = HAL_I2C_Mem_Read(&hi2c_eval, Addr
, Reg, MemAddress, Buffer, Length, hi2c_evalTimeou
t);
00991 
00992 #if defined(BSP_USE_CMSIS_OS)
00993   /* Release semaphore to prevent multiple I
2C access */
00994   osSemaphoreRelease(BspI2cSemaphore);
00995 #endif
00996 
00997   /* Check the communication status */
00998   if(status != HAL_OK)
00999   {
01000     /* Execute user timeout callback */
01001     I2Cx_Error();
01002 
01003   }
01004   return value;
01005 }
01006 
01007 /**
01008   * @brief  Read multiple data on the BUS.
01009   * @param  Addr: I2C Address
01010   * @param  Reg: Reg Address
01011   * @param  RegSize : The target register si
ze (can be 8BIT or 16BIT)
```

```
01012   * @param  pBuffer: pointer to read data buffer
01013   * @param  Length: length of the data
01014   * @retval 0 if no problems to read multiple data
01015   */
01016 static HAL_StatusTypeDef I2Cx_ReadBuffer(uint16_t Addr, uint16_t Reg, uint16_t RegSize, uint8_t *pBuffer, uint16_t Length)
01017 {
01018   HAL_StatusTypeDef status = HAL_OK;
01019
01020 #if defined(BSP_USE_CMSIS_OS)
01021   /* Get semaphore to prevent multiple I2C access */
01022   osSemaphoreWait(BspI2cSemaphore, osWaitForever);
01023 #endif
01024
01025   status = HAL_I2C_Mem_Read(&hi2c_eval, Addr, (uint16_t)Reg, RegSize, pBuffer, Length, hi2c_evalTimeout);
01026
01027 #if defined(BSP_USE_CMSIS_OS)
01028   /* Release semaphore to prevent multiple I2C access */
01029   osSemaphoreRelease(BspI2cSemaphore);
01030 #endif
01031
01032   /* Check the communication status */
01033   if(status != HAL_OK)
01034   {
01035     /* Re-Initiaize the BUS */
01036     I2Cx_Error();
01037   }
01038   return status;
01039 }
```

```
01040
01041 /**
01042 * @brief  Checks if target device is ready f
or communication.
01043 * @note   This function is used with Memory
devices
01044 * @param  DevAddress: Target device address
01045 * @param  Trials: Number of trials
01046 * @retval HAL status
01047 */
01048 static HAL_StatusTypeDef I2Cx_IsDeviceReady(
uint16_t DevAddress, uint32_t Trials)
01049 {
01050   HAL_StatusTypeDef status = HAL_OK;
01051
01052 #if defined(BSP_USE_CMSIS_OS)
01053   /* Get semaphore to prevent multiple I2C a
ccess */
01054   osSemaphoreWait(BspI2cSemaphore, osWaitFor
ever);
01055 #endif
01056
01057   status = HAL_I2C_IsDeviceReady(&hi2c_eval,
 DevAddress, Trials, hi2c_evalTimeout);
01058
01059 #if defined(BSP_USE_CMSIS_OS)
01060   /* Release semaphore to prevent multiple I
2C access */
01061   osSemaphoreRelease(BspI2cSemaphore);
01062 #endif
01063
01064   return status;
01065 }
01066
01067
01068 /**
01069   * @brief Eval I2Cx error treatment function
```

```
  * @retval None
  */
static void I2Cx_Error (void)
{
  uint32_t tmpI2cUsers;

  BSP_ErrorHandler();

  /* De-initialize the I2C communication BUS */
  tmpI2cUsers = v_bspI2cUsers;
  I2Cx_DeInit(BSP_I2C_ALL_USERS);

  /* Re- Initiaize the I2C communication BUS */
  I2Cx_Init(tmpI2cUsers);
}

/*****************************************************************************
                                LINK OPERATIONS
*****************************************************************************/

/****************************** LINK MFX_IO ******************************/

/**
  * @brief  Initialize MFX_IO low level.
  * @retval None
  */
void MFX_IO_Init(void)
{
  I2Cx_Init(BSP_I2C_MFX_USER);
}

```

```
01101 /**
01102   * @brief  Configure MFX_IO low level Inter
rupt.
01103   * @retval None
01104   */
01105 void MFX_IO_ITConfig(void)
01106 {
01107   GPIO_InitTypeDef gpioinitstruct = {0};
01108
01109   /* Enable the GPIO EXTI clock */
01110   MFX_INT_GPIO_CLK_ENABLE();
01111
01112   /* Configure MFX Interrupt GPIO */
01113   gpioinitstruct.Pin   = MFX_INT_PIN;
01114   gpioinitstruct.Pull  = GPIO_NOPULL;
01115   gpioinitstruct.Speed = GPIO_SPEED_FREQ_LOW
;
01116   gpioinitstruct.Mode  = GPIO_MODE_IT_RISING
;
01117   HAL_GPIO_Init(MFX_INT_GPIO_PORT, &gpioinit
struct);
01118
01119   /* Enable and set GPIO EXTI Interrupt to t
he lowest priority */
01120   HAL_NVIC_SetPriority((IRQn_Type)(MFX_INT_E
XTI_IRQn), 0x0F, 0x0F);
01121   HAL_NVIC_EnableIRQ(MFX_INT_EXTI_IRQn);
01122 }
01123
01124 /**
01125   * @brief  MFX_IO writes single data.
01126   * @param  Addr: I2C address
01127   * @param  Reg: Reg address
01128   * @param  Value: Data to be written
01129   * @retval None
01130   */
01131 void MFX_IO_Write(uint16_t Addr, uint8_t Reg
```

```
, uint8_t Value)
01132 {
01133   I2Cx_WriteData(Addr, Reg, I2C_MEMADD_SIZE_
8BIT, Value);
01134 }
01135
01136 /**
01137   * @brief  MFX_IO reads single data.
01138   * @param  Addr: I2C address
01139   * @param  Reg: Reg address
01140   * @retval Read data
01141   */
01142 uint8_t MFX_IO_Read(uint16_t Addr, uint8_t R
eg)
01143 {
01144   return I2Cx_ReadData(Addr, Reg, I2C_MEMADD
_SIZE_8BIT);
01145 }
01146
01147 /**
01148   * @brief  MFX_IO reads multiple data.
01149   * @param  Addr: I2C address
01150   * @param  Reg: Reg address
01151   * @param  Buffer: Pointer to data buffer
01152   * @param  Length: Length of the data
01153   */
01154 uint16_t MFX_IO_ReadMultiple(uint16_t Addr,
uint8_t Reg, uint8_t *Buffer, uint16_t Length)
01155 {
01156   return I2Cx_ReadBuffer(Addr, Reg, I2C_MEMA
DD_SIZE_8BIT, Buffer, Length);
01157 }
01158
01159 /**
01160   * @brief  MFX_IO delay.
01161   * @param  Delay: Delay in ms
01162   * @retval None
```

```
01163   */
01164 void MFX_IO_Delay(uint32_t Delay)
01165 {
01166   HAL_Delay(Delay);
01167 }
01168
01169 /**
01170   * @brief  MFX_IO wakeup.
01171   * @retval None
01172   */
01173 void MFX_IO_Wakeup(void)
01174 {
01175   /* Set Wakeup pin to high to wakeup MFX co
mponent from standby mode */
01176   HAL_GPIO_WritePin(MFX_WAKEUP_GPIO_PORT, MF
X_WAKEUP_PIN, GPIO_PIN_SET);
01177
01178   /* Wait */
01179   HAL_Delay(1);
01180
01181   /* Set gpio pin basck to low */
01182   HAL_GPIO_WritePin(MFX_WAKEUP_GPIO_PORT, MF
X_WAKEUP_PIN, GPIO_PIN_RESET);
01183 }
01184
01185 /**
01186   * @brief  MFX_IO enable wakeup pin.
01187   * @retval None
01188   */
01189 void MFX_IO_EnableWakeupPin(void)
01190 {
01191   GPIO_InitTypeDef  GPIO_InitStruct;
01192
01193   /* Enable wakeup gpio clock */
01194   MFX_WAKEUP_GPIO_CLK_ENABLE();
01195
01196   /* MFX wakeup pin configuration */
```

```
01197   GPIO_InitStruct.Pin   = MFX_WAKEUP_PIN;
01198   GPIO_InitStruct.Mode  = GPIO_MODE_OUTPUT_P
P;
01199   GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HI
GH;
01200   GPIO_InitStruct.Pull  = GPIO_NOPULL;
01201   HAL_GPIO_Init(MFX_WAKEUP_GPIO_PORT, &GPIO_
InitStruct);
01202 }
01203
01204 /**
01205   * @brief  DeInitializes MFX_IO low level.
01206   * @retval None
01207   */
01208 void MFX_IO_DeInit(void)
01209 {
01210   GPIO_InitTypeDef  GPIO_InitStruct;
01211
01212   /* Enable wakeup gpio clock */
01213   MFX_WAKEUP_GPIO_CLK_ENABLE();
01214
01215   /* MFX wakeup pin configuration */
01216   GPIO_InitStruct.Pin   = MFX_WAKEUP_PIN;
01217   GPIO_InitStruct.Mode  = GPIO_MODE_OUTPUT_P
P;
01218   GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LO
W;
01219   GPIO_InitStruct.Pull  = GPIO_PULLDOWN;
01220   HAL_GPIO_Init(MFX_WAKEUP_GPIO_PORT, &GPIO_
InitStruct);
01221
01222   /* DeInit interrupt pin : disable IRQ befo
re to avoid spurious interrupt */
01223   HAL_NVIC_DisableIRQ((IRQn_Type)(MFX_INT_EX
TI_IRQn));
01224   MFX_INT_GPIO_CLK_ENABLE();
01225   HAL_GPIO_DeInit(MFX_INT_GPIO_PORT, MFX_INT
```

```
_PIN);
01226
01227   I2Cx_DeInit(BSP_I2C_MFX_USER);
01228 }
01229
01230 /******************************** LINK TOUC
HSCREEN *********************************/
01231
01232 /**
01233   * @brief  Initializes Touchscreen low leve
l.
01234   * @retval None
01235   */
01236 void TS_IO_Init(void)
01237 {
01238   I2Cx_Init(BSP_I2C_TS_USER);
01239 }
01240
01241 /**
01242   * @brief  Writes a single data.
01243   * @param  Addr: I2C address
01244   * @param  Reg: Reg address
01245   * @param  Value: Data to be written
01246   * @retval None
01247   */
01248 void TS_IO_Write(uint8_t Addr, uint8_t Reg,
uint8_t Value)
01249 {
01250   I2Cx_WriteMultiple(Addr, (uint16_t)Reg, I2
C_MEMADD_SIZE_8BIT,(uint8_t*)&Value, 1);
01251 }
01252
01253 /**
01254   * @brief  Reads a single data.
01255   * @param  Addr: I2C address
01256   * @param  Reg: Reg address
01257   * @retval Data to be read
```

```
01258   */
01259 uint8_t TS_IO_Read(uint8_t Addr, uint8_t Reg
)
01260 {
01261   uint8_t read_value = 0;
01262
01263   I2Cx_ReadMultiple(Addr, Reg, I2C_MEMADD_SI
ZE_8BIT, (uint8_t*)&read_value, 1);
01264
01265   return read_value;
01266 }
01267
01268 /**
01269   * @brief  Reads multiple data with I2C com
munication
01270   *         channel from TouchScreen.
01271   * @param  Addr: I2C address
01272   * @param  Reg: Register address
01273   * @param  Buffer: Pointer to data buffer
01274   * @param  Length: Length of the data
01275   * @retval Number of read data
01276   */
01277 uint16_t TS_IO_ReadMultiple(uint8_t Addr, ui
nt8_t Reg, uint8_t *Buffer, uint16_t Length)
01278 {
01279  return I2Cx_ReadMultiple(Addr, Reg, I2C_MEM
ADD_SIZE_8BIT, Buffer, Length);
01280 }
01281
01282 /**
01283   * @brief  TS delay
01284   * @param  Delay: Delay in ms
01285   * @retval None
01286   */
01287 void TS_IO_Delay(uint32_t Delay)
01288 {
01289   HAL_Delay(Delay);
```

```
01290 }
01291
01292
01293 /********************************** LINK AUDI
O ***********************************/
01294
01295 /**
01296   * @brief  Initialize Audio low level.
01297   * @retval None
01298   */
01299 void AUDIO_IO_Init(void)
01300 {
01301   I2Cx_Init(BSP_I2C_AUDIO_USER);
01302 }
01303
01304 /**
01305   * @brief  Deinitialize Audio low level.
01306   * @retval None
01307   */
01308 void AUDIO_IO_DeInit(void)
01309 {
01310   I2Cx_DeInit(BSP_I2C_AUDIO_USER);
01311 }
01312
01313 /**
01314   * @brief  Writes a single data.
01315   * @param  Addr: I2C address
01316   * @param  Reg: Reg address
01317   * @param  Value: Data to be written
01318   * @retval None
01319   */
01320 void AUDIO_IO_Write(uint8_t Addr, uint16_t R
eg, uint16_t Value)
01321 {
01322   uint16_t tmp = Value;
01323
01324   Value = ((uint16_t)(tmp >> 8) & 0x00FF);
```

```
01325   Value |= ((uint16_t)(tmp << 8)& 0xFF00);
01326
01327   I2Cx_WriteBuffer(Addr, Reg, I2C_MEMADD_SIZ
E_16BIT,(uint8_t*)&Value, 2);
01328 }
01329
01330 /**
01331   * @brief  Reads a single data.
01332   * @param  Addr: I2C address
01333   * @param  Reg: Reg address
01334   * @retval Data to be read
01335   */
01336 uint16_t AUDIO_IO_Read(uint8_t Addr, uint16_
t Reg)
01337 {
01338   uint16_t Read_Value = 0, tmp = 0;
01339
01340   I2Cx_ReadBuffer(Addr, Reg, I2C_MEMADD_SIZE
_16BIT, (uint8_t*)&Read_Value, 2);
01341
01342   tmp = ((uint16_t)(Read_Value >> 8) & 0x00F
F);
01343   tmp |= ((uint16_t)(Read_Value << 8)& 0xFF0
0);
01344
01345   Read_Value = tmp;
01346
01347   return Read_Value;
01348 }
01349
01350 /**
01351   * @brief  AUDIO Codec delay
01352   * @param  Delay: Delay in ms
01353   * @retval None
01354   */
01355 void AUDIO_IO_Delay(uint32_t Delay)
01356 {
```

```
01357   HAL_Delay(Delay);
01358 }
01359
01360 /********************************** LINK I2C
EEPROM ***************************/
01361 /**
01362   * @brief  Initialize peripherals used by t
he I2C EEPROM driver.
01363   * @retval None
01364   */
01365 void EEPROM_IO_Init(void)
01366 {
01367   I2Cx_Init(BSP_I2C_EEPROM_USER);
01368 }
01369
01370 /**
01371   * @brief  Write data to I2C EEPROM driver
01372   * @param  DevAddress: Target device address

01373   * @param  MemAddress: Internal memory addr
ess
01374   * @param  pBuffer: Pointer to data buffer
01375   * @param  BufferSize: Amount of data to be
 sent
01376   * @retval HAL status
01377   */
01378 HAL_StatusTypeDef EEPROM_IO_WriteData(uint16
_t DevAddress, uint16_t MemAddress, uint8_t* pBuff
er, uint32_t BufferSize)
01379 {
01380   return (I2Cx_WriteBuffer(DevAddress, MemAd
dress, I2C_MEMADD_SIZE_16BIT, pBuffer, BufferSize)
);
01381 }
01382
01383 /**
01384   * @brief  Read data from I2C EEPROM driver
```

```
01385   * @param  DevAddress: Target device address
01386   * @param  MemAddress: Internal memory addr
ess
01387   * @param  pBuffer: Pointer to data buffer
01388   * @param  BufferSize: Amount of data to be
 read
01389   * @retval HAL status
01390   */
01391 HAL_StatusTypeDef EEPROM_IO_ReadData(uint16_
t DevAddress, uint16_t MemAddress, uint8_t* pBuffe
r, uint32_t BufferSize)
01392 {
01393   return (I2Cx_ReadBuffer(DevAddress, MemAdd
ress, I2C_MEMADD_SIZE_16BIT, pBuffer, BufferSize))
;
01394 }
01395
01396 /**
01397 * @brief  Checks if target device is ready f
or communication.
01398 * @note   This function is used with Memory
devices
01399 * @param  DevAddress: Target device address
01400 * @param  Trials: Number of trials
01401 * @retval HAL status
01402 */
01403 HAL_StatusTypeDef EEPROM_IO_IsDeviceReady(ui
nt16_t DevAddress, uint32_t Trials)
01404 {
01405   return (I2Cx_IsDeviceReady(DevAddress, Tri
als));
01406 }
01407
01408 #endif /*HAL_I2C_MODULE_ENABLED*/
01409
01410
```

```
01411 /**
01412   * @}
01413   */
01414
01415 /**
01416   * @}
01417   */
01418
01419 /**
01420   * @}
01421   */
01422
01423 /**
01424   * @}
01425   */
01426
01427
01428 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_audio.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_audio.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_audio.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_AUDIO_H
00039 #define __STM32L4R9I_EVAL_AUDIO_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-----------------------------------*/
00046 #if defined(BSP_AUDIO_USE_RTOS)
00047 #include "k_mem.h"
00048 #else
00049 #include <stdlib.h>
00050 #endif
00051 /* Include audio component Driver */
00052 #include "../Components/wm8994/wm8994.h"
00053 #include "stm32l4r9i_eval.h"
00054
00055 /** @addtogroup BSP
00056   * @{
00057   */
00058
00059 /** @addtogroup STM32L4R9I_EVAL
00060   * @{
00061   */
00062
00063 /** @addtogroup STM32L4R9I_EVAL_AUDIO
00064   * @{
00065   */
00066
00067 /** @defgroup STM32L4R9I_EVAL_AUDIO_Exported
_Types Exported Types
```

```
00068   * @{
00069   */
00070 typedef void (*Audio_CallbackTypeDef)(void);
00071
00072 /**
00073   * @}
00074   */
00075
00076 /** @defgroup STM32L4R9I_EVAL_AUDIO_Exported
_Constants Exported Constants
00077   * @{
00078   */
00079 /** @defgroup BSP_Audio_Out_Option BSP Audio
 Out Option
00080   * @{
00081   */
00082 #define BSP_AUDIO_OUT_CIRCULARMODE      ((ui
nt32_t)0x00000001) /* BUFFER CIRCULAR MODE */
00083 #define BSP_AUDIO_OUT_NORMALMODE        ((ui
nt32_t)0x00000002) /* BUFFER NORMAL MODE   */
00084 #define BSP_AUDIO_OUT_STEREOMODE        ((ui
nt32_t)0x00000004) /* STEREO MODE          */
00085 #define BSP_AUDIO_OUT_MONOMODE          ((ui
nt32_t)0x00000008) /* MONO MODE            */
00086 /**
00087   * @}
00088   */
00089
00090 /** @defgroup BSP_Audio_Sample_Rate BSP Audi
o Sample Rate
00091   * @{
00092   */
00093 #define BSP_AUDIO_FREQUENCY_96K         SAI_
AUDIO_FREQUENCY_96K
00094 #define BSP_AUDIO_FREQUENCY_48K         SAI_
AUDIO_FREQUENCY_48K
00095 #define BSP_AUDIO_FREQUENCY_44K         SAI_
```

```
AUDIO_FREQUENCY_44K
00096 #define BSP_AUDIO_FREQUENCY_32K          SAI_
AUDIO_FREQUENCY_32K
00097 #define BSP_AUDIO_FREQUENCY_22K          SAI_
AUDIO_FREQUENCY_22K
00098 #define BSP_AUDIO_FREQUENCY_16K          SAI_
AUDIO_FREQUENCY_16K
00099 #define BSP_AUDIO_FREQUENCY_11K          SAI_
AUDIO_FREQUENCY_11K
00100 #define BSP_AUDIO_FREQUENCY_8K           SAI_
AUDIO_FREQUENCY_8K
00101 /**
00102   * @}
00103   */
00104 /*-----------------------------------------
-----------------------------------
00105                              USER SAI defines p
arameters
00106  -----------------------------------------
-----------------------------------*/
00107 /* SAI peripheral configuration defines */
00108 #define AUDIO_SAIx                       SAI1_Bl
ock_A
00109 #define AUDIO_SAIx_CLK_ENABLE()          __HAL_R
CC_SAI1_CLK_ENABLE()
00110 #define AUDIO_SAIx_CLK_DISABLE()         __HAL_R
CC_SAI1_CLK_DISABLE()
00111 #define AUDIO_SAIx_MCK_SCK_SD_FS_AF      GPIO_AF
13_SAI1
00112
00113 #define AUDIO_SAIx_MCK_CLK_ENABLE()      __HAL_R
CC_GPIOB_CLK_ENABLE()
00114 #define AUDIO_SAIx_MCK_CLK_DISABLE()     __HAL_R
CC_GPIOB_CLK_DISABLE()
00115 #define AUDIO_SAIx_MCK_PIN               GPIO_PI
N_8
00116 #define AUDIO_SAIx_MCK_GPIO_PORT         GPIOB
```

```
00117 #define AUDIO_SAIx_FS_CLK_ENABLE()   __HAL_R
CC_GPIOB_CLK_ENABLE()
00118 #define AUDIO_SAIx_FS_CLK_DISABLE()  __HAL_R
CC_GPIOB_CLK_DISABLE()
00119 #define AUDIO_SAIx_FS_PIN            GPIO_PI
N_9
00120 #define AUDIO_SAIx_FS_GPIO_PORT      GPIOB
00121 #define AUDIO_SAIx_SCK_CLK_ENABLE()  __HAL_R
CC_GPIOA_CLK_ENABLE()
00122 #define AUDIO_SAIx_SCK_CLK_DISABLE() __HAL_R
CC_GPIOA_CLK_DISABLE()
00123 #define AUDIO_SAIx_SCK_PIN           GPIO_PI
N_8
00124 #define AUDIO_SAIx_SCK_GPIO_PORT     GPIOA
00125 #define AUDIO_SAIx_SD_CLK_ENABLE()   __HAL_R
CC_GPIOC_CLK_ENABLE()
00126 #define AUDIO_SAIx_SD_CLK_DISABLE()  __HAL_R
CC_GPIOC_CLK_DISABLE()
00127 #define AUDIO_SAIx_SD_PIN            GPIO_PI
N_1
00128 #define AUDIO_SAIx_SD_GPIO_PORT      GPIOC
00129
00130 /* SAI DMA Channel definitions */
00131 #define AUDIO_SAIx_DMAx_CLK_ENABLE()
 __HAL_RCC_DMA2_CLK_ENABLE()
00132 #define AUDIO_SAIx_DMAx_CLK_DISABLE()
 __HAL_RCC_DMA2_CLK_DISABLE()
00133 #define AUDIO_SAIx_DMAx_REQUEST
 DMA_REQUEST_SAI1_A
00134 #define AUDIO_SAIx_DMAx_CHANNEL
 DMA2_Channel2
00135 #define AUDIO_SAIx_DMAx_IRQ
 DMA2_Channel2_IRQn
00136 #define AUDIO_SAIx_DMAx_PERIPH_DATA_SIZE
 DMA_PDATAALIGN_HALFWORD
00137 #define AUDIO_SAIx_DMAx_MEM_DATA_SIZE
 DMA_MDATAALIGN_HALFWORD
```

```
00138 #define DMA_MAX_SZE
 (uint32_t)0xFFFF
00139
00140 #define AUDIO_SAIx_DMAx_IRQHandler
 DMA2_Channel2_IRQHandler
00141
00142 /* Select the interrupt preemption priority
for the DMA interrupt */
00143 #define AUDIO_OUT_IRQ_PREPRIO           5
/* Select the preemption priority level(0 is the h
ighest) */
00144
00145 /* Disable SAIx PLL */
00146 #define AUDIO_SAIx_PLL_DISABLE()
 HAL_RCCEx_DisablePLLSAI1()
00147
00148
00149 /*-----------------------------------------
-----------------------------------
00150                            AUDIO IN CONFIGURATI
ON
00151 -----------------------------------------
-----------------------------------*/
00152 /* DFSDM Configuration defines */
00153 #define AUDIO_DFSDMx_LEFT_CHANNEL
            DFSDM1_Channel4
00154 #define AUDIO_DFSDMx_RIGHT_CHANNEL
            DFSDM1_Channel3
00155 #define AUDIO_DFSDMx_LEFT_FILTER
            DFSDM1_Filter0
00156 #define AUDIO_DFSDMx_RIGHT_FILTER
            DFSDM1_Filter1
00157 #define AUDIO_DFSDMx_CLK_ENABLE()
            __HAL_RCC_DFSDM1_CLK_ENABLE()
00158 #define AUDIO_DFSDMx_CLK_DISABLE()
            __HAL_RCC_DFSDM1_CLK_DISABLE()
00159 #define AUDIO_DFSDMx_CKOUT_DATIN_AF
```

```
            GPIO_AF6_DFSDM1
00160 #define AUDIO_DFSDMx_CKOUT_CLK_ENABLE()
            __HAL_RCC_GPIOF_CLK_ENABLE()
00161 #define AUDIO_DFSDMx_CKOUT_CLK_DISABLE()
            __HAL_RCC_GPIOF_CLK_DISABLE()
00162 #define AUDIO_DFSDMx_CKOUT_PIN
            GPIO_PIN_10
00163 #define AUDIO_DFSDMx_CKOUT_GPIO_PORT
            GPIOF
00164 #define AUDIO_DFSDMx_DATIN_CLK_ENABLE()
            __HAL_RCC_GPIOC_CLK_ENABLE()
00165 #define AUDIO_DFSDMx_DATIN_CLK_DISABLE()
            __HAL_RCC_GPIOC_CLK_DISABLE()
00166 #define AUDIO_DFSDMx_DATIN_PIN
            GPIO_PIN_0
00167 #define AUDIO_DFSDMx_DATIN_GPIO_PORT
            GPIOC
00168
00169 /* DFSDM DMA Right and Left channels definit
ions */
00170 #define AUDIO_DFSDMx_DMAx_CLK_ENABLE()
            __HAL_RCC_DMA1_CLK_ENABLE()
00171 #define AUDIO_DFSDMx_DMAx_CLK_DISABLE()
            __HAL_RCC_DMA1_CLK_DISABLE()
00172 #define AUDIO_DFSDMx_DMAx_LEFT_CHANNEL
            DMA1_Channel4
00173 #define AUDIO_DFSDMx_DMAx_RIGHT_CHANNEL
            DMA1_Channel5
00174 #define AUDIO_DFSDMx_DMAx_LEFT_IRQ
            DMA1_Channel4_IRQn
00175 #define AUDIO_DFSDMx_DMAx_RIGHT_IRQ
            DMA1_Channel5_IRQn
00176 #define AUDIO_DFSDMx_DMAx_LEFT_REQUEST
            DMA_REQUEST_DFSDM1_FLT0
00177 #define AUDIO_DFSDMx_DMAx_RIGHT_REQUEST
            DMA_REQUEST_DFSDM1_FLT1
00178 #define AUDIO_DFSDMx_DMAx_PERIPH_DATA_SIZE
```

```
            DMA_PDATAALIGN_WORD
00179 #define AUDIO_DFSDMx_DMAx_MEM_DATA_SIZE
            DMA_MDATAALIGN_WORD
00180
00181 #define AUDIO_DFSDM_DMAx_LEFT_IRQHandler
            DMA1_Channel4_IRQHandler
00182 #define AUDIO_DFSDM_DMAx_RIGHT_IRQHandler
            DMA1_Channel5_IRQHandler
00183
00184 /* Select the interrupt preemption priority
and subpriority for the IT/DMA interrupt */
00185 #define AUDIO_IN_IRQ_PREPRIO
6   /* Select the preemption priority level(0 is t
he highest) */
00186
00187 /*-----------------------------------------
-----------------------------------
00188              CONFIGURATION: Audio Driver Con
figuration parameters
00189 -----------------------------------------
-----------------------------------*/
00190
00191 #define AUDIODATA_SIZE
2   /* 16-bits audio data size */
00192
00193 /* Audio status definition */
00194 #define AUDIO_OK                          0

00195 #define AUDIO_ERROR                       1

00196 #define AUDIO_TIMEOUT                     2

00197
00198 /* AudioFreq * DataSize (2 bytes) * NumChann
els (Stereo: 2) */
00199 #define DEFAULT_AUDIO_IN_FREQ
BSP_AUDIO_FREQUENCY_16K
```

```
00200 #define DEFAULT_AUDIO_IN_BIT_RESOLUTION
16
00201 #define DEFAULT_AUDIO_IN_CHANNEL_NBR
2 /* Mono = 1, Stereo = 2 */
00202 #define DEFAULT_AUDIO_IN_VOLUME
64
00203
00204 /* PCM buffer input size in word */
00205 #define INTERNAL_BUFF_SIZE
256
00206
00207 /*----------------------------------------
-----------------------------------
00208                         OPTIONAL Configuration d
efines parameters
00209 ----------------------------------------
-----------------------------------*/
00210
00211 /* Delay for the Codec to be correctly reset
 */
00212 #define CODEC_RESET_DELAY           5
00213
00214 /**
00215   * @}
00216   */
00217
00218 /** @defgroup STM32L4R9I_EVAL_AUDIO_Exported
_Variables Exported Variables
00219   * @{
00220   */
00221 extern SAI_HandleTypeDef          BSP_AUDIO_
hSai;
00222 extern DFSDM_Filter_HandleTypeDef BSP_AUDIO_
hDfsdmLeftFilter;
00223 extern DFSDM_Filter_HandleTypeDef BSP_AUDIO_
hDfsdmRightFilter;
00224
```

```
00225 /**
00226   * @}
00227   */
00228
00229 /** @defgroup STM32L4R9I_EVAL_AUDIO_Exported
_Macros Exported Macros
00230   * @{
00231   */
00232 #define DMA_MAX(_X_)                (((_X_)
<= DMA_MAX_SZE)? (_X_):DMA_MAX_SZE)
00233
00234 /**
00235   * @}
00236   */
00237
00238 /* Exported functions ---------------------
-----------------------------------*/
00239 /** @defgroup STM32L4R9I_EVAL_AUDIO_Exported
_Functions Exported Functions
00240   * @{
00241   */
00242 uint8_t BSP_AUDIO_OUT_Init(uint16_t OutputDe
vice, uint8_t Volume, uint32_t AudioFreq);
00243 uint8_t BSP_AUDIO_OUT_DeInit(void);
00244 uint8_t BSP_AUDIO_OUT_Play(uint16_t* pData,
uint32_t Size);
00245 uint8_t BSP_AUDIO_OUT_ChangeBuffer(uint16_t
*pData, uint16_t Size);
00246 uint8_t BSP_AUDIO_OUT_Pause(void);
00247 uint8_t BSP_AUDIO_OUT_Resume(void);
00248 uint8_t BSP_AUDIO_OUT_Stop(uint32_t Option);
00249 uint8_t BSP_AUDIO_OUT_SetVolume(uint8_t Volu
me);
00250 uint8_t BSP_AUDIO_OUT_SetFrequency(uint32_t
AudioFreq);
00251 void    BSP_AUDIO_OUT_ChangeAudioConfig(uint
32_t AudioOutOption);
```

```
00252 uint8_t BSP_AUDIO_OUT_SetMute(uint32_t Cmd);
00253 uint8_t BSP_AUDIO_OUT_SetOutputMode(uint8_t
Output);
00254 void    BSP_AUDIO_OUT_RegisterCallbacks(Audi
o_CallbackTypeDef ErrorCallback,
00255                                           Audi
o_CallbackTypeDef HalfTransferCallback,
00256                                           Audi
o_CallbackTypeDef TransferCompleteCallback);
00257
00258 uint8_t BSP_AUDIO_IN_Init(uint32_t AudioFreq
, uint32_t BitRes, uint32_t ChnlNbr);
00259 uint8_t BSP_AUDIO_IN_DeInit(void);
00260 uint8_t BSP_AUDIO_IN_Record(uint16_t *pData,
 uint32_t Size);
00261 uint8_t BSP_AUDIO_IN_SetFrequency(uint32_t A
udioFreq);
00262 uint8_t BSP_AUDIO_IN_Stop(void);
00263 uint8_t BSP_AUDIO_IN_Pause(void);
00264 uint8_t BSP_AUDIO_IN_Resume(void);
00265 void    BSP_AUDIO_IN_RegisterCallbacks(Audio
_CallbackTypeDef ErrorCallback,
00266                                         Audio
_CallbackTypeDef HalfTransferCallback,
00267                                         Audio
_CallbackTypeDef TransferCompleteCallback);
00268 /**
00269   * @}
00270   */
00271
00272 /**
00273   * @}
00274   */
00275
00276 /**
00277   * @}
00278   */
```

