# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LOW LEVEL Private TypesDefinitions**

STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LOW LEVEL Private Macros

STM32F429I DISCOVERY LOW LEVEL

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LOW LEVEL Exported Macros**

STM32F429I DISCOVERY LOW LEVEL

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY GYROSCOPE Private Defines**

STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY GYROSCOPE Private Macros

# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY GYROSCOPE Private Function Prototypes**

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY GYROSCOPE**

**Exported Constants**

STM32F429I DISCOVERY GYROSCOPE

### STM32F429I DISCOVERY GYROSCOPE

**Exported Macros**

STM32F429I DISCOVERY GYROSCOPE

## STM32F429I DISCOVERY IO Private Types Definitions

STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY IO Private Macros**

STM32F429I DISCOVERY IO

# STM32F429I DISCOVERY IO Exported Macros

STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LCD Private TypesDefinitions

#include <stm32f429i_discovery_lcd.h>
### Data Fields

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint32_t</code></td>
<td>TextColor</td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td>BackColor</td>
</tr>
<tr>
<td><code>sFONT *</code></td>
<td>pFont</td>
</tr>
</tbody>
</table>
Detailed Description

Definition at line 77 of file stm32f429i_discovery_lcd.h.
Field Documentation

**uint32_t** LCD_DrawPropTypeDef::BackColor

Definition at line 80 of file *stm32f429i_discovery_lcd.h*.

Referenced by **BSP_LCD_ClearStringLine()**, **BSP_LCD_GetBackColor()**, **BSP_LCD_LayerDefaultInit()**, and **BSP_LCD_SetBackColor()**.

**sFONT*** LCD_DrawPropTypeDef::pFont

Definition at line 81 of file *stm32f429i_discovery_lcd.h*.

Referenced by **BSP_LCD_DisplayChar()**, **BSP_LCD_DisplayStringAt()**, **BSP_LCD_GetFont()**, **BSP_LCD_LayerDefaultInit()**, **BSP_LCD_SetFont()**, and **DrawChar()**.

**uint32_t** LCD_DrawPropTypeDef::TextColor

Definition at line 79 of file *stm32f429i_discovery_lcd.h*.

Referenced by **BSP_LCD_ClearStringLine()**, **BSP_LCD_GetTextColor()**, **BSP_LCD_LayerDefaultInit()**, and **BSP_LCD_SetTextColor()**.

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

* stm32f429i_discovery_lcd.h
Point Struct Reference

STM32F429I DISCOVERY LCD Exported Types

#include <stm32f429i_discovery_lcd.h>
### Data Fields

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>int16_t</td>
<td>X</td>
</tr>
<tr>
<td>int16_t</td>
<td>Y</td>
</tr>
</tbody>
</table>
Detailed Description

Definition at line 84 of file `stm32f429i_discovery_lcd.h`.
Field Documentation

int16_t Point::X

Definition at line 86 of file stm32f429i_discovery_lcd.h.

Referenced by BSP_LCD_DrawPolygon(), and BSP_LCD_FillPolygon().

int16_t Point::Y

Definition at line 87 of file stm32f429i_discovery_lcd.h.

Referenced by BSP_LCD_DrawPolygon(), and BSP_LCD_FillPolygon().

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

- stm32f429i_discovery_lcd.h

STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY SDRAM Private Types Definitions

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY SDRAM Private Defines**

STM32F429I DISCOVERY SDRAM

## STM32F429I DISCOVERY SDRAM Private Macros

| STM32F429I DISCOVERY SDRAM |

## STM32F429I DISCOVERY SDRAM Exported Macro

| STM32F429I DISCOVERY SDRAM |

STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY TS Private Types Definitions

STM32F429I DISCOVERY TS

# STM32F429I DISCOVERY TS Private Defines

STM32F429I-DISCOVERY TS Private Macros

STM32F429I DISCOVERY TS

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
</table>

**STM32F429I DISCOVERY TS Private Function Prototypes**

STM32F429I DISCOVERY TS

#include <stm32f429i_discovery_ts.h>
## Data Fields

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16_t</td>
<td>TouchDetected</td>
</tr>
<tr>
<td>uint16_t</td>
<td>X</td>
</tr>
<tr>
<td>uint16_t</td>
<td>Y</td>
</tr>
<tr>
<td>uint16_t</td>
<td>Z</td>
</tr>
</tbody>
</table>
Detailed Description

Definition at line 67 of file `stm32f429i_discovery_ts.h`. 
Field Documentation

**uint16_t** **TS_StateTypeDef::TouchDetected**

Definition at line 69 of file *stm32f429i_discovery_ts.h*.

Referenced by **BSP_TS_GetState()**.

**uint16_t** **TS_StateTypeDef::X**

Definition at line 70 of file *stm32f429i_discovery_ts.h*.

Referenced by **BSP_TS_GetState()**.

**uint16_t** **TS_StateTypeDef::Y**

Definition at line 71 of file *stm32f429i_discovery_ts.h*.

Referenced by **BSP_TS_GetState()**.

**uint16_t** **TS_StateTypeDef::Z**

Definition at line 72 of file *stm32f429i_discovery_ts.h*.

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

- *stm32f429i_discovery_ts.h*

BSP User Manual by [doxygen](https://www.doxygen.org) 1.7.6.1
# STM32F429I DISCOVERY TS Exported Macros

STM32F429I DISCOVERY EEPROM Exported Types

STM32F429I DISCOVERY EEPROM

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

- BackColor : LCD_DrawPropTypeDef
- pFont : LCD_DrawPropTypeDef
- TextColor : LCD_DrawPropTypeDef
- TouchDetected : TS_StateTypeDef
- X : Point, TS_StateTypeDef
- Y : Point, TS_StateTypeDef
- Z : TS_StateTypeDef
- BackColor : `LCD_DrawPropTypeDef`
- pFont : `LCD_DrawPropTypeDef`
- TextColor : `LCD_DrawPropTypeDef`
- TouchDetected : `TS_StateTypeDef`
- X : `Point`, `TS_StateTypeDef`
- Y : `Point`, `TS_StateTypeDef`
- Z : `TS_StateTypeDef`
stm32f429i_discovery_eeprom.c File Reference

This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory. To be able to use this driver, the switch EE_M24LR64 must be defined in your toolchain compiler preprocessor. More...

#include "stm32f429i_discovery_eeprom.h"

Go to the source code of this file.
Detailed Description

This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory. To be able to use this driver, the switch EE_M24LR64 must be defined in your toolchain compiler preprocessor.

Author:
MCD Application Team

Version:
V2.1.3

Date:
13-January-2016

Notes:
- This driver is intended for STM32F4xx families devices only.
- The I2C EEPROM memory (M24LR64) is available on separate daughter board ANT7-M24LR-A, which is not provided with the STM32F429I DISCOVERY board. To use this driver you have to connect the ANT7-M24LR-A to CN3 connector of STM32F429I DISCOVERY board.

It implements a high level communication layer for read and write from/to this memory. The needed STM32F4xx hardware resources (I2C and GPIO) are defined in `stm32f429i_discovery.h` file, and the initialization is performed in `EEPROM_IO_Init()` function declared in `stm32f429i_discovery.c` file. You can easily tailor this driver to any other development board, by just adapting the defines for hardware resources and `EEPROM_IO_Init()` function.

Note:
In this driver, basic read and write functions
(BSP_EEPROM_ReadBuffer() and
BSP_EEPROM_WritePage()) use DMA mode to perform the data transfer to/from EEPROM memory.
Regarding **BSP_EEPROM_WritePage()**, it is an optimized function to perform small write (less than 1 page) but 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 for checking on Page boundaries).

| Pin assignment for M24LR64 EEPROM | +---------------------------------------+-----------+------ |
|------------------------------------|---------------------------------------|------------+------ |
| STM32F4xx 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 | VDD | 8 (3.3V) | +---------------------------------------+ |

**Attention:**
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 stm32f429i_discovery_eeprom.c.
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- __DMAx_CLK_ENABLE : [stm32f429i_discovery_sdram.h](#)
- __STM32F429I_DISCO_BSP_VERSION :
  - [stm32f429i_discovery.c](#)
- __STM32F429I_DISCO_BSP_VERSION_MAIN :
  - [stm32f429i_discovery.c](#)
- __STM32F429I_DISCO_BSP_VERSION_RC :
  - [stm32f429i_discovery.c](#)
- __STM32F429I_DISCO_BSP_VERSION_SUB1 :
  - [stm32f429i_discovery.c](#)
- __STM32F429I_DISCO_BSP_VERSION_SUB2 :
  - [stm32f429i_discovery.c](#)
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

### - a -

- ABS : [stm32f429i_discovery_lcd.c](stm32f429i_discovery_lcd.c)
- ActiveLayer : [stm32f429i_discovery_lcd.c](stm32f429i_discovery_lcd.c)
STM32F429I-Discovery BSP User Manual

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

- **b** -

- BSP_EEPROM_Init() : `stm32f429i_discovery_eeprom.h`
- BSP_EEPROM_ReadBuffer() : `stm32f429i_discovery_eeprom.h`
- BSP_EEPROM_TIMEOUT_UserCallback() : `stm32f429i_discovery_eeprom.h`
- BSP_EEPROM_WaitEepromStandbyState() : `stm32f429i_discovery_eeprom.h`
- BSP_EEPROM_WriteBuffer() : `stm32f429i_discovery_eeprom.h`
- BSP_EEPROM_WritePage() : `stm32f429i_discovery_eeprom.h`
- BSP_GetVersion() : `stm32f429i_discovery.h`, `stm32f429i_discovery.c`
- BSP_GYRO_DisableIT() : `stm32f429i_discovery_gyroscope.c`, `stm32f429i_discovery_gyroscope.h`
- BSP_GYRO_EnableIT() : `stm32f429i_discovery_gyroscope.c`, `stm32f429i_discovery_gyroscope.h`
- BSP_GYRO_GetXYZ() : `stm32f429i_discovery_gyroscope.c`, `stm32f429i_discovery_gyroscope.h`
- BSP_GYRO_Init() : `stm32f429i_discovery_gyroscope.h`, `stm32f429i_discovery_gyroscope.c`
- BSP_GYRO_ITConfig() : `stm32f429i_discovery_gyroscope.c`,
- BSP_GYRO_ReadID(): stm32f429i_discovery_gyroscope.c, stm32f429i_discovery_gyroscope.h
- BSP_GYRO_Reset(): stm32f429i_discovery_gyroscope.c, stm32f429i_discovery_gyroscope.h
- BSP_IO_ConfigPin(): stm32f429i_discovery_io.c, stm32f429i_discovery_io.h
- BSP_IO_Init(): stm32f429i_discovery_io.c, stm32f429i_discovery_io.h
- BSP_IO_ITClear(): stm32f429i_discovery_io.c, stm32f429i_discovery_io.h
- BSP_IO_ITGetStatus(): stm32f429i_discovery_io.c, stm32f429i_discovery_io.h
- BSP_IO_ReadPin(): stm32f429i_discovery_io.h, stm32f429i_discovery_io.c
- BSP_IO_TogglePin(): stm32f429i_discovery_io.c, stm32f429i_discovery_io.h
- BSP_IO_WritePin(): stm32f429i_discovery_io.c, stm32f429i_discovery_io.h
- BSP_LCD_Clear(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_ClearStringLine(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayChar(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayOff(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayOn(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayStringAt(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayStringAtLine(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawBitmap(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawCircle(): stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawEllipse() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawHLine() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawLine() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawPixel() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawPolygon() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawRect() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawVLine() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillCircle() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillEllipse() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillPolygon() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillRect() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillTriangle() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetBackColor() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetFont() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetTextColor() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetXSize() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetYSize() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_Init() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_LayerDefaultInit() : stm32f429i_discovery_lcd.c,
• BSP_LCD_ReadPixel() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
• BSP_LCD_ResetColorKeying() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
• BSP_LCD_SelectLayer() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
• BSP_LCD_SetBackColor() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
• BSP_LCD_SetColorKeying() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
• BSP_LCD_SetFont() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
• BSP_LCD_SetLayerAddress() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
• BSP_LCD_SetLayerVisible() : stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
• BSP_LCD_SetLayerWindow() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
• BSP_LCD_SetTextColor() : stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
• BSP_LCD_SetTransparency() : stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
• BSP_LED_Init() : stm32f429i_discovery.h, stm32f429i_discovery.c
• BSP_LED_Off() : stm32f429i_discovery.h, stm32f429i_discovery.c
• BSP_LED_On() : stm32f429i_discovery.c, stm32f429i_discovery.h
• BSP_LED_Toggle() : stm32f429i_discovery.c, stm32f429i_discovery.h
• BSP_PB_GetState() : stm32f429i_discovery.c, stm32f429i_discovery.h
• BSP_PB_Init() : stm32f429i_discovery.c, stm32f429i_discovery.h
• BSP_SDRAM_DMA_IRQHandler() : stm32f429i_discovery_sdram.h,
stm32f429i_discovery_sdram.c