```
00279
00280 /**
00281   * @}
00282   */
00283
00284 #ifdef __cplusplus
00285 }
00286 #endif
00287
00288 #endif /* __STM32L4R9I_EVAL_AUDIO_H */
00289
00290 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_dsi_lcd.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_dsi_lcd.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_dsi_lcd.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_DSI_LCD_H
00039 #define __STM32L4R9I_EVAL_DSI_LCD_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-----------------------------------*/
00046 /* Include LCD component Driver */
00047
00048 #include "stm32l4r9i_eval.h"
00049
00050 /** @addtogroup BSP
00051   * @{
00052   */
00053
00054 /** @addtogroup STM32L4R9I_EVAL
00055   * @{
00056   */
00057
00058 /** @defgroup STM32L4R9I_EVAL_LCD STM32L4R9I
_EVAL LCD
00059   * @{
00060   */
00061
00062 /** @defgroup STM32L4R9I_EVAL_DSI_LCD STM32L
4R9I_EVAL DSI LCD
00063   * @{
00064   */
00065
00066 /** @addtogroup STM32L4R9I_EVAL_DSI_LCD_Expo
```

```
rted_Variables
00067   * @{
00068   */
00069
00070 /* DMA2D handle variable */
00071 extern DMA2D_HandleTypeDef hdma2d_eval;
00072
00073 /**
00074   * @}
00075   */
00076
00077 /** @addtogroup STM32L4R9I_EVAL_DSI_LCD_Expo
rted_Functions
00078   * @{
00079   */
00080
00081 uint8_t  BSP_DSI_LCD_Init(void);
00082 uint8_t  BSP_DSI_LCD_DeInit(void);
00083
00084 uint32_t BSP_DSI_LCD_GetXSize(void);
00085 uint32_t BSP_DSI_LCD_GetYSize(void);
00086
00087 uint8_t  BSP_DSI_LCD_SetTransparency(uint32_
t LayerIndex, uint8_t Transparency);
00088 uint8_t  BSP_DSI_LCD_SetColorKeying(uint32_t
 LayerIndex, uint32_t RGBValue);
00089 uint8_t  BSP_DSI_LCD_ResetColorKeying(uint32
_t LayerIndex);
00090
00091 uint8_t  BSP_DSI_LCD_SelectLayer(uint32_t La
yerIndex);
00092 uint8_t  BSP_DSI_LCD_SetLayerVisible(uint32_
t LayerIndex, FunctionalState State);
00093
00094 void     BSP_DSI_LCD_SetTextColor(uint32_t C
olor);
00095 uint32_t BSP_DSI_LCD_GetTextColor(void);
```

```
00096 void     BSP_DSI_LCD_SetBackColor(uint32_t C
olor);
00097 uint32_t BSP_DSI_LCD_GetBackColor(void);
00098 void     BSP_DSI_LCD_SetFont(sFONT *fonts);
00099 sFONT    *BSP_DSI_LCD_GetFont(void);
00100
00101 uint32_t BSP_DSI_LCD_ReadPixel(uint16_t Xpos
, uint16_t Ypos);
00102 void     BSP_DSI_LCD_DrawPixel(uint16_t Xpos
, uint16_t Ypos, uint32_t pixel);
00103 void     BSP_DSI_LCD_Clear(uint32_t Color);
00104 void     BSP_DSI_LCD_ClearStringLine(uint32_
t Line);
00105 void     BSP_DSI_LCD_DisplayStringAtLine(uin
t16_t Line, uint8_t *ptr);
00106 void     BSP_DSI_LCD_DisplayStringAt(uint16_
t Xpos, uint16_t Ypos, uint8_t *Text, Text_AlignMo
deTypdef Mode);
00107 void     BSP_DSI_LCD_DisplayChar(uint16_t Xp
os, uint16_t Ypos, uint8_t Ascii);
00108
00109 void     BSP_DSI_LCD_DrawHLine(uint16_t Xpos
, uint16_t Ypos, uint16_t Length);
00110 void     BSP_DSI_LCD_DrawVLine(uint16_t Xpos
, uint16_t Ypos, uint16_t Length);
00111 void     BSP_DSI_LCD_DrawLine(uint16_t x1, u
int16_t y1, uint16_t x2, uint16_t y2);
00112 void     BSP_DSI_LCD_DrawRect(uint16_t Xpos,
 uint16_t Ypos, uint16_t Width, uint16_t Height);
00113 void     BSP_DSI_LCD_DrawCircle(uint16_t Xpo
s, uint16_t Ypos, uint16_t Radius);
00114 void     BSP_DSI_LCD_DrawPolygon(pPoint Poin
ts, uint16_t PointCount);
00115 void     BSP_DSI_LCD_DrawEllipse(int Xpos, i
nt Ypos, int XRadius, int YRadius);
00116 void     BSP_DSI_LCD_DrawBitmap(uint32_t Xpo
s, uint32_t Ypos, uint8_t *pbmp);
```

```
00117
00118 void     BSP_DSI_LCD_FillRect(uint16_t Xpos,
 uint16_t Ypos, uint16_t Width, uint16_t Height);
00119 void     BSP_DSI_LCD_FillCircle(uint16_t Xpo
s, uint16_t Ypos, uint16_t Radius);
00120 void     BSP_DSI_LCD_FillPolygon(pPoint Poin
ts, uint16_t PointCount);
00121 void     BSP_DSI_LCD_FillEllipse(int Xpos, i
nt Ypos, int XRadius, int YRadius);
00122
00123 void     BSP_DSI_LCD_DisplayOff(void);
00124 void     BSP_DSI_LCD_DisplayOn(void);
00125
00126 void     BSP_DSI_LCD_Refresh(void);
00127 uint8_t  BSP_DSI_LCD_IsFrameBufferAvailable(
void);
00128
00129 void     BSP_DSI_LCD_SetBrightness(uint8_t B
rightnessValue);
00130
00131 /* These __weak functions can be surcharged
by application code for specific application needs
 */
00132 void     BSP_DSI_LCD_MspInit(void);
00133 void     BSP_DSI_LCD_MspDeInit(void);
00134 void     BSP_DSI_LCD_DMA2D_IRQHandler(void);
00135 void     BSP_DSI_LCD_DSI_IRQHandler(void);
00136 void     BSP_DSI_LCD_LTDC_IRQHandler(void);
00137 void     BSP_DSI_LCD_LTDC_ER_IRQHandler(void
);
00138
00139 /**
00140   * @}
00141   */
00142
00143 /**
00144   * @}
```

```
00145   */
00146
00147 /**
00148   * @}
00149   */
00150
00151 /**
00152   * @}
00153   */
00154
00155 /**
00156   * @}
00157   */
00158
00159 #ifdef __cplusplus
00160 }
00161 #endif
00162
00163 #endif /* __STM32L4R9I_EVAL_DSI_LCD_H */
00164
00165 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_gfxmmu_lut.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_gfxmmu_lut.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains GFXMMU look-up-table for DSI round LCD.
00006   ******************************************************************************
00007   * @attention
00008   *
00009   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00010   *
00011   * Redistribution and use in source and binary forms, with or without modification,
00012   * are permitted provided that the following conditions are met:
00013   *   1. Redistributions of source code must retain the above copyright notice,
00014   *      this list of conditions and the following disclaimer.
```

```
00015   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00016   *      this list of conditions and the fol
lowing disclaimer in the documentation
00017   *      and/or other materials provided wit
h the distribution.
00018   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00019   *      may be used to endorse or promote p
roducts derived from this software
00020   *      without specific prior written perm
ission.
00021   *
00022   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00023   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00024   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00025   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00026   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00027   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00028   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00029   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00030   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00031   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00032   *
00033   ****************************************
**********************************
00034   */
```

```
00035
00036 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00037 #ifndef __STM32L4R9I_EVAL_GFXMMU_LUT_H
00038 #define __STM32L4R9I_EVAL_GFXMMU_LUT_H
00039
00040 #ifdef __cplusplus
00041  extern "C" {
00042 #endif
00043
00044 /* GFXMMU look-up table for ARGB8888 */
00045 const uint32_t gfxmmu_lut_config_argb8888[78
0] = {
00046     0x00342d01, // GFXMMU_LUT0L
00047     0x003ffd30, // GFXMMU_LUT0H
00048     0x00362b01, // GFXMMU_LUT1L
00049     0x003ffdd0, // GFXMMU_LUT1H
00050     0x00382901, // GFXMMU_LUT2L
00051     0x003ffeb0, // GFXMMU_LUT2H
00052     0x00392701, // GFXMMU_LUT3L
00053     0x003fffd0, // GFXMMU_LUT3H
00054     0x003b2601, // GFXMMU_LUT4L
00055     0x00000110, // GFXMMU_LUT4H
00056     0x003c2501, // GFXMMU_LUT5L
00057     0x00000280, // GFXMMU_LUT5H
00058     0x003d2401, // GFXMMU_LUT6L
00059     0x00000410, // GFXMMU_LUT6H
00060     0x003d2301, // GFXMMU_LUT7L
00061     0x000005c0, // GFXMMU_LUT7H
00062     0x003e2201, // GFXMMU_LUT8L
00063     0x00000780, // GFXMMU_LUT8H
00064     0x003f2101, // GFXMMU_LUT9L
00065     0x00000960, // GFXMMU_LUT9H
00066     0x00402101, // GFXMMU_LUT10L
00067     0x00000b50, // GFXMMU_LUT10H
00068     0x00402001, // GFXMMU_LUT11L
00069     0x00000d60, // GFXMMU_LUT11H
```

```
00070     0x00411f01, // GFXMMU_LUT12L
00071     0x00000f80, // GFXMMU_LUT12H
00072     0x00421f01, // GFXMMU_LUT13L
00073     0x000011b0, // GFXMMU_LUT13H
00074     0x00431e01, // GFXMMU_LUT14L
00075     0x00001400, // GFXMMU_LUT14H
00076     0x00431d01, // GFXMMU_LUT15L
00077     0x00001670, // GFXMMU_LUT15H
00078     0x00441d01, // GFXMMU_LUT16L
00079     0x000018e0, // GFXMMU_LUT16H
00080     0x00441c01, // GFXMMU_LUT17L
00081     0x00001b70, // GFXMMU_LUT17H
00082     0x00451c01, // GFXMMU_LUT18L
00083     0x00001e00, // GFXMMU_LUT18H
00084     0x00451b01, // GFXMMU_LUT19L
00085     0x000020b0, // GFXMMU_LUT19H
00086     0x00461b01, // GFXMMU_LUT20L
00087     0x00002360, // GFXMMU_LUT20H
00088     0x00461a01, // GFXMMU_LUT21L
00089     0x00002630, // GFXMMU_LUT21H
00090     0x00471a01, // GFXMMU_LUT22L
00091     0x00002900, // GFXMMU_LUT22H
00092     0x00471901, // GFXMMU_LUT23L
00093     0x00002bf0, // GFXMMU_LUT23H
00094     0x00481901, // GFXMMU_LUT24L
00095     0x00002ee0, // GFXMMU_LUT24H
00096     0x00481801, // GFXMMU_LUT25L
00097     0x000031f0, // GFXMMU_LUT25H
00098     0x00491801, // GFXMMU_LUT26L
00099     0x00003500, // GFXMMU_LUT26H
00100     0x00491701, // GFXMMU_LUT27L
00101     0x00003830, // GFXMMU_LUT27H
00102     0x00491701, // GFXMMU_LUT28L
00103     0x00003b60, // GFXMMU_LUT28H
00104     0x004a1701, // GFXMMU_LUT29L
00105     0x00003e90, // GFXMMU_LUT29H
00106     0x004a1601, // GFXMMU_LUT30L
```

```
00107     0x000041e0, // GFXMMU_LUT30H
00108     0x004b1601, // GFXMMU_LUT31L
00109     0x00004530, // GFXMMU_LUT31H
00110     0x004b1501, // GFXMMU_LUT32L
00111     0x000048a0, // GFXMMU_LUT32H
00112     0x004b1501, // GFXMMU_LUT33L
00113     0x00004c10, // GFXMMU_LUT33H
00114     0x004c1501, // GFXMMU_LUT34L
00115     0x00004f80, // GFXMMU_LUT34H
00116     0x004c1401, // GFXMMU_LUT35L
00117     0x00005310, // GFXMMU_LUT35H
00118     0x004c1401, // GFXMMU_LUT36L
00119     0x000056a0, // GFXMMU_LUT36H
00120     0x004d1401, // GFXMMU_LUT37L
00121     0x00005a30, // GFXMMU_LUT37H
00122     0x004d1301, // GFXMMU_LUT38L
00123     0x00005de0, // GFXMMU_LUT38H
00124     0x004e1301, // GFXMMU_LUT39L
00125     0x00006190, // GFXMMU_LUT39H
00126     0x004e1301, // GFXMMU_LUT40L
00127     0x00006550, // GFXMMU_LUT40H
00128     0x004e1201, // GFXMMU_LUT41L
00129     0x00006920, // GFXMMU_LUT41H
00130     0x004e1201, // GFXMMU_LUT42L
00131     0x00006cf0, // GFXMMU_LUT42H
00132     0x004f1201, // GFXMMU_LUT43L
00133     0x000070c0, // GFXMMU_LUT43H
00134     0x004f1101, // GFXMMU_LUT44L
00135     0x000074b0, // GFXMMU_LUT44H
00136     0x004f1101, // GFXMMU_LUT45L
00137     0x000078a0, // GFXMMU_LUT45H
00138     0x00501101, // GFXMMU_LUT46L
00139     0x00007c90, // GFXMMU_LUT46H
00140     0x00501101, // GFXMMU_LUT47L
00141     0x00008090, // GFXMMU_LUT47H
00142     0x00501001, // GFXMMU_LUT48L
00143     0x000084a0, // GFXMMU_LUT48H
```

```
00144     0x00511001, // GFXMMU_LUT49L
00145     0x000088b0, // GFXMMU_LUT49H
00146     0x00511001, // GFXMMU_LUT50L
00147     0x00008cd0, // GFXMMU_LUT50H
00148     0x00510f01, // GFXMMU_LUT51L
00149     0x00009100, // GFXMMU_LUT51H
00150     0x00510f01, // GFXMMU_LUT52L
00151     0x00009530, // GFXMMU_LUT52H
00152     0x00520f01, // GFXMMU_LUT53L
00153     0x00009960, // GFXMMU_LUT53H
00154     0x00520f01, // GFXMMU_LUT54L
00155     0x00009da0, // GFXMMU_LUT54H
00156     0x00520e01, // GFXMMU_LUT55L
00157     0x0000a1f0, // GFXMMU_LUT55H
00158     0x00520e01, // GFXMMU_LUT56L
00159     0x0000a640, // GFXMMU_LUT56H
00160     0x00530e01, // GFXMMU_LUT57L
00161     0x0000aa90, // GFXMMU_LUT57H
00162     0x00530e01, // GFXMMU_LUT58L
00163     0x0000aef0, // GFXMMU_LUT58H
00164     0x00530d01, // GFXMMU_LUT59L
00165     0x0000b360, // GFXMMU_LUT59H
00166     0x00530d01, // GFXMMU_LUT60L
00167     0x0000b7d0, // GFXMMU_LUT60H
00168     0x00540d01, // GFXMMU_LUT61L
00169     0x0000bc40, // GFXMMU_LUT61H
00170     0x00540d01, // GFXMMU_LUT62L
00171     0x0000c0c0, // GFXMMU_LUT62H
00172     0x00540c01, // GFXMMU_LUT63L
00173     0x0000c550, // GFXMMU_LUT63H
00174     0x00540c01, // GFXMMU_LUT64L
00175     0x0000c9e0, // GFXMMU_LUT64H
00176     0x00550c01, // GFXMMU_LUT65L
00177     0x0000ce70, // GFXMMU_LUT65H
00178     0x00550c01, // GFXMMU_LUT66L
00179     0x0000d310, // GFXMMU_LUT66H
00180     0x00550c01, // GFXMMU_LUT67L
```

```
00181     0x0000d7b0, // GFXMMU_LUT67H
00182     0x00550b01, // GFXMMU_LUT68L
00183     0x0000dc60, // GFXMMU_LUT68H
00184     0x00550b01, // GFXMMU_LUT69L
00185     0x0000e110, // GFXMMU_LUT69H
00186     0x00560b01, // GFXMMU_LUT70L
00187     0x0000e5c0, // GFXMMU_LUT70H
00188     0x00560b01, // GFXMMU_LUT71L
00189     0x0000ea80, // GFXMMU_LUT71H
00190     0x00560b01, // GFXMMU_LUT72L
00191     0x0000ef40, // GFXMMU_LUT72H
00192     0x00560a01, // GFXMMU_LUT73L
00193     0x0000f410, // GFXMMU_LUT73H
00194     0x00560a01, // GFXMMU_LUT74L
00195     0x0000f8e0, // GFXMMU_LUT74H
00196     0x00570a01, // GFXMMU_LUT75L
00197     0x0000fdb0, // GFXMMU_LUT75H
00198     0x00570a01, // GFXMMU_LUT76L
00199     0x00010290, // GFXMMU_LUT76H
00200     0x00570901, // GFXMMU_LUT77L
00201     0x00010780, // GFXMMU_LUT77H
00202     0x00570901, // GFXMMU_LUT78L
00203     0x00010c70, // GFXMMU_LUT78H
00204     0x00570901, // GFXMMU_LUT79L
00205     0x00011160, // GFXMMU_LUT79H
00206     0x00580901, // GFXMMU_LUT80L
00207     0x00011650, // GFXMMU_LUT80H
00208     0x00580901, // GFXMMU_LUT81L
00209     0x00011b50, // GFXMMU_LUT81H
00210     0x00580901, // GFXMMU_LUT82L
00211     0x00012050, // GFXMMU_LUT82H
00212     0x00580801, // GFXMMU_LUT83L
00213     0x00012560, // GFXMMU_LUT83H
00214     0x00580801, // GFXMMU_LUT84L
00215     0x00012a70, // GFXMMU_LUT84H
00216     0x00580801, // GFXMMU_LUT85L
00217     0x00012f80, // GFXMMU_LUT85H
```

```
00218     0x00590801, // GFXMMU_LUT86L
00219     0x00013490, // GFXMMU_LUT86H
00220     0x00590801, // GFXMMU_LUT87L
00221     0x000139b0, // GFXMMU_LUT87H
00222     0x00590801, // GFXMMU_LUT88L
00223     0x00013ed0, // GFXMMU_LUT88H
00224     0x00590701, // GFXMMU_LUT89L
00225     0x00014400, // GFXMMU_LUT89H
00226     0x00590701, // GFXMMU_LUT90L
00227     0x00014930, // GFXMMU_LUT90H
00228     0x00590701, // GFXMMU_LUT91L
00229     0x00014e60, // GFXMMU_LUT91H
00230     0x005a0701, // GFXMMU_LUT92L
00231     0x00015390, // GFXMMU_LUT92H
00232     0x005a0701, // GFXMMU_LUT93L
00233     0x000158d0, // GFXMMU_LUT93H
00234     0x005a0701, // GFXMMU_LUT94L
00235     0x00015e10, // GFXMMU_LUT94H
00236     0x005a0601, // GFXMMU_LUT95L
00237     0x00016360, // GFXMMU_LUT95H
00238     0x005a0601, // GFXMMU_LUT96L
00239     0x000168b0, // GFXMMU_LUT96H
00240     0x005a0601, // GFXMMU_LUT97L
00241     0x00016e00, // GFXMMU_LUT97H
00242     0x005a0601, // GFXMMU_LUT98L
00243     0x00017350, // GFXMMU_LUT98H
00244     0x005b0601, // GFXMMU_LUT99L
00245     0x000178a0, // GFXMMU_LUT99H
00246     0x005b0601, // GFXMMU_LUT100L
00247     0x00017e00, // GFXMMU_LUT100H
00248     0x005b0601, // GFXMMU_LUT101L
00249     0x00018360, // GFXMMU_LUT101H
00250     0x005b0601, // GFXMMU_LUT102L
00251     0x000188c0, // GFXMMU_LUT102H
00252     0x005b0501, // GFXMMU_LUT103L
00253     0x00018e30, // GFXMMU_LUT103H
00254     0x005b0501, // GFXMMU_LUT104L
```

```
00255     0x000193a0, // GFXMMU_LUT104H
00256     0x005b0501, // GFXMMU_LUT105L
00257     0x00019910, // GFXMMU_LUT105H
00258     0x005c0501, // GFXMMU_LUT106L
00259     0x00019e80, // GFXMMU_LUT106H
00260     0x005c0501, // GFXMMU_LUT107L
00261     0x0001a400, // GFXMMU_LUT107H
00262     0x005c0501, // GFXMMU_LUT108L
00263     0x0001a980, // GFXMMU_LUT108H
00264     0x005c0501, // GFXMMU_LUT109L
00265     0x0001af00, // GFXMMU_LUT109H
00266     0x005c0401, // GFXMMU_LUT110L
00267     0x0001b490, // GFXMMU_LUT110H
00268     0x005c0401, // GFXMMU_LUT111L
00269     0x0001ba20, // GFXMMU_LUT111H
00270     0x005c0401, // GFXMMU_LUT112L
00271     0x0001bfb0, // GFXMMU_LUT112H
00272     0x005c0401, // GFXMMU_LUT113L
00273     0x0001c540, // GFXMMU_LUT113H
00274     0x005d0401, // GFXMMU_LUT114L
00275     0x0001cad0, // GFXMMU_LUT114H
00276     0x005d0401, // GFXMMU_LUT115L
00277     0x0001d070, // GFXMMU_LUT115H
00278     0x005d0401, // GFXMMU_LUT116L
00279     0x0001d610, // GFXMMU_LUT116H
00280     0x005d0401, // GFXMMU_LUT117L
00281     0x0001dbb0, // GFXMMU_LUT117H
00282     0x005d0401, // GFXMMU_LUT118L
00283     0x0001e150, // GFXMMU_LUT118H
00284     0x005d0301, // GFXMMU_LUT119L
00285     0x0001e700, // GFXMMU_LUT119H
00286     0x005d0301, // GFXMMU_LUT120L
00287     0x0001ecb0, // GFXMMU_LUT120H
00288     0x005d0301, // GFXMMU_LUT121L
00289     0x0001f260, // GFXMMU_LUT121H
00290     0x005d0301, // GFXMMU_LUT122L
00291     0x0001f810, // GFXMMU_LUT122H
```

```
00292     0x005d0301, // GFXMMU_LUT123L
00293     0x0001fdc0, // GFXMMU_LUT123H
00294     0x005e0301, // GFXMMU_LUT124L
00295     0x00020370, // GFXMMU_LUT124H
00296     0x005e0301, // GFXMMU_LUT125L
00297     0x00020930, // GFXMMU_LUT125H
00298     0x005e0301, // GFXMMU_LUT126L
00299     0x00020ef0, // GFXMMU_LUT126H
00300     0x005e0301, // GFXMMU_LUT127L
00301     0x000214b0, // GFXMMU_LUT127H
00302     0x005e0301, // GFXMMU_LUT128L
00303     0x00021a70, // GFXMMU_LUT128H
00304     0x005e0301, // GFXMMU_LUT129L
00305     0x00022030, // GFXMMU_LUT129H
00306     0x005e0201, // GFXMMU_LUT130L
00307     0x00022600, // GFXMMU_LUT130H
00308     0x005e0201, // GFXMMU_LUT131L
00309     0x00022bd0, // GFXMMU_LUT131H
00310     0x005e0201, // GFXMMU_LUT132L
00311     0x000231a0, // GFXMMU_LUT132H
00312     0x005e0201, // GFXMMU_LUT133L
00313     0x00023770, // GFXMMU_LUT133H
00314     0x005e0201, // GFXMMU_LUT134L
00315     0x00023d40, // GFXMMU_LUT134H
00316     0x005f0201, // GFXMMU_LUT135L
00317     0x00024310, // GFXMMU_LUT135H
00318     0x005f0201, // GFXMMU_LUT136L
00319     0x000248f0, // GFXMMU_LUT136H
00320     0x005f0201, // GFXMMU_LUT137L
00321     0x00024ed0, // GFXMMU_LUT137H
00322     0x005f0201, // GFXMMU_LUT138L
00323     0x000254b0, // GFXMMU_LUT138H
00324     0x005f0201, // GFXMMU_LUT139L
00325     0x00025a90, // GFXMMU_LUT139H
00326     0x005f0201, // GFXMMU_LUT140L
00327     0x00026070, // GFXMMU_LUT140H
00328     0x005f0201, // GFXMMU_LUT141L
```

```
00329     0x00026650, // GFXMMU_LUT141H
00330     0x005f0101, // GFXMMU_LUT142L
00331     0x00026c40, // GFXMMU_LUT142H
00332     0x005f0101, // GFXMMU_LUT143L
00333     0x00027230, // GFXMMU_LUT143H
00334     0x005f0101, // GFXMMU_LUT144L
00335     0x00027820, // GFXMMU_LUT144H
00336     0x005f0101, // GFXMMU_LUT145L
00337     0x00027e10, // GFXMMU_LUT145H
00338     0x005f0101, // GFXMMU_LUT146L
00339     0x00028400, // GFXMMU_LUT146H
00340     0x005f0101, // GFXMMU_LUT147L
00341     0x000289f0, // GFXMMU_LUT147H
00342     0x005f0101, // GFXMMU_LUT148L
00343     0x00028fe0, // GFXMMU_LUT148H
00344     0x00600101, // GFXMMU_LUT149L
00345     0x000295d0, // GFXMMU_LUT149H
00346     0x00600101, // GFXMMU_LUT150L
00347     0x00029bd0, // GFXMMU_LUT150H
00348     0x00600101, // GFXMMU_LUT151L
00349     0x0002a1d0, // GFXMMU_LUT151H
00350     0x00600101, // GFXMMU_LUT152L
00351     0x0002a7d0, // GFXMMU_LUT152H
00352     0x00600101, // GFXMMU_LUT153L
00353     0x0002add0, // GFXMMU_LUT153H
00354     0x00600101, // GFXMMU_LUT154L
00355     0x0002b3d0, // GFXMMU_LUT154H
00356     0x00600101, // GFXMMU_LUT155L
00357     0x0002b9d0, // GFXMMU_LUT155H
00358     0x00600101, // GFXMMU_LUT156L
00359     0x0002bfd0, // GFXMMU_LUT156H
00360     0x00600101, // GFXMMU_LUT157L
00361     0x0002c5d0, // GFXMMU_LUT157H
00362     0x00600001, // GFXMMU_LUT158L
00363     0x0002cbe0, // GFXMMU_LUT158H
00364     0x00600001, // GFXMMU_LUT159L
00365     0x0002d1f0, // GFXMMU_LUT159H
```

```
00366     0x00600001, // GFXMMU_LUT160L
00367     0x0002d800, // GFXMMU_LUT160H
00368     0x00600001, // GFXMMU_LUT161L
00369     0x0002de10, // GFXMMU_LUT161H
00370     0x00600001, // GFXMMU_LUT162L
00371     0x0002e420, // GFXMMU_LUT162H
00372     0x00600001, // GFXMMU_LUT163L
00373     0x0002ea30, // GFXMMU_LUT163H
00374     0x00600001, // GFXMMU_LUT164L
00375     0x0002f040, // GFXMMU_LUT164H
00376     0x00600001, // GFXMMU_LUT165L
00377     0x0002f650, // GFXMMU_LUT165H
00378     0x00600001, // GFXMMU_LUT166L
00379     0x0002fc60, // GFXMMU_LUT166H
00380     0x00600001, // GFXMMU_LUT167L
00381     0x00030270, // GFXMMU_LUT167H
00382     0x00600001, // GFXMMU_LUT168L
00383     0x00030880, // GFXMMU_LUT168H
00384     0x00600001, // GFXMMU_LUT169L
00385     0x00030e90, // GFXMMU_LUT169H
00386     0x00600001, // GFXMMU_LUT170L
00387     0x000314a0, // GFXMMU_LUT170H
00388     0x00600001, // GFXMMU_LUT171L
00389     0x00031ab0, // GFXMMU_LUT171H
00390     0x00610001, // GFXMMU_LUT172L
00391     0x000320c0, // GFXMMU_LUT172H
00392     0x00610001, // GFXMMU_LUT173L
00393     0x000326e0, // GFXMMU_LUT173H
00394     0x00610001, // GFXMMU_LUT174L
00395     0x00032d00, // GFXMMU_LUT174H
00396     0x00610001, // GFXMMU_LUT175L
00397     0x00033320, // GFXMMU_LUT175H
00398     0x00610001, // GFXMMU_LUT176L
00399     0x00033940, // GFXMMU_LUT176H
00400     0x00610001, // GFXMMU_LUT177L
00401     0x00033f60, // GFXMMU_LUT177H
00402     0x00610001, // GFXMMU_LUT178L
```

```
00403     0x00034580, // GFXMMU_LUT178H
00404     0x00610001, // GFXMMU_LUT179L
00405     0x00034ba0, // GFXMMU_LUT179H
00406     0x00610001, // GFXMMU_LUT180L
00407     0x000351c0, // GFXMMU_LUT180H
00408     0x00610001, // GFXMMU_LUT181L
00409     0x000357e0, // GFXMMU_LUT181H
00410     0x00610001, // GFXMMU_LUT182L
00411     0x00035e00, // GFXMMU_LUT182H
00412     0x00610001, // GFXMMU_LUT183L
00413     0x00036420, // GFXMMU_LUT183H
00414     0x00610001, // GFXMMU_LUT184L
00415     0x00036a40, // GFXMMU_LUT184H
00416     0x00610001, // GFXMMU_LUT185L
00417     0x00037060, // GFXMMU_LUT185H
00418     0x00610001, // GFXMMU_LUT186L
00419     0x00037680, // GFXMMU_LUT186H
00420     0x00610001, // GFXMMU_LUT187L
00421     0x00037ca0, // GFXMMU_LUT187H
00422     0x00610001, // GFXMMU_LUT188L
00423     0x000382c0, // GFXMMU_LUT188H
00424     0x00610001, // GFXMMU_LUT189L
00425     0x000388e0, // GFXMMU_LUT189H
00426     0x00610001, // GFXMMU_LUT190L
00427     0x00038f00, // GFXMMU_LUT190H
00428     0x00610001, // GFXMMU_LUT191L
00429     0x00039520, // GFXMMU_LUT191H
00430     0x00610001, // GFXMMU_LUT192L
00431     0x00039b40, // GFXMMU_LUT192H
00432     0x00610001, // GFXMMU_LUT193L
00433     0x0003a160, // GFXMMU_LUT193H
00434     0x00610001, // GFXMMU_LUT194L
00435     0x0003a780, // GFXMMU_LUT194H
00436     0x00610001, // GFXMMU_LUT195L
00437     0x0003ada0, // GFXMMU_LUT195H
00438     0x00610001, // GFXMMU_LUT196L
00439     0x0003b3c0, // GFXMMU_LUT196H
```

```
00440     0x00610001, // GFXMMU_LUT197L
00441     0x0003b9e0, // GFXMMU_LUT197H
00442     0x00610001, // GFXMMU_LUT198L
00443     0x0003c000, // GFXMMU_LUT198H
00444     0x00610001, // GFXMMU_LUT199L
00445     0x0003c620, // GFXMMU_LUT199H
00446     0x00610001, // GFXMMU_LUT200L
00447     0x0003cc40, // GFXMMU_LUT200H
00448     0x00610001, // GFXMMU_LUT201L
00449     0x0003d260, // GFXMMU_LUT201H
00450     0x00610001, // GFXMMU_LUT202L
00451     0x0003d880, // GFXMMU_LUT202H
00452     0x00610001, // GFXMMU_LUT203L
00453     0x0003dea0, // GFXMMU_LUT203H
00454     0x00610001, // GFXMMU_LUT204L
00455     0x0003e4c0, // GFXMMU_LUT204H
00456     0x00610001, // GFXMMU_LUT205L
00457     0x0003eae0, // GFXMMU_LUT205H
00458     0x00610001, // GFXMMU_LUT206L
00459     0x0003f100, // GFXMMU_LUT206H
00460     0x00610001, // GFXMMU_LUT207L
00461     0x0003f720, // GFXMMU_LUT207H
00462     0x00610001, // GFXMMU_LUT208L
00463     0x0003fd40, // GFXMMU_LUT208H
00464     0x00610001, // GFXMMU_LUT209L
00465     0x00040360, // GFXMMU_LUT209H
00466     0x00610001, // GFXMMU_LUT210L
00467     0x00040980, // GFXMMU_LUT210H
00468     0x00610001, // GFXMMU_LUT211L
00469     0x00040fa0, // GFXMMU_LUT211H
00470     0x00610001, // GFXMMU_LUT212L
00471     0x000415c0, // GFXMMU_LUT212H
00472     0x00610001, // GFXMMU_LUT213L
00473     0x00041be0, // GFXMMU_LUT213H
00474     0x00610001, // GFXMMU_LUT214L
00475     0x00042200, // GFXMMU_LUT214H
00476     0x00610001, // GFXMMU_LUT215L
```