- BSP_SDRAM_Init() : stm32f429i_discovery_sdram.c, stm32f429i_discovery_sdram.h
- BSP_SDRAM_Initialization_sequence() : 
  stm32f429i_discovery_sdram.h, stm32f429i_discovery_sdram.c
- BSP_SDRAM_ReadData() : stm32f429i_discovery_sdram.c, 
  stm32f429i_discovery_sdram.h
- BSP_SDRAM_ReadData_DMA() : 
  stm32f429i_discovery_sdram.h, 
  stm32f429i_discovery_sdram.c
- BSP_SDRAM_Sendcmd() : 
  stm32f429i_discovery_sdram.c, 
  stm32f429i_discovery_sdram.h
- BSP_SDRAM_WriteData() : 
  stm32f429i_discovery_sdram.h, 
  stm32f429i_discovery_sdram.c
- BSP_SDRAM_WriteData_DMA() : 
  stm32f429i_discovery_sdram.c, 
  stm32f429i_discovery_sdram.h
- BSP_TS_GetState() : 
  stm32f429i_discovery_ts.h, 
  stm32f429i_discovery_ts.c
- BSP_TS_Init() : 
  stm32f429i_discovery_ts.h, 
  stm32f429i_discovery_ts.c
- BSP_TS_ITClear() : 
  stm32f429i_discovery_ts.c, 
  stm32f429i_discovery_ts.h
- BSP_TS_ITConfig() : 
  stm32f429i_discovery_ts.c, 
  stm32f429i_discovery_ts.h
- BSP_TS_ITGetStatus() : 
  stm32f429i_discovery_ts.h, 
  stm32f429i_discovery_ts.c
- BUFFER_OFFSET : 
  stm32f429i_discovery_lcd.h
- BUTTONIRQn : 
  stm32f429i_discovery.c
- BUTTONKEY : 
  stm32f429i_discovery.h
- BUTTONMODE_EXTI : 
  stm32f429i_discovery.h
- BUTTONMODE_GPIO : 
  stm32f429i_discovery.h
- BUTTON_PIN : 
  stm32f429i_discovery.c
- BUTTON_PORT : 
  stm32f429i_discovery.c
- Button_TypeDef : 
  stm32f429i_discovery.c
- ButtonMode_TypeDef : 
  stm32f429i_discovery.h
- BUTTONn : *stm32f429i_discovery.h*
- BUTTONx_GPIO_CLK_DISABLE : *stm32f429i_discovery.h*
- BUTTONx_GPIO_CLK_ENABLE : *stm32f429i_discovery.h*
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- c -

- CENTER_MODE : [stm32f429i_discovery_lcd.h](#)
- Command : [stm32f429i_discovery_sdram.c](#)
- ConvertLineToARGB8888() : [stm32f429i_discovery_lcd.c](#)
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- d -

- DISCOVERY_I2Cx : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_CLOCK_ENABLE : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_ER_IRQn : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_EV_IRQn : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_FORCE_RESET : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_RELEASE_RESET : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_SCL_GPIO_PORT : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_SCL_PIN : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_SCL_SDA_AF : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_SDA_GPIO_PORT : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_SDA_PIN : [stm32f429i_discovery.h](#)
- DISCOVERY_SPIx : [stm32f429i_discovery.h](#)
- DISCOVERY_SPIx_AF : [stm32f429i_discovery.h](#)
- DISCOVERY_SPIx_CLK_ENABLE: stm32f429i_discovery.h
- DISCOVERY_SPIx_GPIO_CLK_DISABLE: stm32f429i_discovery.h
- DISCOVERY_SPIx_GPIO_CLK_ENABLE: stm32f429i_discovery.h
- DISCOVERY_SPIx_GPIO_PORT: stm32f429i_discovery.h
- DISCOVERY_SPIx_MISO_PIN: stm32f429i_discovery.h
- DISCOVERY_SPIx_MOSI_PIN: stm32f429i_discovery.h
- DISCOVERY_SPIx_SCK_PIN: stm32f429i_discovery.h
- Dma2dHandler: stm32f429i_discovery_lcd.c
- DrawChar(): stm32f429i_discovery_lcd.c
- DrawProp: stm32f429i_discovery_lcd.c
- DUMMY_BYTE: stm32f429i_discovery.h
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- e -

- EEPROM_FAIL : **stm32f429i_discovery_eeprom.h**
- EEPROM_IO_Init() : **stm32f429i_discovery_eeprom.h**
- EEPROM_IO_IsDeviceReady() : **stm32f429i_discovery_eeprom.h**
- EEPROM_IO_ReadData() : **stm32f429i_discovery_eeprom.h**
- EEPROM_IO_WriteData() : **stm32f429i_discovery_eeprom.h**
- EEPROM_MAX_SIZE : **stm32f429i_discovery_eeprom.h**
- EEPROM_MAX_TRIALS : **stm32f429i_discovery_eeprom.h**
- EEPROM_OK : **stm32f429i_discovery_eeprom.h**
- EEPROM_PAGESIZE : **stm32f429i_discovery_eeprom.h**
- EEPROM_READ_TIMEOUT : **stm32f429i_discovery_eeprom.h**
- EEPROM_TIMEOUT : **stm32f429i_discovery_eeprom.h**
- EEPROM_WRITE_TIMEOUT : **stm32f429i_discovery_eeprom.h**
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- f -

- FillBuffer() : `stm32f429i_discovery_lcd.c`
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- **g** -

- `GPIO_PIN` : [stm32f429i_discovery.c](#)
- `GPIO_PORT` : [stm32f429i_discovery.c](#)
- `GYRO_CS_GPIO_CLK_DISABLE` : [stm32f429i_discovery.h](#)
- `GYRO_CS_GPIO_CLK_ENABLE` : [stm32f429i_discovery.h](#)
- `GYRO_CS_GPIO_PORT` : [stm32f429i_discovery.h](#)
- `GYRO_CS_HIGH` : [stm32f429i_discovery.h](#)
- `GYRO_CS_LOW` : [stm32f429i_discovery.h](#)
- `GYRO_CS_PIN` : [stm32f429i_discovery.h](#)
- `GYRO_ERROR` : [stm32f429i_discovery_gyroscope.h](#)
- `GYRO_INT1_EXTI_IRQn` : [stm32f429i_discovery.h](#)
- `GYRO_INT1_PIN` : [stm32f429i_discovery.h](#)
- `GYRO_INT2_EXTI_IRQn` : [stm32f429i_discovery.h](#)
- `GYRO_INT2_PIN` : [stm32f429i_discovery.h](#)
- `GYRO_INT_GPIO_CLK_DISABLE` : [stm32f429i_discovery.h](#)
- `GYRO_INT_GPIO_CLK_ENABLE` : [stm32f429i_discovery.h](#)
- `GYRO_INT_GPIO_PORT` : [stm32f429i_discovery.h](#)
- `GYRO_IO_Init()` : [stm32f429i_discovery.c](#)
- `GYRO_IO_Read()` : [stm32f429i_discovery.c](#)
- `GYRO_IO_Write()` : [stm32f429i_discovery.c](#)
- `GYRO_OK`: `stm32f429i_discovery_gyroscope.h`
- `GYRO_StatusTypeDef`: `stm32f429i_discovery_gyroscope.h`
- `GYRO_TIMEOUT`: `stm32f429i_discovery_gyroscope.h`
- `GyroscopeDrv`: `stm32f429i_discovery_gyroscope.c`
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- **i** -

- l2cHandle : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2Cx_Error() : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2Cx_Init() : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2Cx_ITConfig() : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2Cx_MspInit() : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2Cx_ReadBuffer() : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2Cx_ReadData() : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2Cx_TIMEOUT_MAX : [stm32f429i_discovery.h](stm32f429i_discovery.h)
- l2Cx_WriteBuffer() : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2Cx_WriteData() : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- l2cxTimeout : [stm32f429i_discovery.c](stm32f429i_discovery.c)
- IO_ERROR : [stm32f429i_discovery_io.h](stm32f429i_discovery_io.h)
- IO_I2C_ADDRESS : [stm32f429i_discovery.h](stm32f429i_discovery.h)
- IO_OK : [stm32f429i_discovery_io.h](stm32f429i_discovery_io.h)
- IO_PIN_0 : [stm32f429i_discovery_io.h](stm32f429i_discovery_io.h)
- IO_PIN_1 : [stm32f429i_discovery_io.h](stm32f429i_discovery_io.h)
- IO_PIN_2 : [stm32f429i_discovery_io.h](stm32f429i_discovery_io.h)
- IO_PIN_3 : [stm32f429i_discovery_io.h](stm32f429i_discovery_io.h)
- IO_PIN_4 : [stm32f429i_discovery_io.h](stm32f429i_discovery_io.h)
- IO_PIN_5 : stm32f429i_discovery_io.h
- IO_PIN_6 : stm32f429i_discovery_io.h
- IO_PIN_7 : stm32f429i_discovery_io.h
- IO_PIN_ALL : stm32f429i_discovery_io.h
- IO_StatusTypeDef : stm32f429i_discovery_io.h
- IO_TIMEOUT : stm32f429i_discovery_io.h
- IoDrv : stm32f429i_discovery_io.c
- IOE_Delay() : stm32f429i_discovery.c
- IOE_Init() : stm32f429i_discovery.c
- IOE_ITConfig() : stm32f429i_discovery.c
- IOE_Read() : stm32f429i_discovery.c
- IOE_ReadMultiple() : stm32f429i_discovery.c
- IOE_Write() : stm32f429i_discovery.c
- IOE_WriteMultiple() : stm32f429i_discovery.c
- Is_LCD_IO_Initialized : stm32f429i_discovery.c
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- k -

- **KEY_BUTTON_EXTI_IRQn**: `stm32f429i_discovery.h`
- **KEY_BUTTON_GPIO_CLK_DISABLE**: `stm32f429i_discovery.h`
- **KEY_BUTTON_GPIO_CLK_ENABLE**: `stm32f429i_discovery.h`
- **KEY_BUTTON_GPIO_PORT**: `stm32f429i_discovery.h`
- **KEY_BUTTON_PIN**: `stm32f429i_discovery.h`
STM32F429I-Discovery BSP User Manual

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

- [ ]

- **LCD_BACKGROUND_LAYER**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_BLACK**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_BLUE**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_BROWN**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_CYAN**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_DARKBLUE**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_DARKCYAN**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_DARKGRAY**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_DARKGREEN**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_DARKMAGENTA**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_DARKRED**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_DARKYELLOW**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_GRAY**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_GREEN**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_LIGHTBLUE**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_LIGHTCYAN**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_LIGHTGRAY**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_LIGHTGREEN**: [stm32f429i_discovery_lcd.h](#)
- **LCD_COLOR_LIGHTMAGENTA**: [stm32f429i_discovery_lcd.h](#)
- LCD_COLOR_LIGHTRED : stm32f429i_discovery_lcd.h
- LCD_COLOR_LIGHTYELLOW : stm32f429i_discovery_lcd.h
- LCD_COLOR_MAGENTA : stm32f429i_discovery_lcd.h
- LCD_COLOR_ORANGE : stm32f429i_discovery_lcd.h
- LCD_COLOR_RED : stm32f429i_discovery_lcd.h
- LCD_COLOR_TRANSPARENT : stm32f429i_discovery_lcd.h
- LCD_COLOR_WHITE : stm32f429i_discovery_lcd.h
- LCD_COLOR_YELLOW : stm32f429i_discovery_lcd.h
- LCD_CS_HIGH : stm32f429i_discovery.h
- LCD_CS_LOW : stm32f429i_discovery.h
- LCD_DEFAULT_FONT : stm32f429i_discovery_lcd.h
- LCD_Delay() : stm32f429i_discovery.c
- LCD_ERROR : stm32f429i_discovery_lcd.h
- LCD_FOREGROUND_LAYER : stm32f429i_discovery_lcd.h
- LCD_FRAME_BUFFER : stm32f429i_discovery_lcd.h
- LCD_IO_Init() : stm32f429i_discovery.c
- LCD_IO_ReadData() : stm32f429i_discovery.c
- LCD_IO_WriteData() : stm32f429i_discovery.c
- LCD_IO_WriteReg() : stm32f429i_discovery.c
- LCD_LayerCfgTypeDef : stm32f429i_discovery_lcd.h
- LCD_NCS_GPIO_CLK_DISABLE : stm32f429i_discovery.h
- LCD_NCS_GPIO_CLK_ENABLE : stm32f429i_discovery.h
- LCD_NCS_GPIO_PORT : stm32f429i_discovery.h
- LCD_NCS_PIN : stm32f429i_discovery.h
- LCD_OK : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_AL44 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_AL88 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_ARGB1555 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_ARGB4444 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_ARGB8888 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_L8 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_RGB565 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_RGB888 : stm32f429i_discovery_lcd.h
- LCD_RDX_GPIO_CLK_DISABLE : stm32f429i_discovery_h
• LCD_RDX_GPIO_CLK_ENABLE : stm32f429i_discovery.h
• LCD_RDX_GPIO_PORT : stm32f429i_discovery.h
• LCD_RDX_HIGH : stm32f429i_discovery.h
• LCD_RDX_LOW : stm32f429i_discovery.h
• LCD_RDX_PIN : stm32f429i_discovery.h
• LCD_StatusTypeDef : stm32f429i_discovery_lcd.h
• LCD_TIMEOUT : stm32f429i_discovery_lcd.h
• LCD_WRX_GPIO_CLK_DISABLE : stm32f429i_discovery.h
• LCD_WRX_GPIO_CLK_ENABLE : stm32f429i_discovery.h
• LCD_WRX_GPIO_PORT : stm32f429i_discovery.h
• LCD_WRX_HIGH : stm32f429i_discovery.h
• LCD_WRX_LOW : stm32f429i_discovery.h
• LCD_WRX_PIN : stm32f429i_discovery.h
• LcdDrv : stm32f429i_discovery_lcd.c
• LED3 : stm32f429i_discovery.h
• LED3_GPIO_CLK_DISABLE : stm32f429i_discovery.h
• LED3_GPIO_CLK_ENABLE : stm32f429i_discovery.h
• LED3_GPIO_PORT : stm32f429i_discovery.h
• LED3_PIN : stm32f429i_discovery.h
• LED4 : stm32f429i_discovery.h
• LED4_GPIO_CLK_DISABLE : stm32f429i_discovery.h
• LED4_GPIO_CLK_ENABLE : stm32f429i_discovery.h
• LED4_GPIO_PORT : stm32f429i_discovery.h
• LED4_PIN : stm32f429i_discovery.h
• Led_TypeDef : stm32f429i_discovery.h
• LEDn : stm32f429i_discovery.h
• LEDx_GPIO_CLK_DISABLE : stm32f429i_discovery.h
• LEDx_GPIO_CLK_ENABLE : stm32f429i_discovery.h
• LEFT_MODE : stm32f429i_discovery_lcd.h
• LtducHandler : stm32f429i_discovery_lcd.c
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- m -

- MAX_LAYER_NUMBER : [stm32f429i_discovery_lcd.h](#)
-MspInit() : [stm32f429i_discovery_lcd.c](#), [stm32f429i_discovery_sdram.c](#)
- MULTIPLEBYTE_CMD : [stm32f429i_discovery.h](#)
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- p -

- PeriphClkInitStruct : [stm32f429i_discovery_lcd.c](stm32f429i_discovery_lcd.c)
- POLY_X : [stm32f429i_discovery_lcd.c](stm32f429i_discovery_lcd.c)
- POLY_Y : [stm32f429i_discovery_lcd.c](stm32f429i_discovery_lcd.c)
- pPoint : [stm32f429i_discovery_lcd.h](stm32f429i_discovery_lcd.h)
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- r -

- **READWRITE_CMD** : [stm32f429i_discovery.h](#)
- **REFRESH_COUNT** : [stm32f429i_discovery_sdram.h](#)
- **RIGHT_MODE** : [stm32f429i_discovery_lcd.h](#)
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- **s** -

- SDCLOCK_PERIOD : [stm32f429i_discovery_sdram.h](#)
- SDRAM_CAS_LATENCY : [stm32f429i_discovery_sdram.h](#)
- SDRAM_DEVICE_ADDR : [stm32f429i_discovery_sdram.h](#)
- SDRAM_DEVICE_SIZE : [stm32f429i_discovery_sdram.h](#)
- SDRAM_DMAx_CHANNEL : [stm32f429i_discovery_sdram.h](#)
- SDRAM_DMAx_IRQHandler : [stm32f429i_discovery_sdram.h](#)
- SDRAM_DMAx_IRQn : [stm32f429i_discovery_sdram.h](#)
- SDRAM_DMAx_STREAM : [stm32f429i_discovery_sdram.h](#)
- SDRAM_MEMORY_WIDTH : [stm32f429i_discovery_sdram.h](#)
- SDRAM_MODEREG_BURST_LENGTH_1 : [stm32f429i_discovery_sdram.h](#)
- SDRAM_MODEREG_BURST_LENGTH_2 : [stm32f429i_discovery_sdram.h](#)
- SDRAM_MODEREG_BURST_LENGTH_4 : [stm32f429i_discovery_sdram.h](#)
- SDRAM_MODEREG_BURST_LENGTH_8 : [stm32f429i_discovery_sdram.h](#)
- SDRAM_MODEREG_BURST_TYPE_INTERLEAVED : [stm32f429i_discovery_sdram.h](#)
- SDRAM_MODEREG_BURST_TYPE_SEQUENTIAL: \texttt{stm32f429i\_discovery\_sdram.h}
- SDRAM_MODEREG_CAS_LATENCY_2: \texttt{stm32f429i\_discovery\_sdram.h}
- SDRAM_MODEREG_CAS_LATENCY_3: \texttt{stm32f429i\_discovery\_sdram.h}
- SDRAM_MODEREG_OPERATING_MODE_STANDARD: \texttt{stm32f429i\_discovery\_sdram.h}
- SDRAM_MODEREG_WRITEBURST_MODE_PROGRAMMED: \texttt{stm32f429i\_discovery\_sdram.h}
- SDRAM_MODEREG_WRITEBURST_MODE_SINGLE: \texttt{stm32f429i\_discovery\_sdram.h}
- SDRAM_READBURST: \texttt{stm32f429i\_discovery\_sdram.h}
- SDRAM_TIMEOUT: \texttt{stm32f429i\_discovery\_sdram.h}
- SdramHandle: \texttt{stm32f429i\_discovery\_sdram.c}
- SpiHandle: \texttt{stm32f429i\_discovery.c}
- SPIx\_Error(): \texttt{stm32f429i\_discovery.c}
- SPIx\_Init(): \texttt{stm32f429i\_discovery.c}
- SPIx\_MspInit(): \texttt{stm32f429i\_discovery.c}
- SPIx\_Read(): \texttt{stm32f429i\_discovery.c}
- SPIx\_TIMEOUT\_MAX: \texttt{stm32f429i\_discovery.h}
- SPIx\_Write(): \texttt{stm32f429i\_discovery.c}
- SPIx\_WriteRead(): \texttt{stm32f429i\_discovery.c}
- SpixTimeout: \texttt{stm32f429i\_discovery.c}
- STMPE811\_INT\_CLK\_DISABLE: \texttt{stm32f429i\_discovery.h}
- STMPE811\_INT\_CLK\_ENABLE: \texttt{stm32f429i\_discovery.h}
- STMPE811\_INT\_EXTI: \texttt{stm32f429i\_discovery.h}
- STMPE811\_INT\_EXTIHandler: \texttt{stm32f429i\_discovery.h}
- STMPE811\_INT\_GPIO\_PORT: \texttt{stm32f429i\_discovery.h}
- STMPE811\_INT\_PIN: \texttt{stm32f429i\_discovery.h}
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- **Text_ALIGNModeTypdef**: [stm32f429i_discovery_lcd.h](#)
- **Timing**: [stm32f429i_discovery_sdram.c](#)
- **TS_ERROR**: [stm32f429i_discovery_ts.h](#)
- **TS_I2C_ADDRESS**: [stm32f429i_discovery.h](#)
- **TS_OK**: [stm32f429i_discovery_ts.h](#)
- **TS_StatusTypeDef**: [stm32f429i_discovery_ts.h](#)
- **TS_SWAP_NONE**: [stm32f429i_discovery_ts.h](#)
- **TS_SWAP_X**: [stm32f429i_discovery_ts.h](#)
- **TS_SWAP_XY**: [stm32f429i_discovery_ts.h](#)
- **TS_SWAP_Y**: [stm32f429i_discovery_ts.h](#)
- **TS_TIMEOUT**: [stm32f429i_discovery_ts.h](#)
- **TsDrv**: [stm32f429i_discovery_ts.c](#)
- **TsXBoundary**: [stm32f429i_discovery_ts.c](#)
- **TsYBoundary**: [stm32f429i_discovery_ts.c](#)
STM32F429I-Discovery BSP User Manual