```
00477     0x00042820, // GFXMMU_LUT215H
00478     0x00610001, // GFXMMU_LUT216L
00479     0x00042e40, // GFXMMU_LUT216H
00480     0x00610001, // GFXMMU_LUT217L
00481     0x00043460, // GFXMMU_LUT217H
00482     0x00600001, // GFXMMU_LUT218L
00483     0x00043a80, // GFXMMU_LUT218H
00484     0x00600001, // GFXMMU_LUT219L
00485     0x00044090, // GFXMMU_LUT219H
00486     0x00600001, // GFXMMU_LUT220L
00487     0x000446a0, // GFXMMU_LUT220H
00488     0x00600001, // GFXMMU_LUT221L
00489     0x00044cb0, // GFXMMU_LUT221H
00490     0x00600001, // GFXMMU_LUT222L
00491     0x000452c0, // GFXMMU_LUT222H
00492     0x00600001, // GFXMMU_LUT223L
00493     0x000458d0, // GFXMMU_LUT223H
00494     0x00600001, // GFXMMU_LUT224L
00495     0x00045ee0, // GFXMMU_LUT224H
00496     0x00600001, // GFXMMU_LUT225L
00497     0x000464f0, // GFXMMU_LUT225H
00498     0x00600001, // GFXMMU_LUT226L
00499     0x00046b00, // GFXMMU_LUT226H
00500     0x00600001, // GFXMMU_LUT227L
00501     0x00047110, // GFXMMU_LUT227H
00502     0x00600001, // GFXMMU_LUT228L
00503     0x00047720, // GFXMMU_LUT228H
00504     0x00600001, // GFXMMU_LUT229L
00505     0x00047d30, // GFXMMU_LUT229H
00506     0x00600001, // GFXMMU_LUT230L
00507     0x00048340, // GFXMMU_LUT230H
00508     0x00600001, // GFXMMU_LUT231L
00509     0x00048950, // GFXMMU_LUT231H
00510     0x00600101, // GFXMMU_LUT232L
00511     0x00048f50, // GFXMMU_LUT232H
00512     0x00600101, // GFXMMU_LUT233L
00513     0x00049550, // GFXMMU_LUT233H
```

```
00514     0x00600101, // GFXMMU_LUT234L
00515     0x00049b50, // GFXMMU_LUT234H
00516     0x00600101, // GFXMMU_LUT235L
00517     0x0004a150, // GFXMMU_LUT235H
00518     0x00600101, // GFXMMU_LUT236L
00519     0x0004a750, // GFXMMU_LUT236H
00520     0x00600101, // GFXMMU_LUT237L
00521     0x0004ad50, // GFXMMU_LUT237H
00522     0x00600101, // GFXMMU_LUT238L
00523     0x0004b350, // GFXMMU_LUT238H
00524     0x00600101, // GFXMMU_LUT239L
00525     0x0004b950, // GFXMMU_LUT239H
00526     0x00600101, // GFXMMU_LUT240L
00527     0x0004bf50, // GFXMMU_LUT240H
00528     0x005f0101, // GFXMMU_LUT241L
00529     0x0004c550, // GFXMMU_LUT241H
00530     0x005f0101, // GFXMMU_LUT242L
00531     0x0004cb40, // GFXMMU_LUT242H
00532     0x005f0101, // GFXMMU_LUT243L
00533     0x0004d130, // GFXMMU_LUT243H
00534     0x005f0101, // GFXMMU_LUT244L
00535     0x0004d720, // GFXMMU_LUT244H
00536     0x005f0101, // GFXMMU_LUT245L
00537     0x0004dd10, // GFXMMU_LUT245H
00538     0x005f0101, // GFXMMU_LUT246L
00539     0x0004e300, // GFXMMU_LUT246H
00540     0x005f0101, // GFXMMU_LUT247L
00541     0x0004e8f0, // GFXMMU_LUT247H
00542     0x005f0201, // GFXMMU_LUT248L
00543     0x0004eed0, // GFXMMU_LUT248H
00544     0x005f0201, // GFXMMU_LUT249L
00545     0x0004f4b0, // GFXMMU_LUT249H
00546     0x005f0201, // GFXMMU_LUT250L
00547     0x0004fa90, // GFXMMU_LUT250H
00548     0x005f0201, // GFXMMU_LUT251L
00549     0x00050070, // GFXMMU_LUT251H
00550     0x005f0201, // GFXMMU_LUT252L
```

```
00551     0x00050650, // GFXMMU_LUT252H
00552     0x005f0201, // GFXMMU_LUT253L
00553     0x00050c30, // GFXMMU_LUT253H
00554     0x005f0201, // GFXMMU_LUT254L
00555     0x00051210, // GFXMMU_LUT254H
00556     0x005e0201, // GFXMMU_LUT255L
00557     0x000517f0, // GFXMMU_LUT255H
00558     0x005e0201, // GFXMMU_LUT256L
00559     0x00051dc0, // GFXMMU_LUT256H
00560     0x005e0201, // GFXMMU_LUT257L
00561     0x00052390, // GFXMMU_LUT257H
00562     0x005e0201, // GFXMMU_LUT258L
00563     0x00052960, // GFXMMU_LUT258H
00564     0x005e0201, // GFXMMU_LUT259L
00565     0x00052f30, // GFXMMU_LUT259H
00566     0x005e0301, // GFXMMU_LUT260L
00567     0x000534f0, // GFXMMU_LUT260H
00568     0x005e0301, // GFXMMU_LUT261L
00569     0x00053ab0, // GFXMMU_LUT261H
00570     0x005e0301, // GFXMMU_LUT262L
00571     0x00054070, // GFXMMU_LUT262H
00572     0x005e0301, // GFXMMU_LUT263L
00573     0x00054630, // GFXMMU_LUT263H
00574     0x005e0301, // GFXMMU_LUT264L
00575     0x00054bf0, // GFXMMU_LUT264H
00576     0x005e0301, // GFXMMU_LUT265L
00577     0x000551b0, // GFXMMU_LUT265H
00578     0x005d0301, // GFXMMU_LUT266L
00579     0x00055770, // GFXMMU_LUT266H
00580     0x005d0301, // GFXMMU_LUT267L
00581     0x00055d20, // GFXMMU_LUT267H
00582     0x005d0301, // GFXMMU_LUT268L
00583     0x000562d0, // GFXMMU_LUT268H
00584     0x005d0301, // GFXMMU_LUT269L
00585     0x00056880, // GFXMMU_LUT269H
00586     0x005d0301, // GFXMMU_LUT270L
00587     0x00056e30, // GFXMMU_LUT270H
```

```
00588     0x005d0401, // GFXMMU_LUT271L
00589     0x000573d0, // GFXMMU_LUT271H
00590     0x005d0401, // GFXMMU_LUT272L
00591     0x00057970, // GFXMMU_LUT272H
00592     0x005d0401, // GFXMMU_LUT273L
00593     0x00057f10, // GFXMMU_LUT273H
00594     0x005d0401, // GFXMMU_LUT274L
00595     0x000584b0, // GFXMMU_LUT274H
00596     0x005d0401, // GFXMMU_LUT275L
00597     0x00058a50, // GFXMMU_LUT275H
00598     0x005c0401, // GFXMMU_LUT276L
00599     0x00058ff0, // GFXMMU_LUT276H
00600     0x005c0401, // GFXMMU_LUT277L
00601     0x00059580, // GFXMMU_LUT277H
00602     0x005c0401, // GFXMMU_LUT278L
00603     0x00059b10, // GFXMMU_LUT278H
00604     0x005c0401, // GFXMMU_LUT279L
00605     0x0005a0a0, // GFXMMU_LUT279H
00606     0x005c0501, // GFXMMU_LUT280L
00607     0x0005a620, // GFXMMU_LUT280H
00608     0x005c0501, // GFXMMU_LUT281L
00609     0x0005aba0, // GFXMMU_LUT281H
00610     0x005c0501, // GFXMMU_LUT282L
00611     0x0005b120, // GFXMMU_LUT282H
00612     0x005c0501, // GFXMMU_LUT283L
00613     0x0005b6a0, // GFXMMU_LUT283H
00614     0x005b0501, // GFXMMU_LUT284L
00615     0x0005bc20, // GFXMMU_LUT284H
00616     0x005b0501, // GFXMMU_LUT285L
00617     0x0005c190, // GFXMMU_LUT285H
00618     0x005b0501, // GFXMMU_LUT286L
00619     0x0005c700, // GFXMMU_LUT286H
00620     0x005b0601, // GFXMMU_LUT287L
00621     0x0005cc60, // GFXMMU_LUT287H
00622     0x005b0601, // GFXMMU_LUT288L
00623     0x0005d1c0, // GFXMMU_LUT288H
00624     0x005b0601, // GFXMMU_LUT289L
```

```
00625     0x0005d720, // GFXMMU_LUT289H
00626     0x005b0601, // GFXMMU_LUT290L
00627     0x0005dc80, // GFXMMU_LUT290H
00628     0x005a0601, // GFXMMU_LUT291L
00629     0x0005e1e0, // GFXMMU_LUT291H
00630     0x005a0601, // GFXMMU_LUT292L
00631     0x0005e730, // GFXMMU_LUT292H
00632     0x005a0601, // GFXMMU_LUT293L
00633     0x0005ec80, // GFXMMU_LUT293H
00634     0x005a0601, // GFXMMU_LUT294L
00635     0x0005f1d0, // GFXMMU_LUT294H
00636     0x005a0701, // GFXMMU_LUT295L
00637     0x0005f710, // GFXMMU_LUT295H
00638     0x005a0701, // GFXMMU_LUT296L
00639     0x0005fc50, // GFXMMU_LUT296H
00640     0x005a0701, // GFXMMU_LUT297L
00641     0x00060190, // GFXMMU_LUT297H
00642     0x00590701, // GFXMMU_LUT298L
00643     0x000606d0, // GFXMMU_LUT298H
00644     0x00590701, // GFXMMU_LUT299L
00645     0x00060c00, // GFXMMU_LUT299H
00646     0x00590701, // GFXMMU_LUT300L
00647     0x00061130, // GFXMMU_LUT300H
00648     0x00590801, // GFXMMU_LUT301L
00649     0x00061650, // GFXMMU_LUT301H
00650     0x00590801, // GFXMMU_LUT302L
00651     0x00061b70, // GFXMMU_LUT302H
00652     0x00590801, // GFXMMU_LUT303L
00653     0x00062090, // GFXMMU_LUT303H
00654     0x00580801, // GFXMMU_LUT304L
00655     0x000625b0, // GFXMMU_LUT304H
00656     0x00580801, // GFXMMU_LUT305L
00657     0x00062ac0, // GFXMMU_LUT305H
00658     0x00580801, // GFXMMU_LUT306L
00659     0x00062fd0, // GFXMMU_LUT306H
00660     0x00580901, // GFXMMU_LUT307L
00661     0x000634d0, // GFXMMU_LUT307H
```

```
00662     0x00580901, // GFXMMU_LUT308L
00663     0x000639d0, // GFXMMU_LUT308H
00664     0x00580901, // GFXMMU_LUT309L
00665     0x00063ed0, // GFXMMU_LUT309H
00666     0x00570901, // GFXMMU_LUT310L
00667     0x000643d0, // GFXMMU_LUT310H
00668     0x00570901, // GFXMMU_LUT311L
00669     0x000648c0, // GFXMMU_LUT311H
00670     0x00570901, // GFXMMU_LUT312L
00671     0x00064db0, // GFXMMU_LUT312H
00672     0x00570a01, // GFXMMU_LUT313L
00673     0x00065290, // GFXMMU_LUT313H
00674     0x00570a01, // GFXMMU_LUT314L
00675     0x00065770, // GFXMMU_LUT314H
00676     0x00560a01, // GFXMMU_LUT315L
00677     0x00065c50, // GFXMMU_LUT315H
00678     0x00560a01, // GFXMMU_LUT316L
00679     0x00066120, // GFXMMU_LUT316H
00680     0x00560b01, // GFXMMU_LUT317L
00681     0x000665e0, // GFXMMU_LUT317H
00682     0x00560b01, // GFXMMU_LUT318L
00683     0x00066aa0, // GFXMMU_LUT318H
00684     0x00560b01, // GFXMMU_LUT319L
00685     0x00066f60, // GFXMMU_LUT319H
00686     0x00550b01, // GFXMMU_LUT320L
00687     0x00067420, // GFXMMU_LUT320H
00688     0x00550b01, // GFXMMU_LUT321L
00689     0x000678d0, // GFXMMU_LUT321H
00690     0x00550c01, // GFXMMU_LUT322L
00691     0x00067d70, // GFXMMU_LUT322H
00692     0x00550c01, // GFXMMU_LUT323L
00693     0x00068210, // GFXMMU_LUT323H
00694     0x00550c01, // GFXMMU_LUT324L
00695     0x000686b0, // GFXMMU_LUT324H
00696     0x00540c01, // GFXMMU_LUT325L
00697     0x00068b50, // GFXMMU_LUT325H
00698     0x00540c01, // GFXMMU_LUT326L
```

```
00699     0x00068fe0, // GFXMMU_LUT326H
00700     0x00540d01, // GFXMMU_LUT327L
00701     0x00069460, // GFXMMU_LUT327H
00702     0x00540d01, // GFXMMU_LUT328L
00703     0x000698e0, // GFXMMU_LUT328H
00704     0x00530d01, // GFXMMU_LUT329L
00705     0x00069d60, // GFXMMU_LUT329H
00706     0x00530d01, // GFXMMU_LUT330L
00707     0x0006a1d0, // GFXMMU_LUT330H
00708     0x00530e01, // GFXMMU_LUT331L
00709     0x0006a630, // GFXMMU_LUT331H
00710     0x00530e01, // GFXMMU_LUT332L
00711     0x0006aa90, // GFXMMU_LUT332H
00712     0x00520e01, // GFXMMU_LUT333L
00713     0x0006aef0, // GFXMMU_LUT333H
00714     0x00520e01, // GFXMMU_LUT334L
00715     0x0006b340, // GFXMMU_LUT334H
00716     0x00520f01, // GFXMMU_LUT335L
00717     0x0006b780, // GFXMMU_LUT335H
00718     0x00520f01, // GFXMMU_LUT336L
00719     0x0006bbc0, // GFXMMU_LUT336H
00720     0x00510f01, // GFXMMU_LUT337L
00721     0x0006c000, // GFXMMU_LUT337H
00722     0x00510f01, // GFXMMU_LUT338L
00723     0x0006c430, // GFXMMU_LUT338H
00724     0x00511001, // GFXMMU_LUT339L
00725     0x0006c850, // GFXMMU_LUT339H
00726     0x00511001, // GFXMMU_LUT340L
00727     0x0006cc70, // GFXMMU_LUT340H
00728     0x00501001, // GFXMMU_LUT341L
00729     0x0006d090, // GFXMMU_LUT341H
00730     0x00501101, // GFXMMU_LUT342L
00731     0x0006d490, // GFXMMU_LUT342H
00732     0x00501101, // GFXMMU_LUT343L
00733     0x0006d890, // GFXMMU_LUT343H
00734     0x004f1101, // GFXMMU_LUT344L
00735     0x0006dc90, // GFXMMU_LUT344H
```

```
00736     0x004f1101, // GFXMMU_LUT345L
00737     0x0006e080, // GFXMMU_LUT345H
00738     0x004f1201, // GFXMMU_LUT346L
00739     0x0006e460, // GFXMMU_LUT346H
00740     0x004e1201, // GFXMMU_LUT347L
00741     0x0006e840, // GFXMMU_LUT347H
00742     0x004e1201, // GFXMMU_LUT348L
00743     0x0006ec10, // GFXMMU_LUT348H
00744     0x004e1301, // GFXMMU_LUT349L
00745     0x0006efd0, // GFXMMU_LUT349H
00746     0x004e1301, // GFXMMU_LUT350L
00747     0x0006f390, // GFXMMU_LUT350H
00748     0x004d1301, // GFXMMU_LUT351L
00749     0x0006f750, // GFXMMU_LUT351H
00750     0x004d1401, // GFXMMU_LUT352L
00751     0x0006faf0, // GFXMMU_LUT352H
00752     0x004c1401, // GFXMMU_LUT353L
00753     0x0006fe90, // GFXMMU_LUT353H
00754     0x004c1401, // GFXMMU_LUT354L
00755     0x00070220, // GFXMMU_LUT354H
00756     0x004c1501, // GFXMMU_LUT355L
00757     0x000705a0, // GFXMMU_LUT355H
00758     0x004b1501, // GFXMMU_LUT356L
00759     0x00070920, // GFXMMU_LUT356H
00760     0x004b1501, // GFXMMU_LUT357L
00761     0x00070c90, // GFXMMU_LUT357H
00762     0x004b1601, // GFXMMU_LUT358L
00763     0x00070ff0, // GFXMMU_LUT358H
00764     0x004a1601, // GFXMMU_LUT359L
00765     0x00071350, // GFXMMU_LUT359H
00766     0x004a1701, // GFXMMU_LUT360L
00767     0x00071690, // GFXMMU_LUT360H
00768     0x00491701, // GFXMMU_LUT361L
00769     0x000719d0, // GFXMMU_LUT361H
00770     0x00491701, // GFXMMU_LUT362L
00771     0x00071d00, // GFXMMU_LUT362H
00772     0x00491801, // GFXMMU_LUT363L
```

```
00773     0x00072020, // GFXMMU_LUT363H
00774     0x00481801, // GFXMMU_LUT364L
00775     0x00072340, // GFXMMU_LUT364H
00776     0x00481901, // GFXMMU_LUT365L
00777     0x00072640, // GFXMMU_LUT365H
00778     0x00471901, // GFXMMU_LUT366L
00779     0x00072940, // GFXMMU_LUT366H
00780     0x00471a01, // GFXMMU_LUT367L
00781     0x00072c20, // GFXMMU_LUT367H
00782     0x00461a01, // GFXMMU_LUT368L
00783     0x00072f00, // GFXMMU_LUT368H
00784     0x00461b01, // GFXMMU_LUT369L
00785     0x000731c0, // GFXMMU_LUT369H
00786     0x00451b01, // GFXMMU_LUT370L
00787     0x00073480, // GFXMMU_LUT370H
00788     0x00451c01, // GFXMMU_LUT371L
00789     0x00073720, // GFXMMU_LUT371H
00790     0x00441c01, // GFXMMU_LUT372L
00791     0x000739c0, // GFXMMU_LUT372H
00792     0x00441d01, // GFXMMU_LUT373L
00793     0x00073c40, // GFXMMU_LUT373H
00794     0x00431d01, // GFXMMU_LUT374L
00795     0x00073ec0, // GFXMMU_LUT374H
00796     0x00431e01, // GFXMMU_LUT375L
00797     0x00074120, // GFXMMU_LUT375H
00798     0x00421f01, // GFXMMU_LUT376L
00799     0x00074370, // GFXMMU_LUT376H
00800     0x00411f01, // GFXMMU_LUT377L
00801     0x000745b0, // GFXMMU_LUT377H
00802     0x00402001, // GFXMMU_LUT378L
00803     0x000747d0, // GFXMMU_LUT378H
00804     0x00402101, // GFXMMU_LUT379L
00805     0x000749d0, // GFXMMU_LUT379H
00806     0x003f2101, // GFXMMU_LUT380L
00807     0x00074bd0, // GFXMMU_LUT380H
00808     0x003e2201, // GFXMMU_LUT381L
00809     0x00074db0, // GFXMMU_LUT381H
```

```
00810     0x003d2301, // GFXMMU_LUT382L
00811     0x00074f70, // GFXMMU_LUT382H
00812     0x003d2401, // GFXMMU_LUT383L
00813     0x00075110, // GFXMMU_LUT383H
00814     0x003c2501, // GFXMMU_LUT384L
00815     0x000752a0, // GFXMMU_LUT384H
00816     0x003b2601, // GFXMMU_LUT385L
00817     0x00075410, // GFXMMU_LUT385H
00818     0x00392701, // GFXMMU_LUT386L
00819     0x00075560, // GFXMMU_LUT386H
00820     0x00382901, // GFXMMU_LUT387L
00821     0x00075670, // GFXMMU_LUT387H
00822     0x00362b01, // GFXMMU_LUT388L
00823     0x00075750, // GFXMMU_LUT388H
00824     0x00342d01, // GFXMMU_LUT389L
00825     0x000757f0  // GFXMMU_LUT389H
00826 };
00827
00828 #ifdef __cplusplus
00829 }
00830 #endif
00831
00832 #endif /* __STM32L4R9I_EVAL_GFXMMU_LUT_H */
00833
00834 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_dsi_ts.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_dsi_ts.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_dsi_ts.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_DSI_TS_H
00039 #define __STM32L4R9I_EVAL_DSI_TS_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-----------------------------------*/
00046 #include "stm32l4r9i_eval.h"
00047
00048 /* Include TouchScreen component driver */
00049 #include "../Components/ft3x67/ft3x67.h"
00050
00051 /** @addtogroup BSP
00052   * @{
00053   */
00054
00055 /** @addtogroup STM32L4R9I_EVAL
00056   * @{
00057   */
00058
00059 /** @defgroup STM32L4R9I_EVAL_TS STM32L4R9I_
EVAL TS
00060   * @{
00061   */
00062
00063 /** @defgroup STM32L4R9I-EVAL_DSI_TS STM32L4
R9I_EVAL DSI TS
00064   * @{
00065   */
00066
```

```
00067 /** @defgroup STM32L4R9I-EVAL_DSI_TS_Exporte
d_Functions Exported Functions
00068   * @{
00069   */
00070
00071 uint8_t BSP_DSI_TS_Init(uint16_t ts_SizeX, u
int16_t ts_SizeY);
00072 uint8_t BSP_DSI_TS_DeInit(void);
00073
00074 uint8_t BSP_DSI_TS_GetState(TS_StateTypeDef
*TS_State);
00075 uint8_t BSP_DSI_TS_GestureConfig(FunctionalS
tate State);
00076 uint8_t BSP_DSI_TS_Get_GestureId(TS_StateTyp
eDef *TS_State);
00077 uint8_t BSP_DSI_TS_ResetTouchData(TS_StateTy
peDef *TS_State);
00078
00079 uint8_t BSP_DSI_TS_ITConfig(void);
00080 uint8_t BSP_DSI_TS_ITGetStatus(void);
00081 void    BSP_DSI_TS_ITClear(void);
00082
00083 /**
00084   * @}
00085   */
00086
00087 /**
00088   * @}
00089   */
00090
00091 /**
00092   * @}
00093   */
00094
00095 /**
00096   * @}
00097   */
```

```
00098
00099 /**
00100   * @}
00101   */
00102
00103 #ifdef __cplusplus
00104 }
00105 #endif
00106
00107 #endif /* __STM32L4R9I_EVAL_DSI_TS_H */
00108
00109 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_eeprom.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_eeprom.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains all the functions prototypes for
00006   *          the stm32l4r9i_eval_eeprom.c firmware driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_EEPROM_H
00039 #define __STM32L4R9I_EVAL_EEPROM_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-------------------------------------*/
00046 #include "stm32l4r9i_eval.h"
00047
00048 /** @addtogroup BSP
00049   * @{
00050   */
00051
00052 /** @addtogroup STM32L4R9I_EVAL
00053   * @{
00054   */
00055
00056 /** @addtogroup STM32L4R9I_EVAL_EEPROM
00057   * @brief This file includes the I2C EEPROM
 driver of STM32L4R9I_EVAL evaluation board.
00058   * @{
00059   */
00060
00061 /** @defgroup STM32L4R9I_EVAL_EEPROM_Exporte
d_Constants Exported Constants
00062   * @{
00063   */
00064 /* EEPROM hardware address and page size */
00065 #define EEPROM_PAGESIZE             ((uint8_
t)4)
```

```
00066 #define EEPROM_MAX_SIZE             ((uint16
_t)0x2000) /* 64Kbit */
00067
00068
00069 /* Maximum number of trials for BSP_EEPROM_W
aitEepromStandbyState() function */
00070 #define EEPROM_MAX_TRIALS           ((uint32
_t)3000)
00071
00072 #define EEPROM_OK                   ((uint32
_t)0)
00073 #define EEPROM_FAIL                 ((uint32
_t)1)
00074 #define EEPROM_TIMEOUT              ((uint32
_t)2)
00075 /**
00076   * @}
00077   */
00078
00079 /** @defgroup STM32L4R9I_EVAL_EEPROM_Exporte
d_Functions Exported Functions
00080   * @{
00081   */
00082 uint32_t BSP_EEPROM_Init(void);
00083 uint8_t  BSP_EEPROM_DeInit(void);
00084 uint32_t BSP_EEPROM_ReadBuffer(uint8_t* pBuf
fer, uint16_t ReadAddr, uint16_t* NumByteToRead);
00085 uint32_t BSP_EEPROM_WritePage(uint8_t* pBuff
er, uint16_t WriteAddr, uint8_t* NumByteToWrite);
00086 uint32_t BSP_EEPROM_WriteBuffer(uint8_t* pBu
ffer, uint16_t WriteAddr, uint16_t NumByteToWrite)
;
00087 uint32_t BSP_EEPROM_WaitEepromStandbyState(v
oid);
00088
00089 /* USER Callbacks: This function is declared
 as __weak in EEPROM driver and
```

```
00090    should be implemented into user applicati
on.
00091    BSP_EEPROM_TIMEOUT_UserCallback() functio
n is called whenever a timeout condition
00092    occurs during communication (waiting on a
n event that doesn't occur, bus
00093    errors, busy devices ...). */
00094 void     BSP_EEPROM_TIMEOUT_UserCallback(void
);
00095
00096 /* Link function for I2C EEPROM peripheral */

00097 void              EEPROM_IO_Init(void);
00098 HAL_StatusTypeDef EEPROM_IO_WriteData(uint16
_t DevAddress, uint16_t MemAddress, uint8_t *pBuff
er, uint32_t BufferSize);
00099 HAL_StatusTypeDef EEPROM_IO_ReadData(uint16_
t DevAddress, uint16_t MemAddress, uint8_t *pBuffe
r, uint32_t BufferSize);
00100 HAL_StatusTypeDef EEPROM_IO_IsDeviceReady(ui
nt16_t DevAddress, uint32_t Trials);
00101
00102 /**
00103   * @}
00104   */
00105
00106 /**
00107   * @}
00108   */
00109
00110 /**
00111   * @}
00112   */
00113
00114 /**
00115   * @}
00116   */
```

```
00117
00118 #ifdef __cplusplus
00119 }
00120 #endif
00121
00122 #endif /* __STM32L4R9I_EVAL_EEPROM_H */
00123
00124 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_eeprom.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_eeprom.c
00004   * @author  MCD Application Team
00005   * @brief   This file provides a set of functions needed to manage an I2C M24LR64
00006   *          EEPROM memory.
00007   @verbatim
00008
00009              ===================================
00010              Notes:
00011               - The I2C EEPROM memory (M24LR64) is available on separate daughter
00012                 board ANT7-M24LR-A, which is not provided with the STM32L4R9I_EVAL
00013                 board.
00014                 To use this driver you have to connect the ANT7-M24LR-A to CN5
00015                 connector of STM32L4R9I_EVAL board.
```

```
00016             ================================
==================================
00017
00018             It implements a high level commu
nication layer for read and write
00019             from/to this memory. The needed
STM32L4xx hardware resources (I2C and
00020             GPIO) are defined in stm32l4r9i_
eval.h file, and the initialization is
00021             performed in EEPROM_IO_Init() fu
nction declared in stm32l4r9i_eval.c
00022             file.
00023             You can easily tailor this drive
r to any other development board,
00024             by just adapting the defines for
 hardware resources and
00025             EEPROM_IO_Init() function.
00026
00027             @note In this driver, basic read
 and write functions (BSP_EEPROM_ReadBuffer()
00028                   and BSP_EEPROM_WritePage()
) use DMA mode to perform the data
00029                   transfer to/from EEPROM me
mory.
00030
00031           @note   Regarding BSP_EEPROM_Writ
ePage(), it is an optimized function to perform
00032                   small write (less than 1 p
age) BUT the number of bytes (combined to write st
art address) must not
00033                   cross the EEPROM page boun
dary. This function can only writes into
00034                   the boundaries of an EEPRO
M page.
00035                   This function doesn't chec
k on boundaries condition (in this driver
00036                   the function BSP_EEPROM_Wr
```

```
iteBuffer() which calls BSP_EEPROM_WritePage() is
00037                   responsible of checking on
 Page boundaries).
00038
00039
00040        +-----------------------------------
-----------------------------+
00041        |               Pin assignment for M2
4LR64 EEPROM                 |
00042        +-----------------------------------
---+-----------+-------------+
00043        |  STM32L4xx I2C Pins
   |   EEPROM  |   Pin       |
00044        +-----------------------------------
---+-----------+-------------+
00045        | .
   |   E0(GND) |    1  (0V)  |
00046        | .
   |   AC0     |    2        |
00047        | .
   |   AC1     |    3        |
00048        | .
   |   VSS     |    4  (0V)  |
00049        | SDA
   |   SDA     |    5        |
00050        | SCL
   |   SCL     |    6        |
00051        | .
   |   E1(GND) |    7  (0V)  |
00052        | .
   |   VDD     |    8 (3.3V) |
00053        +-----------------------------------
---+-----------+-------------+
00054   @endverbatim
00055   ******************************************
*********************************
00056   * @attention
```

```
00057   *
00058   * <h2><center>&copy; COPYRIGHT(c) 2017 STM
icroelectronics</center></h2>
00059   *
00060   * Redistribution and use in source and bin
ary forms, with or without modification,
00061   * are permitted provided that the followin
g conditions are met:
00062   *   1. Redistributions of source code must
 retain the above copyright notice,
00063   *      this list of conditions and the fol
lowing disclaimer.
00064   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00065   *      this list of conditions and the fol
lowing disclaimer in the documentation
00066   *      and/or other materials provided wit
h the distribution.
00067   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00068   *      may be used to endorse or promote p
roducts derived from this software
00069   *      without specific prior written perm
ission.
00070   *
00071   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00072   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00073   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00074   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00075   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00076   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 PROCUREMENT OF SUBSTITUTE GOODS OR
```

```
00077   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00078   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00079   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00080   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00081   *
00082   ****************************************
**********************************
00083   */
00084 /* Includes ------------------------------
-----------------------------------*/
00085 #include "stm32l4r9i_eval_eeprom.h"
00086
00087 /** @addtogroup BSP
00088   * @{
00089   */
00090
00091 /** @addtogroup STM32L4R9I_EVAL
00092   * @{
00093   */
00094
00095 /** @defgroup STM32L4R9I_EVAL_EEPROM STM32L4
R9I_EVAL EEPROM
00096   * @brief This file includes the I2C EEPROM
 driver of STM32L4R9I-EVAL evaluation board.
00097   * @{
00098   */
00099
00100 /** @defgroup STM32L4R9I_EVAL_EEPROM_Private
_Variables Private Variables
00101   * @{
00102   */
00103 __IO uint16_t EEPROMAddress = 0;
00104 __IO uint16_t EEPROMDataRead;
```

```
00105 __IO uint8_t  EEPROMDataWrite;
00106 /**
00107   * @}
00108   */
00109
00110 /** @defgroup STM32L4R9I_EVAL_EEPROM_Exporte
d_Functions Exported Functions
00111   * @{
00112   */
00113
00114 /**
00115   * @brief  Initializes peripherals used by
the I2C EEPROM driver.
00116   * @retval EEPROM_OK (0) if operation is co
rrectly performed, else return value
00117   *         different from EEPROM_OK (0)
00118   */
00119 uint32_t BSP_EEPROM_Init(void)
00120 {
00121   /* I2C Initialization */
00122   EEPROM_IO_Init();
00123
00124   /* Select the EEPROM address check if OK */

00125   EEPROMAddress = EEPROM_I2C_ADDRESS;
00126   if(EEPROM_IO_IsDeviceReady(EEPROMAddress,
EEPROM_MAX_TRIALS) != HAL_OK)
00127   {
00128     return EEPROM_FAIL;
00129   }
00130   return EEPROM_OK;
00131 }
00132
00133 /**
00134   * @brief  DeInitializes the EEPROM.
00135   * @retval EEPROM state
00136   */
```

```
00137 uint8_t BSP_EEPROM_DeInit(void)
00138 {
00139   /* I2C is not disabled because common to o
ther functionalities */
00140   return EEPROM_OK;
00141 }
00142
00143 /**
00144   * @brief  Reads a block of data from the E
EPROM.
00145   * @param  pBuffer: pointer to the buffer t
hat receives the data read from
00146   *         the EEPROM.
00147   * @param  ReadAddr: EEPROM's internal addr
ess to start reading from.
00148   * @param  NumByteToRead: pointer to the va
riable holding number of bytes to
00149   *         be read from the EEPROM.
00150   *
00151   *        @note The variable pointed by Num
ByteToRead is reset to 0 when all the
00152   *              data are read from the EEPR
OM. Application should monitor this
00153   *              variable in order know when
 the transfer is complete.
00154   *
00155   * @retval EEPROM_OK (0) if operation is co
rrectly performed, else return value
00156   *         different from EEPROM_OK (0) or
the timeout user callback.
00157   */
00158 uint32_t BSP_EEPROM_ReadBuffer(uint8_t* pBuf
fer, uint16_t ReadAddr, uint16_t* NumByteToRead)
00159 {
00160   uint32_t buffersize = *NumByteToRead;
00161
00162   /* Set the pointer to the Number of data t
```

```
o be read. This pointer will be used
00163      by the DMA Transfer Completer interrupt
 Handler in order to reset the
00164      variable to 0. User should check on thi
s variable in order to know if the
00165      DMA transfer has been complete or not.
*/
00166   EEPROMDataRead = *NumByteToRead;
00167
00168   if(EEPROM_IO_ReadData(EEPROMAddress, ReadA
ddr, pBuffer, buffersize) != HAL_OK)
00169   {
00170     BSP_EEPROM_TIMEOUT_UserCallback();
00171     return EEPROM_FAIL;
00172   }
00173
00174   /* If all operations OK, return EEPROM_OK
(0) */
00175   return EEPROM_OK;
00176 }
00177
00178 /**
00179   * @brief  Writes more than one byte to the
 EEPROM with a single WRITE cycle.
00180   *
00181   * @note   The number of bytes (combined to
 write start address) must not
00182   *         cross the EEPROM page boundary.
This function can only write into
00183   *         the boundaries of an EEPROM page.