- b -

- BSP_EEPROM_Init() : stm32f429i_discovery_eeprom.h
- BSP_EEPROM_ReadBuffer() : stm32f429i_discovery_eeprom.h
- BSP_EEPROM_TIMEOUT_UserCallback() :
  stm32f429i_discovery_eeprom.h
- BSP_EEPROM_WaitEepromStandbyState() :
  stm32f429i_discovery_eeprom.h
- BSP_EEPROM_WriteBuffer() : stm32f429i_discovery_eeprom.h
- BSP_EEPROM_WritePage() : stm32f429i_discovery_eeprom.h
- BSP_GetVersion() : stm32f429i_discovery.h,
  stm32f429i_discovery.c
- BSP_GYRO_DisableIT() : stm32f429i_discovery_gyroscope.c,
  stm32f429i_discovery_gyroscope.h
- BSP_GYRO_EnableIT() : stm32f429i_discovery_gyroscope.c,
  stm32f429i_discovery_gyroscope.h
- BSP_GYRO_GetXYZ() : stm32f429i_discovery_gyroscope.c,
  stm32f429i_discovery_gyroscope.h
- BSP_GYRO_Init() : stm32f429i_discovery_gyroscope.h,
  stm32f429i_discovery_gyroscope.c
- BSP_GYRO_ITConfig() : stm32f429i_discovery_gyroscope.c,
  stm32f429i_discovery_gyroscope.h
- BSP_GYRO_ReadID() : stm32f429i_discovery_gyroscope.c ,
  stm32f429i_discovery_gyroscope.h
- BSP_GYRO_Reset() : stm32f429i_discovery_gyroscope.c ,
  stm32f429i_discovery_gyroscope.h
- BSP_IO_ConfigPin() : stm32f429i_discovery_io.c ,
  stm32f429i_discovery_io.h
- BSP_IO_Init() : stm32f429i_discovery_io.c ,
  stm32f429i_discovery_io.h
- BSP_IO_ITClear() : stm32f429i_discovery_io.c ,
  stm32f429i_discovery_io.h
- BSP_IO_ITGetStatus() : stm32f429i_discovery_io.c ,
  stm32f429i_discovery_io.h
- BSP_IO_ReadPin() : stm32f429i_discovery_io.h ,
  stm32f429i_discovery_io.c
- BSP_IO_TogglePin() : stm32f429i_discovery_io.c ,
  stm32f429i_discovery_io.h
- BSP_IO_WritePin() : stm32f429i_discovery_io.c ,
  stm32f429i_discovery_io.h
- BSP_LCD_Clear() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_ClearStringLine() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayChar() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayOff() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayOn() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayStringAt() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_DisplayStringAtLine() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_DrawBitmap() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_DrawCircle() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
- BSP_LCD_DrawEllipse() : stm32f429i_discovery_lcd.c ,
  stm32f429i_discovery_lcd.h
stm32f429i_discovery_lcd.h

- BSP_LCD_DrawHLine() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawLine() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawPixel() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawPolygon() : stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
- BSP_LCD_DrawRect() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_DrawVLine() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillCircle() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillEllipse() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillPolygon() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillRect() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_FillTriangle() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetBackColor() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetFont() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetTextColor() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetXSize() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_GetYSize() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_Init() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_LayerDefaultInit() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_ReadPixel() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_ResetColorKeying() : stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
- BSP_LCD_SelectLayer() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetBackColor() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetColorKeying() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetFont() : stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
- BSP_LCD_SetLayerAddress() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerVisible() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerWindow() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetTextColor() : stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
- BSP_LCD_SetTransparency() : stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
- BSP_LED_Init() : stm32f429i_discovery.c, stm32f429i_discovery.h
- BSP_LED_Off() : stm32f429i_discovery.c, stm32f429i_discovery.h
- BSP_LED_On() : stm32f429i_discovery.c, stm32f429i_discovery.h
- BSP_LED_Toggle() : stm32f429i_discovery.c, stm32f429i_discovery.h
- BSP_PB_GetState() : stm32f429i_discovery.c, stm32f429i_discovery.h
- BSP_PB_Init() : stm32f429i_discovery.c, stm32f429i_discovery.h
- BSP_SDRAM_DMA_IRQHandler() : stm32f429i_discovery_sdram.h, stm32f429i_discovery_sdram.c
- c -

- BSP_SDRAM_Init() : stm32f429i_discovery_sdram.h, stm32f429i_discovery_sdram.c
- BSP_SDRAM_Initialization_sequence() :
  stm32f429i_discovery_sdram.h, stm32f429i_discovery_sdram.c
- BSP_SDRAM_ReadData() : stm32f429i_discovery_sdram.h, stm32f429i_discovery_sdram.c
- BSP_SDRAM_ReadData_DMA() :
  stm32f429i_discovery_sdram.c, stm32f429i_discovery_sdram.h
- BSP_SDRAM_Sendcmd() : stm32f429i_discovery_sdram.c, stm32f429i_discovery_sdram.h
- BSP_SDRAM_WriteData() : stm32f429i_discovery_sdram.c, stm32f429i_discovery_sdram.h
- BSP_SDRAM_WriteData_DMA() :
  stm32f429i_discovery_sdram.c, stm32f429i_discovery_sdram.h
- BSP_TS_GetState() : stm32f429i_discovery_ts.c, stm32f429i_discovery_ts.h
- BSP_TS_Init() : stm32f429i_discovery_ts.c, stm32f429i_discovery_ts.h
- BSP_TS_ITClear() : stm32f429i_discovery_ts.c, stm32f429i_discovery_ts.h
- BSP_TS_ITConfig() : stm32f429i_discovery_ts.c, stm32f429i_discovery_ts.h
- BSP_TS_ITGetStatus() : stm32f429i_discovery_ts.c, stm32f429i_discovery_ts.h

- d -

- DrawChar() : stm32f429i_discovery_lcd.c

- e -
- EEPROM_IO_Init() : stm32f429i_discovery_eeprom.h
- EEPROM_IO_IsDeviceReady() : stm32f429i_discovery_eeprom.h
- EEPROM_IO_ReadData() : stm32f429i_discovery_eeprom.h
- EEPROM_IO_WriteData() : stm32f429i_discovery_eeprom.h

- f -

- FillBuffer() : stm32f429i_discovery_lcd.c

- g -

- GYRO_IO_Init() : stm32f429i_discovery.c
- GYRO_IO_Read() : stm32f429i_discovery.c
- GYRO_IO_Write() : stm32f429i_discovery.c

- i -

- I2Cx_Error() : stm32f429i_discovery.c
- I2Cx_Init() : stm32f429i_discovery.c
- I2Cx_ITConfig() : stm32f429i_discovery.c
- I2Cx_MspInit() : stm32f429i_discovery.c
- I2Cx_ReadBuffer() : stm32f429i_discovery.c
- I2Cx_ReadData() : stm32f429i_discovery.c
- I2Cx_WriteBuffer() : stm32f429i_discovery.c
- I2Cx_WriteData() : stm32f429i_discovery.c
- IOE_Delay() : stm32f429i_discovery.c
- IOE_Init() : stm32f429i_discovery.c
- IOE_ITConfig() : stm32f429i_discovery.c
- IOE_Read() : stm32f429i_discovery.c
- IOE_ReadMultiple() : stm32f429i_discovery.c
- IOE_Write() : stm32f429i_discovery.c
- IOE_WriteMultiple() : stm32f429i_discovery.c

- l -

- LCD_Delay() : stm32f429i_discovery.c
- LCD_IO_Init() : stm32f429i_discovery.c
- m -

- LCD_IO_ReadData() : stm32f429i_discovery.c
- LCD_IO_WriteData() : stm32f429i_discovery.c
- LCD_IO_WriteReg() : stm32f429i_discovery.c

- m -

- MspInit() : stm32f429i_discovery_lcd.c,          
  stm32f429i_discovery_sdram.c

- s -

- SPIx_Error() : stm32f429i_discovery.c
- SPIx_Init() : stm32f429i_discovery.c
- SPIx_MspInit() : stm32f429i_discovery.c
- SPIx_Read() : stm32f429i_discovery.c
- SPIx_Write() : stm32f429i_discovery.c
- SPIx_WriteRead() : stm32f429i_discovery.c
STM32F429I-Discovery BSP User Manual

- ActiveLayer : `stm32f429i_discovery_lcd.c`
- BUTTON_IRQn : `stm32f429i_discovery.c`
- BUTTON_PIN : `stm32f429i_discovery.c`
- BUTTON_PORT : `stm32f429i_discovery.c`
- Command : `stm32f429i_discovery_sdram.c`
- Dma2dHandler : `stm32f429i_discovery_lcd.c`
- DrawProp : `stm32f429i_discovery_lcd.c`
- GPIO_PIN : `stm32f429i_discovery.c`
- GPIO_PORT : `stm32f429i_discovery.c`
- GyroscopeDrv : `stm32f429i_discovery_gyroscope.c`
- I2cHandle : `stm32f429i_discovery.c`
- I2cxTimeout : `stm32f429i_discovery.c`
- IoDrv : `stm32f429i_discovery_io.c`
- Is_LCD_IO_Initialized : `stm32f429i_discovery.c`
- LcdDrv : `stm32f429i_discovery_lcd.c`
- LtdcHandler : `stm32f429i_discovery_lcd.c`
- PeriphClkInitStruct : `stm32f429i_discovery_lcd.c`
- SdramHandle : `stm32f429i_discovery_sdram.c`
- SpiHandle : `stm32f429i_discovery.c`
- SpixTimeout : `stm32f429i_discovery.c`
- Timing : `stm32f429i_discovery_sdram.c`
- TsDrv : `stm32f429i_discovery_ts.c`
- TsXBoundary : `stm32f429i_discovery_ts.c`
• TsYBoundary : stm32f429i_discovery_ts.c

BSP User Manual by doxygen 1.7.6.1
**pPoint**: `stm32f429i_discovery_lcd.h`
STM32F429I-Discovery BSP User Manual

- Button_TypeDef: `stm32f429i_discovery.h`
- ButtonMode_TypeDef: `stm32f429i_discovery.h`
- GYRO_StatusTypeDef: `stm32f429i_discovery_gyroscope.h`
- IO_StatusTypeDef: `stm32f429i_discovery_io.h`
- LCD_StatusTypeDef: `stm32f429i_discovery_lcd.h`
- Led_TypeDef: `stm32f429i_discovery.h`
- Text_AlignModeTypdef: `stm32f429i_discovery_lcd.h`
- TS_StatusTypeDef: `stm32f429i_discovery_ts.h`

BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

- BUTTON_KEY : `stm32f429i_discovery.h`
- BUTTON_MODE_EXTI : `stm32f429i_discovery.h`
- BUTTON_MODE_GPIO : `stm32f429i_discovery.h`
- CENTER_MODE : `stm32f429i_discovery_lcd.h`
- GYRO_ERROR : `stm32f429i_discovery_gyroscope.h`
- GYRO_OK : `stm32f429i_discovery_gyroscope.h`
- GYRO_TIMEOUT : `stm32f429i_discovery_gyroscope.h`
- IO_ERROR : `stm32f429i_discovery_io.h`
- IO_OK : `stm32f429i_discovery_io.h`
- IO_TIMEOUT : `stm32f429i_discovery_io.h`
- LCD_ERROR : `stm32f429i_discovery_lcd.h`
- LCD_OK : `stm32f429i_discovery_lcd.h`
- LCD_TIMEOUT : `stm32f429i_discovery_lcd.h`
- LED3 : `stm32f429i_discovery.h`
- LED4 : `stm32f429i_discovery.h`
- LEFT_MODE : `stm32f429i_discovery_lcd.h`
- RIGHT_MODE : `stm32f429i_discovery_lcd.h`
- TS_ERROR : `stm32f429i_discovery_ts.h`
- TS_OK : `stm32f429i_discovery_ts.h`
- TS_TIMEOUT : `stm32f429i_discovery_ts.h`
### STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File List</td>
<td>Globals</td>
<td></td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>Functions</td>
<td>Variables</td>
<td>Typedefs</td>
</tr>
</tbody>
</table>

### Defines

- `__DMAx_CLK_ENABLE` : `stm32f429i_discovery_sdram.h`
- `__STM32F429I_DISCO_BSP_VERSION` : `stm32f429i_discovery.c`
- `__STM32F429I_DISCO_BSP_VERSION_MAIN` : `stm32f429i_discovery.c`
- `__STM32F429I_DISCO_BSP_VERSION_RC` : `stm32f429i_discovery.c`
- `__STM32F429I_DISCO_BSP_VERSION_SUB1` : `stm32f429i_discovery.c`
- `__STM32F429I_DISCO_BSP_VERSION_SUB2` : `stm32f429i_discovery.c`

- **a** -
  - `ABS` : `stm32f429i_discovery_lcd.c`

- **b** -
  - `BUFFER_OFFSET` : `stm32f429i_discovery_lcd.h`
  - `BUTTONn` : `stm32f429i_discovery.h`
- d -

- e -

- EEPROM_FAIL : stm32f429i_discovery_eeprom.h
- g -

- i -

- g -

- i -
- k -

- KEY_BUTTON_EXTIIRQn : stm32f429i_discovery.h
- KEY_BUTTON_GPIO_CLKDISABLE : stm32f429i_discovery.h
- KEY_BUTTON_GPIO_CLKENABLE : stm32f429i_discovery.h
- KEY_BUTTON_GPIO_PORT : stm32f429i_discovery.h
- KEY_BUTTON_PIN : stm32f429i_discovery.h

- l -

- LCD_BACKGROUND_LAYER : stm32f429i_discovery_lcd.h
- LCD_COLOR_BLACK : stm32f429i_discovery_lcd.h
- LCD_COLOR_BLUE : stm32f429i_discovery_lcd.h
- LCD_COLOR_BROWN : stm32f429i_discovery_lcd.h
- LCD_COLOR_CYAN : stm32f429i_discovery_lcd.h
- LCD_COLOR_DARKBLUE : stm32f429i_discovery_lcd.h
- LCD_COLOR_DARKCYAN : stm32f429i_discovery_lcd.h
- LCD_COLOR_DARKGRAY : stm32f429i_discovery_lcd.h
- LCD_COLOR_DARKGREEN : stm32f429i_discovery_lcd.h
- LCD_COLOR_DARKMAGENTA : stm32f429i_discovery_lcd.h
- LCD_COLOR_DARKRED : stm32f429i_discovery_lcd.h
- LCD_COLOR_DARKYELLOW : stm32f429i_discovery_lcd.h
- LCD_COLOR_GRAY : stm32f429i_discovery_lcd.h
- LCD_COLOR_GREEN : stm32f429i_discovery_lcd.h
- LCD_COLOR_LIGHTBLUE : stm32f429i_discovery_lcd.h
- LCD_COLOR_LIGHTCYAN : stm32f429i_discovery_lcd.h
- LCD_COLOR_LIGHTGRAY : stm32f429i_discovery_lcd.h
- LCD_COLOR_LIGHTGREEN : stm32f429i_discovery_lcd.h
- LCD_COLOR_LIGHTMAGENTA : stm32f429i_discovery_lcd.h
- LCD_COLOR_LIGHTRED : stm32f429i_discovery_lcd.h
- LCD_COLOR_LIGHTYELLOW : stm32f429i_discovery_lcd.h
- LCD_COLOR_MAGENTA : stm32f429i_discovery_lcd.h
- LCD_COLOR_ORANGE : stm32f429i_discovery_lcd.h
- LCD_COLOR_TRANSPARENT : stm32f429i_discovery_lcd.h
- LCD_COLOR_WHITE : stm32f429i_discovery_lcd.h
- LCD_COLOR_YELLOW : stm32f429i_discovery_lcd.h
- LCD_CS_HIGH : stm32f429i_discovery.h
- LCD_CS_LOW : stm32f429i_discovery.h
- LCD_DEFAULT_FONT : stm32f429i_discovery_lcd.h
- LCD_FOREGROUND_LAYER : stm32f429i_discovery_lcd.h
- LCD_FRAME_BUFFER : stm32f429i_discovery_lcd.h
- LCD_LayerCfgTypeDef : stm32f429i_discovery_lcd.h
- LCD_NCS_GPIO_CLK_DISABLE : stm32f429i_discovery.h
- LCD_NCS_GPIO_CLK_ENABLE : stm32f429i_discovery.h
- LCD_NCS_GPIO_PORT : stm32f429i_discovery.h
- LCD_NCS_PIN : stm32f429i_discovery.h
- LCD_PIXEL_FORMAT_AL44 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_AL88 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_ARGB1555 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_ARGB4444 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_ARGB8888 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_L8 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_RGB565 : stm32f429i_discovery_lcd.h
- LCD_PIXEL_FORMAT_RGB888 : stm32f429i_discovery_lcd.h
- LCD_RDX_GPIO_CLK_DISABLE : stm32f429i_discovery.h
- LCD_RDX_GPIO_CLK_ENABLE : stm32f429i_discovery.h
- LCD_RDX_GPIO_PORT : stm32f429i_discovery.h
- LCD_RDX_HIGH : stm32f429i_discovery.h
- LCD_RDX_LOW : stm32f429i_discovery.h
- LCD_RDX_PIN : stm32f429i_discovery.h
- LCD_WRX_GPIO_CLK_DISABLE : stm32f429i_discovery.h
- LCD_WRX_GPIO_CLK_ENABLE : stm32f429i_discovery.h
- LCD_WRX_GPIO_PORT : stm32f429i_discovery.h
- LCD_WRX_HIGH : stm32f429i_discovery.h
- LCD_WRX_LOW : stm32f429i_discovery.h
- LCD_WRX_PIN : stm32f429i_discovery.h
- LED3_GPIO_CLK_DISABLE : stm32f429i_discovery.h
- LED3_GPIO_CLK_ENABLE : stm32f429i_discovery.h
- LED3_GPIO_PORT : stm32f429i_discovery.h
- LED3_PIN : stm32f429i_discovery.h
- LED4_GPIO_CLK_DISABLE : stm32f429i_discovery.h
- m -

- MAX_LAYER_NUMBER : stm32f429i_discovery_lcd.h
- MULTIPLEBYTE_CMD : stm32f429i_discovery.h

- p -

- POLY_X : stm32f429i_discovery_lcd.c
- POLY_Y : stm32f429i_discovery_lcd.c

- r -

- READWRITE_CMD : stm32f429i_discovery.h
- REFRESH_COUNT : stm32f429i_discovery_sdram.h

- s -

- SDCLOCK_PERIOD : stm32f429i_discovery_sdram.h
- SDRAM_CAS_LATENCY : stm32f429i_discovery_sdram.h
- SDRAM_DEVICE_ADDR : stm32f429i_discovery_sdram.h
- SDRAM_DEVICE_SIZE : stm32f429i_discovery_sdram.h
- SDRAM_DMAx_CHANNEL : stm32f429i_discovery_sdram.h
- SDRAM_DMAx_IRQHandler : stm32f429i_discovery_sdram.h
- SDRAM_DMAx_IRQHandler : stm32f429i_discovery_sdram.h
- SDRAM_DMAx_IRQn : stm32f429i_discovery_sdram.h
- SDRAM_DMAx_STREAM : stm32f429i_discovery_sdram.h
- SDRAM_MEMORY_WIDTH : stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_BURST_LENGTH_1 : stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_BURST_LENGTH_2 : stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_BURST_LENGTH_4 :
stm32f429i_discovery.c File Reference

This file provides set of firmware functions to manage Leds and push-button available on STM32F429I-Discovery Kit from STMicroelectronics. More...

```c
#include "stm32f429i_discovery.h"

Go to the source code of this file.
# Defines

```
#define __STM32F429I_DISCO_BSP_VERSION_MAIN  (0x02)
STM32F429I DISCO BSP Driver version number V2.1.3.