00184   *         This function doesn't check on b
oundaries condition (in this driver
00185   *         the function BSP_EEPROM_WriteBuf
fer() which calls BSP_EEPROM_WritePage() is
00186   *         responsible of checking on Page
boundaries).
```

```
00187   *
00188   * @param  pBuffer: pointer to the buffer c
ontaining the data to be written to
00189   *         the EEPROM.
00190   * @param  WriteAddr: EEPROM's internal add
ress to write to.
00191   * @param  NumByteToWrite: pointer to the v
ariable holding number of bytes to
00192   *         be written into the EEPROM.
00193   *
00194   *        @note The variable pointed by Num
ByteToWrite is reset to 0 when all the
00195   *              data are written to the EEP
ROM. Application should monitor this
00196   *              variable in order know when
 the transfer is complete.
00197   *
00198   *        @note This function just configur
e the communication and enable the DMA
00199   *              channel to transfer data. M
eanwhile, the user application may perform
00200   *              other tasks in parallel.
00201   *
00202   * @retval EEPROM_OK (0) if operation is co
rrectly performed, else return value
00203   *         different from EEPROM_OK (0) or
the timeout user callback.
00204   */
00205 uint32_t BSP_EEPROM_WritePage(uint8_t* pBuff
er, uint16_t WriteAddr, uint8_t* NumByteToWrite)
00206 {
00207   uint32_t buffersize = *NumByteToWrite;
00208   uint32_t status = EEPROM_OK;
00209
00210   /* Set the pointer to the Number of data t
o be written. This pointer will be used
00211       by the DMA Transfer Completer interrup
```

```
t Handler in order to reset the
00212       variable to 0. User should check on th
is variable in order to know if the
00213       DMA transfer has been complete or not.
 */
00214   EEPROMDataWrite = *NumByteToWrite;
00215
00216   if(EEPROM_IO_WriteData(EEPROMAddress, Writ
eAddr, pBuffer, buffersize) != HAL_OK)
00217   {
00218     BSP_EEPROM_TIMEOUT_UserCallback();
00219     status = EEPROM_FAIL;
00220   }
00221
00222   if(BSP_EEPROM_WaitEepromStandbyState() !=
EEPROM_OK)
00223   {
00224     return EEPROM_FAIL;
00225   }
00226
00227   /* If all operations OK, return EEPROM_OK
(0) */
00228   return status;
00229 }
00230
00231 /**
00232   * @brief  Writes buffer of data to the I2C
 EEPROM.
00233   * @param  pBuffer: pointer to the buffer
containing the data to be written
00234   *         to the EEPROM.
00235   * @param  WriteAddr: EEPROM's internal add
ress to write to.
00236   * @param  NumByteToWrite: number of bytes
to write to the EEPROM.
00237   * @retval EEPROM_OK (0) if operation is co
rrectly performed, else return value
```

```
00238   *         different from EEPROM_OK (0) or
the timeout user callback.
00239   */
00240 uint32_t BSP_EEPROM_WriteBuffer(uint8_t *pBu
ffer, uint16_t WriteAddr, uint16_t NumByteToWrite)
00241 {
00242   uint16_t numofpage = 0, numofsingle = 0, c
ount = 0;
00243   uint16_t addr = 0;
00244   uint8_t  dataindex = 0;
00245   uint32_t status = EEPROM_OK;
00246
00247   addr = WriteAddr % EEPROM_PAGESIZE;
00248   count = EEPROM_PAGESIZE - addr;
00249   numofpage =  NumByteToWrite / EEPROM_PAGES
IZE;
00250   numofsingle = NumByteToWrite % EEPROM_PAGE
SIZE;
00251
00252   /* If WriteAddr is EEPROM_PAGESIZE aligned
 */
00253   if(addr == 0)
00254   {
00255     /* If NumByteToWrite < EEPROM_PAGESIZE */

00256     if(numofpage == 0)
00257     {
00258       /* Store the number of data to be writ
ten */
00259       dataindex = numofsingle;
00260       /* Start writing data */
00261       status = BSP_EEPROM_WritePage(pBuffer,
 WriteAddr, (uint8_t*)(&dataindex));
00262       if(status != EEPROM_OK)
00263       {
00264         return status;
00265       }
```

```
00266     }
00267     /* If NumByteToWrite > EEPROM_PAGESIZE */

00268     else
00269     {
00270       while(numofpage--)
00271       {
00272         /* Store the number of data to be wr
itten */
00273         dataindex = EEPROM_PAGESIZE;
00274         status = BSP_EEPROM_WritePage(pBuffe
r, WriteAddr, (uint8_t*)(&dataindex));
00275         if(status != EEPROM_OK)
00276         {
00277           return status;
00278         }
00279
00280         WriteAddr +=  EEPROM_PAGESIZE;
00281         pBuffer += EEPROM_PAGESIZE;
00282       }
00283
00284       if(numofsingle!=0)
00285       {
00286         /* Store the number of data to be wr
itten */
00287         dataindex = numofsingle;
00288         status = BSP_EEPROM_WritePage(pBuffe
r, WriteAddr, (uint8_t*)(&dataindex));
00289         if(status != EEPROM_OK)
00290         {
00291           return status;
00292         }
00293       }
00294     }
00295   }
00296   /* If WriteAddr is not EEPROM_PAGESIZE ali
gned */
```

```
00297   else
00298   {
00299     /* If NumByteToWrite < EEPROM_PAGESIZE */

00300     if(numofpage== 0)
00301     {
00302       /* If the number of data to be written
 is more than the remaining space
00303       in the current page: */
00304       if(NumByteToWrite > count)
00305       {
00306         /* Store the number of data to be wr
itten */
00307         dataindex = count;
00308         /* Write the data contained in same
page */
00309         status = BSP_EEPROM_WritePage(pBuffe
r, WriteAddr, (uint8_t*)(&dataindex));
00310         if(status != EEPROM_OK)
00311         {
00312           return status;
00313         }
00314
00315         /* Store the number of data to be wr
itten */
00316         dataindex = (NumByteToWrite - count)
;
00317         /* Write the remaining data in the f
ollowing page */
00318         status = BSP_EEPROM_WritePage((uint8
_t*)(pBuffer + count), (WriteAddr + count), (uint8
_t*)(&dataindex));
00319         if(status != EEPROM_OK)
00320         {
00321           return status;
00322         }
00323       }
```

```
00324       else
00325       {
00326         /* Store the number of data to be wr
itten */
00327         dataindex = numofsingle;
00328         status = BSP_EEPROM_WritePage(pBuffe
r, WriteAddr, (uint8_t*)(&dataindex));
00329         if(status != EEPROM_OK)
00330         {
00331           return status;
00332         }
00333       }
00334     }
00335     /* If NumByteToWrite > EEPROM_PAGESIZE */

00336     else
00337     {
00338       NumByteToWrite -= count;
00339       numofpage =  NumByteToWrite / EEPROM_P
AGESIZE;
00340       numofsingle = NumByteToWrite % EEPROM_
PAGESIZE;
00341
00342       if(count != 0)
00343       {
00344         /* Store the number of data to be wr
itten */
00345         dataindex = count;
00346         status = BSP_EEPROM_WritePage(pBuffe
r, WriteAddr, (uint8_t*)(&dataindex));
00347         if(status != EEPROM_OK)
00348         {
00349           return status;
00350         }
00351         WriteAddr += count;
00352         pBuffer += count;
00353       }
```

```
00354
00355       while(numofpage--)
00356       {
00357         /* Store the number of data to be wr
itten */
00358         dataindex = EEPROM_PAGESIZE;
00359         status = BSP_EEPROM_WritePage(pBuffe
r, WriteAddr, (uint8_t*)(&dataindex));
00360         if(status != EEPROM_OK)
00361         {
00362           return status;
00363         }
00364         WriteAddr +=  EEPROM_PAGESIZE;
00365         pBuffer += EEPROM_PAGESIZE;
00366       }
00367       if(numofsingle != 0)
00368       {
00369         /* Store the number of data to be wr
itten */
00370         dataindex = numofsingle;
00371         status = BSP_EEPROM_WritePage(pBuffe
r, WriteAddr, (uint8_t*)(&dataindex));
00372         if(status != EEPROM_OK)
00373         {
00374           return status;
00375         }
00376       }
00377     }
00378   }
00379
00380   /* If all operations OK, return EEPROM_OK
(0) */
00381   return EEPROM_OK;
00382 }
00383
00384 /**
00385   * @brief  Wait for EEPROM Standby state.
```

```
00386   *
00387   * @note  This function allows to wait and
check that EEPROM has finished the
00388   *        last operation. It is mostly used
 after Write operation: after receiving
00389   *        the buffer to be written, the EEP
ROM may need additional time to actually
00390   *        perform the write operation. Duri
ng this time, it doesn't answer to
00391   *        I2C packets addressed to it. Once
 the write operation is complete
00392   *        the EEPROM responds to its addres
s.
00393   *
00394   * @retval EEPROM_OK (0) if operation is co
rrectly performed, else return value
00395   *         different from EEPROM_OK (0) or
the timeout user callback.
00396   */
00397 uint32_t BSP_EEPROM_WaitEepromStandbyState(v
oid)
00398 {
00399   /* Check if the maximum allowed number of
trials has bee reached */
00400   if(EEPROM_IO_IsDeviceReady(EEPROMAddress,
EEPROM_MAX_TRIALS) != HAL_OK)
00401   {
00402     /* If the maximum number of trials has b
een reached, exit the function */
00403     BSP_EEPROM_TIMEOUT_UserCallback();
00404     return EEPROM_TIMEOUT;
00405   }
00406   return EEPROM_OK;
00407 }
00408
00409 /**
00410   * @brief  Basic management of the timeout
```

```
situation.
00411   * @retval None
00412   */
00413 __weak void BSP_EEPROM_TIMEOUT_UserCallback(
void)
00414 {
00415 }
00416
00417 /**
00418   * @}
00419   */
00420
00421 /**
00422   * @}
00423   */
00424
00425 /**
00426   * @}
00427   */
00428
00429 /**
00430   * @}
00431   */
00432
00433 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_idd.h

Go to the documentation of this file.

```
00001  /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_idd.h
00004   * @author  MCD Application Team
00005   * @brief   Header file for stm32l4r9i_eval_idd.c module.
00006   ******************************************************************************
00007   * @attention
00008   *
00009   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00010   *
00011   * Redistribution and use in source and binary forms, with or without modification,
00012   * are permitted provided that the following conditions are met:
00013   *   1. Redistributions of source code must retain the above copyright notice,
00014   *      this list of conditions and the following disclaimer.
```

```
00015   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00016   *      this list of conditions and the fol
lowing disclaimer in the documentation
00017   *      and/or other materials provided wit
h the distribution.
00018   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00019   *      may be used to endorse or promote p
roducts derived from this software
00020   *      without specific prior written perm
ission.
00021   *
00022   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00023   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00024   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00025   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00026   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00027   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00028   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00029   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00030   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00031   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00032   *
00033   ****************************************
**********************************
00034   */
```

```
00035
00036 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00037 #ifndef __STM32L4R9I_EVAL_IDD_H
00038 #define __STM32L4R9I_EVAL_IDD_H
00039
00040 #ifdef __cplusplus
00041  extern "C" {
00042 #endif
00043
00044 /* Includes ------------------------------
-------------------------------------*/
00045 #include "stm32l4r9i_eval.h"
00046 /* Include Idd measurement component driver
*/
00047 #include "../Components/mfxstm32l152/mfxstm3
2l152.h"
00048
00049 /** @addtogroup BSP
00050   * @{
00051   */
00052
00053 /** @addtogroup STM32L4R9I_EVAL
00054   * @{
00055   */
00056
00057 /** @addtogroup STM32L4R9I_EVAL_IDD
00058   * @{
00059   */
00060
00061 /** @defgroup STM32L4R9I_EVAL_IDD_Exported_D
efines  Exported Defines
00062   * @{
00063   */
00064 /**
00065   * @brief  IDD Status values
00066   */
```

```
00067 typedef enum
00068 {
00069   IDD_OK = 0,
00070   IDD_TIMEOUT = 1,
00071   IDD_ZERO_VALUE = 2,
00072   IDD_ERROR = 0xFF
00073 }
00074 IDD_StatusTypeDef;
00075
00076 /**
00077   * @brief  Shunt values on evaluation board
 in milli ohms
00078   */
00079 #define EVAL_IDD_SHUNT0_VALUE
   ((uint16_t) 1000)     /*!< value in milliohm */
00080 #define EVAL_IDD_SHUNT1_VALUE
   ((uint16_t) 24)       /*!< value in ohm */
00081 #define EVAL_IDD_SHUNT2_VALUE
   ((uint16_t) 620)      /*!< value in ohm */
00082 #define EVAL_IDD_SHUNT4_VALUE
   ((uint16_t) 10000)    /*!< value in ohm */
00083
00084 /**
00085   * @brief  Shunt stabilization delay on eva
luation board in milli ohms
00086   */
00087 #define EVAL_IDD_SHUNT0_STABDELAY
   ((uint8_t) 149)       /*!< value in millisec */
00088 #define EVAL_IDD_SHUNT1_STABDELAY
   ((uint8_t) 149)       /*!< value in millisec */
00089 #define EVAL_IDD_SHUNT2_STABDELAY
   ((uint8_t) 149)       /*!< value in millisec */
00090 #define EVAL_IDD_SHUNT4_STABDELAY
   ((uint8_t) 255)       /*!< value in millisec */
00091
00092 /**
00093   * @brief  IDD Ampli Gain on evaluation boa
```

```
rd
00094   */
00095 #define EVAL_IDD_AMPLI_GAIN
   ((uint16_t) 4967)     /*!< value is gain * 100
*/
00096
00097 /**
00098   * @brief  IDD Vdd Min on evaluation board
00099   */
00100 #define EVAL_IDD_VDD_MIN
   ((uint16_t) 1710)     /*!< value in millivolt */

00101
00102 /**
00103   * @}
00104   */
00105
00106 /* Exported functions ----------------------
-----------------------------------*/
00107 /** @defgroup STM32L4R9I_EVAL_IDD_Exported_F
unctions  Exported Functions
00108   * @{
00109   */
00110 uint8_t   BSP_IDD_Init(void);
00111 void      BSP_IDD_DeInit(void);
00112 void      BSP_IDD_Reset(void);
00113 void      BSP_IDD_LowPower(void);
00114 void      BSP_IDD_WakeUp(void);
00115 void      BSP_IDD_StartMeasure(void);
00116 void      BSP_IDD_Config(IDD_ConfigTypeDef I
ddConfig);
00117 void      BSP_IDD_GetValue(uint32_t *IddValu
e);
00118 void      BSP_IDD_EnableIT(void);
00119 void      BSP_IDD_ClearIT(void);
00120 uint8_t   BSP_IDD_GetITStatus(void);
00121 void      BSP_IDD_DisableIT(void);
```

```
00122 uint8_t   BSP_IDD_ErrorGetCode(void);
00123 void      BSP_IDD_ErrorEnableIT(void);
00124 void      BSP_IDD_ErrorClearIT(void);
00125 uint8_t   BSP_IDD_ErrorGetITStatus(void);
00126 void      BSP_IDD_ErrorDisableIT(void);
00127
00128 /**
00129   * @}
00130   */
00131
00132 /**
00133   * @}
00134   */
00135
00136 /**
00137   * @}
00138   */
00139
00140 /**
00141   * @}
00142   */
00143
00144 #ifdef __cplusplus
00145 }
00146 #endif
00147
00148 #endif /* __STM32L4R9I_EVAL_IDD_H */
00149
00150 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_idd.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_idd.c
00004   * @author  MCD Application Team
00005   * @brief   This file provides a set of firmware functions to manage the
00006   *          Idd measurement driver for STM32L4R9I_EVAL board.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Includes ------------------------------
-----------------------------------*/
00038 #include "stm32l4r9i_eval_idd.h"
00039 #include "stm32l4r9i_eval_io.h"
00040
00041 /** @addtogroup BSP
00042   * @{
00043   */
00044
00045 /** @addtogroup STM32L4R9I_EVAL
00046   * @{
00047   */
00048
00049 /** @defgroup STM32L4R9I_EVAL_IDD STM32L4R9I
_EVAL IDD
00050   * @brief This file includes the Idd driver
 for STM32L4R9I_EVAL board.
00051   *        It allows user to measure MCU Idd
 current on board, especially in
00052   *        different low power modes.
00053   * @{
00054   */
00055
00056 /** @defgroup STM32L4R9I_EVAL_IDD_Private_Va
riables Private Variables
00057   * @{
00058   */
00059 static IDD_DrvTypeDef *IddDrv = NULL;
00060
00061 /**
00062   * @}
00063   */
00064
00065 /** @defgroup STM32L4R9I_EVAL_IDD_Exported_F
```

```
unctions Exported Functions
00066   * @{
00067   */
00068
00069 /**
00070   * @brief  Configures IDD measurement compo
nent.
00071   * @retval IDD_OK if no problem during init
ialization
00072   */
00073 uint8_t BSP_IDD_Init(void)
00074 {
00075   IDD_ConfigTypeDef iddconfig = {0};
00076   uint8_t mfxstm32l152_id = 0;
00077   uint8_t ret = 0;
00078
00079   if (BSP_IO_Init() == IO_ERROR)
00080   {
00081     BSP_ErrorHandler();
00082   }
00083
00084   /* wake up mfx component in case it went t
o standby mode */
00085   mfxstm32l152_idd_drv.WakeUp(IDD_I2C_ADDRESS
);
00086   HAL_Delay(5);
00087
00088   /* Read ID and verify if the MFX is ready
*/
00089   mfxstm32l152_id = mfxstm32l152_idd_drv.Rea
dID(IDD_I2C_ADDRESS);
00090
00091   if((mfxstm32l152_id == MFXSTM32L152_ID_1)
|| (mfxstm32l152_id == MFXSTM32L152_ID_2))
00092   {
00093     /* Initialize the Idd driver structure */
```

```
00094     IddDrv = &mfxstm32l152_idd_drv;
00095
00096     /* Initialize the Idd driver */
00097     if(IddDrv->Init != NULL)
00098     {
00099       IddDrv->Init(IDD_I2C_ADDRESS);
00100     }
00101
00102     /* Configure Idd component with default
values */
00103     iddconfig.AmpliGain = EVAL_IDD_AMPLI_GAIN
;
00104     iddconfig.VddMin = EVAL_IDD_VDD_MIN;
00105     iddconfig.Shunt0Value = EVAL_IDD_SHUNT0_
VALUE;
00106     iddconfig.Shunt1Value = EVAL_IDD_SHUNT1_
VALUE;
00107     iddconfig.Shunt2Value = EVAL_IDD_SHUNT2_
VALUE;
00108     iddconfig.Shunt3Value = 0;
00109     iddconfig.Shunt4Value = EVAL_IDD_SHUNT4_
VALUE;
00110     iddconfig.Shunt0StabDelay = EVAL_IDD_SHU
NT0_STABDELAY;
00111     iddconfig.Shunt1StabDelay = EVAL_IDD_SHU
NT1_STABDELAY;
00112     iddconfig.Shunt2StabDelay = EVAL_IDD_SHU
NT2_STABDELAY;
00113     iddconfig.Shunt3StabDelay = 0;
00114     iddconfig.Shunt4StabDelay = EVAL_IDD_SHU
NT4_STABDELAY;
00115     iddconfig.ShuntNbOnBoard = MFXSTM32L152_
IDD_SHUNT_NB_4;
00116     iddconfig.ShuntNbUsed = MFXSTM32L152_IDD
_SHUNT_NB_4;
00117     iddconfig.VrefMeasurement = MFXSTM32L152
_IDD_VREF_AUTO_MEASUREMENT_ENABLE;
```

```
00118     iddconfig.Calibration = MFXSTM32L152_IDD
_AUTO_CALIBRATION_ENABLE;
00119     iddconfig.PreDelayUnit = MFXSTM32L152_ID
D_PREDELAY_20_MS;
00120     iddconfig.PreDelayValue = 0x7F;
00121     iddconfig.MeasureNb = 100;
00122     iddconfig.DeltaDelayUnit= MFXSTM32L152_I
DD_DELTADELAY_0_5_MS;
00123     iddconfig.DeltaDelayValue = 10;
00124     BSP_IDD_Config(iddconfig);
00125
00126     ret = IDD_OK;
00127   }
00128   else
00129   {
00130     ret = IDD_ERROR;
00131   }
00132
00133   return ret;
00134 }
00135
00136 /**
00137   * @brief  Unconfigures IDD measurement com
ponent.
00138   * @retval IDD_OK if no problem during dein
itialization
00139   */
00140 void BSP_IDD_DeInit(void)
00141 {
00142   if(IddDrv->DeInit!= NULL)
00143   {
00144     IddDrv->DeInit(IDD_I2C_ADDRESS);
00145   }
00146 }
00147
00148 /**
00149   * @brief  Reset Idd measurement component.
```

```
00150   * @retval None
00151   */
00152 void BSP_IDD_Reset(void)
00153 {
00154   if(IddDrv->Reset != NULL)
00155   {
00156     IddDrv->Reset(IDD_I2C_ADDRESS);
00157   }
00158 }
00159
00160 /**
00161   * @brief  Turn Idd measurement component i
n low power (standby/sleep) mode
00162   * @retval None
00163   */
00164 void BSP_IDD_LowPower(void)
00165 {
00166   if(IddDrv->LowPower != NULL)
00167   {
00168     IddDrv->LowPower(IDD_I2C_ADDRESS);
00169   }
00170 }
00171
00172 /**
00173   * @brief  Start Measurement campaign
00174   * @retval None
00175   */
00176 void BSP_IDD_StartMeasure(void)
00177 {
00178
00179   /* Activate the OPAMP used ny the MFX to m
easure the current consumption */
00180   BSP_IO_ConfigPin(IDD_AMP_CONTROL_PIN, IO_M
ODE_OUTPUT);
00181   BSP_IO_WritePin(IDD_AMP_CONTROL_PIN, GPIO_
PIN_RESET);
00182
```

```
00183   if(IddDrv->Start != NULL)
00184   {
00185     IddDrv->Start(IDD_I2C_ADDRESS);
00186   }
00187 }
00188
00189 /**
00190   * @brief  Configure Idd component
00191   * @param  IddConfig: structure of idd para
meters
00192   * @retval None
00193   */
00194 void BSP_IDD_Config(IDD_ConfigTypeDef IddCon
fig)
00195 {
00196   if(IddDrv->Config != NULL)
00197   {
00198     IddDrv->Config(IDD_I2C_ADDRESS, IddConfi
g);
00199   }
00200 }
00201
00202 /**
00203   * @brief  Get Idd current value.
00204   * @param  IddValue: Pointer on u32 to stor
e Idd. Value unit is 10 nA.
00205   * @retval None
00206   */
00207 void BSP_IDD_GetValue(uint32_t *IddValue)
00208 {
00209   /* De-activate the OPAMP used ny the MFX t
o measure the current consumption */
00210   BSP_IO_ConfigPin(IDD_AMP_CONTROL_PIN, IO_M
ODE_INPUT);
00211
00212   if(IddDrv->GetValue != NULL)
00213   {
```

```
00214     IddDrv->GetValue(IDD_I2C_ADDRESS, IddVal
ue);
00215   }
00216 }
00217
00218 /**
00219   * @brief  Enable Idd interrupt that warn e
nd of measurement
00220   * @retval None
00221   */
00222 void BSP_IDD_EnableIT(void)
00223 {
00224   if(IddDrv->EnableIT != NULL)
00225   {
00226     IddDrv->EnableIT(IDD_I2C_ADDRESS);
00227   }
00228 }
00229
00230 /**
00231   * @brief  Clear Idd interrupt that warn en
d of measurement
00232   * @retval None
00233   */
00234 void BSP_IDD_ClearIT(void)
00235 {
00236   if(IddDrv->ClearIT != NULL)
00237   {
00238     IddDrv->ClearIT(IDD_I2C_ADDRESS);
00239   }
00240 }
00241
00242 /**
00243   * @brief  Get Idd interrupt status
00244   * @retval status
00245   */
00246 uint8_t BSP_IDD_GetITStatus(void)
00247 {
```

```
00248   if(IddDrv->GetITStatus != NULL)
00249   {
00250     return (IddDrv->GetITStatus(IDD_I2C_ADDR
ESS));
00251   }
00252   else
00253   {
00254     return IDD_ERROR;
00255   }
00256 }
00257
00258 /**
00259   * @brief  Disable Idd interrupt that warn
end of measurement
00260   * @retval None
00261   */
00262 void BSP_IDD_DisableIT(void)
00263 {
00264   if(IddDrv->DisableIT != NULL)
00265   {
00266     IddDrv->DisableIT(IDD_I2C_ADDRESS);
00267   }
00268 }
00269
00270 /**
00271   * @brief  Get Error Code .
00272   * @retval Error code or error status
00273   */
00274 uint8_t BSP_IDD_ErrorGetCode(void)
00275 {
00276   if(IddDrv->ErrorGetSrc != NULL)
00277   {
00278     if((IddDrv->ErrorGetSrc(IDD_I2C_ADDRESS)
 & MFXSTM32L152_IDD_ERROR_SRC) != RESET)
00279     {
00280       if(IddDrv->ErrorGetCode != NULL)
00281       {
```

```
00282         return IddDrv->ErrorGetCode(IDD_I2C_A
DDRESS);
00283       }
00284       else
00285       {
00286         return IDD_ERROR;
00287       }
00288     }
00289     else
00290     {
00291       return IDD_ERROR;
00292     }
00293   }
00294   else
00295   {
00296     return IDD_ERROR;
00297   }
00298 }
00299
00300
00301 /**
00302   * @brief  Enable error interrupt that warn
 end of measurement
00303   * @retval None
00304   */
00305 void BSP_IDD_ErrorEnableIT(void)
00306 {
00307   if(IddDrv->ErrorEnableIT != NULL)
00308   {
00309     IddDrv->ErrorEnableIT(IDD_I2C_ADDRESS);
00310   }
00311 }
00312
00313 /**
00314   * @brief  Clear Error interrupt that warn
end of measurement
00315   * @retval None
```

```
00316   */
00317 void BSP_IDD_ErrorClearIT(void)
00318 {
00319   if(IddDrv->ErrorClearIT != NULL)
00320   {
00321     IddDrv->ErrorClearIT(IDD_I2C_ADDRESS);
00322   }
00323 }
00324
00325 /**
00326   * @brief  Get Error interrupt status
00327   * @retval Status
00328   */
00329 uint8_t BSP_IDD_ErrorGetITStatus(void)
00330 {
00331   if(IddDrv->ErrorGetITStatus != NULL)
00332   {
00333     return (IddDrv->ErrorGetITStatus(IDD_I2C
_ADDRESS));
00334   }
00335   else
00336   {
00337     return 0;
00338   }
00339 }
00340
00341 /**
00342   * @brief  Disable Error interrupt
00343   * @retval None
00344   */
00345 void BSP_IDD_ErrorDisableIT(void)
00346 {
00347   if(IddDrv->ErrorDisableIT != NULL)
00348   {
00349     IddDrv->ErrorDisableIT(IDD_I2C_ADDRESS);
00350   }
00351 }
```

```
00352
00353 /**
00354   * @brief  Wake up Idd measurement componen
t.
00355   * @retval None
00356   */
00357 void BSP_IDD_WakeUp(void)
00358 {
00359   if(IddDrv->WakeUp != NULL)
00360   {
00361     IddDrv->WakeUp(IDD_I2C_ADDRESS);
00362   }
00363 }
00364
00365 /**
00366   * @}
00367   */
00368
00369 /**
00370   * @}
00371   */
00372
00373 /**
00374   * @}
00375   */
00376
00377 /**
00378   * @}
00379   */
00380
00381 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
00382
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_io.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_io.c
00004   * @author  MCD Application Team
00005   * @brief   This file provides a set of functions needed to manage the IO pins
00006   *          on STM32L4R9I_EVAL evaluation board.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
************************************
00035   */
00036
00037 /* File Info : ------------------------------
-------------------------------------
00038                                         User NOTES

00039 1. How To use this driver:
00040 --------------------------
00041    - This driver is used to drive the IO mod
ule of the STM32L4R9I_EVAL evaluation
00042      board.
00043    - The MFXSTM32L152 IO expander device com
ponent driver must be included with this
00044      driver in order to run the IO functiona
lities commanded by the IO expander
00045      device mounted on the evaluation board.
00046
00047 2. Driver description:
00048 ---------------------
00049   + Initialization steps:
00050      o Initialize the IO module using the BS
P_IO_Init() function. This
00051        function includes the MSP layer hardw
are resources initialization and the
00052        communication layer configuration to
start the IO functionalities use.
00053
00054   + IO functionalities use
00055      o The IO pin mode is configured when ca
lling the function BSP_IO_ConfigPin(), you
00056        must specify the desired IO mode by c
hoosing the "IO_ModeTypedef" parameter
00057        predefined value.
00058      o If an IO pin is used in interrupt mod
e, the function BSP_IO_ITGetStatus() is
00059        needed to get the interrupt status. T
```

```
o clear the IT pending bits, you should
00060        call the function BSP_IO_ITClear() wi
th specifying the IO pending bit to clear.
00061      o The IT is handled using the correspon
ding external interrupt IRQ handler,
00062        the user IT callback treatment is imp
lemented on the same external interrupt
00063        callback.
00064      o To get/set an IO pin combination stat
e you can use the functions
00065        BSP_IO_ReadPin()/BSP_IO_WritePin() or
 the function BSP_IO_TogglePin() to toggle the pin
00066        state.
00067
00068 ----------------------------------------
----------------------------------*/
00069
00070 /* Includes ------------------------------
----------------------------------*/
00071 #include "stm32l4r9i_eval_io.h"
00072
00073 /** @addtogroup BSP
00074   * @{
00075   */
00076
00077 /** @addtogroup STM32L4R9I_EVAL
00078   * @{
00079   */
00080
00081 /** @defgroup STM32L4R9I_EVAL_IO STM32L4R9I_
EVAL IO
00082   * @{
00083   */
00084
00085 /* Private constants ---------------------
----------------------------------*/
00086 /* Private macros ------------------------
```

```
----------------------------------*/
00087 /* Private variables ---------------------
----------------------------------*/
00088
00089 /** @defgroup STM32L4R9I_EVAL_IO_Private_Var
iables Private Variables
00090   * @{
00091   */
00092 static IO_DrvTypeDef *io_driver;
00093
00094 /**
00095   * @}
00096   */
00097
00098 /* Private function prototypes -------------
----------------------------------*/
00099 /* Exported functions ----------------------
----------------------------------*/
00100
00101 /** @addtogroup STM32L4R9I_EVAL_IO_Exported_
Functions
00102   * @{
00103   */
00104
00105 /**
00106   * @brief  Initialize and configure the IO
functionalities and configures all
00107   *         necessary hardware resources (GP
IOs, clocks..).
00108   * @note   BSP_IO_Init() is using HAL_Delay
() function to ensure that stmpe811
00109   *         IO Expander is correctly reset.
HAL_Delay() function provides accurate
00110   *         delay (in milliseconds) based on
 variable incremented in SysTick ISR.
00111   *         This implies that if BSP_IO_Init
() is called from a peripheral ISR process,
```

```
00112   *         then the SysTick interrupt must
have higher priority (numerically lower)
00113   *         than the peripheral interrupt. O
therwise the caller ISR process will be blocked.
00114   * @retval IO_OK: if all initializations ar
e OK. Other value if error.
00115   */
00116 uint8_t BSP_IO_Init(void)
00117 {
00118   uint8_t ret = IO_OK;
00119   uint8_t mfxstm32l152_id = 0;
00120
00121   if (io_driver == NULL) /* Checks if MFX in
itialization has been already done */
00122   {
00123     mfxstm32l152_idd_drv.WakeUp(IO_I2C_ADDRE
SS);
00124
00125     HAL_Delay(10);
00126
00127     /* Read ID and verify the IO expander is
 ready */
00128     mfxstm32l152_id = mfxstm32l152_io_drv.Re
adID(IO_I2C_ADDRESS);
00129
00130     if((mfxstm32l152_id == MFXSTM32L152_ID_1
) || (mfxstm32l152_id == MFXSTM32L152_ID_2))
00131     {
00132       /* Initialize the MFX */
00133       io_driver = &mfxstm32l152_io_drv;
00134
00135       /* Initialize the MFX IO driver struct
ure  */
00136       if(io_driver->Init != NULL)
00137       {
00138         io_driver->Init(IO_I2C_ADDRESS);
00139         io_driver->Start(IO_I2C_ADDRESS, IO_
```

```
PIN_ALL);
00140       }
00141       else
00142       {
00143         ret = IO_ERROR;
00144       }
00145     }
00146     else
00147     {
00148       ret = IO_ERROR;
00149     }
00150   }
00151   else
00152   {
00153     /* MFX initialization already done : rei
nit I2C if needed */
00154     MFX_IO_Init();
00155   }
00156
00157   return ret;
00158 }
00159
00160 /**
00161   * @brief  DeInitialize the IO to allow Mfx
 Initialization to be executed again
00162   * @note   BSP_IO_Init() has no effect if t
he io_driver is already initialized
00163   *         BSP_IO_DeInit() allows to erase
the pointer such to allow init to be effective
00164   * @retval IO_OK
00165   */
00166 uint8_t BSP_IO_DeInit(void)
00167 {
00168   if (io_driver != NULL)
00169   {
00170     if(io_driver->Reset != NULL)
00171     {
```

```
00172       io_driver->Reset(IO_I2C_ADDRESS);
00173     }
00174
00175     io_driver = NULL;
00176   }
00177   return IO_OK;
00178 }
00179
00180 /**
00181   * @brief  Get the selected pins IT status.
00182   * @param  IO_Pin: Selected pin(s) to check
 the status.
00183   *          This parameter can be any combi
nation of the IO pins.
00184   * @retval Status of the checked IO pin(s).
00185   */
00186 uint32_t BSP_IO_ITGetStatus(uint32_t IO_Pin)
00187 {
00188   /* Return the IO Pin IT status */
00189   return (io_driver->ITStatus(IO_I2C_ADDRESS
, IO_Pin));
00190 }
00191
00192 /**
00193   * @brief  Clear the selected IO IT pending
 bit.
00194   * @param  IO_Pin: Selected pin(s) to clear
 the status.
00195   *          This parameter can be any combi
nation of the IO pins.
00196   * @retval None
00197   */
00198 void BSP_IO_ITClear(uint32_t IO_Pin)
00199 {
00200   /* Clear the selected IO IT pending bits */

00201   io_driver->ClearIT(IO_I2C_ADDRESS, IO_Pin)
```

```
;
00202 }
00203
00204 /**
00205   * @brief  Configure the IO pin(s) accordin
g to IO mode structure value.
00206   * @param  IO_Pin: Output IO pin(s) to be s
et or reset.
00207   *          This parameter can be any combi
nation of the IO pin(s).
00208   * @param  IO_Mode: IO pin mode to configure

00209   *          This parameter can be one of th
e following values:
00210   *            @arg  IO_MODE_INPUT
00211   *            @arg  IO_MODE_OUTPUT
00212   *            @arg  IO_MODE_IT_RISING_EDGE
00213   *            @arg  IO_MODE_IT_FALLING_EDGE
00214   *            @arg  IO_MODE_IT_LOW_LEVEL
00215   *            @arg  IO_MODE_IT_HIGH_LEVEL
00216   * @retval IO_OK: if all initializations ar
e OK. Other value if error.
00217   */
00218 uint8_t BSP_IO_ConfigPin(uint32_t IO_Pin, IO
_ModeTypedef IO_Mode)
00219 {
00220   /* Configure the selected IO pin(s) mode */

00221   io_driver->Config(IO_I2C_ADDRESS, IO_Pin,
IO_Mode);
00222
00223   return IO_OK;
00224 }
00225
00226 /**
00227   * @brief  Set the selected IO pin(s) state.
```

```
00228   * @param  IO_Pin: Selected IO pin(s) to wr
ite.
00229   *          This parameter can be any combi
nation of the IO pin(s).
00230   * @param  PinState: New pin state to write
00231   * @retval None
00232   */
00233 void BSP_IO_WritePin(uint32_t IO_Pin, uint8_
t PinState)
00234 {
00235   /* Set the IO pin(s) state */
00236   io_driver->WritePin(IO_I2C_ADDRESS, IO_Pin
, PinState);
00237 }
00238
00239 /**
00240   * @brief  Get the selected IO pin(s) curre
nt state.
00241   * @param  IO_Pin: Selected pin(s) to read.
00242   *          This parameter can be any combi
nation of the IO pin(s).
00243   * @retval The current pins state
00244   */
00245 uint32_t BSP_IO_ReadPin(uint32_t IO_Pin)
00246 {
00247   return(io_driver->ReadPin(IO_I2C_ADDRESS,
IO_Pin));
00248 }
00249
00250 /**
00251   * @brief  Toggle the selected IO pin(s) st
ate
00252   * @param  IO_Pin: Selected IO pin(s) to to
ggle.
00253   *          This parameter can be any combi
nation of the IO pin(s).
00254   * @retval None
```

```
00255   */
00256 void BSP_IO_TogglePin(uint32_t IO_Pin)
00257 {
00258   /* Toggle the IO selected pin(s) state */
00259   if(io_driver->ReadPin(IO_I2C_ADDRESS, IO_P
in) != 0) /* Set */
00260   {
00261     io_driver->WritePin(IO_I2C_ADDRESS, IO_P
in, 0); /* Reset */
00262   }
00263   else
00264   {
00265     io_driver->WritePin(IO_I2C_ADDRESS, IO_P
in, 1);  /* Set */
00266   }
00267
00268 }
00269
00270 /**
00271   * @}
00272   */
00273
00274 /**
00275   * @}
00276   */
00277
00278 /**
00279   * @}
00280   */
00281
00282 /**
00283   * @}
00284   */
00285
00286 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

Generated on Thu Oct 12 2017 10:53:59 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_nor.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_nor.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_nor.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /** @addtogroup BSP
00038   * @{
00039   */
00040
00041 /** @addtogroup STM32L4R9I_EVAL
00042   * @{
00043   */
00044
00045 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00046 #ifndef __STM32L4R9I_EVAL_NOR_H
00047 #define __STM32L4R9I_EVAL_NOR_H
00048
00049 #ifdef __cplusplus
00050  extern "C" {
00051 #endif
00052
00053 /* Includes ------------------------------
-------------------------------------*/
00054 #include "stm32l4xx_hal.h"
00055
00056 /** @addtogroup STM32L4R9I_EVAL_NOR
00057   * @{
00058   */
00059
00060
00061 /* Exported constants ---------------------
-------------------------------------*/
00062 /** @defgroup STM32L4R9I_EVAL_NOR_Exported_C
onstants Exported Constants
00063   * @{
00064   */
00065
00066 /**
```

```
00067   * @brief  NOR status structure definition
00068   */
00069 #define NOR_STATUS_OK           0x00
00070 #define NOR_STATUS_ERROR        0x01
00071
00072 #define NOR_DEVICE_ADDR         ((uint32_t)0
x68000000)
00073
00074 #define NOR_MEMORY_WIDTH        FMC_NORSRAM_
MEM_BUS_WIDTH_16
00075
00076 #define NOR_BURSTACCESS         FMC_BURST_AC
CESS_MODE_DISABLE
00077
00078 #define NOR_WRITEBURST          FMC_WRITE_BU
RST_DISABLE
00079
00080 /* NOR operations Timeout definitions */
00081 #define BLOCKERASE_TIMEOUT      2000U  /* NO
R block erase timeout : 2 s */
00082 #define CHIPERASE_TIMEOUT       400000U  /*
NOR chip erase timeout : 400 s */
00083 #define PROGRAM_TIMEOUT         13U  /* NOR
program timeout : 64 bytes * 200 us = 12.8 ms */
00084
00085 /* NOR Ready/Busy signal GPIO definitions */
00086 #define NOR_READY_BUSY_PIN      GPIO_PIN_6
00087 #define NOR_READY_BUSY_GPIO     GPIOD
00088 #define NOR_READY_STATE         GPIO_PIN_SET
00089 #define NOR_BUSY_STATE          GPIO_PIN_RES
ET
00090
00091 /**
00092   * @}
00093   */
00094
00095
```

```
00096 /* Exported functions ---------------------
---------------------------------*/
00097
00098 /** @addtogroup STM32L4R9I_EVAL_NOR_Exported
_Functions
00099   * @{
00100   */
00101 uint8_t BSP_NOR_Init(void);
00102 uint8_t BSP_NOR_ReadData(uint32_t uwStartAdd
ress, uint16_t *pData, uint32_t uwDataSize);
00103 uint8_t BSP_NOR_WriteData(uint32_t uwStartAd
dress, uint16_t *pData, uint32_t uwDataSize);
00104 uint8_t BSP_NOR_ProgramData(uint32_t uwStart
Address, uint16_t *pData, uint32_t uwDataSize);
00105 uint8_t BSP_NOR_Erase_Block(uint32_t BlockAd
dress);
00106 uint8_t BSP_NOR_Erase_Chip(void);
00107 uint8_t BSP_NOR_Read_ID(NOR_IDTypeDef *pNOR_
ID);
00108 void    BSP_NOR_ReturnToReadMode(void);
00109
00110 /**
00111   * @}
00112   */
00113
00114 /**
00115   * @}
00116   */
00117
00118 #ifdef __cplusplus
00119 }
00120 #endif
00121
00122 #endif /* __STM32L4R9I_EVAL_NOR_H */
00123 /**
00124   * @}
00125   */
```

```
00126
00127 /**
00128   * @}
00129   */
00130
00131 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_nor.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_nor.c
00004   * @author  MCD Application Team
00005   * @brief   This file includes a standard driver for the M29W128GL70ZA6E NOR
00006   *          memories mounted on STM32L4R9I-EVAL board.
00007   @verbatim
00008   ==============================================================================
00009                         ##### How to use this driver #####
00010   ==============================================================================
00011   [..]
00012    (#) This driver is used to drive the M29W128GL70ZA6E NOR flash external
00013        memory mounted on STM32L4R9I-EVAL evaluation board.
00014
```

```
00015    (#) This driver does not need a specific component driver for the NOR device
00016        to be included with.
00017
00018    (#) Initialization steps:
00019        (++) Initialize the NOR external memory using the BSP_NOR_Init() function. This
00020             function includes the MSP layer hardware resources initialization and the
00021             FMC controller configuration to interface with the external NOR memory.
00022
00023    (#) NOR flash operations
00024        (++) NOR external memory can be accessed with read/write operations once it is
00025             initialized.
00026             Read/write operation can be performed with AHB access using the functions
00027             BSP_NOR_ReadData()/BSP_NOR_WriteData(). The BSP_NOR_WriteData() performs write operation
00028             of an amount of data by unit (halfword). You can also perform a program data
00029             operation of an amount of data using the function BSP_NOR_ProgramData().
00030        (++) The function BSP_NOR_Read_ID() returns the chip IDs stored in the structure
00031             "NOR_IDTypeDef". (see the NOR IDs in the memory data sheet)
00032        (++) Perform erase block operation using the function BSP_NOR_Erase_Block() and by
00033             specifying the block address. You can perform an erase operation of the whole
00034             chip by calling the function BSP_NOR_Erase_Chip().
00035        (++) After other operations, the function BSP_NOR_ReturnToReadMode() allows the NOR
```

```
00036             flash to return to read mode to perform read operations on it.
00037   @endverbatim
00038   ******************************************************************************
00039   * @attention
00040   *
00041   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00042   *
00043   * Redistribution and use in source and binary forms, with or without modification,
00044   * are permitted provided that the following conditions are met:
00045   *   1. Redistributions of source code must retain the above copyright notice,
00046   *      this list of conditions and the following disclaimer.
00047   *   2. Redistributions in binary form must reproduce the above copyright notice,
00048   *      this list of conditions and the following disclaimer in the documentation
00049   *      and/or other materials provided with the distribution.
00050   *   3. Neither the name of STMicroelectronics nor the names of its contributors
00051   *      may be used to endorse or promote products derived from this software
00052   *      without specific prior written permission.
00053   *
00054   * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
00055   * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
00056   * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
```

```
00057   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00058   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00059   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00060   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00061   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00062   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00063   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00064   *
00065   ******************************************
**********************************
00066   */
00067
00068 /* Includes ------------------------------
-----------------------------------*/
00069 #include "stm32l4r9i_eval_nor.h"
00070
00071 /** @addtogroup BSP
00072   * @{
00073   */
00074
00075 /** @addtogroup STM32L4R9I_EVAL
00076   * @{
00077   */
00078
00079 /** @defgroup STM32L4R9I_EVAL_NOR STM32L4R9I
_EVAL NOR
00080   * @{
00081   */
00082
00083 /* Private constants ----------------------
```

```
-----------------------------------*/
00084
00085 /** @defgroup STM32L4R9I_EVAL_NOR_Private_Co
nstants Private Constants
00086   * @{
00087   */
00088 /* Timings for NOR M29W128GL70ZA6E */
00089 #define NOR_ADDR_SETUP_TIME  9 /* 70ns with
a clock at 120 MHz (period of 8.33 ns) */
00090 #define NOR_DATA_SETUP_TIME  5 /* 45ns with
a clock at 120 MHz (period of 8.33 ns) */
00091 #define NOR_DATA_HOLD_TIME   0
00092 #define NOR_TURN_AROUND_TIME 4 /* 30ns with
a clock at 120 MHz (perido of 8.33 ns) */
00093
00094 #define NOR_MAX_BUFFER_SIZE  32 /* 16-bits b
us so 32 words */
00095
00096 /**
00097   * @}
00098   */
00099
00100 /* Private variables ----------------------
-----------------------------------*/
00101
00102 /** @defgroup STM32L4R9I_EVAL_NOR_Private_Va
riables Private Variables
00103   * @{
00104   */
00105 static NOR_HandleTypeDef norHandle;
00106 static FMC_NORSRAM_TimingTypeDef Timing;
00107
00108 /**
00109   * @}
00110   */
00111
00112 /* Private functions ----------------------
```

```
----------------------------------*/
00113
00114 /** @defgroup STM32L4R9I_EVAL_NOR_Private_Fu
nctions Private Functions
00115   * @{
00116   */
00117 static void NOR_MspInit(void);
00118
00119 /**
00120   * @}
00121   */
00122
00123 /** @defgroup STM32L4R9I_EVAL_NOR_Exported_F
unctions Exported Functions
00124   * @{
00125   */
00126
00127 /**
00128   * @brief  Initializes the NOR device.
00129   * @retval NOR memory status
00130   */
00131 uint8_t BSP_NOR_Init(void)
00132 {
00133   norHandle.Instance  = FMC_NORSRAM_DEVICE;
00134   norHandle.Extended  = FMC_NORSRAM_EXTENDED
_DEVICE;
00135
00136   /* NOR device configuration */
00137   Timing.AddressSetupTime      = NOR_ADDR_SE
TUP_TIME;
00138   Timing.AddressHoldTime       = 1;
          /* Min value, Don't care on NOR Access
mode B */
00139   Timing.DataSetupTime         = NOR_DATA_SE
TUP_TIME;
00140   Timing.DataHoldTime          = NOR_DATA_HO
LD_TIME;
```

```
00141   Timing.CLKDivision           = 2;
           /* Min value, Don't care on NOR Access
mode B */
00142   Timing.DataLatency           = 2;
           /* Min value, Don't care on NOR Access
mode B */
00143   Timing.BusTurnAroundDuration = NOR_TURN_AR
OUND_TIME;
00144   Timing.AccessMode            = FMC_ACCESS_
MODE_B;
00145
00146   norHandle.Init.NSBank              = FMC_NO
RSRAM_BANK3;
00147   norHandle.Init.DataAddressMux      = FMC_DA
TA_ADDRESS_MUX_DISABLE;
00148   norHandle.Init.MemoryType          = FMC_ME
MORY_TYPE_NOR;
00149   norHandle.Init.MemoryDataWidth     = FMC_NO
RSRAM_MEM_BUS_WIDTH_16;
00150   norHandle.Init.BurstAccessMode     = FMC_BU
RST_ACCESS_MODE_DISABLE;
00151   norHandle.Init.WaitSignal          = FMC_WA
IT_SIGNAL_DISABLE;
00152   norHandle.Init.WaitSignalActive    = FMC_WA
IT_TIMING_BEFORE_WS;
00153   norHandle.Init.WaitSignalPolarity = FMC_WA
IT_SIGNAL_POLARITY_LOW;
00154   norHandle.Init.WriteOperation      = FMC_WR
ITE_OPERATION_ENABLE;
00155   norHandle.Init.ExtendedMode        = FMC_EX
TENDED_MODE_DISABLE;
00156   norHandle.Init.AsynchronousWait    = FMC_AS
YNCHRONOUS_WAIT_ENABLE;
00157   norHandle.Init.WriteBurst          = FMC_BU
RST_ACCESS_MODE_DISABLE;
00158   norHandle.Init.ContinuousClock     = FMC_CO
NTINUOUS_CLOCK_SYNC_ONLY;
```

```
00159   norHandle.Init.WriteFifo          = FMC_WR
ITE_FIFO_DISABLE;
00160   norHandle.Init.NBLSetupTime       = 0;
00161   norHandle.Init.PageSize           = FMC_PA
GE_SIZE_NONE;
00162
00163   /* NOR controller initialization */
00164   NOR_MspInit();
00165
00166   if(HAL_NOR_Init(&norHandle, &Timing, &Timi
ng) != HAL_OK)
00167   {
00168     return NOR_STATUS_ERROR;
00169   }
00170   else
00171   {
00172     return NOR_STATUS_OK;
00173   }
00174 }
00175
00176 /**
00177   * @brief  Reads an amount of data from the
 NOR device.
00178   * @param  uwStartAddress: Read start addre
ss
00179   * @param  pData: Pointer to data to be read

00180   * @param  uwDataSize: Size of data to read
00181   * @retval NOR memory status
00182   */
00183 uint8_t BSP_NOR_ReadData(uint32_t uwStartAdd
ress, uint16_t* pData, uint32_t uwDataSize)
00184 {
00185   if(HAL_NOR_ReadBuffer(&norHandle, NOR_DEVI
CE_ADDR + uwStartAddress, pData, uwDataSize) != HA
L_OK)
00186   {
```

```
00187     return NOR_STATUS_ERROR;
00188   }
00189   else
00190   {
00191     return NOR_STATUS_OK;
00192   }
00193 }
00194
00195 /**
00196   * @brief  Returns the NOR memory to read m
ode.
00197   * @retval None
00198   */
00199 void BSP_NOR_ReturnToReadMode(void)
00200 {
00201    HAL_NOR_ReturnToReadMode(&norHandle);
00202 }
00203
00204 /**
00205   * @brief  Writes an amount of data to the
NOR device.
00206   * @param  uwStartAddress: Write start addr
ess
00207   * @param  pData: Pointer to data to be wri
tten
00208   * @param  uwDataSize: Size of data to write

00209   * @retval NOR memory status
00210   */
00211 uint8_t BSP_NOR_WriteData(uint32_t uwStartAd
dress, uint16_t* pData, uint32_t uwDataSize)
00212 {
00213   uint32_t index = uwDataSize;
00214
00215   while(index > 0)
00216   {
00217     /* Write data to NOR */
```

```
00218     HAL_NOR_Program(&norHandle, (uint32_t *)(
NOR_DEVICE_ADDR + uwStartAddress), pData);
00219
00220     /* Read NOR device status */
00221     if(HAL_NOR_GetStatus(&norHandle, NOR_DEV
ICE_ADDR, PROGRAM_TIMEOUT) != HAL_NOR_STATUS_SUCCE
SS)
00222     {
00223       return NOR_STATUS_ERROR;
00224     }
00225
00226     /* Update the counters */
00227     index--;
00228     uwStartAddress += 2;
00229     pData++;
00230   }
00231
00232   return NOR_STATUS_OK;
00233 }
00234
00235 /**
00236   * @brief  Programs an amount of data to th
e NOR device.
00237   * @param  uwStartAddress: Write start addr
ess
00238   * @param  pData: Pointer to data to be wri
tten
00239   * @param  uwDataSize: Size of data to write

00240   * @retval NOR memory status
00241   */
00242 uint8_t BSP_NOR_ProgramData(uint32_t uwStart
Address, uint16_t* pData, uint32_t uwDataSize)
00243 {
00244   uint32_t buffer_size, data_nb;
00245   uint16_t *src_addr, *dest_addr;
00246   src_addr = pData;
```

```
00247   dest_addr = (uint16_t *)(NOR_DEVICE_ADDR +
 uwStartAddress);
00248   data_nb = uwDataSize;
00249
00250   do
00251   {
00252     if (uwDataSize < NOR_MAX_BUFFER_SIZE)
00253     {
00254       buffer_size = data_nb;
00255     }
00256     else
00257     {
00258       buffer_size = NOR_MAX_BUFFER_SIZE;
00259     }
00260
00261     /* Send NOR program buffer operation */
00262     HAL_NOR_ProgramBuffer(&norHandle, (uint3
2_t)dest_addr, src_addr, buffer_size);
00263
00264     /* Return the NOR memory status */
00265     if(HAL_NOR_GetStatus(&norHandle, NOR_DEV
ICE_ADDR, PROGRAM_TIMEOUT) != HAL_NOR_STATUS_SUCCE
SS)
00266     {
00267       return NOR_STATUS_ERROR;
00268     }
00269
00270     data_nb -= buffer_size;
00271     src_addr += buffer_size;
00272     dest_addr += buffer_size;
00273   } while(data_nb > 0);
00274
00275   return NOR_STATUS_OK;
00276 }
00277
00278 /**
00279   * @brief  Erases the specified block of th
```

```
e NOR device.
00280   * @param  BlockAddress: Block address to e
rase
00281   * @retval NOR memory status
00282   */
00283 uint8_t BSP_NOR_Erase_Block(uint32_t BlockAd
dress)
00284 {
00285   /* Send NOR erase block operation */
00286   HAL_NOR_Erase_Block(&norHandle, BlockAddre
ss, NOR_DEVICE_ADDR);
00287
00288   /* Return the NOR memory status */
00289   if(HAL_NOR_GetStatus(&norHandle, NOR_DEVIC
E_ADDR, BLOCKERASE_TIMEOUT) != HAL_NOR_STATUS_SUCC
ESS)
00290   {
00291     return NOR_STATUS_ERROR;
00292   }
00293   else
00294   {
00295     return NOR_STATUS_OK;
00296   }
00297 }
00298
00299 /**
00300   * @brief  Erases the entire NOR chip.
00301   * @retval NOR memory status
00302   */
00303 uint8_t BSP_NOR_Erase_Chip(void)
00304 {
00305   /* Send NOR Erase chip operation */
00306   HAL_NOR_Erase_Chip(&norHandle, NOR_DEVICE_
ADDR);
00307
00308   /* Return the NOR memory status */
00309   if(HAL_NOR_GetStatus(&norHandle, NOR_DEVIC
```

```
E_ADDR, CHIPERASE_TIMEOUT) != HAL_NOR_STATUS_SUCCE
SS)
00310   {
00311     return NOR_STATUS_ERROR;
00312   }
00313   else
00314   {
00315     return NOR_STATUS_OK;
00316   }
00317 }
00318
00319 /**
00320   * @brief  Reads NOR flash IDs.
00321   * @param  pNOR_ID : Pointer to NOR ID stru
cture
00322   * @retval NOR memory status
00323   */
00324 uint8_t BSP_NOR_Read_ID(NOR_IDTypeDef *pNOR_
ID)
00325 {
00326   if(HAL_NOR_Read_ID(&norHandle, pNOR_ID) !=
 HAL_OK)
00327   {
00328     return NOR_STATUS_ERROR;
00329   }
00330   else
00331   {
00332     return NOR_STATUS_OK;
00333   }
00334 }
00335
00336 /**
00337   * @}
00338   */
00339
00340 /** @addtogroup STM32L4R9I_EVAL_NOR_Private_
Functions
```

```
00341   * @{
00342   */
00343
00344 /**
00345   * @brief  Initializes the NOR MSP.
00346   * @retval None
00347   */
00348 static void NOR_MspInit(void)
00349 {
00350   GPIO_InitTypeDef gpioinitstruct = {0};
00351
00352   /* Enable FMC clock */
00353   __HAL_RCC_FMC_CLK_ENABLE();
00354
00355   /* Enable GPIOs clock */
00356   __HAL_RCC_GPIOD_CLK_ENABLE();
00357   __HAL_RCC_GPIOE_CLK_ENABLE();
00358   __HAL_RCC_GPIOF_CLK_ENABLE();
00359   __HAL_RCC_GPIOG_CLK_ENABLE();
00360   __HAL_RCC_PWR_CLK_ENABLE();
00361   HAL_PWREx_EnableVddIO2();
00362
00363   /* Common GPIO configuration */
00364   gpioinitstruct.Mode      = GPIO_MODE_AF_PP
;
00365   gpioinitstruct.Speed     = GPIO_SPEED_FREQ
_VERY_HIGH;
00366   gpioinitstruct.Alternate = GPIO_AF12_FMC;
00367
00368   /*## NE configuration #######*/
00369   /* NE1 : SRAM */
00370   gpioinitstruct.Pull = GPIO_PULLUP;
00371   gpioinitstruct.Pin = GPIO_PIN_7;
00372   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00373   /* NE3 : NOR */
00374   /* NE4 : TFT LCD */
00375   gpioinitstruct.Pin = GPIO_PIN_10 | GPIO_PI
```

```
N_12;
00376   HAL_GPIO_Init(GPIOG, &gpioinitstruct);
00377
00378   /*## NOE and NWE configuration #######*/
00379   gpioinitstruct.Pin  = GPIO_PIN_4 | GPIO_PI
N_5;
00380   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00381
00382   /*## Ready/Busy configuration #######*/
00383   gpioinitstruct.Pin = NOR_READY_BUSY_PIN;
00384   HAL_GPIO_Init(NOR_READY_BUSY_GPIO, &gpioin
itstruct);
00385
00386   /*## Address Bus #######*/
00387   /* GPIOD configuration */
00388   gpioinitstruct.Pull = GPIO_NOPULL;
00389   gpioinitstruct.Pin  = GPIO_PIN_11 | GPIO_P
IN_12 | GPIO_PIN_13;
00390   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00391
00392   /* GPIOE configuration */
00393   gpioinitstruct.Pin = GPIO_PIN_2 | GPIO_PIN
_3 | GPIO_PIN_4 |
00394                        GPIO_PIN_5 | GPIO_PIN
_6;
00395   HAL_GPIO_Init(GPIOE, &gpioinitstruct);
00396
00397   /* GPIOF configuration */
00398   gpioinitstruct.Pin = GPIO_PIN_0 | GPIO_PIN
_1 | GPIO_PIN_2 | GPIO_PIN_3 |
00399                        GPIO_PIN_4 | GPIO_PIN
_5 | GPIO_PIN_12 | GPIO_PIN_13 |
00400                        GPIO_PIN_14 | GPIO_PI
N_15;
00401   HAL_GPIO_Init(GPIOF, &gpioinitstruct);
00402
00403   /* GPIOG configuration */
```

```
00404   gpioinitstruct.Pin = GPIO_PIN_0 | GPIO_PIN
_1 | GPIO_PIN_2 |
00405                        GPIO_PIN_3 | GPIO_PIN
_4 | GPIO_PIN_5;
00406   HAL_GPIO_Init(GPIOG, &gpioinitstruct);
00407
00408   /*## Data Bus #######*/
00409   /* GPIOD configuration */
00410   gpioinitstruct.Pin = GPIO_PIN_0 | GPIO_PIN
_1 | GPIO_PIN_8 | GPIO_PIN_9 |
00411                        GPIO_PIN_10 | GPIO_PI
N_14 | GPIO_PIN_15;
00412   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00413
00414   /* GPIOE configuration */
00415   gpioinitstruct.Pin = GPIO_PIN_7 | GPIO_PIN
_8 | GPIO_PIN_9 | GPIO_PIN_10 |
00416                        GPIO_PIN_11 | GPIO_PI
N_12 | GPIO_PIN_13 |
00417                        GPIO_PIN_14 | GPIO_PI
N_15;
00418   HAL_GPIO_Init(GPIOE, &gpioinitstruct);
00419 }
00420
00421 /**
00422   * @brief  NOR BSP Wait for Ready/Busy sign
al.
00423   * @param  hnor: Pointer to NOR handle
00424   * @param  Timeout: Timeout duration
00425   * @retval None
00426   */
00427 void HAL_NOR_MspWait(NOR_HandleTypeDef *hnor
, uint32_t Timeout)
00428 {
00429 #if 0
00430   uint32_t timeout = Timeout;
00431
```

```
00432   /* Polling on Ready/Busy signal */
00433   while((HAL_GPIO_ReadPin(NOR_READY_BUSY_GPIO
, NOR_READY_BUSY_PIN) != NOR_BUSY_STATE) && (timeo
ut > 0))
00434   {
00435     timeout--;
00436   }
00437
00438   timeout = Timeout;
00439
00440   /* Polling on Ready/Busy signal */
00441   while((HAL_GPIO_ReadPin(NOR_READY_BUSY_GPIO
, NOR_READY_BUSY_PIN) != NOR_READY_STATE) && (time
out > 0))
00442   {
00443     timeout--;
00444   }
00445 #endif
00446 }
00447
00448 /**
00449   * @}
00450   */
00451
00452 /**
00453   * @}
00454   */
00455
00456 /**
00457   * @}
00458   */
00459
00460 /**
00461   * @}
00462   */
00463
00464 /************************ (C) COPYRIGHT STMi
```

```
croelectronics *****END OF FILE****/
00465
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ospi_ram.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_ospi_ram.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_ospi_ram.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_OSPI_RAM_H
00039 #define __STM32L4R9I_EVAL_OSPI_RAM_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-----------------------------------*/
00046 #include "stm32l4xx_hal.h"
00047 #include "../Components/iss66wvh8m8/iss66wvh
8m8.h"
00048
00049 /** @addtogroup BSP
00050   * @{
00051   */
00052
00053 /** @addtogroup STM32L4R9I_EVAL
00054   * @{
00055   */
00056
00057 /** @addtogroup STM32L4R9I_EVAL_OSPI_RAM
00058   * @{
00059   */
00060
00061 /* Exported constants --------------------
-----------------------------------*/
00062 /** @defgroup STM32L4R9I_EVAL_OSPI_RAM_Expor
ted_Constants Exported Constants
00063   * @{
00064   */
00065 /* OSPI Error codes */
```

```
00066 #define OSPI_RAM_OK            ((uint8_t)0x0
0)
00067 #define OSPI_RAM_ERROR         ((uint8_t)0x0
1)
00068
00069 /**
00070   * @}
00071   */
00072
00073 /* Exported types ------------------------
-----------------------------------*/
00074
00075 /* Exported functions ---------------------
-----------------------------------*/
00076 /** @defgroup STM32L4R9I_EVAL_OSPI_RAM_Expor
ted_Functions Exported Functions
00077   * @{
00078   */
00079 uint8_t BSP_OSPI_RAM_Init                  (
void);
00080 uint8_t BSP_OSPI_RAM_DeInit                (
void);
00081 uint8_t BSP_OSPI_RAM_Read                  (
uint8_t* pData, uint32_t ReadAddr, uint32_t Size);
00082 uint8_t BSP_OSPI_RAM_Read_DMA              (
uint8_t* pData, uint32_t ReadAddr, uint32_t Size);
00083 uint8_t BSP_OSPI_RAM_Write                 (
uint8_t* pData, uint32_t WriteAddr, uint32_t Size)
;
00084 uint8_t BSP_OSPI_RAM_Write_DMA             (
uint8_t* pData, uint32_t WriteAddr, uint32_t Size)
;
00085 uint8_t BSP_OSPI_RAM_EnableMemoryMappedMode(
void);
00086 uint8_t BSP_OSPI_RAM_EnterDeepPowerDown    (
void);
00087 uint8_t BSP_OSPI_RAM_LeaveDeepPowerDown    (
```

```
void);
00088 void    BSP_OSPI_RAM_DMA_IRQHandler         (
void);
00089 void    BSP_OSPI_RAM_IRQHandler             (
void);
00090
00091 /**
00092   * @}
00093   */
00094
00095 /**
00096   * @}
00097   */
00098
00099 /**
00100   * @}
00101   */
00102
00103 /**
00104   * @}
00105   */
00106
00107 #ifdef __cplusplus
00108 }
00109 #endif
00110
00111 #endif /* __STM32L4R9I_EVAL_OSPI__RAM_H */
00112
00113 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_ospi_ram.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_ospi_ram.c
00004   * @author  MCD Application Team
00005   * @brief   This file includes a standard driver for the ISS66WVH8M8 HyperRAM
00006   *          memory mounted on STM32L4R9I-EVAL board.
00007   @verbatim
00008   ==============================================================================
00009                     ##### How to use this driver #####
00010   ==============================================================================
00011   [..]
00012    (#) This driver is used to drive the ISS66WVH8M8 HyperRAM external
00013        memory mounted on STM32L4R9I-EVAL evaluation board.
00014
```

```
00015    (#) This driver need a specific component
 driver (ISS66WVH8M8) to be included with.
00016
00017    (#) Initialization steps:
00018        (++) Initialize the HyperRAM external
 memory using the BSP_OSPI_RAM_Init() function. Th
is
00019             function includes the MSP layer
hardware resources initialization and the
00020             OSPI interface with the external
 memory.
00021
00022    (#) HyperRAM memory operations
00023        (++) HyperRAM memory can be accessed
with read/write operations once it is
00024             initialized.
00025             Read/write operation can be perf
ormed with AHB access using the functions
00026             BSP_OSPI_RAM_Read()/BSP_OSPI_RAM
_Write().
00027        (++) The function BSP_OSPI_RAM_GetInf
o() returns the configuration of the HyperRAM memo
ry.
00028             (see the HyperRAM memory data sh
eet)
00029        (++) Perform erase block operation us
ing the function BSP_OSPI_RAM_Erase_Block() and by
00030             specifying the block address. Yo
u can perform an erase operation of the whole
00031             chip by calling the function BSP
_OSPI_RAM_Erase_Chip().
00032        (++) The function BSP_OSPI_RAM_GetSta
tus() returns the current status of the HyperRAM m
emory.
00033             (see the HyperRAM memory data sh
eet)
00034   @endverbatim
```

```
00035   ******************************************************************************
00036   * @attention
00037   *
00038   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00039   *
00040   * Redistribution and use in source and binary forms, with or without modification,
00041   * are permitted provided that the following conditions are met:
00042   *   1. Redistributions of source code must retain the above copyright notice,
00043   *      this list of conditions and the following disclaimer.
00044   *   2. Redistributions in binary form must reproduce the above copyright notice,
00045   *      this list of conditions and the following disclaimer in the documentation
00046   *      and/or other materials provided with the distribution.
00047   *   3. Neither the name of STMicroelectronics nor the names of its contributors
00048   *      may be used to endorse or promote products derived from this software
00049   *      without specific prior written permission.
00050   *
00051   * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
00052   * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
00053   * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
00054   * DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
00055   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
```

```
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00056   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00057   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00058   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00059   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00060   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00061   *
00062   ****************************************
**********************************
00063   */
00064
00065 /* Includes ------------------------------
----------------------------------*/
00066 #include "stm32l4r9i_eval_ospi_ram.h"
00067
00068 /** @addtogroup BSP
00069   * @{
00070   */
00071
00072 /** @addtogroup STM32L4R9I_EVAL
00073   * @{
00074   */
00075
00076 /** @defgroup STM32L4R9I_EVAL_OSPI_RAM STM32
L4R9I_EVAL OSPI RAM
00077   * @{
00078   */
00079
00080 /* Private constants ---------------------
----------------------------------*/
00081 /** @defgroup STM32L4R9I_EVAL_OSPI_RAM_Priva
te_Constants Private Constants
```

```
00082   * @{
00083   */
00084 #define OSPI_RAM_QUAD_DISABLE       0x0
00085 #define OSPI_RAM_QUAD_ENABLE        0x1
00086
00087 #define OSPI_RAM_HIGH_PERF_DISABLE  0x0
00088 #define OSPI_RAM_HIGH_PERF_ENABLE   0x1
00089 /**
00090   * @}
00091   */
00092 /* Private variables ---------------------
---------------------------------*/
00093
00094 /** @defgroup STM32L4R9I_EVAL_OSPI_RAM_Priva
te_Variables Private Variables
00095   * @{
00096   */
00097 OSPI_HandleTypeDef OspiRamHandle;
00098 DMA_HandleTypeDef DmaOspiRamHandle;
00099
00100 /**
00101   * @}
00102   */
00103
00104
00105 /* Private functions ---------------------
---------------------------------*/
00106
00107 /** @defgroup STM32L4R9I_EVAL_OSPI_RAM_Priva
te_Functions Private Functions
00108   * @{
00109   */
00110 static void OSPI_RAM_MspInit  (void);
00111 static void OSPI_RAM_MspDeInit(void);
00112
00113 /**
00114   * @}
```

```
00115   */
00116
00117 /* Exported functions ---------------------
---------------------------------*/
00118
00119 /** @addtogroup STM32L4R9I_EVAL_OSPI_RAM_Exp
orted_Functions
00120   * @{
00121   */
00122
00123 /**
00124   * @brief  Initializes the OSPI interface.
00125   * @retval HyperRAM memory status
00126   */
00127 uint8_t BSP_OSPI_RAM_Init(void)
00128 {
00129   OSPI_HyperbusCfgTypeDef  sHyperbusCfg;
00130
00131   OspiRamHandle.Instance = OCTOSPI2;
00132
00133   /* Call the DeInit function to reset the d
river */
00134   if (HAL_OSPI_DeInit(&OspiRamHandle) != HAL
_OK)
00135   {
00136     return OSPI_RAM_ERROR;
00137   }
00138
00139   /* System level initialization */
00140   OSPI_RAM_MspInit();
00141
00142   /* OctoSPI initialization */
00143   OspiRamHandle.Init.FifoThreshold         =
 4;
00144   OspiRamHandle.Init.DualQuad              =
 HAL_OSPI_DUALQUAD_DISABLE;
00145   OspiRamHandle.Init.MemoryType            =
```

```
 HAL_OSPI_MEMTYPE_HYPERBUS;
00146   OspiRamHandle.Init.DeviceSize            =
 POSITION_VAL(ISS66WVH8M8_RAM_SIZE); /* 64 MBits */

00147   OspiRamHandle.Init.ChipSelectHighTime    =
 2;
00148   OspiRamHandle.Init.FreeRunningClock      =
 HAL_OSPI_FREERUNCLK_DISABLE;
00149   OspiRamHandle.Init.ClockMode             =
 HAL_OSPI_CLOCK_MODE_0;
00150   OspiRamHandle.Init.WrapSize              =
 HAL_OSPI_WRAP_NOT_SUPPORTED;
00151   OspiRamHandle.Init.ClockPrescaler        =
 3; /* OctoSPI clock = 110MHz / ClockPrescaler = 3
6.67 MHz */
00152   OspiRamHandle.Init.SampleShifting        =
 HAL_OSPI_SAMPLE_SHIFTING_NONE;
00153   OspiRamHandle.Init.DelayHoldQuarterCycle =
 HAL_OSPI_DHQC_ENABLE;
00154   OspiRamHandle.Init.ChipSelectBoundary    =
 0;
00155
00156   if (HAL_OSPI_Init(&OspiRamHandle) != HAL_O
K)
00157   {
00158     return OSPI_RAM_ERROR;
00159   }
00160
00161   /* OctoSPI Hyperbus configuration to acces
s memory space */
00162   sHyperbusCfg.RWRecoveryTime   = 3; /* 40 n
s @ 60MHz */
00163   sHyperbusCfg.AccessTime       = ISS66WVH8M
8_LATENCY_166M;
00164   sHyperbusCfg.WriteZeroLatency = HAL_OSPI_L
ATENCY_ON_WRITE;
00165   sHyperbusCfg.LatencyMode      = HAL_OSPI_F
```

```
IXED_LATENCY;
00166