#define __STM32F429I_DISCO_BSP_VERSION_SUB1  (0x01)
#define __STM32F429I_DISCO_BSP_VERSION_SUB2  (0x03)
#define __STM32F429I_DISCO_BSP_VERSION_RC     (0x00)
#define __STM32F429I_DISCO_BSP_VERSION
```
## Functions

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static void</td>
<td><strong>I2Cx_Init</strong> (void)</td>
<td>I2Cx Bus initialization.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>I2Cx_ITConfig</strong> (void)</td>
<td>Configures Interruption pin for I2C communication.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>I2Cx_WriteData</strong> (uint8_t Addr, uint8_t Reg, uint8_t Value)</td>
<td>Writes a value in a register of the device through BUS.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>I2Cx_WriteBuffer</strong> (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)</td>
<td>Writes a value in a register of the device through BUS.</td>
</tr>
<tr>
<td>static uint8_t</td>
<td><strong>I2Cx_ReadData</strong> (uint8_t Addr, uint8_t Reg)</td>
<td>Reads a register of the device through BUS.</td>
</tr>
<tr>
<td>static uint8_t</td>
<td><strong>I2Cx_ReadBuffer</strong> (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)</td>
<td>Reads multiple data on the BUS.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>I2Cx_Error</strong> (void)</td>
<td>I2Cx error treatment function.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>I2Cx_MspInit</strong> (I2C_HandleTypeDef *hi2c)</td>
<td>I2Cx MSP Initialization.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>SPIx_Init</strong> (void)</td>
<td>SPIx Bus initialization.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>SPIx_Write</strong> (uint16_t Value)</td>
<td>Writes a byte to device.</td>
</tr>
<tr>
<td>static uint32_t</td>
<td><strong>SPIx_Read</strong> (uint8_t ReadSize)</td>
<td>Reads 4 bytes from device.</td>
</tr>
<tr>
<td>static uint8_t</td>
<td><strong>SPIx_WriteRead</strong> (uint8_t Byte)</td>
<td>Sends a Byte through the SPI interface and return the Byte received from the SPI bus.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>SPIx_Error</strong> (void)</td>
<td>SPIx error treatment function.</td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>static void SPIx_MspInit</td>
<td>SPI MSP Init.</td>
<td></td>
</tr>
<tr>
<td>void LCD_IO_Init (void)</td>
<td>Configures the LCD_SPI interface.</td>
<td></td>
</tr>
<tr>
<td>void LCD_IO_WriteData (uint16_t RegValue)</td>
<td>Writes register value.</td>
<td></td>
</tr>
<tr>
<td>void LCD_IO_WriteReg (uint8_t Reg)</td>
<td>Writes register address.</td>
<td></td>
</tr>
<tr>
<td>uint32_t LCD_IO_ReadData (uint16_t RegValue, uint8_t ReadSize)</td>
<td>Reads register value.</td>
<td></td>
</tr>
<tr>
<td>void LCD_Delay (uint32_t Delay)</td>
<td>Wait for loop in ms.</td>
<td></td>
</tr>
<tr>
<td>void IOE_Init (void)</td>
<td>IOE Low Level Initialization.</td>
<td></td>
</tr>
<tr>
<td>void IOE_ITConfig (void)</td>
<td>IOE Low Level Interrupt configuration.</td>
<td></td>
</tr>
<tr>
<td>void IOE_Delay (uint32_t Delay)</td>
<td>IOE Delay.</td>
<td></td>
</tr>
<tr>
<td>void IOE_Write (uint8_t Addr, uint8_t Reg, uint8_t Value)</td>
<td>IOE Writes single data operation.</td>
<td></td>
</tr>
<tr>
<td>uint8_t IOE_Read (uint8_t Addr, uint8_t Reg)</td>
<td>IOE Reads single data.</td>
<td></td>
</tr>
<tr>
<td>uint16_t IOE_ReadMultiple (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)</td>
<td>IOE Reads multiple data.</td>
<td></td>
</tr>
<tr>
<td>void IOE_WriteMultiple (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)</td>
<td>IOE Writes multiple data.</td>
<td></td>
</tr>
<tr>
<td>void GYRO_IO_Init (void)</td>
<td>Configures the Gyroscope SPI interface.</td>
<td></td>
</tr>
<tr>
<td>void GYRO_IO_Write (uint8_t *pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite)</td>
<td>Writes one byte to the Gyroscope.</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>void GYRO_IO_Read (uint8_t *pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead)</code></td>
<td>Reads a block of data from the Gyroscope.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_GetVersion (void)</code></td>
<td>This method returns the STM32F429I DISCO BSP Driver revision.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LED_Init (Led_TypeDef Led)</code></td>
<td>Configures LED GPIO.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LED_On (Led_TypeDef Led)</code></td>
<td>Turns selected LED On.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LED_Off (Led_TypeDef Led)</code></td>
<td>Turns selected LED Off.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LED_Toggle (Led_TypeDef Led)</code></td>
<td>Toggles the selected LED.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_PB_Init (Button_TypeDef Button, ButtonMode_TypeDef ButtonMode)</code></td>
<td>Configures Button GPIO and EXTI Line.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_PB_GetState (Button_TypeDef Button)</code></td>
<td>Returns the selected Button state.</td>
<td></td>
</tr>
</tbody>
</table>
**Variables**

<table>
<thead>
<tr>
<th>Type</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIO_TypeDef *</td>
<td>GPIO_PORT [LEDn]</td>
</tr>
<tr>
<td>const uint16_t</td>
<td>GPIO_PIN [LEDn]</td>
</tr>
<tr>
<td>GPIO_TypeDef *</td>
<td>BUTTON_PORT [BUTTONn] = {KEY/Button_GPIO_PORT}</td>
</tr>
<tr>
<td>const uint16_t</td>
<td>BUTTON_PIN [BUTTONn] = {KEY/Button_PIN}</td>
</tr>
<tr>
<td>const uint8_t</td>
<td>BUTTON_IRQn [BUTTONn] = {KEY/Button_EXTI_IRQn}</td>
</tr>
<tr>
<td>uint32_t</td>
<td>I2cxTimeout = I2Cx_TIMEOUT_MAX</td>
</tr>
<tr>
<td>uint32_t</td>
<td>SpixTimeout = SPIx_TIMEOUT_MAX</td>
</tr>
<tr>
<td>I2C_HandleTypeDef</td>
<td>I2cHandle</td>
</tr>
<tr>
<td>static SPI_HandleTypeDef</td>
<td>SpiHandle</td>
</tr>
<tr>
<td>static uint8_t</td>
<td>Is_LCD_IO_Initialized = 0</td>
</tr>
</tbody>
</table>
Detailed Description

This file provides set of firmware functions to manage Leds and push-button available on STM32F429I-Discovery Kit from STMicroelectronics.

Author:
   MCD Application Team

Version:
   V2.1.3

Date:
   13-January-2016

Attention:
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 stm32f429i_discovery.c.
STM32F429I-Discovery BSP User Manual

stm32f429i_discovery.h File Reference

This file contains definitions for STM32F429I-Discovery Kit LEDs, push-buttons hardware resources. More...

#include "stm32f4xx_hal.h"

Go to the source code of this file.
Defines

#define LEDn 2
#define LED3_PIN GPIO_PIN_13
#define LED3_GPIO_PORT GPIOG
#define LED3_GPIO_CLK_ENABLE() __GPIOG_CLK_ENABLE()
#define LED3_GPIO_CLK_DISABLE() __GPIOG_CLK_DISABLE()
#define LED4_PIN GPIO_PIN_14
#define LED4_GPIO_PORT GPIOG
#define LED4_GPIO_CLK_ENABLE() __GPIOG_CLK_ENABLE()
#define LED4_GPIO_CLK_DISABLE() __GPIOG_CLK_DISABLE()
#define LEDx_GPIO_CLK_ENABLE(__INDEX__) (__INDEX__) __GPIOG_CLK_ENABLE()
#define LEDx_GPIO_CLK_DISABLE(__INDEX__) (__INDEX__) __GPIOG_CLK_DISABLE()
#define BUTTONn 1
#define KEY_BUTTON_PIN GPIO_PIN_0
Wakeup push-button.
#define KEY_BUTTON_GPIO_PORT GPIOA
#define KEY_BUTTON_GPIO_CLK_ENABLE() __GPIOA_CLK_ENABLE()
#define KEY_BUTTON_GPIO_CLK_DISABLE() __GPIOA_CLK_DISABLE()
#define KEY_BUTTON_EXTI_IRQn EXTI0_IRQn
#define BUTTONx_GPIO_CLK_ENABLE(__INDEX__) (__INDEX__) __GPIOA_CLK_ENABLE()
#define BUTTONx_GPIO_CLK_DISABLE(__INDEX__) (__INDEX__) __GPIOA_CLK_DISABLE()
#define IO_I2C_ADDRESS 0x82
#define TS_I2C_ADDRESS 0x82
#define DISCOVERY_I2Cx I2C3
#define DISCOVERY_I2Cx_CLOCK_ENABLE() __I2C3_CLK_ENABLE
#define DISCOVERY_I2Cx_FORCE_RESET() __I2C3_FORCE_RESET
#define DISCOVERY_I2Cx_RELEASE_RESET() __I2C3_RELEASE_RESET
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE() __GPIOC_
#define DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE() __GPIOA_
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE() __GPIOC_
#define DISCOVERY_I2Cx_SCL_PIN GPIO_PIN_8
#define DISCOVERY_I2Cx_SCL_GPIO_PORT GPIOA
```c
#define DISCOVERY_I2Cx_SCL_SDA_AF  GPIO_AF4_I2C3
#define DISCOVERY_I2Cx_SDA_PIN  GPIO_PIN_9
#define DISCOVERY_I2Cx_SDA_GPIO_PORT  GPIOC
#define DISCOVERY_I2Cx_EV_IRQn  I2C3_EV_IRQn
#define DISCOVERY_I2Cx_ER_IRQn  I2C3_ER_IRQn

#define I2Cx_TIMEOUT_MAX  0x3000  /*! The value of the maximal timeout waiting loops */
#define DISCOVERY_SPIx  SPI5
#define DISCOVERY_SPIx_CLK_ENABLE()  __SPI5_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_PORT  GPIOF /* GPIOF */
#define DISCOVERY_SPIx_AF  GPIO_AF5_SPI5
#define DISCOVERY_SPIx_GPIO_CLK_ENABLE()  __GPIOF_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_CLK_DISABLE()  __GPIOF_CLK_DISABLE()
#define DISCOVERY_SPIx_SCK_PIN  GPIO_PIN_7 /* PF.07 */
#define DISCOVERY_SPIx_MISO_PIN  GPIO_PIN_8 /* PF.08 */
#define DISCOVERY_SPIx_MOSI_PIN  GPIO_PIN_9 /* PF.09 */
#define SPIx_TIMEOUT_MAX  ((uint32_t)0x1000)
#define STMPE811_INT_PIN  GPIO_PIN_15
#define STMPE811_INT_GPIO_PORT  GPIOA
#define STMPE811_INT_CLK_ENABLE()  __GPIOA_CLK_ENABLE()
#define STMPE811_INT_CLK_DISABLE()  __GPIOA_CLK_DISABLE()
#define STMPE811_INT_EXTI  EXTI15_10_IRQn
#define STMPE811_INT_EXTIHandler  EXTI15_10_IRQHandler
#define LCD_CS_LOW()  HAL_GPIO_WritePin(LCD_NCS_GPIO_PORT, LCD_NCS_PIN, GPIO_PIN_RESET)
#define LCD_CS_HIGH()  HAL_GPIO_WritePin(LCD_NCS_GPIO_PORT, LCD_NCS_PIN, GPIO_PIN_SET)
#define LCD_WRX_LOW()  HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, LCD_WRX_PIN, GPIO_PIN_RESET)
#define LCD_WRX_HIGH()  HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, LCD_WRX_PIN, GPIO_PIN_SET)
#define LCD_RDX_LOW()  HAL_GPIO_WritePin(LCD_RDX_GPIO_PORT, LCD_RDX_PIN, GPIO_PIN_RESET)
#define LCD_RDX_HIGH()  HAL_GPIO_WritePin(LCD_RDX_GPIO_PORT, LCD_RDX_PIN, GPIO_PIN_SET)
```
#define LCD_RDX_PIN, GPIO_PIN_SET)

#define LCD_NCS_GPIO_PORT  GPIOC
#define LCD_NCS_GPIO_CLK_ENABLE() __GPIOC_CLK_ENABLE()
#define LCD_NCS_GPIO_CLK_DISABLE() __GPIOC_CLK_DISABLE()

#define LCD_WRX_GPIO_PORT  GPIOD
#define LCD_WRX_GPIO_CLK_ENABLE() __GPIOD_CLK_ENABLE()
#define LCD_WRX_GPIO_CLK_DISABLE() __GPIOD_CLK_DISABLE()

#define LCD_RDX_GPIO_PORT  GPIOD
#define LCD_RDX_GPIO_CLK_ENABLE() __GPIOD_CLK_ENABLE()
#define LCD_RDX_GPIO_CLK_DISABLE() __GPIOD_CLK_DISABLE()

#define READWRITE_CMD ( (uint8_t)0x80)
#define MULTIPLEBYTE_CMD ( (uint8_t)0x40)
#define DUMMY_BYTE ( (uint8_t)0x00)

#define GYRO_CS_LOW() HAL_GPIO_WritePin( GYRO_CS_GPIO_PORT, GYRO_CS_PIN, GPIO_PIN_RESET)
#define GYRO_CS_HIGH() HAL_GPIO_WritePin( GYRO_CS_GPIO_PORT, GYRO_CS_PIN, GPIO_PIN_SET)

#define GYRO_CS_GPIO_PORT  GPIOC /* GPIOC */
#define GYRO_CS_GPIO_CLK_ENABLE() __GPIOC_CLK_ENABLE()
#define GYRO_CS_GPIO_CLK_DISABLE() __GPIOC_CLK_DISABLE()
#define GYRO_INT_GPIO_CLK_ENABLE() __GPIOA_CLK_ENABLE()
#define GYRO_INT_GPIO_CLK_DISABLE() __GPIOA_CLK_DISABLE()

#define GYRO_INT_GPIO_PORT  GPIOA /* GPIOA */
#define GYRO_INT1_PIN GPIO_PIN_1 /* PA.01 */
#define GYRO_INT1_EXTIIRQn EXTI1_IRQn
#define GYRO_INT2_PIN GPIO_PIN_2 /* PA.02 */
#define GYRO_INT2_EXTI_IRQn EXTI2_IRQn
## Enumerations

<table>
<thead>
<tr>
<th>Enum Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Led_TypeDef</td>
<td>{ LED3 = 0, LED4 = 1 }</td>
</tr>
<tr>
<td>Button_TypeDef</td>
<td>{ BUTTON_KEY = 0 }</td>
</tr>
<tr>
<td>ButtonMode_TypeDef</td>
<td>{ BUTTON_MODE_GPIO = 0, BUTTON_MODE EXTI = 1 }</td>
</tr>
</tbody>
</table>
### Functions

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint32_t</td>
<td><code>BSP_GetVersion (void)</code></td>
<td>This method returns the STM32F429I DISCO BSP Driver revision.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LED_Init (Led_TypeDef Led)</code></td>
<td>Configures LED GPIO.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LED_On (Led_TypeDef Led)</code></td>
<td>Turns selected LED On.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LED_Off (Led_TypeDef Led)</code></td>
<td>Turns selected LED Off.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LED_Toggle (Led_TypeDef Led)</code></td>
<td>Toggles the selected LED.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_PB_Init (Button_TypeDef Button, ButtonMode_TypeDef ButtonMode)</code></td>
<td>Configures Button GPIO and EXTI Line.</td>
</tr>
<tr>
<td>uint32_t</td>
<td><code>BSP_PB_GetState (Button_TypeDef Button)</code></td>
<td>Returns the selected Button state.</td>
</tr>
</tbody>
</table>
Detailed Description

This file contains definitions for STM32F429I-Discovery Kit LEDs, push-buttons hardware resources.

Author:
MCD Application Team

Version:
V2.1.3

Date:
13-January-2016

Attention:
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 stm32f429i_discovery.h.
This file contains all the functions prototypes for the `stm32f429i_discovery_eeprom.c` firmware driver. More...