00167   if (HAL_OSPI_HyperbusCfg(&OspiRamHandle, &
sHyperbusCfg, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != H
AL_OK)
00168   {
00169     return OSPI_RAM_ERROR;
00170   }
00171
00172   return OSPI_RAM_OK;
00173 }
00174
00175 /**
00176   * @brief  De-Initializes the OSPI interfac
e.
00177   * @retval HyperRAM memory status
00178   */
00179 uint8_t BSP_OSPI_RAM_DeInit(void)
00180 {
00181   OspiRamHandle.Instance = OCTOSPI1;
00182
00183   /* Call the DeInit function to reset the d
river */
00184   if (HAL_OSPI_DeInit(&OspiRamHandle) != HAL
_OK)
00185   {
00186     return OSPI_RAM_ERROR;
00187   }
00188
00189   /* System level De-initialization */
00190   OSPI_RAM_MspDeInit();
00191
00192   return OSPI_RAM_OK;
00193 }
00194
00195 /**
00196   * @brief  Reads an amount of data from the
```

```
 HyperRAM memory.
00197   * @param  pData: Pointer to data to be read

00198   * @param  ReadAddr: Read start address
00199   * @param  Size: Size of data to read
00200   * @retval HyperRAM memory status
00201   */
00202 uint8_t BSP_OSPI_RAM_Read(uint8_t* pData, ui
nt32_t ReadAddr, uint32_t Size)
00203 {
00204   OSPI_HyperbusCmdTypeDef  sCommand;
00205
00206   /* Initialize the read command */
00207   sCommand.AddressSpace = HAL_OSPI_MEMORY_AD
DRESS_SPACE;
00208   sCommand.AddressSize  = HAL_OSPI_ADDRESS_3
2_BITS;
00209   sCommand.Address      = ReadAddr;
00210   sCommand.DQSMode      = HAL_OSPI_DQS_ENABL
E;
00211   sCommand.NbData       = Size;
00212
00213   /* Configure the command */
00214   if (HAL_OSPI_HyperbusCmd(&OspiRamHandle, &
sCommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_O
K)
00215   {
00216     return OSPI_RAM_ERROR;
00217   }
00218
00219   /* Reception of the data */
00220   if (HAL_OSPI_Receive(&OspiRamHandle, pData
, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00221   {
00222     return OSPI_RAM_ERROR;
00223   }
00224
```

```
00225   return OSPI_RAM_OK;
00226 }
00227
00228 /**
00229   * @brief  Reads an amount of data from the
 HyperRAM memory in DMA mode.
00230   * @param  pData: Pointer to data to be read

00231   * @param  ReadAddr: Read start address
00232   * @param  Size: Size of data to read
00233   * @retval HyperRAM memory status
00234   */
00235 uint8_t BSP_OSPI_RAM_Read_DMA(uint8_t* pData
, uint32_t ReadAddr, uint32_t Size)
00236 {
00237   OSPI_HyperbusCmdTypeDef  sCommand;
00238
00239   /* Initialize the read command */
00240   sCommand.AddressSpace = HAL_OSPI_MEMORY_AD
DRESS_SPACE;
00241   sCommand.AddressSize  = HAL_OSPI_ADDRESS_3
2_BITS;
00242   sCommand.Address      = ReadAddr;
00243   sCommand.DQSMode      = HAL_OSPI_DQS_ENABL
E;
00244   sCommand.NbData       = Size;
00245
00246   /* Configure the command */
00247   if (HAL_OSPI_HyperbusCmd(&OspiRamHandle, &
sCommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_O
K)
00248   {
00249     return OSPI_RAM_ERROR;
00250   }
00251
00252   /* Reception of the data */
00253   if (HAL_OSPI_Receive_DMA(&OspiRamHandle, p
```

```
Data) != HAL_OK)
00254   {
00255     return OSPI_RAM_ERROR;
00256   }
00257
00258   return OSPI_RAM_OK;
00259 }
00260
00261 /**
00262   * @brief  Writes an amount of data to the
HyperRAM memory.
00263   * @param  pData: Pointer to data to be wri
tten
00264   * @param  WriteAddr: Write start address
00265   * @param  Size: Size of data to write
00266   * @retval HyperRAM memory status
00267   */
00268 uint8_t BSP_OSPI_RAM_Write(uint8_t* pData, u
int32_t WriteAddr, uint32_t Size)
00269 {
00270   OSPI_HyperbusCmdTypeDef  sCommand;
00271
00272   /* Initialize the write command */
00273   sCommand.AddressSpace = HAL_OSPI_MEMORY_AD
DRESS_SPACE;
00274   sCommand.AddressSize  = HAL_OSPI_ADDRESS_3
2_BITS;
00275   sCommand.Address      = WriteAddr;
00276   sCommand.DQSMode      = HAL_OSPI_DQS_ENABL
E;
00277   sCommand.NbData       = Size;
00278
00279   /* Configure the command */
00280   if (HAL_OSPI_HyperbusCmd(&OspiRamHandle, &
sCommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_O
K)
00281   {
```

```
00282     return OSPI_RAM_ERROR;
00283   }
00284
00285   /* Transmission of the data */
00286   if (HAL_OSPI_Transmit(&OspiRamHandle, pDat
a, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_OK)
00287   {
00288     return OSPI_RAM_ERROR;
00289   }
00290
00291   return OSPI_RAM_OK;
00292 }
00293
00294 /**
00295   * @brief  Writes an amount of data to the
HyperRAM memory in DMA mode.
00296   * @param  pData: Pointer to data to be wri
tten
00297   * @param  WriteAddr: Write start address
00298   * @param  Size: Size of data to write
00299   * @retval HyperRAM memory status
00300   */
00301 uint8_t BSP_OSPI_RAM_Write_DMA(uint8_t* pDat
a, uint32_t WriteAddr, uint32_t Size)
00302 {
00303   OSPI_HyperbusCmdTypeDef  sCommand;
00304
00305   /* Initialize the write command */
00306   sCommand.AddressSpace = HAL_OSPI_MEMORY_AD
DRESS_SPACE;
00307   sCommand.AddressSize  = HAL_OSPI_ADDRESS_3
2_BITS;
00308   sCommand.Address      = WriteAddr;
00309   sCommand.DQSMode      = HAL_OSPI_DQS_ENABL
E;
00310   sCommand.NbData       = Size;
00311
```

```
00312   /* Configure the command */
00313   if (HAL_OSPI_HyperbusCmd(&OspiRamHandle, &
sCommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_O
K)
00314   {
00315     return OSPI_RAM_ERROR;
00316   }
00317
00318   /* Transmission of the data */
00319   if (HAL_OSPI_Transmit_DMA(&OspiRamHandle,
pData) != HAL_OK)
00320   {
00321     return OSPI_RAM_ERROR;
00322   }
00323
00324   return OSPI_RAM_OK;
00325 }
00326
00327 /**
00328   * @brief  Configure the OSPI in memory-map
ped mode
00329   * @retval HyperRAM memory status
00330   */
00331 uint8_t BSP_OSPI_RAM_EnableMemoryMappedMode(
void)
00332 {
00333   OSPI_HyperbusCmdTypeDef  sCommand;
00334   OSPI_MemoryMappedTypeDef sMemMappedCfg;
00335
00336   /* OctoSPI Hyperbus command configuration
*/
00337   sCommand.AddressSpace = HAL_OSPI_MEMORY_AD
DRESS_SPACE;
00338   sCommand.AddressSize  = HAL_OSPI_ADDRESS_3
2_BITS;
00339   sCommand.Address      = 0;
00340   sCommand.DQSMode      = HAL_OSPI_DQS_ENABL
```

```
E;
00341   sCommand.NbData       = 1;
00342
00343   if (HAL_OSPI_HyperbusCmd(&OspiRamHandle, &
sCommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_O
K)
00344   {
00345     return OSPI_RAM_ERROR;
00346   }
00347
00348   /* OctoSPI activation of memory-mapped mod
e */
00349   sMemMappedCfg.TimeOutActivation = HAL_OSPI
_TIMEOUT_COUNTER_DISABLE;
00350
00351   if (HAL_OSPI_MemoryMapped(&OspiRamHandle,
&sMemMappedCfg) != HAL_OK)
00352   {
00353     return OSPI_RAM_ERROR;
00354   }
00355
00356   return OSPI_RAM_OK;
00357 }
00358
00359 /**
00360   * @brief  This function enter the HyperRAM
 memory in deep power down mode.
00361   * @retval HyperRAM memory status
00362   */
00363 uint8_t BSP_OSPI_RAM_EnterDeepPowerDown(void
)
00364 {
00365   OSPI_HyperbusCfgTypeDef  sHyperbusCfg;
00366   OSPI_HyperbusCmdTypeDef  sCommand;
00367   uint16_t reg;
00368
00369   /* To write register, no latency is requir
```

```
ed. So a reconfiguration is performed */
00370   sHyperbusCfg.RWRecoveryTime   = 3; /* 40 n
s @ 60MHz */
00371   sHyperbusCfg.AccessTime       = ISS66WVH8M
8_LATENCY_166M;
00372   sHyperbusCfg.WriteZeroLatency = HAL_OSPI_N
O_LATENCY_ON_WRITE;
00373   sHyperbusCfg.LatencyMode      = HAL_OSPI_F
IXED_LATENCY;
00374
00375   if (HAL_OSPI_HyperbusCfg(&OspiRamHandle, &
sHyperbusCfg, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != H
AL_OK)
00376   {
00377     return OSPI_RAM_ERROR;
00378   }
00379
00380   /* Read the value of the CR0 register */
00381   sCommand.AddressSpace = HAL_OSPI_REGISTER_
ADDRESS_SPACE;
00382   sCommand.AddressSize  = HAL_OSPI_ADDRESS_3
2_BITS;
00383   sCommand.Address      = ISS66WVH8M8_CR0_AD
DRESS;
00384   sCommand.DQSMode      = HAL_OSPI_DQS_ENABL
E;
00385   sCommand.NbData       = 2;
00386
00387   if (HAL_OSPI_HyperbusCmd(&OspiRamHandle, &
sCommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_O
K)
00388   {
00389     return OSPI_RAM_ERROR;
00390   }
00391
00392   if (HAL_OSPI_Receive(&OspiRamHandle, (uint
8_t *)(&reg), HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != H
```

```
AL_OK)
00393   {
00394     return OSPI_RAM_ERROR;
00395   }
00396
00397   /* Update the deep power down value of the
 CR0 register */
00398   SET_BIT(reg, ISS66WVH8M8_CR0_DPDE);
00399
00400   if (HAL_OSPI_HyperbusCmd(&OspiRamHandle, &
sCommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_O
K)
00401   {
00402     return OSPI_RAM_ERROR;
00403   }
00404
00405   if (HAL_OSPI_Transmit(&OspiRamHandle, (uin
t8_t *)(&reg), HAL_OSPI_TIMEOUT_DEFAULT_VALUE) !=
HAL_OK)
00406   {
00407     return OSPI_RAM_ERROR;
00408   }
00409
00410   /* ---          Memory takes 10us min to e
nter deep power down          --- */
00411   /* --- At least 30us should be respected b
efore leaving deep power down --- */
00412
00413   /* A reconfiguration is performed to re-en
able the latency on write access */
00414   sHyperbusCfg.WriteZeroLatency = HAL_OSPI_L
ATENCY_ON_WRITE;
00415
00416   if (HAL_OSPI_HyperbusCfg(&OspiRamHandle, &
sHyperbusCfg, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != H
AL_OK)
00417   {
```

```
00418     return OSPI_RAM_ERROR;
00419   }
00420
00421   return OSPI_RAM_OK;
00422 }
00423
00424 /**
00425   * @brief  This function leave the HyperRAM
 memory from deep power down mode.
00426   * @retval HyperRAM memory status
00427   */
00428 uint8_t BSP_OSPI_RAM_LeaveDeepPowerDown(void
)
00429 {
00430   /* --- A dummy command is sent to the memo
ry, as the nCS should be low for at least 200 ns -
-- */
00431   /* ---                    Memory takes 150us
 max to leave deep power down                    -
-- */
00432
00433    OSPI_HyperbusCmdTypeDef  sCommand;
00434    uint32_t data;
00435
00436   /* Initialize the read command */
00437   sCommand.AddressSpace = HAL_OSPI_MEMORY_AD
DRESS_SPACE;
00438   sCommand.AddressSize  = HAL_OSPI_ADDRESS_3
2_BITS;
00439   sCommand.Address      = 0;
00440   sCommand.DQSMode      = HAL_OSPI_DQS_ENABL
E;
00441   sCommand.NbData       = 4;
00442
00443   /* Configure the command */
00444   if (HAL_OSPI_HyperbusCmd(&OspiRamHandle, &
sCommand, HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != HAL_O
```

```
K)
00445   {
00446     return OSPI_RAM_ERROR;
00447   }
00448
00449   /* Reception of the data */
00450   if (HAL_OSPI_Receive(&OspiRamHandle, (uint
8_t*)(&data), HAL_OSPI_TIMEOUT_DEFAULT_VALUE) != H
AL_OK)
00451   {
00452     return OSPI_RAM_ERROR;
00453   }
00454
00455   return OSPI_RAM_OK;
00456 }
00457
00458 /**
00459   * @brief  Handles OctoSPI HyperRAM DMA tra
nsfer interrupt request.
00460   * @retval None
00461   */
00462 void BSP_OSPI_RAM_DMA_IRQHandler(void)
00463 {
00464   HAL_DMA_IRQHandler(OspiRamHandle.hdma);
00465 }
00466
00467 /**
00468   * @brief  Handles OctoSPI HyperRAM interru
pt request.
00469   * @retval None
00470   */
00471 void BSP_OSPI_RAM_IRQHandler(void)
00472 {
00473   HAL_OSPI_IRQHandler(&OspiRamHandle);
00474 }
00475 /**
00476   * @}
```

```
00477   */
00478
00479 /** @addtogroup STM32L4R9I_EVAL_OSPI_RAM_Pri
vate_Functions
00480   * @{
00481   */
00482
00483 /**
00484   * @brief  Initializes the OSPI MSP.
00485   * @retval None
00486   */
00487 static void OSPI_RAM_MspInit(void)
00488 {
00489   GPIO_InitTypeDef GPIO_InitStruct;
00490
00491   /* Enable the OctoSPI memory interface, DM
A, DMAMUX and GPIO clocks */
00492   __HAL_RCC_OSPIM_CLK_ENABLE();
00493   __HAL_RCC_OSPI2_CLK_ENABLE();
00494   __HAL_RCC_DMAMUX1_CLK_ENABLE();
00495   __HAL_RCC_DMA1_CLK_ENABLE();
00496   __HAL_RCC_GPIOG_CLK_ENABLE();
00497   __HAL_RCC_GPIOH_CLK_ENABLE();
00498   __HAL_RCC_GPIOI_CLK_ENABLE();
00499
00500   /* Reset the OctoSPI memory interface */
00501   __HAL_RCC_OSPI2_FORCE_RESET();
00502   __HAL_RCC_OSPI2_RELEASE_RESET();
00503
00504   /* IOSV bit MUST be set to access GPIO por
t G[2:15] */
00505   __HAL_RCC_PWR_CLK_ENABLE();
00506   SET_BIT(PWR->CR2, PWR_CR2_IOSV);
00507
00508   /* OctoSPI CS GPIO pin configuration  */
00509   GPIO_InitStruct.Pin       = GPIO_PIN_5;
00510   GPIO_InitStruct.Mode      = GPIO_MODE_AF_P
```

```
P;
00511   GPIO_InitStruct.Pull      = GPIO_PULLUP;
00512   GPIO_InitStruct.Speed     = GPIO_SPEED_FRE
Q_VERY_HIGH;
00513   GPIO_InitStruct.Alternate = GPIO_AF5_OCTOS
PIM_P2;
00514   HAL_GPIO_Init(GPIOI, &GPIO_InitStruct);
00515
00516   /* OctoSPI DQS GPIO pins configuration  */
00517   GPIO_InitStruct.Pin  = GPIO_PIN_15;
00518   GPIO_InitStruct.Pull = GPIO_NOPULL;
00519   HAL_GPIO_Init(GPIOG, &GPIO_InitStruct);
00520
00521   /* OctoSPI D3, D4, D5, D6 and D7 GPIO pins
 configuration  */
00522   GPIO_InitStruct.Pin  = (GPIO_PIN_8 | GPIO_
PIN_9 | GPIO_PIN_10 | GPIO_PIN_11 | GPIO_PIN_12);
00523   HAL_GPIO_Init(GPIOH, &GPIO_InitStruct);
00524
00525   /* OctoSPI CK, nCK, D2, D1 and D0 GPIO pin
s configuration  */
00526   GPIO_InitStruct.Pin = (GPIO_PIN_6 | GPIO_P
IN_7 | GPIO_PIN_9 | GPIO_PIN_10 | GPIO_PIN_11);
00527   HAL_GPIO_Init(GPIOI, &GPIO_InitStruct);
00528
00529   /* Configure the OctoSPI DMA */
00530   DmaOspiRamHandle.Instance = DMA1_Channel2;
00531   DmaOspiRamHandle.Init.Request
= DMA_REQUEST_OCTOSPI2;
00532   DmaOspiRamHandle.Init.PeriphInc
= DMA_PINC_DISABLE;
00533   DmaOspiRamHandle.Init.MemInc
= DMA_MINC_ENABLE;
00534   DmaOspiRamHandle.Init.Mode
= DMA_NORMAL;
00535   DmaOspiRamHandle.Init.Priority
= DMA_PRIORITY_LOW;
```

```
00536   DmaOspiRamHandle.Init.PeriphDataAlignment
= DMA_PDATAALIGN_BYTE;
00537   DmaOspiRamHandle.Init.MemDataAlignment
= DMA_MDATAALIGN_BYTE;
00538
00539   __HAL_LINKDMA(&OspiRamHandle, hdma, DmaOsp
iRamHandle);
00540   HAL_DMA_Init(&DmaOspiRamHandle);
00541
00542   /* Enable and set priority of the OctoSPI
and DMA interrupts */
00543   HAL_NVIC_SetPriority(OCTOSPI2_IRQn, 0x0F,
0);
00544   HAL_NVIC_SetPriority(DMA1_Channel2_IRQn, 0
x00, 0);
00545
00546   HAL_NVIC_EnableIRQ(OCTOSPI2_IRQn);
00547   HAL_NVIC_EnableIRQ(DMA1_Channel2_IRQn);
00548 }
00549
00550 /**
00551   * @brief  De-Initializes the OSPI MSP.
00552   * @retval None
00553   */
00554 static void OSPI_RAM_MspDeInit(void)
00555 {
00556   GPIO_InitTypeDef GPIO_InitStruct;
00557
00558   /* Disable DMA and OctoSPI interrupts */
00559   HAL_NVIC_DisableIRQ(DMA1_Channel2_IRQn);
00560   HAL_NVIC_DisableIRQ(OCTOSPI2_IRQn);
00561
00562   /* De-configure the OctoSPI DMA */
00563   DmaOspiRamHandle.Instance = DMA1_Channel2;
00564   HAL_DMA_DeInit(&DmaOspiRamHandle);
00565
00566   /* OctoSPI CLK, CS, D0-D7, DQS GPIO pins d
```

```
e-configuration  */
00567   HAL_GPIO_DeInit(GPIOG, GPIO_PIN_15);
00568   HAL_GPIO_DeInit(GPIOH, (GPIO_PIN_8 | GPIO_
PIN_9 | GPIO_PIN_10 | GPIO_PIN_11 | GPIO_PIN_12));
00569   HAL_GPIO_DeInit(GPIOI, (GPIO_PIN_5 | GPIO_
PIN_6 | GPIO_PIN_7 | GPIO_PIN_9 | GPIO_PIN_10 | GP
IO_PIN_11));
00570
00571   /* Set GPIOI pin 5 in pull up mode (optimu
m default setting) */
00572   GPIO_InitStruct.Mode      = GPIO_MODE_INPU
T;
00573   GPIO_InitStruct.Pin       = GPIO_PIN_5;
00574   GPIO_InitStruct.Pull      = GPIO_NOPULL;
00575   GPIO_InitStruct.Speed     = GPIO_SPEED_FRE
Q_LOW;
00576   HAL_GPIO_Init(GPIOI, &GPIO_InitStruct);
00577
00578   /* Set GPIOI pin 6 and 7 in no pull, low s
tate (optimum default setting) */
00579   GPIO_InitStruct.Mode      = GPIO_MODE_OUTP
UT_PP;
00580   GPIO_InitStruct.Pull      = GPIO_NOPULL;
00581   GPIO_InitStruct.Pin       = (GPIO_PIN_6 |
GPIO_PIN_7);
00582   HAL_GPIO_Init(GPIOI, &GPIO_InitStruct);
00583   HAL_GPIO_WritePin(GPIOI, GPIO_PIN_6, GPIO_
PIN_RESET);
00584   HAL_GPIO_WritePin(GPIOI, GPIO_PIN_7, GPIO_
PIN_RESET);
00585
00586   /* Reset the OctoSPI memory interface */
00587   __HAL_RCC_OSPI2_FORCE_RESET();
00588   __HAL_RCC_OSPI2_RELEASE_RESET();
00589
00590   /* Disable the OctoSPI memory interface cl
ock */
```

```
00591   __HAL_RCC_OSPI2_CLK_DISABLE();
00592 }
00593
00594 /**
00595   * @}
00596   */
00597
00598 /**
00599   * @}
00600   */
00601
00602 /**
00603   * @}
00604   */
00605
00606 /**
00607   * @}
00608   */
00609
00610 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
00611
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_rgb_lcd.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_rgb_lcd.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_rgb_lcd.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-----------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_RGB_LCD_H
00039 #define __STM32L4R9I_EVAL_RGB_LCD_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-----------------------------------*/
00046 #include "stm32l4r9i_eval.h"
00047 /* Include LCD component Driver */
00048 /* LCD RK043FN48H-CT672B 4,3" 480x272 pixels
 */
00049 #include "../Components/rk043fn48h/rk043fn48
h.h"
00050
00051
00052 /** @addtogroup BSP
00053   * @{
00054   */
00055
00056 /** @addtogroup STM32L4R9I_EVAL
00057   * @{
00058   */
00059
00060 /** @defgroup STM32L4R9I_EVAL_LCD STM32L4R9I
_EVAL LCD
00061   * @{
00062   */
00063
00064 /** @defgroup STM32L4R9I_EVAL_RGB_LCD STM32L
4R9I_EVAL RGB LCD
```

```
00065   * @{
00066   */
00067
00068 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Export
ed_Constants Exported Constants
00069   * @{
00070   */
00071 /**
00072   * @brief  LCD FB_StartAddress
00073   */
00074 #define LCD_FB_START_ADDRESS     (uint32_t)P
hysFrameBuffer  /* Internal SRAM used for LCD Fram
e buffer */
00075
00076 /**
00077   * @brief  LCD Reload Types
00078   */
00079 #define LCD_RELOAD_IMMEDIATE                (
(uint32_t)LTDC_SRCR_IMR)
00080 #define LCD_RELOAD_VERTICAL_BLANKING        (
(uint32_t)LTDC_SRCR_VBR)
00081
00082 /**
00083   * @}
00084   */
00085
00086 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Export
ed_Variables Exported Variables
00087   * @{
00088   */
00089 extern LTDC_HandleTypeDef         hltdc_eval
;
00090 extern uint32_t                   PhysFrameB
uffer[];
00091
00092 /**
00093   * @}
```

```
00094   */
00095
00096 /** @defgroup STM32L4R9I_EVAL_RGB_LCD_Export
ed_Functions Exported Functions
00097   * @{
00098   */
00099 uint8_t  BSP_RGB_LCD_Init(void);
00100 uint8_t  BSP_RGB_LCD_DeInit(void);
00101 uint32_t BSP_RGB_LCD_GetXSize(void);
00102 uint32_t BSP_RGB_LCD_GetYSize(void);
00103 void     BSP_RGB_LCD_SetXSize(uint32_t image
WidthPixels);
00104 void     BSP_RGB_LCD_SetYSize(uint32_t image
HeightPixels);
00105
00106 /* Functions using the LTDC controller */
00107 void     BSP_RGB_LCD_LayerDefaultInit(uint16
_t LayerIndex, uint32_t FrameBuffer);
00108 void     BSP_RGB_LCD_LayerRgb565Init(uint16_
t LayerIndex, uint32_t FB_Address);
00109 void     BSP_RGB_LCD_SetTransparency(uint32_
t LayerIndex, uint8_t Transparency);
00110 void     BSP_RGB_LCD_SetTransparency_NoReload
(uint32_t LayerIndex, uint8_t Transparency);
00111 void     BSP_RGB_LCD_SetLayerAddress(uint32_
t LayerIndex, uint32_t Address);
00112 void     BSP_RGB_LCD_SetLayerAddress_NoReload
(uint32_t LayerIndex, uint32_t Address);
00113 void     BSP_RGB_LCD_SetColorKeying(uint32_t
 LayerIndex, uint32_t RGBValue);
00114 void     BSP_RGB_LCD_SetColorKeying_NoReload
(uint32_t LayerIndex, uint32_t RGBValue);
00115 void     BSP_RGB_LCD_ResetColorKeying(uint32
_t LayerIndex);
00116 void     BSP_RGB_LCD_ResetColorKeying_NoRelo
ad(uint32_t LayerIndex);
00117 void     BSP_RGB_LCD_SetLayerWindow(uint16_t
```

```
 LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_
t Width, uint16_t Height);
00118 void     BSP_RGB_LCD_SetLayerWindow_NoReload
(uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos
, uint16_t Width, uint16_t Height);
00119 void     BSP_RGB_LCD_SelectLayer(uint32_t La
yerIndex);
00120 void     BSP_RGB_LCD_SetLayerVisible(uint32_
t LayerIndex, FunctionalState State);
00121 void     BSP_RGB_LCD_SetLayerVisible_NoReload
(uint32_t LayerIndex, FunctionalState State);
00122 void     BSP_RGB_LCD_Reload(uint32_t ReloadT
ype);
00123
00124 void     BSP_RGB_LCD_SetTextColor(uint32_t C
olor);
00125 uint32_t BSP_RGB_LCD_GetTextColor(void);
00126 void     BSP_RGB_LCD_SetBackColor(uint32_t C
olor);
00127 uint32_t BSP_RGB_LCD_GetBackColor(void);
00128 void     BSP_RGB_LCD_SetFont(sFONT *fonts);
00129 sFONT    *BSP_RGB_LCD_GetFont(void);
00130
00131 uint32_t BSP_RGB_LCD_ReadPixel(uint16_t Xpos
, uint16_t Ypos);
00132 void     BSP_RGB_LCD_DrawPixel(uint16_t Xpos
, uint16_t Ypos, uint32_t pixel);
00133 void     BSP_RGB_LCD_Clear(uint32_t Color);
00134 void     BSP_RGB_LCD_ClearStringLine(uint32_
t Line);
00135 void     BSP_RGB_LCD_DisplayStringAtLine(uin
t16_t Line, uint8_t *ptr);
00136 void     BSP_RGB_LCD_DisplayStringAt(uint16_
t Xpos, uint16_t Ypos, uint8_t *Text, Text_AlignMo
deTypdef Mode);
00137 void     BSP_RGB_LCD_DisplayChar(uint16_t Xp
os, uint16_t Ypos, uint8_t Ascii);
```

```
00138
00139 void     BSP_RGB_LCD_DrawHLine(uint16_t Xpos
, uint16_t Ypos, uint16_t Length);
00140 void     BSP_RGB_LCD_DrawVLine(uint16_t Xpos
, uint16_t Ypos, uint16_t Length);
00141 void     BSP_RGB_LCD_DrawLine(uint16_t x1, u
int16_t y1, uint16_t x2, uint16_t y2);
00142 void     BSP_RGB_LCD_DrawRect(uint16_t Xpos,
 uint16_t Ypos, uint16_t Width, uint16_t Height);
00143 void     BSP_RGB_LCD_DrawCircle(uint16_t Xpo
s, uint16_t Ypos, uint16_t Radius);
00144 void     BSP_RGB_LCD_DrawPolygon(pPoint Poin
ts, uint16_t PointCount);
00145 void     BSP_RGB_LCD_DrawEllipse(int Xpos, i
nt Ypos, int XRadius, int YRadius);
00146 void     BSP_RGB_LCD_DrawBitmap(uint32_t Xpo
s, uint32_t Ypos, uint8_t *pbmp);
00147
00148 void     BSP_RGB_LCD_FillRect(uint16_t Xpos,
 uint16_t Ypos, uint16_t Width, uint16_t Height);
00149 void     BSP_RGB_LCD_FillCircle(uint16_t Xpo
s, uint16_t Ypos, uint16_t Radius);
00150 void     BSP_RGB_LCD_FillPolygon(pPoint Poin
ts, uint16_t PointCount);
00151 void     BSP_RGB_LCD_FillEllipse(int Xpos, i
nt Ypos, int XRadius, int YRadius);
00152
00153 void     BSP_RGB_LCD_DisplayOff(void);
00154 void     BSP_RGB_LCD_DisplayOn(void);
00155
00156 /* These __weak functions can be surcharged
by application code for specific application needs
 */
00157 void     BSP_RGB_LCD_MspInit(LTDC_HandleType
Def *hltdc, void *Params);
00158 void     BSP_RGB_LCD_MspDeInit(LTDC_HandleTy
peDef *hltdc, void *Params);
```

```
00159 void     BSP_RGB_LCD_ClockConfig(LTDC_Handle
TypeDef *hltdc, void *Params);
00160 void     BSP_RGB_LCD_LTDC_IRQHandler(void);
00161 void     BSP_RGB_LCD_LTDC_ER_IRQHandler(void
);
00162 void     BSP_RGB_LCD_DMA2D_IRQHandler(void);
00163
00164 /**
00165   * @}
00166   */
00167
00168 /**
00169   * @}
00170   */
00171
00172 /**
00173   * @}
00174   */
00175
00176 /**
00177   * @}
00178   */
00179
00180 /**
00181   * @}
00182   */
00183
00184 #ifdef __cplusplus
00185 }
00186 #endif
00187
00188 #endif /* __STM32L4R9I_EVAL_RGB_LCD_H */
00189
00190 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

Generated on Thu Oct 12 2017 10:53:59 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_rgb_ts.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_rgb_ts.h
00004   * @brief   This file contains the common defines and functions prototypes for
00005   *          the stm32l4r9i_eval_rgb_ts.c driver.
00006   ******************************************************************************
00007   * @attention
00008   *
00009   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00010   *
00011   * Redistribution and use in source and binary forms, with or without modification,
00012   * are permitted provided that the following conditions are met:
00013   *   1. Redistributions of source code must retain the above copyright notice,
00014   *      this list of conditions and the fol
```

```
lowing disclaimer.
00015   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00016   *      this list of conditions and the fol
lowing disclaimer in the documentation
00017   *      and/or other materials provided wit
h the distribution.
00018   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00019   *      may be used to endorse or promote p
roducts derived from this software
00020   *      without specific prior written perm
ission.
00021   *
00022   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00023   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00024   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00025   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00026   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00027   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00028   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00029   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00030   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00031   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00032   *
00033   ****************************************
**********************************
```

```
00034   */
00035
00036 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00037 #ifndef __STM32L4R9I_EVAL_RGB_TS_H
00038 #define __STM32L4R9I_EVAL_RGB_TS_H
00039
00040 #ifdef __cplusplus
00041  extern "C" {
00042 #endif
00043
00044 /* Includes ------------------------------
-------------------------------------*/
00045 #include "stm32l4r9i_eval.h"
00046 /* Include touch screen FT5336 component Dri
ver */
00047 #include "../Components/ft5336/ft5336.h"
00048
00049 /** @addtogroup BSP
00050   * @{
00051   */
00052
00053 /** @addtogroup STM32L4R9I_EVAL
00054   * @{
00055   */
00056
00057 /** @defgroup STM32L4R9I_EVAL_TS STM32L4R9I_
EVAL TS
00058   * @{
00059   */
00060
00061 /** @defgroup STM32L4R9I_EVAL_RGB_TS STM32L4
R9I_EVAL RGB TS
00062   * @{
00063   */
00064
00065 /** @defgroup STM32L4R9I_EVAL_RGB_TS_Importe
```

```
d_Variables Imported Variables
00066   * @{
00067   */
00068 /**
00069  *  @brief Table for touchscreen event infor
mation display on LCD :
00070  *  table indexed on enum @ref TS_TouchEvent
TypeDef information
00071  */
00072 extern char * ts_event_string_tab[TOUCH_EVEN
T_NB_MAX];
00073
00074 /**
00075  *  @brief Table for touchscreen gesture Id
information display on LCD : table indexed
00076  *  on enum @ref TS_GestureIdTypeDef informa
tion
00077  */
00078 extern char * ts_gesture_id_string_tab[GEST_
ID_NB_MAX];
00079 /**
00080   * @}
00081   */
00082
00083 /** @addtogroup STM32L4R9I_EVAL_RGB_TS_Expor
ted_Functions
00084   * @{
00085   */
00086 uint8_t BSP_RGB_TS_Init(uint16_t ts_SizeX, u
int16_t ts_SizeY);
00087 uint8_t BSP_RGB_TS_DeInit(void);
00088 uint8_t BSP_RGB_TS_GetState(TS_StateTypeDef
*TS_State);
00089
00090 #if (TS_MULTI_TOUCH_SUPPORTED == 1)
00091 uint8_t BSP_RGB_TS_Get_GestureId(TS_StateTyp
eDef *TS_State);
```

```
00092 #endif /* TS_MULTI_TOUCH_SUPPORTED == 1 */
00093
00094 uint8_t BSP_RGB_TS_ITConfig(void);
00095 uint8_t BSP_RGB_TS_ITGetStatus(void);
00096 void    BSP_RGB_TS_ITClear(void);
00097 uint8_t BSP_RGB_TS_ResetTouchData(TS_StateTy
peDef *TS_State);
00098
00099 /**
00100   * @}
00101   */
00102
00103 /**
00104   * @}
00105   */
00106
00107 /**
00108   * @}
00109   */
00110
00111 /**
00112   * @}
00113   */
00114
00115 /**
00116   * @}
00117   */
00118
00119 #ifdef __cplusplus
00120 }
00121 #endif
00122
00123 #endif /* __STM32L4R9I_EVAL_RGB_TS_H */
00124
00125 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

Generated on Thu Oct 12 2017 10:53:59 for STM32L4R9I_EVAL BSP
User Manual by  1.7.6.1

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_sd.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_sd.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_sd.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00038 #ifndef __STM32L4R9I_EVAL_SD_H
00039 #define __STM32L4R9I_EVAL_SD_H
00040
00041 #ifdef __cplusplus
00042  extern "C" {
00043 #endif
00044
00045 /* Includes ------------------------------
-------------------------------------*/
00046 #include "stm32l4xx_hal.h"
00047
00048 /** @addtogroup BSP
00049   * @{
00050   */
00051
00052 /** @addtogroup STM32L4R9I_EVAL
00053   * @{
00054   */
00055
00056 /** @addtogroup STM32L4R9I_EVAL_SD
00057   * @{
00058   */
00059
00060 /* Exported types ------------------------
-------------------------------------*/
00061
00062 /** @defgroup STM32L4R9I_EVAL_SD_Exported_Ty
pes Exported Types
00063   * @{
00064   */
00065
00066 /**
```

```
00067   * @brief SD Card information structure
00068   */
00069 #define BSP_SD_CardInfo HAL_SD_CardInfoTypeD
ef
00070 /**
00071   * @}
00072   */
00073
00074 /* Exported constants ---------------------
---------------------------------*/
00075
00076 /** @defgroup STM32L4R9I_EVAL_SD_Exported_Co
nstants  Exported Constants
00077   * @{
00078   */
00079 /**
00080   * @brief  SD status structure definition
00081   */
00082 #define   MSD_OK                        ((ui
nt8_t)0x00)
00083 #define   MSD_ERROR                     ((ui
nt8_t)0x01)
00084 #define   MSD_ERROR_SD_NOT_PRESENT      ((ui
nt8_t)0x02)
00085
00086 /**
00087   * @brief  SD transfer state definition
00088   */
00089 #define   SD_TRANSFER_OK                ((ui
nt8_t)0x00)
00090 #define   SD_TRANSFER_BUSY              ((ui
nt8_t)0x01)
00091 #define   SD_TRANSFER_ERROR             ((ui
nt8_t)0x02)
00092
00093 #define SD_DATATIMEOUT           ((uint32_t)
100000000)
```

```
00094
00095 #define SD_PRESENT                ((uint8_t)0
x01)
00096 #define SD_NOT_PRESENT            ((uint8_t)0
x00)
00097
00098 /* SD IRQ handler */
00099 #define SDMMCx_IRQHandler         SDMMC1_IRQH
andler
00100 #define SDMMCx_IRQn               SDMMC1_IRQn
00101
00102 /**
00103   * @}
00104   */
00105
00106 /* Exported variables ---------------------
-----------------------------------*/
00107
00108 /** @defgroup STM32L4R9I_EVAL_SD_Exported_Va
riables Exported Variables
00109   * @{
00110   */
00111 extern SD_HandleTypeDef           hsd_eval;
00112
00113 /**
00114   * @}
00115   */
00116
00117 /* Exported functions ---------------------
-----------------------------------*/
00118
00119 /** @defgroup STM32L4R9I_EVAL_SD_Exported_Fu
nctions Exported Functions
00120   * @{
00121   */
00122 uint8_t BSP_SD_Init(void);
00123 uint8_t BSP_SD_DeInit(void);
```

```
00124 uint8_t BSP_SD_ITConfig(void);
00125 uint8_t BSP_SD_ReadBlocks(uint32_t *pData, u
int32_t ReadAddr, uint32_t NumOfBlocks, uint32_t T
imeout);
00126 uint8_t BSP_SD_WriteBlocks(uint32_t *pData,
uint32_t WriteAddr, uint32_t NumOfBlocks, uint32_t
 Timeout);
00127 uint8_t BSP_SD_ReadBlocks_DMA(uint32_t *pDat
a, uint32_t ReadAddr, uint32_t NumOfBlocks);
00128 uint8_t BSP_SD_WriteBlocks_DMA(uint32_t *pDa
ta, uint32_t WriteAddr, uint32_t NumOfBlocks);
00129 uint8_t BSP_SD_Erase(uint32_t StartAddr, uin
t32_t EndAddr);
00130 uint8_t BSP_SD_GetCardState(void);
00131 void    BSP_SD_GetCardInfo(BSP_SD_CardInfo *
CardInfo);
00132 uint8_t BSP_SD_IsDetected(void);
00133
00134 /* These __weak functions can be surcharged
by application code in case the current settings
00135    (eg. interrupt priority, callbacks implem
entation) need to be changed for specific applicat
ion needs */
00136 void    BSP_SD_MspInit(SD_HandleTypeDef *hsd
, void *Params);
00137 void    BSP_SD_MspDeInit(SD_HandleTypeDef *h
sd, void *Params);
00138 void    BSP_SD_AbortCallback(void);
00139 void    BSP_SD_WriteCpltCallback(void);
00140 void    BSP_SD_ReadCpltCallback(void);
00141
00142 /**
00143   * @}
00144   */
00145
00146 /**
00147   * @}
```

```
00148   */
00149
00150 /**
00151   * @}
00152   */
00153
00154 /**
00155   * @}
00156   */
00157
00158 #ifdef __cplusplus
00159 }
00160 #endif
00161
00162 #endif /* __STM32L4R9I_EVAL_SD_H */
00163
00164 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_sd.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_sd.c
00004   * @author  MCD Application Team
00005   * @brief   This file includes the uSD card driver.
00006   @verbatim
00007   ==============================================================================
00008                         ##### How to use this driver #####
00009   ==============================================================================
00010   (#) This driver is used to drive the micro SD external card mounted on STM32L4R9I_EVAL
00011       evaluation board.
00012
00013   (#) This driver does not need a specific component driver for the micro SD device
00014       to be included with.
00015
```

```
00016   (#) Initialization steps:
00017         (++) Initialize the micro SD card usi
ng the BSP_SD_Init() function. This
00018              function includes the MSP layer
hardware resources initialization (BSP_SD_MspInit(
))
00019              and the SDMMC1 interface configu
ration to interface with the external micro SD. It
00020              also includes the micro SD initi
alization sequence.
00021         (++) To check the SD card presence yo
u can use the function BSP_SD_IsDetected() which
00022              returns the detection status.
00023         (++) If SD presence detection interru
pt mode is desired, you must configure the
00024              SD detection interrupt mode by c
alling the functions BSP_SD_ITConfig().
00025              The interrupt is generated as an
 external interrupt whenever the
00026              micro SD card is plugged/unplugg
ed in/from the evaluation board. The SD detection
00027              is managed by MFX, so the SD det
ection interrupt has to be treated by MFX_IRQOUT
00028              gpio pin IRQ handler.
00029         (++) The function BSP_SD_GetCardInfo(
) is used to get the micro SD card information
00030              which is stored in the structure
 "HAL_SD_CardInfoTypedef".
00031
00032   (#) Micro SD card operations
00033         (++) The micro SD card can be accesse
d with read/write block(s) operations once
00034              it is reay for access. The acces
s can be performed whether using the polling
00035              mode by calling the functions BS
P_SD_ReadBlocks()/BSP_SD_WriteBlocks(),
00036              or by DMA transfer using the fun
```

```
ctions BSP_SD_ReadBlocks_DMA()/BSP_SD_WriteBlocks_
DMA().
00037        (++) The DMA transfer complete is use
d with interrupt mode. Once the SD transfer
00038             is complete, the DMA Tx/Rx trans
fer complete are handled using the
00039             BSP_SD_WriteCpltCallback()/BSP_S
D_ReadCpltCallback() user callback functions imple
mented
00040             by the user at application level.

00041        (++) The SD erase block(s) is perform
ed using the function BSP_SD_Erase() with specifyi
ng
00042             the number of blocks to erase.
00043        (++) The SD runtime status is returne
d when calling the function BSP_SD_GetStatus().
00044    [..]
00045   @endverbatim
00046   ******************************************
**********************************
00047   * @attention
00048   *
00049   * <h2><center>&copy; COPYRIGHT(c) 2017 STM
icroelectronics</center></h2>
00050   *
00051   * Redistribution and use in source and bin
ary forms, with or without modification,
00052   * are permitted provided that the followin
g conditions are met:
00053   *   1. Redistributions of source code must
 retain the above copyright notice,
00054   *      this list of conditions and the fol
lowing disclaimer.
00055   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00056   *      this list of conditions and the fol
```

```
lowing disclaimer in the documentation
00057   *      and/or other materials provided wit
h the distribution.
00058   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00059   *      may be used to endorse or promote p
roducts derived from this software
00060   *      without specific prior written perm
ission.
00061   *
00062   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00063   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00064   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00065   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00066   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00067   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00068   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00069   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00070   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00071   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00072   *
00073   ****************************************
**********************************
00074   */
00075
00076 /* Includes ------------------------------
---------------------------------*/
```

```
00077 #include "stm32l4r9i_eval_io.h"
00078 #include "stm32l4r9i_eval_sd.h"
00079
00080 /** @addtogroup BSP
00081   * @{
00082   */
00083
00084 /** @addtogroup STM32L4R9I_EVAL
00085   * @{
00086   */
00087
00088 /** @defgroup STM32L4R9I_EVAL_SD STM32L4R9I_
EVAL SD
00089   * @{
00090   */
00091
00092 /* Private variables ----------------------
-----------------------------------*/
00093
00094 /** @defgroup STM32L4R9I_EVAL_SD_Private_Var
iables Private Variables
00095   * @{
00096   */
00097 SD_HandleTypeDef hsd_eval;
00098 static uint8_t UseExtiModeDetection = 0;
00099 /**
00100   * @}
00101   */
00102
00103 /* Private function prototypes ------------
-----------------------------------*/
00104
00105 /* Exported functions ---------------------
-----------------------------------*/
00106
00107 /** @addtogroup STM32L4R9I_EVAL_SD_Exported_
Functions
```

```
00108   * @{
00109   */
00110
00111 /**
00112   * @brief  Initializes the SD card device.
00113   * @retval SD status
00114   */
00115 uint8_t BSP_SD_Init(void)
00116 {
00117   uint8_t sd_state = MSD_OK;
00118
00119   /* uSD device interface configuration */
00120   hsd_eval.Instance = SDMMC1;
00121   hsd_eval.Init.ClockEdge           = SDMMC_
CLOCK_EDGE_RISING;
00122   hsd_eval.Init.ClockPowerSave      = SDMMC_
CLOCK_POWER_SAVE_DISABLE;
00123   hsd_eval.Init.BusWide             = SDMMC_
BUS_WIDE_4B;
00124   hsd_eval.Init.HardwareFlowControl = SDMMC_
HARDWARE_FLOW_CONTROL_ENABLE;
00125   hsd_eval.Init.ClockDiv            = 1;
00126   hsd_eval.Init.Transceiver         = SDMMC_
TRANSCEIVER_DISABLE;
00127
00128   /* Initialize IO functionalities (MFX) use
d by SD detect pin */
00129   BSP_IO_Init();
00130
00131   /* Check if the SD card is plugged in the
slot */
00132   BSP_IO_ConfigPin(SD_DETECT_PIN, IO_MODE_IN
PUT_PU);
00133
00134   /* Check if the SD card is plugged in the
slot */
00135   if(BSP_SD_IsDetected() != SD_PRESENT)
```

```
00136   {
00137     return MSD_ERROR_SD_NOT_PRESENT;
00138   }
00139
00140   /* Msp SD initialization */
00141   BSP_SD_MspInit(&hsd_eval, NULL);
00142
00143   /* HAL SD initialization */
00144   if(HAL_SD_Init(&hsd_eval) != HAL_OK)
00145   {
00146     sd_state = MSD_ERROR;
00147   }
00148
00149   return  sd_state;
00150 }
00151
00152 /**
00153   * @brief  DeInitializes the SD card device.

00154   * @retval SD status
00155   */
00156 uint8_t BSP_SD_DeInit(void)
00157 {
00158   uint8_t sd_state = MSD_OK;
00159
00160   hsd_eval.Instance = SDMMC1;
00161
00162   /* Set back Mfx pin to INPUT mode in case
it was in exti */
00163   UseExtiModeDetection = 0;
00164
00165   /* HAL SD deinitialization */
00166   if(HAL_SD_DeInit(&hsd_eval) != HAL_OK)
00167   {
00168     sd_state = MSD_ERROR;
00169   }
00170
```

```
00171   /* Msp SD deinitialization */
00172   BSP_SD_MspDeInit(&hsd_eval, NULL);
00173
00174   return  sd_state;
00175 }
00176
00177 /**
00178   * @brief  Configures Interrupt mode for SD
 detection pin.
00179   * @retval IO_OK: if all initializations ar
e OK. Other value if error.
00180   */
00181 uint8_t BSP_SD_ITConfig(void)
00182 {
00183   /* Configure Interrupt mode for SD detecti
on pin */
00184   /* Note: disabling exti mode can be done c
alling BSP_SD_DeInit() */
00185   UseExtiModeDetection = 1;
00186
00187   /* Check SD card detect pin */
00188   if(BSP_IO_ReadPin(SD_DETECT_PIN) == IO_PIN
_RESET)
00189   {
00190     /* Card present */
00191     return BSP_IO_ConfigPin(SD_DETECT_PIN, I
O_MODE_IT_RISING_EDGE_PU);
00192   }
00193   else
00194   {
00195     /* Card not present */
00196     return BSP_IO_ConfigPin(SD_DETECT_PIN, I
O_MODE_IT_FALLING_EDGE_PU);
00197   }
00198 }
00199
00200 /**
```

```
00201  * @brief  Detects if SD card is correctly p
lugged in the memory slot or not.
00202  * @retval Returns if SD is detected or not
00203  */
00204 uint8_t BSP_SD_IsDetected(void)
00205 {
00206   __IO uint8_t status = SD_PRESENT;
00207
00208   /* Check SD card detect pin */
00209   if (BSP_IO_ReadPin(SD_DETECT_PIN) == IO_PI
N_RESET)
00210   {
00211     /* Card present */
00212     if (UseExtiModeDetection)
00213     {
00214       BSP_IO_ConfigPin(SD_DETECT_PIN, IO_MOD
E_IT_RISING_EDGE_PU);
00215     }
00216   }
00217   else
00218   {
00219     /* Card not present */
00220     status = SD_NOT_PRESENT;
00221     if (UseExtiModeDetection)
00222     {
00223       BSP_IO_ConfigPin(SD_DETECT_PIN, IO_MOD
E_IT_FALLING_EDGE_PU);
00224     }
00225   }
00226
00227   return status;
00228 }
00229
00230 /**
00231   * @brief  Reads block(s) from a specified
address in an SD card, in polling mode.
00232   * @param  pData: Pointer to the buffer tha
```

```
t will contain the data to transmit
00233   * @param  ReadAddr: Address from where dat
a is to be read
00234   * @param  NumOfBlocks: Number of SD blocks
 to read
00235   * @param  Timeout: Timeout for read operat
ion
00236   * @retval SD status
00237   */
00238 uint8_t BSP_SD_ReadBlocks(uint32_t *pData, u
int32_t ReadAddr, uint32_t NumOfBlocks, uint32_t T
imeout)
00239 {
00240   HAL_StatusTypeDef  sd_state = HAL_OK;
00241
00242   sd_state =  HAL_SD_ReadBlocks(&hsd_eval, (
uint8_t *)pData, ReadAddr, NumOfBlocks, Timeout);
00243
00244   if  (sd_state == HAL_OK)
00245   {
00246     return MSD_OK;
00247   }
00248   else
00249   {
00250     return MSD_ERROR;
00251   }
00252 }
00253
00254 /**
00255   * @brief  Writes block(s) to a specified a
ddress in an SD card, in polling mode.
00256   * @param  pData: Pointer to the buffer tha
t will contain the data to transmit
00257   * @param  WriteAddr: Address from where da
ta is to be written
00258   * @param  NumOfBlocks: Number of SD blocks
 to write
```

```
00259   * @param  Timeout: Timeout for write opera
tion
00260   * @retval SD status
00261   */
00262 uint8_t BSP_SD_WriteBlocks(uint32_t *pData,
uint32_t WriteAddr, uint32_t NumOfBlocks, uint32_t
 Timeout)
00263 {
00264   HAL_StatusTypeDef  sd_state = HAL_OK;
00265
00266   sd_state = HAL_SD_WriteBlocks(&hsd_eval, (
uint8_t *)pData, WriteAddr, NumOfBlocks, Timeout);
00267
00268   if( sd_state == HAL_OK)
00269   {
00270     return MSD_OK;
00271   }
00272   else
00273   {
00274     return MSD_ERROR;
00275   }
00276 }
00277
00278 /**
00279   * @brief  Reads block(s) from a specified
address in an SD card, in DMA mode.
00280   * @param  pData: Pointer to the buffer tha
t will contain the data to transmit
00281   * @param  ReadAddr: Address from where dat
a is to be read
00282   * @param  NumOfBlocks: Number of SD blocks
 to read
00283   * @retval SD status
00284   */
00285 uint8_t BSP_SD_ReadBlocks_DMA(uint32_t *pDat
a, uint32_t ReadAddr, uint32_t NumOfBlocks)
00286 {
```

```
00287   HAL_StatusTypeDef  sd_state = HAL_OK;
00288
00289   /* Read block(s) in DMA transfer mode */
00290   sd_state = HAL_SD_ReadBlocks_DMA(&hsd_eval
, (uint8_t *)pData, ReadAddr, NumOfBlocks);
00291
00292   if( sd_state == HAL_OK)
00293   {
00294     return MSD_OK;
00295   }
00296   else
00297   {
00298     return MSD_ERROR;
00299   }
00300 }
00301
00302 /**
00303   * @brief  Writes block(s) to a specified a
ddress in an SD card, in DMA mode.
00304   * @param  pData: Pointer to the buffer tha
t will contain the data to transmit
00305   * @param  WriteAddr: Address from where da
ta is to be written
00306   * @param  NumOfBlocks: Number of SD blocks
 to write
00307   * @retval SD status
00308   */
00309 uint8_t BSP_SD_WriteBlocks_DMA(uint32_t *pDa
ta, uint32_t WriteAddr, uint32_t NumOfBlocks)
00310 {
00311   HAL_StatusTypeDef  sd_state = HAL_OK;
00312
00313   /* Write block(s) in DMA transfer mode */
00314   sd_state = HAL_SD_WriteBlocks_DMA(&hsd_eval
, (uint8_t *)pData, WriteAddr, NumOfBlocks);
00315
00316   if( sd_state == HAL_OK)
```

```
00317   {
00318     return MSD_OK;
00319   }
00320   else
00321   {
00322     return MSD_ERROR;
00323   }
00324 }
00325
00326 /**
00327   * @brief  Erases the specified memory area
 of the given SD card.
00328   * @param  StartAddr: Start byte address
00329   * @param  EndAddr: End byte address
00330   * @retval SD status
00331   */
00332 uint8_t BSP_SD_Erase(uint32_t StartAddr, uin
t32_t EndAddr)
00333 {
00334   HAL_StatusTypeDef  sd_state = HAL_OK;
00335
00336   sd_state = HAL_SD_Erase(&hsd_eval, StartAd
dr, EndAddr);
00337
00338   if( sd_state == HAL_OK)
00339   {
00340     return MSD_OK;
00341   }
00342   else
00343   {
00344     return MSD_ERROR;
00345   }
00346 }
00347
00348 /**
00349   * @brief  Gets the current SD card data st
atus.
```

```
00350   * @retval Data transfer state.
00351   *          This value can be one of the fo
llowing values:
00352   *            @arg  SD_TRANSFER_OK: No data
 transfer is acting
00353   *            @arg  SD_TRANSFER_BUSY: Data
transfer is acting
00354   */
00355 uint8_t BSP_SD_GetCardState(void)
00356 {
00357   return((HAL_SD_GetCardState(&hsd_eval) ==
HAL_SD_CARD_TRANSFER ) ? SD_TRANSFER_OK : SD_TRANS
FER_BUSY);
00358 }
00359
00360 /**
00361   * @brief  Get SD information about specifi
c SD card.
00362   * @param  CardInfo: Pointer to HAL_SD_Card
InfoTypedef structure
00363   * @retval None
00364   */
00365 void BSP_SD_GetCardInfo(BSP_SD_CardInfo *Car
dInfo)
00366 {
00367   /* Get SD card Information */
00368   HAL_SD_GetCardInfo(&hsd_eval, CardInfo);
00369 }
00370
00371 /**
00372   * @brief  Initializes the SD MSP.
00373   * @retval None
00374   */
00375 __weak void BSP_SD_MspInit(SD_HandleTypeDef
*hsd, void *Params)
00376 {
00377   GPIO_InitTypeDef gpioinitstruct = {0};
```

```
00378   RCC_OscInitTypeDef        RCC_OscInitStruc
t;
00379   RCC_PeriphCLKInitTypeDef  RCC_PeriphClkIni
t;
00380
00381   /* Check whether HSI48 is enabled or not */

00382   HAL_RCC_GetOscConfig(&RCC_OscInitStruct);
00383   if(RCC_OscInitStruct.HSI48State != RCC_HSI
48_ON)
00384   {
00385     RCC_OscInitStruct.OscillatorType = RCC_O
SCILLATORTYPE_HSI48;
00386     RCC_OscInitStruct.HSI48State     = RCC_H
SI48_ON;
00387     RCC_OscInitStruct.PLL.PLLState   = RCC_P
LL_NONE;
00388     if(HAL_RCC_OscConfig(&RCC_OscInitStruct)
 != HAL_OK)
00389     {
00390       while(1) {}
00391     }
00392   }
00393
00394   /* Configure the Eval SDMMC1 clock source.
 The clock is derived from the HSI48 */
00395   RCC_PeriphClkInit.PeriphClockSelection = R
CC_PERIPHCLK_SDMMC1;
00396   RCC_PeriphClkInit.Sdmmc1ClockSelection = R
CC_SDMMC1CLKSOURCE_HSI48;
00397   if(HAL_RCCEx_PeriphCLKConfig(&RCC_PeriphCl
kInit) != HAL_OK)
00398   {
00399     while(1) {}
00400   }
00401
00402   /* Enable SDMMC1 clock */
```

```
00403   __HAL_RCC_SDMMC1_CLK_ENABLE();
00404
00405   /* Enable GPIOs clock */
00406   __HAL_RCC_GPIOC_CLK_ENABLE();
00407   __HAL_RCC_GPIOD_CLK_ENABLE();
00408
00409   /* Common GPIO configuration */
00410   gpioinitstruct.Mode      = GPIO_MODE_AF_PP
;
00411   gpioinitstruct.Pull      = GPIO_PULLUP;
00412   gpioinitstruct.Speed     = GPIO_SPEED_FREQ
_VERY_HIGH;
00413   gpioinitstruct.Alternate = GPIO_AF12_SDMMC
1;
00414
00415   /* GPIOC configuration */
00416   gpioinitstruct.Pin = GPIO_PIN_8 | GPIO_PIN
_9 | GPIO_PIN_10 | GPIO_PIN_11 | GPIO_PIN_12;
00417   HAL_GPIO_Init(GPIOC, &gpioinitstruct);
00418
00419   /* GPIOD configuration */
00420   gpioinitstruct.Pin = GPIO_PIN_2;
00421   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00422
00423   /* NVIC configuration for SDMMC1 interrupt
s */
00424   HAL_NVIC_SetPriority(SDMMCx_IRQn, 5, 0);
00425   HAL_NVIC_EnableIRQ(SDMMCx_IRQn);
00426 }
00427
00428 /**
00429   * @brief  De-Initializes the SD MSP.
00430   * @retval None
00431   */
00432 __weak void BSP_SD_MspDeInit(SD_HandleTypeDe
f *hsd, void *Params)
00433 {
```

```
00434   GPIO_InitTypeDef gpioinitstruct = {0};
00435
00436   /* NVIC configuration for SDMMC1 interrupt
s */
00437   HAL_NVIC_DisableIRQ(SDMMCx_IRQn);
00438
00439   /* Disable SDMMC1 clock */
00440   __HAL_RCC_SDMMC1_CLK_DISABLE();
00441
00442   /* Enable GPIOs clock */
00443   __HAL_RCC_GPIOC_CLK_ENABLE();
00444   __HAL_RCC_GPIOD_CLK_ENABLE();
00445
00446   /* Common GPIO configuration */
00447   gpioinitstruct.Mode      = GPIO_MODE_ANALO
G;
00448   gpioinitstruct.Pull      = GPIO_NOPULL;
00449   gpioinitstruct.Speed     = GPIO_SPEED_FREQ
_LOW;
00450   gpioinitstruct.Alternate = 0;
00451
00452   /* GPIOC configuration */
00453   gpioinitstruct.Pin = GPIO_PIN_8 | GPIO_PIN
_9 | GPIO_PIN_10 | GPIO_PIN_11 | GPIO_PIN_12;
00454   HAL_GPIO_Init(GPIOC, &gpioinitstruct);
00455
00456   /* GPIOD configuration */
00457   gpioinitstruct.Pin = GPIO_PIN_2;
00458   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00459 }
00460
00461 /**
00462   * @brief BSP SD Abort callback
00463   * @retval None
00464   */
00465 __weak void BSP_SD_AbortCallback(void)
00466 {
```

```
00467
00468 }
00469
00470 /**
00471   * @brief BSP Tx Transfer completed callback

00472   * @retval None
00473   */
00474 __weak void BSP_SD_WriteCpltCallback(void)
00475 {
00476
00477 }
00478
00479 /**
00480   * @brief BSP Rx Transfer completed callback

00481   * @retval None
00482   */
00483 __weak void BSP_SD_ReadCpltCallback(void)
00484 {
00485
00486 }
00487
00488 /**
00489   * @brief SD Abort callback
00490   * @param hsd: SD handle
00491   * @retval None
00492   */
00493 void HAL_SD_AbortCallback(SD_HandleTypeDef *
hsd)
00494 {
00495   BSP_SD_AbortCallback();
00496 }
00497
00498 /**
00499   * @brief Tx Transfer completed callback
00500   * @param hsd: SD handle
```

```
00501   * @retval None
00502   */
00503 void HAL_SD_TxCpltCallback(SD_HandleTypeDef
*hsd)
00504 {
00505   BSP_SD_WriteCpltCallback();
00506 }
00507
00508 /**
00509   * @brief Rx Transfer completed callback
00510   * @param hsd: SD handle
00511   * @retval None
00512   */
00513 void HAL_SD_RxCpltCallback(SD_HandleTypeDef
*hsd)
00514 {
00515   BSP_SD_ReadCpltCallback();
00516 }
00517
00518 /**
00519   * @}
00520   */
00521
00522 /**
00523   * @}
00524   */
00525
00526 /**
00527   * @}
00528   */
00529
00530 /**
00531   * @}
00532   */
00533
00534 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```



STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_sram.h

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_sram.h
00004   * @author  MCD Application Team
00005   * @brief   This file contains the common defines and functions prototypes for
00006   *          the stm32l4r9i_eval_sram.c driver.
00007   ******************************************************************************
00008   * @attention
00009   *
00010   * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00011   *
00012   * Redistribution and use in source and binary forms, with or without modification,
00013   * are permitted provided that the following conditions are met:
00014   *   1. Redistributions of source code must retain the above copyright notice,
```

```
00015   *      this list of conditions and the fol
lowing disclaimer.
00016   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00017   *      this list of conditions and the fol
lowing disclaimer in the documentation
00018   *      and/or other materials provided wit
h the distribution.
00019   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00020   *      may be used to endorse or promote p
roducts derived from this software
00021   *      without specific prior written perm
ission.
00022   *
00023   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00024   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00025   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00026   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00027   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
00028   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00029   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00030   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00031   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00032   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00033   *
00034   ****************************************
```

```
**********************************
00035   */
00036
00037 /** @addtogroup BSP
00038   * @{
00039   */
00040
00041 /** @addtogroup STM32L4R9I_EVAL
00042   * @{
00043   */
00044
00045 /* Define to prevent recursive inclusion ---
-------------------------------------*/
00046 #ifndef __STM32L4R9I_EVAL_SRAM_H
00047 #define __STM32L4R9I_EVAL_SRAM_H
00048
00049 #ifdef __cplusplus
00050  extern "C" {
00051 #endif
00052
00053 /* Includes ------------------------------
-------------------------------------*/
00054 #include "stm32l4xx_hal.h"
00055
00056 /** @addtogroup STM32L4R9I_EVAL_SRAM
00057   * @{
00058   */
00059
00060
00061 /* Exported constants ---------------------
-------------------------------------*/
00062
00063 /** @defgroup STM32L4R9I_EVAL_SRAM_Exported_
Constants Exported Constants
00064   * @{
00065   */
00066
```

```
00067 /**
00068   * @brief  SRAM status structure definition
00069   */
00070 #define SRAM_OK              0x00
00071 #define SRAM_ERROR           0x01
00072
00073 #define SRAM_DEVICE_ADDR     ((uint32_t)0x60
000000)
00074 #define SRAM_DEVICE_SIZE     ((uint32_t)0x20
0000)  /* SRAM device size : 1024 words of 16-bits
 */
00075
00076 #define SRAM_MEMORY_WIDTH    FMC_NORSRAM_MEM
_BUS_WIDTH_16
00077
00078 #define SRAM_BURSTACCESS     FMC_BURST_ACCES
S_MODE_DISABLE
00079
00080 #define SRAM_WRITEBURST      FMC_WRITE_BURST
_DISABLE
00081
00082 /* DMA definitions for SRAM DMA transfer */
00083 #define SRAM_DMAx_CLK_ENABLE __HAL_RCC_DMA1_
CLK_ENABLE
00084 #define SRAM_DMAx_CHANNEL    DMA1_Channel1
00085 #define SRAM_DMAx_IRQn       DMA1_Channel1_I
RQn
00086 #define SRAM_DMAx_IRQHandler DMA1_Channel1_I
RQHandler
00087
00088 /**
00089   * @}
00090   */
00091
00092
00093 /* Exported functions ---------------------
-----------------------------------*/
```

```
00094
00095 /** @addtogroup STM32L4R9I_EVAL_SRAM_Exporte
d_Functions
00096   * @{
00097   */
00098 uint8_t BSP_SRAM_Init(void);
00099 uint8_t BSP_SRAM_ReadData(uint32_t uwStartAd
dress, uint16_t *pData, uint32_t uwDataSize);
00100 uint8_t BSP_SRAM_ReadData_DMA(uint32_t uwSta
rtAddress, uint16_t *pData, uint32_t uwDataSize);
00101 uint8_t BSP_SRAM_WriteData(uint32_t uwStartA
ddress, uint16_t *pData, uint32_t uwDataSize);
00102 uint8_t BSP_SRAM_WriteData_DMA(uint32_t uwSt
artAddress, uint16_t *pData, uint32_t uwDataSize);
00103 void BSP_SRAM_DMA_IRQHandler(void);
00104
00105 /**
00106   * @}
00107   */
00108
00109 /**
00110   * @}
00111   */
00112
00113 #ifdef __cplusplus
00114 }
00115 #endif
00116
00117 #endif /* __STM32L4R9I_EVAL_SRAM_H */
00118
00119 /**
00120   * @}
00121   */
00122
00123 /**
00124   * @}
00125   */
```

```
00126
00127 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

stm32l4r9i_eval_sram.c

Go to the documentation of this file.

```
00001 /**
00002   ******************************************************************************
00003   * @file    stm32l4r9i_eval_sram.c
00004   * @author  MCD Application Team
00005   * @brief   This file provides a set of functions needed to drive the
00006   *          IS61WV102416BLL-10MLI SRAM memory mounted on STM32L4R9I-EVAL board.
00007   @verbatim
00008   ==============================================================================
00009                         ##### How to use this driver #####
00010   ==============================================================================
00011   [..]
00012    (#) This driver is used to drive the IS61WV102416BLL-10MLI SRAM external memory mounted
00013        on STM32L4R9I-EVAL evaluation board.
00014
00015    (#) This driver does not need a specific
```

```
component driver for the SRAM device
00016        to be included with.
00017
00018    (#) Initialization steps:
00019        (++) Initialize the SRAM external mem
ory using the BSP_SRAM_Init() function. This
00020             function includes the MSP layer
 hardware resources initialization and the
00021             FMC controller configuration to
 interface with the external SRAM memory.
00022
00023    (#) SRAM read/write operations
00024        (++) SRAM external memory can be acce
ssed with read/write operations once it is
00025             initialized.
00026             Read/write operation can be perf
ormed with AHB access using the functions
00027             BSP_SRAM_ReadData()/BSP_SRAM_Wri
teData(), or by DMA transfer using the functions
00028             BSP_SRAM_ReadData_DMA()/BSP_SRAM
_WriteData_DMA().
00029        (++) The AHB access is performed with
 16-bit width transaction, the DMA transfer
00030             configuration is fixed at single
 (no burst) halfword transfer
00031             (see the SRAM_MspInit() static f
unction).
00032        (++) User can implement his own funct
ions for read/write access with his desired
00033             configurations.
00034        (++) If interrupt mode is used for DM
A transfer, the function BSP_SRAM_DMA_IRQHandler()
00035             is called in IRQ handler file, t
o serve the generated interrupt once the DMA
00036             transfer is complete.
00037   @endverbatim
00038   ******************************************
```