```c
#include "stm32f429i_discovery.h"
```

Go to the source code of this file.
Defines

#define EEPROM_PAGESIZE 4
#define EEPROM_MAX_SIZE 0x2000 /* 64Kbit*/
#define EEPROM_READ_TIMEOUT ((uint32_t)(1000))
#define EEPROM_WRITE_TIMEOUT ((uint32_t)(1))
#define EEPROM_MAX_TRIALS 300
#define EEPROM_OK 0
#define EEPROM_FAIL 1
#define EEPROM_TIMEOUT 2
### Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint32_t</code></td>
<td><code>BSP_EEPROM_Init</code></td>
<td>(void)</td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><code>BSP_EEPROM_ReadBuffer</code></td>
<td>(uint8_t *pBuffer, uint16_t ReadAddr, uint16_t *NumByteToRead)</td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><code>BSP_EEPROM_WritePage</code></td>
<td>(uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite)</td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><code>BSP_EEPROM_WriteBuffer</code></td>
<td>(uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite)</td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><code>BSP_EEPROM_WaitEepromStandbyState</code></td>
<td>(void)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_EEPROM_TIMEOUT_UserCallback</code></td>
<td>(void)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>EEPROM_IO_Init</code></td>
<td>(void)</td>
</tr>
<tr>
<td><code>HAL_StatusTypeDef</code></td>
<td><code>EEPROM_IO_WriteData</code></td>
<td>(uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize)</td>
</tr>
<tr>
<td><code>HAL_StatusTypeDef</code></td>
<td><code>EEPROM_IO_ReadData</code></td>
<td>(uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize)</td>
</tr>
<tr>
<td><code>HAL_StatusTypeDef</code></td>
<td><code>EEPROM_IO_IsDeviceReady</code></td>
<td>(uint16_t DevAddress, uint32_t Trials)</td>
</tr>
</tbody>
</table>
Detailed Description

This file contains all the functions prototypes for the stm32f429i_discovery_eeprom.c firmware driver.

Author:
    MCD Application Team

Version:
    V2.1.3

Date:
    13-January-2016

Attention:
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 `stm32f429i_discovery_eeprom.h`. 

stm32f429i_discovery_gyroscope.c File Reference

This file provides a set of functions needed to manage the MEMS gyroscope available on STM32F429I-Discovery Kit. More...

#include "stm32f429i_discovery_gyroscope.h"

Go to the source code of this file.
### Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_GYRO_Init (void)</code></td>
<td>Set Gyroscope Initialization.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_GYRO_ReadID (void)</code></td>
<td>Read ID of Gyroscope component.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_Reset (void)</code></td>
<td>Reboot memory content of Gyroscope.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_ITConfig (GYRO_InterruptConfigTypeDef *pIntConfig)</code></td>
<td>Configures INT1 interrupt.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_EnableIT (uint8_t IntPin)</code></td>
<td>Enables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_DisableIT (uint8_t IntPin)</code></td>
<td>Disables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_GetXYZ (float *pfData)</code></td>
<td>Gets XYZ angular acceleration.</td>
</tr>
</tbody>
</table>
Variables

static GYRO_DrvTypeDef * GyroscopeDrv
Detailed Description

This file provides a set of functions needed to manage the MEMS gyroscope available on STM32F429I-Discovery Kit.

Author:
MCD Application Team

Version:
V2.1.3

Date:
13-January-2016

Attention:
© COPYRIGHT(c) 2015 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 stm32f429i_discovery_gyroscope.c.
This file contains definitions for `stm32f429i_discovery_gyroscope.c` firmware driver. More...

```c
#include "stm32f429i_discovery.h" #include "../Components/l3gd20/l3gd20.h"
```

Go to the source code of this file.


Enumerations

```c
enum GYRO_StatusTypeDef { GYRO_OK = 0, GYRO_ERROR = 1, GYRO_TIMEOUT = 2 }
```
# Functions

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><strong>BSP_GYRO_Init</strong> (void)</td>
<td>Set Gyroscope Initialization.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_Reset</strong> (void)</td>
<td>Reboot memory content of Gyroscope.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><strong>BSP_GYRO_ReadID</strong> (void)</td>
<td>Read ID of Gyroscope component.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_ITConfig</strong> (GYRO_InterruptConfigTypeDef *pIntConfigStruct)</td>
<td>Configures INT1 interrupt.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_EnableIT</strong> (uint8_t IntPin)</td>
<td>Enables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_DisableIT</strong> (uint8_t IntPin)</td>
<td>Disables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_GetXYZ</strong> (float *pfData)</td>
<td>Gets XYZ angular acceleration/.</td>
</tr>
</tbody>
</table>
Detailed Description

This file contains definitions for `stm32f429i_discovery_gyroscope.c` firmware driver.

**Author:**
MCD Application Team

**Version:**
V2.1.3

**Date:**
13-January-2016

**Attention:**
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 `stm32f429i_discovery_gyroscope.h`. 

stm32f429i_discovery_io.c File Reference

This file provides a set of functions needed to manage the STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit. More...

#include "stm32f429i_discovery_io.h"

Go to the source code of this file.
### Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_IO_Init</code> (void)</td>
<td>Initializes and configures the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_IO_ITGetStatus</code> (uint16_t IoPin)</td>
<td>Gets the selected pins IT status.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_IO_ITClear</code> (void)</td>
<td>Clears all the IO IT pending bits.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_IO_ConfigPin</code> (uint16_t IoPin, IO_Mode_TypeDef IoMode)</td>
<td>Configures the IO pin(s) according to IO mode structure value.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_IO_WritePin</code> (uint16_t IoPin, uint8_t PinState)</td>
<td>Sets the selected pins state.</td>
</tr>
<tr>
<td><code>uint16_t</code></td>
<td><code>BSP_IO_ReadPin</code> (uint16_t IoPin)</td>
<td>Gets the selected pins current state.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_IO_TogglePin</code> (uint16_t IoPin)</td>
<td>Toggles the selected pins state.</td>
</tr>
</tbody>
</table>
## Variables

| static IO_DrvTypeDef * | IoDrv |
Detailed Description

This file provides a set of functions needed to manage the STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit.

Author:
   MCD Application Team

Version:
   V2.1.3

Date:
   13-January-2016

Attention:
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 stm32f429i_discovery_io.c.
STM32F429I-Discovery BSP User Manual

stm32f429i_discovery_io.h File Reference

This file contains all the functions prototypes for the
stm32f429i_discovery_io.c driver. More...

#include "stm32f429i_discovery.h" #include "../Components/stmpe811/stmpe811.h"

Go to the source code of this file.
## Defines

<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IO_PIN_0</td>
<td>0x01</td>
</tr>
<tr>
<td>IO_PIN_1</td>
<td>0x02</td>
</tr>
<tr>
<td>IO_PIN_2</td>
<td>0x04</td>
</tr>
<tr>
<td>IO_PIN_3</td>
<td>0x08</td>
</tr>
<tr>
<td>IO_PIN_4</td>
<td>0x10</td>
</tr>
<tr>
<td>IO_PIN_5</td>
<td>0x20</td>
</tr>
<tr>
<td>IO_PIN_6</td>
<td>0x40</td>
</tr>
<tr>
<td>IO_PIN_7</td>
<td>0x80</td>
</tr>
<tr>
<td>IO_PIN_ALL</td>
<td>0xFF</td>
</tr>
</tbody>
</table>
Enumerations

```
enum IO_StatusTypeDef { IO_OK = 0, IO_ERROR = 1, IO_TIMEOUT = 2 }
```
Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><strong>BSP_IO_Init</strong> (void)</td>
<td>Initializes and configures the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td>uint8_t</td>
<td><strong>BSP_IO_ITGetStatus</strong> (uint16_t IoPin)</td>
<td>Gets the selected pins IT status.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_IO_ITClear</strong> (void)</td>
<td>Clears all the IO IT pending bits.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_IO_ConfigPin</strong> (uint16_t IoPin, IO_ModeTypedef IoMode)</td>
<td>Configures the IO pin(s) according to IO mode structure value.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_IO_WritePin</strong> (uint16_t IoPin, uint8_t PinState)</td>
<td>Sets the selected pins state.</td>
</tr>
<tr>
<td>uint16_t</td>
<td><strong>BSP_IO_ReadPin</strong> (uint16_t IoPin)</td>
<td>Gets the selected pins current state.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_IO_TogglePin</strong> (uint16_t IoPin)</td>
<td>Toggles the selected pins state.</td>
</tr>
</tbody>
</table>
Detailed Description

This file contains all the functions prototypes for the stm32f429i_discovery_io.c driver.

Author:
MCD Application Team

Version:
V2.1.3

Date:
13-January-2016

Attention:
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 stm32f429i_discovery_io.h.
stm32f429i_discovery_lcd.c File Reference

This file includes the LCD driver for ILI9341 Liquid Crystal Display Modules of STM32F429I-Discovery kit (MB1075). More...