```
**********************************
00039   * @attention
00040   *
00041   * <h2><center>&copy; COPYRIGHT(c) 2017 STM
icroelectronics</center></h2>
00042   *
00043   * Redistribution and use in source and bin
ary forms, with or without modification,
00044   * are permitted provided that the followin
g conditions are met:
00045   *   1. Redistributions of source code must
 retain the above copyright notice,
00046   *      this list of conditions and the fol
lowing disclaimer.
00047   *   2. Redistributions in binary form must
 reproduce the above copyright notice,
00048   *      this list of conditions and the fol
lowing disclaimer in the documentation
00049   *      and/or other materials provided wit
h the distribution.
00050   *   3. Neither the name of STMicroelectron
ics nor the names of its contributors
00051   *      may be used to endorse or promote p
roducts derived from this software
00052   *      without specific prior written perm
ission.
00053   *
00054   * THIS SOFTWARE IS PROVIDED BY THE COPYRIG
HT HOLDERS AND CONTRIBUTORS "AS IS"
00055   * AND ANY EXPRESS OR IMPLIED WARRANTIES, I
NCLUDING, BUT NOT LIMITED TO, THE
00056   * IMPLIED WARRANTIES OF MERCHANTABILITY AN
D FITNESS FOR A PARTICULAR PURPOSE ARE
00057   * DISCLAIMED. IN NO EVENT SHALL THE COPYRI
GHT HOLDER OR CONTRIBUTORS BE LIABLE
00058   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SP
ECIAL, EXEMPLARY, OR CONSEQUENTIAL
```

```
00059   * DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR
00060   * SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER
00061   * CAUSED AND ON ANY THEORY OF LIABILITY, W
HETHER IN CONTRACT, STRICT LIABILITY,
00062   * OR TORT (INCLUDING NEGLIGENCE OR OTHERWI
SE) ARISING IN ANY WAY OUT OF THE USE
00063   * OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 POSSIBILITY OF SUCH DAMAGE.
00064   *
00065   ****************************************
**********************************
00066   */
00067
00068 /* Includes ------------------------------
----------------------------------*/
00069 #include "stm32l4r9i_eval_sram.h"
00070
00071 /** @addtogroup BSP
00072   * @{
00073   */
00074
00075 /** @addtogroup STM32L4R9I_EVAL
00076   * @{
00077   */
00078
00079 /** @defgroup STM32L4R9I_EVAL_SRAM STM32L4R9
I_EVAL SRAM
00080   * @{
00081   */
00082
00083
00084 /* Private variables ---------------------
----------------------------------*/
00085
00086 /** @defgroup STM32L4R9I_EVAL_SRAM_Private_V
```

```
ariables Private Variables
00087   * @{
00088   */
00089 static SRAM_HandleTypeDef sramHandle;
00090 static FMC_NORSRAM_TimingTypeDef Timing;
00091
00092 /**
00093   * @}
00094   */
00095
00096 /* Private macros ------------------------
-----------------------------------*/
00097 /* Private constants ---------------------
-----------------------------------*/
00098
00099 /** @defgroup STM32L4R9I_EVAL_SRAM_Private_C
onstants Private Constants
00100   * @{
00101   */
00102 /* Timings for SRAM IS61WV102416BLL-10MLI */
00103 #define SRAM_ADDR_SETUP_TIME       2
/* 10ns with a clock at 120 MHz (period of 8.33 ns
) */
00104 #define SRAM_DATA_SETUP_TIME       1
/* 6ns with a clock at 120 MHz (period of 8.33 ns)
 */
00105 #define SRAM_DATA_HOLD_TIME        0
00106 #define SRAM_TURN_AROUND_TIME      1
/* 4ns with a clock at 120 MHz (period of 8.33 ns)
 */
00107
00108 /**
00109   * @}
00110   */
00111
00112 /* Private function prototypes ------------
-----------------------------------*/
```

```
00113
00114 /** @defgroup STM32L4R9I_EVAL_SRAM_Private_F
unctions Private Functions
00115   * @{
00116   */
00117 static void SRAM_MspInit(void);
00118
00119 /**
00120   * @}
00121   */
00122
00123 /* Private functions ---------------------
-----------------------------------*/
00124
00125 /** @defgroup STM32L4R9I_EVAL_SRAM_Exported_
Functions Exported Functions
00126   * @{
00127   */
00128
00129 /**
00130   * @brief  Initializes the SRAM device.
00131   * @retval SRAM status
00132   */
00133 uint8_t BSP_SRAM_Init(void)
00134 {
00135   sramHandle.Instance  = FMC_NORSRAM_DEVICE;
00136   sramHandle.Extended  = FMC_NORSRAM_EXTENDE
D_DEVICE;
00137
00138   /* SRAM device configuration */
00139   Timing.AddressSetupTime      = SRAM_ADDR_S
ETUP_TIME;
00140   Timing.AddressHoldTime       = 1;
          /* Min value, Don't care on SRAM Access
 mode A */
00141   Timing.DataSetupTime         = SRAM_DATA_S
ETUP_TIME;
```

```
00142   Timing.DataHoldTime          = SRAM_DATA_H
OLD_TIME;
00143   Timing.CLKDivision           = 2;
           /* Min value, Don't care on SRAM Access
 mode A */
00144   Timing.DataLatency           = 2;
           /* Min value, Don't care on SRAM Access
 mode A */
00145   Timing.BusTurnAroundDuration = SRAM_TURN_A
ROUND_TIME;
00146   Timing.AccessMode            = FMC_ACCESS_
MODE_A;
00147
00148   sramHandle.Init.NSBank             = FMC_N
ORSRAM_BANK1;
00149   sramHandle.Init.DataAddressMux     = FMC_D
ATA_ADDRESS_MUX_DISABLE;
00150   sramHandle.Init.MemoryType         = FMC_M
EMORY_TYPE_SRAM;
00151   sramHandle.Init.MemoryDataWidth    = FMC_N
ORSRAM_MEM_BUS_WIDTH_16;
00152   sramHandle.Init.BurstAccessMode    = FMC_B
URST_ACCESS_MODE_DISABLE;
00153   sramHandle.Init.WriteOperation     = FMC_W
RITE_OPERATION_ENABLE;
00154   sramHandle.Init.ExtendedMode       = FMC_E
XTENDED_MODE_DISABLE;
00155   sramHandle.Init.AsynchronousWait   = FMC_A
SYNCHRONOUS_WAIT_DISABLE;
00156   sramHandle.Init.WriteBurst         = FMC_W
RITE_BURST_DISABLE;
00157   sramHandle.Init.ContinuousClock    = FMC_C
ONTINUOUS_CLOCK_SYNC_ONLY;
00158   sramHandle.Init.WriteFifo          = FMC_W
RITE_FIFO_DISABLE;
00159   sramHandle.Init.NBLSetupTime       = 0;
00160   sramHandle.Init.PageSize           = FMC_P
```

```
AGE_SIZE_NONE;
00161
00162   /* SRAM controller initialization */
00163   SRAM_MspInit();
00164   if(HAL_SRAM_Init(&sramHandle, &Timing, &Ti
ming) != HAL_OK)
00165   {
00166     return SRAM_ERROR;
00167   }
00168   else
00169   {
00170     return SRAM_OK;
00171   }
00172 }
00173
00174 /**
00175   * @brief  Reads an amount of data from the
 SRAM device in polling mode.
00176   * @param  uwStartAddress: Read start addre
ss
00177   * @param  pData: Pointer to data to be read

00178   * @param  uwDataSize: Size of read data fr
om the memory
00179   * @retval SRAM status
00180   */
00181 uint8_t BSP_SRAM_ReadData(uint32_t uwStartAd
dress, uint16_t *pData, uint32_t uwDataSize)
00182 {
00183   if(HAL_SRAM_Read_16b(&sramHandle, (uint32_
t *)uwStartAddress, pData, uwDataSize) != HAL_OK)
00184   {
00185     return SRAM_ERROR;
00186   }
00187   else
00188   {
00189     return SRAM_OK;
```

```
00190   }
00191 }
00192
00193 /**
00194   * @brief  Reads an amount of data from the
 SRAM device in DMA mode.
00195   * @param  uwStartAddress: Read start addre
ss
00196   * @param  pData: Pointer to data to be read

00197   * @param  uwDataSize: Size of read data fr
om the memory
00198   * @retval SRAM status
00199   */
00200 uint8_t BSP_SRAM_ReadData_DMA(uint32_t uwSta
rtAddress, uint16_t *pData, uint32_t uwDataSize)
00201 {
00202   if(HAL_SRAM_Read_DMA(&sramHandle, (uint32_
t *)uwStartAddress, (uint32_t *)pData, uwDataSize)
 != HAL_OK)
00203   {
00204     return SRAM_ERROR;
00205   }
00206   else
00207   {
00208     return SRAM_OK;
00209   }
00210 }
00211
00212 /**
00213   * @brief  Writes an amount of data from th
e SRAM device in polling mode.
00214   * @param  uwStartAddress: Write start addr
ess
00215   * @param  pData: Pointer to data to be wri
tten
00216   * @param  uwDataSize: Size of written data
```

```
 from the memory
00217   * @retval SRAM status
00218   */
00219 uint8_t BSP_SRAM_WriteData(uint32_t uwStartA
ddress, uint16_t *pData, uint32_t uwDataSize)
00220 {
00221   if(HAL_SRAM_Write_16b(&sramHandle, (uint32
_t *)uwStartAddress, pData, uwDataSize) != HAL_OK)
00222   {
00223     return SRAM_ERROR;
00224   }
00225   else
00226   {
00227     return SRAM_OK;
00228   }
00229 }
00230
00231 /**
00232   * @brief  Writes an amount of data from th
e SRAM device in DMA mode.
00233   * @param  uwStartAddress: Write start addr
ess
00234   * @param  pData: Pointer to data to be wri
tten
00235   * @param  uwDataSize: Size of written data
 from the memory
00236   * @retval SRAM status
00237   */
00238 uint8_t BSP_SRAM_WriteData_DMA(uint32_t uwSt
artAddress, uint16_t *pData, uint32_t uwDataSize)
00239 {
00240   if(HAL_SRAM_Write_DMA(&sramHandle, (uint32
_t *)uwStartAddress, (uint32_t *)pData, uwDataSize
) != HAL_OK)
00241   {
00242     return SRAM_ERROR;
00243   }
```

```
00244   else
00245   {
00246     return SRAM_OK;
00247   }
00248 }
00249
00250 /**
00251   * @brief  Handles SRAM DMA transfer interr
upt request.
00252   * @retval None
00253   */
00254 void BSP_SRAM_DMA_IRQHandler(void)
00255 {
00256   HAL_DMA_IRQHandler(sramHandle.hdma);
00257 }
00258
00259 /** @addtogroup STM32L476G_EVAL_SRAM_Private
_Functions
00260   * @{
00261   */
00262
00263 /**
00264   * @brief  Initializes SRAM MSP.
00265   * @retval None
00266   */
00267 static void SRAM_MspInit(void)
00268 {
00269   static DMA_HandleTypeDef dmaHandle;
00270   GPIO_InitTypeDef gpioinitstruct;
00271   SRAM_HandleTypeDef *hsram = &sramHandle;
00272
00273   /* Enable FMC clock */
00274   __HAL_RCC_FMC_CLK_ENABLE();
00275
00276   /* Enable chosen DMAx clock */
00277   SRAM_DMAx_CLK_ENABLE();
00278
```

```
00279   /* Enable GPIOs clock */
00280   __HAL_RCC_GPIOD_CLK_ENABLE();
00281   __HAL_RCC_GPIOE_CLK_ENABLE();
00282   __HAL_RCC_GPIOF_CLK_ENABLE();
00283   __HAL_RCC_GPIOG_CLK_ENABLE();
00284   __HAL_RCC_PWR_CLK_ENABLE();
00285   HAL_PWREx_EnableVddIO2();
00286
00287   /* Common GPIO configuration */
00288   gpioinitstruct.Mode      = GPIO_MODE_AF_PP
;
00289   gpioinitstruct.Speed     = GPIO_SPEED_FREQ
_VERY_HIGH;
00290   gpioinitstruct.Alternate = GPIO_AF12_FMC;
00291
00292   /*## NE configuration #######*/
00293   /* NE1 : SRAM */
00294   gpioinitstruct.Pull = GPIO_PULLUP;
00295   gpioinitstruct.Pin  = GPIO_PIN_7;
00296   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00297   /* NE3 : NOR */
00298   /* NE4 : TFT LCD */
00299   gpioinitstruct.Pin = GPIO_PIN_10 | GPIO_PI
N_12;
00300   HAL_GPIO_Init(GPIOG, &gpioinitstruct);
00301
00302   /*## NOE and NWE configuration #######*/
00303   gpioinitstruct.Pin = GPIO_PIN_4 | GPIO_PIN
_5;
00304   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00305
00306   /*## NBL0, NBL1 configuration #######*/
00307   gpioinitstruct.Pin = GPIO_PIN_0 | GPIO_PIN
_1;
00308   HAL_GPIO_Init(GPIOE, &gpioinitstruct);
00309
00310   /*## Address Bus #######*/
```

```
00311   /* GPIOD configuration */
00312   gpioinitstruct.Pull = GPIO_NOPULL;
00313   gpioinitstruct.Pin  = GPIO_PIN_11 | GPIO_P
IN_12 | GPIO_PIN_13;
00314   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00315
00316   /* GPIOE configuration */
00317   gpioinitstruct.Pin = GPIO_PIN_3 | GPIO_PIN
_4 | GPIO_PIN_5;
00318   HAL_GPIO_Init(GPIOE, &gpioinitstruct);
00319
00320   /* GPIOF configuration */
00321   gpioinitstruct.Pin = GPIO_PIN_0 | GPIO_PIN
_1 | GPIO_PIN_2 | GPIO_PIN_3 |
00322                        GPIO_PIN_4 | GPIO_PIN
_5 | GPIO_PIN_12 | GPIO_PIN_13 |
00323                        GPIO_PIN_14 | GPIO_PI
N_15;
00324   HAL_GPIO_Init(GPIOF, &gpioinitstruct);
00325
00326   /* GPIOG configuration */
00327   gpioinitstruct.Pin = GPIO_PIN_0 | GPIO_PIN
_1 | GPIO_PIN_2 |
00328                        GPIO_PIN_3 | GPIO_PIN
_4 | GPIO_PIN_5;
00329   HAL_GPIO_Init(GPIOG, &gpioinitstruct);
00330
00331   /*## Data Bus #######*/
00332   /* GPIOD configuration */
00333   gpioinitstruct.Pin = GPIO_PIN_0 | GPIO_PIN
_1 | GPIO_PIN_8 | GPIO_PIN_9 |
00334                        GPIO_PIN_10 | GPIO_PI
N_14 | GPIO_PIN_15;
00335   HAL_GPIO_Init(GPIOD, &gpioinitstruct);
00336
00337   /* GPIOE configuration */
00338   gpioinitstruct.Pin = GPIO_PIN_7 | GPIO_PIN
```

```
_8 | GPIO_PIN_9 | GPIO_PIN_10 |
00339                           GPIO_PIN_11 | GPIO_PI
N_12 | GPIO_PIN_13 |
00340                           GPIO_PIN_14 | GPIO_PI
N_15;
00341   HAL_GPIO_Init(GPIOE, &gpioinitstruct);
00342
00343   /* Configure common DMA parameters */
00344   dmaHandle.Init.Direction           = DMA_M
EMORY_TO_MEMORY;
00345   dmaHandle.Init.PeriphInc           = DMA_P
INC_ENABLE;
00346   dmaHandle.Init.MemInc              = DMA_M
INC_ENABLE;
00347   dmaHandle.Init.PeriphDataAlignment = DMA_P
DATAALIGN_HALFWORD;
00348   dmaHandle.Init.MemDataAlignment    = DMA_M
DATAALIGN_HALFWORD;
00349   dmaHandle.Init.Mode                = DMA_N
ORMAL;
00350   dmaHandle.Init.Priority            = DMA_P
RIORITY_HIGH;
00351
00352   dmaHandle.Instance = SRAM_DMAx_CHANNEL;
00353
00354    /* Associate the DMA handle */
00355   __HAL_LINKDMA(hsram, hdma, dmaHandle);
00356
00357   /* Deinitialize the DMA channel for new tr
ansfer */
00358   HAL_DMA_DeInit(&dmaHandle);
00359
00360   /* Configure the DMA channel */
00361   HAL_DMA_Init(&dmaHandle);
00362
00363   /* NVIC configuration for DMA transfer com
plete interrupt */
```

```
00364   HAL_NVIC_SetPriority(SRAM_DMAx_IRQn, 5, 0)
;
00365   HAL_NVIC_EnableIRQ(SRAM_DMAx_IRQn);
00366 }
00367
00368 /**
00369   * @}
00370   */
00371
00372 /**
00373   * @}
00374   */
00375
00376 /**
00377   * @}
00378   */
00379
00380 /**
00381   * @}
00382   */
00383
00384 /**
00385   * @}
00386   */
00387
00388 /************************ (C) COPYRIGHT STMi
croelectronics *****END OF FILE****/
```

STM32L4R9I_EVAL BSP User Manual

BSP

Define for STM32L4R9I_EVAL board. More...

Modules

STM32L4R9I_EVAL

Detailed Description

Define for STM32L4R9I_EVAL board.

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL

BSP

Modules

STM32L4R9I_EVAL Common

STM32L4R9I_EVAL AUDIO

This file includes the low layer driver for wm8994 Audio Codec available on STM32L4R9I-EVAL evaluation board(MB1313).

STM32L4R9I_EVAL LCD

STM32L4R9I_EVAL TS

STM32L4R9I_EVAL EEPROM

This file includes the I2C EEPROM driver of STM32L4R9I-EVAL evaluation board.

STM32L4R9I_EVAL IDD

This file includes the Idd driver for STM32L4R9I_EVAL board.

STM32L4R9I_EVAL IO

STM32L4R9I_EVAL NOR

STM32L4R9I_EVAL OSPI NOR

STM32L4R9I_EVAL OSPI RAM

STM32L4R9I_EVAL SD

STM32L4R9I_EVAL SRAM

STM32L4R9I_EVAL BSP User Manual

Exported Constants

STM32L4R9I_EVAL Common

Modules

- LED Constants
- BUTTON Constants
- COM Constants
- POTENTIOMETER Constants
- BUS Constants

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL AUDIO

STM32L4R9I_EVAL

This file includes the low layer driver for wm8994 Audio Codec available on STM32L4R9I-EVAL evaluation board(MB1313). More...

Modules

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

Detailed Description

This file includes the low layer driver for wm8994 Audio Codec available on STM32L4R9I-EVAL evaluation board(MB1313).

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL LCD

STM32L4R9I_EVAL

Modules

- **STM32L4R9I_EVAL DSI LCD**
- **Exported Constants**
- **Exported Types**
- **STM32L4R9I_EVAL RGB LCD**
- **Exported Functions**

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL RGB LCD

STM32L4R9I_EVAL LCD

Modules

- **Exported Constants**
- **Exported Variables**
- **Exported Functions**

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL DSI TS

STM32L4R9I_EVAL TS

Modules

Exported Functions

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL RGB TS

STM32L4R9I_EVAL TS

Modules

Imported Variables

STM32L4R9I_EVAL_RGB_TS_Exported_Functions

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL EEPROM

This file includes the I2C EEPROM driver of STM32L4R9I-EVAL evaluation board. More...

Modules

- Private Variables
- Exported Functions
- Exported Constants

Detailed Description

This file includes the I2C EEPROM driver of STM32L4R9I-EVAL evaluation board.

This file includes the I2C EEPROM driver of STM32L4R9I_EVAL evaluation board.

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL IDD

STM32L4R9I_EVAL

This file includes the Idd driver for STM32L4R9I_EVAL board. More...

Modules

- Private Variables
- Exported Functions
- Exported Defines

Detailed Description

This file includes the Idd driver for STM32L4R9I_EVAL board.

It allows user to measure MCU Idd current on board, especially in different low power modes.

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL IO

STM32L4R9I_EVAL

Modules

- Private Variables
- Exported Types
- Exported Constants
- Exported Functions

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL NOR

STM32L4R9I_EVAL

Modules

- Private Constants
- Private Variables
- Private Functions
- Exported Functions
- Exported Constants

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL
OSPI NOR

STM32L4R9I_EVAL

Modules

- Private Constants
- Private Variables
- Private Functions
- Exported Constants
- Exported Types
- Exported Functions

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL OSPI RAM

STM32L4R9I_EVAL

Modules

- Private Constants
- Private Variables
- Private Functions
- Exported Constants
- Exported Functions

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL SD

STM32L4R9I_EVAL

Modules

- Private Variables
- Exported Types
- Exported Constants
- Exported Variables
- Exported Functions

STM32L4R9I_EVAL BSP User Manual

STM32L4R9I_EVAL SRAM

STM32L4R9I_EVAL

Modules

- Private Variables
- Private Constants
- Private Functions
- Exported Functions
- Exported Constants