```
#include "stm32f429i_discovery_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.
Defines

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

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>MspInit</code> (void)</td>
<td>Initializes the LTDC MSP.</td>
</tr>
<tr>
<td><code>DrawChar (uint16_t Xpos, uint16_t Ypos, const uint8_t *c)</code></td>
<td>Draws a character on LCD.</td>
</tr>
<tr>
<td><code>FillBuffer (uint32_t LayerIndex, void *pDst, uint32_t xSize, uint32_t ySize, uint32_t OffLine, uint32_t ColorIndex)</code></td>
<td>Fills buffer.</td>
</tr>
<tr>
<td><code>ConvertLineToARGB8888 (void *pSrc, void *pDst, uint32_t xSize, uint32_t ColorMode)</code></td>
<td>Converts Line to ARGB8888 pixel format.</td>
</tr>
<tr>
<td><code>BSP_LCD_Init (void)</code></td>
<td>Initializes the LCD.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetXSize (void)</code></td>
<td>Gets the LCD X size.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetYSize (void)</code></td>
<td>Gets the LCD Y size.</td>
</tr>
<tr>
<td><code>BSP_LCD_LayerDefaultInit (uint16_t LayerIndex, uint32_t FB_Address)</code></td>
<td>Initializes the LCD layers.</td>
</tr>
<tr>
<td><code>BSP_LCD_SelectLayer (uint32_t LayerIndex)</code></td>
<td>Selects the LCD Layer.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetLayerVisible (uint32_t LayerIndex, FunctionalState state)</code></td>
<td>Sets a LCD Layer visible.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetTransparency (uint32_t LayerIndex, uint8_t Transparency)</code></td>
<td>Configures the Transparency.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetLayerAddress (uint32_t LayerIndex, uint32_t Address)</code></td>
<td>Sets a LCD layer frame buffer address.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetLayerWindow</strong> (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetColorKeying</strong> (uint32_t LayerIndex, uint32_t RGBValue)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_ResetColorKeying</strong> (uint32_t LayerIndex)</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_LCD_GetTextColor</strong> (void)</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_LCD_GetBackColor</strong> (void)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetTextColor</strong> (uint32_t Color)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetBackColor</strong> (uint32_t Color)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetFont</strong> (sFONT *pFonts)</td>
</tr>
<tr>
<td>sFONT *</td>
<td><strong>BSP_LCD_GetFont</strong> (void)</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_LCD_ReadPixel</strong> (uint16_t Xpos, uint16_t Ypos)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_Clear</strong> (uint32_t Color)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_ClearStringLine</strong> (uint32_t Line)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DisplayChar</strong> (uint16_t Xpos, uint16_t Ypos, uint8_t Ascii)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DisplayStringAt</strong> (uint16_t X, uint16_t Y, uint8_t *pText, Text_AlignModeTypdef mode)</td>
</tr>
<tr>
<td>Function</td>
<td>Parameters</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>void *ptr)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawHLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawVLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawLine(uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawRect(uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawCircle(uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawPolygon(pPoint Points, uint16_t PointCount)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawEllipse(int Xpos, int Ypos, int XRADIUS, int Yradius)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawBitmap(uint32_t X, uint32_t Y, uint8_t *pBmp)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_FillRect(uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_FillCircle(uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</code></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_FillTriangle(uint16_t X1, uint16_t X2, uint16_t</code></td>
<td></td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td><code>void X3, uint16_t Y1, uint16_t Y2, uint16_t Y3)</code></td>
<td>Fill triangle.</td>
</tr>
<tr>
<td><code>void BSP_LCD_FillPolygon (pPoint Points, uint16_t PointCount)</code></td>
<td>Displays a full poly-line (between many points).</td>
</tr>
<tr>
<td><code>void BSP_LCD_FillEllipse (int Xpos, int Ypos, int XRADIUS, int YRADIUS)</code></td>
<td>Draw a full ellipse.</td>
</tr>
<tr>
<td><code>void BSP_LCD_DisplayOn (void)</code></td>
<td>Enables the Display.</td>
</tr>
<tr>
<td><code>void BSP_LCD_DisplayOff (void)</code></td>
<td>Disables the Display.</td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawPixel (uint16_t Xpos, uint16_t Ypos, uint32_t RGB_Code)</code></td>
<td>Writes Pixel.</td>
</tr>
</tbody>
</table>
### Variables

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>static LTDC_HandleTypeDef</td>
<td>LtdcHandler</td>
</tr>
<tr>
<td>static DMA2D_HandleTypeDef</td>
<td>Dma2dHandler</td>
</tr>
<tr>
<td>static RCC_PeriphCLKInitTypeDef</td>
<td>PeriphClkInitStruct</td>
</tr>
<tr>
<td>static uint32_t</td>
<td>ActiveLayer = 0</td>
</tr>
<tr>
<td>static LCD_DrawPropTypeDef</td>
<td>DrawProp [MAX_LAYER_NUMBER]</td>
</tr>
<tr>
<td>LCD_DrvTypeDef *</td>
<td>LcdDrv</td>
</tr>
</tbody>
</table>
Detailed Description

This file includes the LCD driver for ILI9341 Liquid Crystal Display Modules of STM32F429I-Discovery kit (MB1075).

Author:
MCD Application Team

Version:
V2.1.3

Date:
13-January-2016

Attention:
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 \texttt{stm32f429i_discovery_lcd.c}.
stm32f429i_discovery_lcd.h File Reference

This file contains all the functions prototypes for the stm32f429i_discovery_lcd.c driver. More...

#include "stm32f429i_discovery.h" #include "stm32f429i_discovery_sdram.h"
#include "../../../Utilities/Fonts/fonts.h"
#include "../Components/ili9341/ili9341.h"

Go to the source code of this file.
Data Structures

<table>
<thead>
<tr>
<th>struct</th>
<th>LCD_DrawPropTypeDef</th>
</tr>
</thead>
<tbody>
<tr>
<td>struct</td>
<td>Point</td>
</tr>
</tbody>
</table>
Defines

#define LCD_LayerCfgTypeDef LTDC_LayerCfgTypeDef
#define MAX_LAYER_NUMBER 2

LCD status structure definition.

#define LCD_FRAME_BUFFER ((uint32_t)0xD0000000)
#define BUFFER_OFFSET ((uint32_t)0x50000)
#define LCD_COLOR_BLUE 0xFF0000FF

LCD color.

#define LCD_COLOR_GREEN 0xFF00FF00
#define LCD_COLOR_RED 0xFFFF0000
#define LCD_COLOR_CYAN 0xFF00FFFF
#define LCD_COLOR_MAGENTA 0xFFFF00FF
#define LCD_COLOR_YELLOW 0xFFFFFF00
#define LCD_COLOR_LIGHTBLUE 0xFF8080FF
#define LCD_COLOR_LIGHTGREEN 0xFF80FF80
#define LCD_COLOR_LIGHTRED 0xFFFF8080
#define LCD_COLOR_LIGHTCYAN 0xFF80FFFF
#define LCD_COLOR_LIGHTMAGENTA 0xFFFF80FF
#define LCD_COLOR_LIGHTYELLOW 0xFFFFFF80
#define LCD_COLOR_DARKBLUE 0xFF000080
#define LCD_COLOR_DARKGREEN 0xFF008000
#define LCD_COLOR_DARKRED 0xFF800000
#define LCD_COLOR_DARKCYAN 0xFF008080
#define LCD_COLOR_DARKMAGENTA 0xFF800080
#define LCD_COLOR_DARKYELLOW 0xFF808000
#define LCD_COLOR_WHITE 0xFFFFFFFF
#define LCD_COLOR_LIGHTGRAY 0xFFD3D3D3
#define LCD_COLOR_GRAY 0xFF808080
#define LCD_COLOR_DARKGRAY 0xFF404040
#define LCD_COLOR_BLACK 0xFF000000
#define LCD_COLOR_BROWN 0xFFA52A2A
#define LCD_COLOR_ORANGE 0xFFFFA500

#define LCD_COLOR_LIGHTGRAY 0xFFD3D3D3
#define LCD_COLOR_GRAY 0xFF808080
#define LCD_COLOR_DARKGRAY 0xFF404040
#define LCD_COLOR_BLACK 0xFF000000
#define LCD_COLOR_BROWN 0xFFA52A2A
#define LCD_COLOR_ORANGE 0xFFFFA500
<table>
<thead>
<tr>
<th>Definition</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#define LCD_COLOR_TRANSPARENT</code></td>
<td>0xFF000000</td>
<td>LCD background layer</td>
</tr>
<tr>
<td><code>#define LCD_DEFAULT_FONT</code></td>
<td>Font24</td>
<td>LCD default font.</td>
</tr>
<tr>
<td><code>#define LCD_BACKGROUND_LAYER</code></td>
<td>0x0000</td>
<td>LCD background layer</td>
</tr>
<tr>
<td><code>#define LCD_FOREGROUND_LAYER</code></td>
<td>0x0001</td>
<td>LCD foreground layer</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_ARGB8888</code></td>
<td>LTDC_PIXEL_FORMAT_ARGB8888</td>
<td>LCD pixel format.</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_RGB888</code></td>
<td>LTDC_PIXEL_FORMAT_RGB888</td>
<td>LCD pixel format.</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_RGB565</code></td>
<td>LTDC_PIXEL_FORMAT_RGB565</td>
<td>LCD pixel format.</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_ARGB1555</code></td>
<td>LTDC_PIXEL_FORMAT_ARGB1555</td>
<td>LCD pixel format.</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_ARGB4444</code></td>
<td>LTDC_PIXEL_FORMAT_ARGB4444</td>
<td>LCD pixel format.</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_L8</code></td>
<td>LTDC_PIXEL_FORMAT_L8</td>
<td>LCD pixel format.</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_AL44</code></td>
<td>LTDC_PIXEL_FORMAT_AL44</td>
<td>LCD pixel format.</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_AL88</code></td>
<td>LTDC_PIXEL_FORMAT_AL88</td>
<td>LCD pixel format.</td>
</tr>
</tbody>
</table>
typedef struct Point * pPoint
### Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_StatusTypeDef</td>
<td>- <strong>LCD_OK</strong> = 0, <strong>LCD_ERROR</strong> = 1, <strong>LCD_TIMEOUT</strong> = 2</td>
</tr>
<tr>
<td>Text_AlignModeTypdef</td>
<td>- <strong>CENTER_MODE</strong> = 0x01, <strong>RIGHT_MODE</strong> = 0x02, <strong>LEFT_MODE</strong> = 0x03</td>
</tr>
</tbody>
</table>

Line mode structures definition. More...
## Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void BSP_LCD_Init (void)</code></td>
<td>Initializes the LCD.</td>
</tr>
<tr>
<td><code>uint8_t BSP_LCD_Init (void)</code></td>
<td>Initializes the LCD.</td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_GetXSize (void)</code></td>
<td>Gets the LCD X size.</td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_GetYSize (void)</code></td>
<td>Gets the LCD Y size.</td>
</tr>
<tr>
<td><code>void BSP_LCD_LayerDefaultInit (uint16_t LayerIndex, uint32_t FrameBuffer)</code></td>
<td>Initializes the LCD layers.</td>
</tr>
<tr>
<td><code>void BSP_LCD_SetTransparency (uint32_t LayerIndex, uint8_t Transparency)</code></td>
<td>Configures the Transparency.</td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerAddress (uint32_t LayerIndex, uint32_t Address)</code></td>
<td>Sets a LCD layer frame buffer address.</td>
</tr>
<tr>
<td><code>void BSP_LCD_SetColorKeying (uint32_t LayerIndex, uint32_t RGBValue)</code></td>
<td>Configures and sets the color Keying.</td>
</tr>
<tr>
<td><code>void BSP_LCD_ResetColorKeying (uint32_t LayerIndex)</code></td>
<td>Disables the color Keying.</td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerWindow (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</code></td>
<td>Sets the Display window.</td>
</tr>
<tr>
<td><code>void BSP_LCD_SelectLayer (uint32_t LayerIndex)</code></td>
<td>Selects the LCD Layer.</td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerVisible (uint32_t LayerIndex, FunctionalState state)</code></td>
<td>Sets a LCD Layer visible.</td>
</tr>
<tr>
<td><code>void BSP_LCD_SetTextColor (uint32_t Color)</code></td>
<td>Sets the Text color.</td>
</tr>
<tr>
<td>Function name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>void BSP_LCD_SetBackColor (uint32_t Color)</strong></td>
<td>Sets the Background color.</td>
</tr>
<tr>
<td><strong>uint32_t BSP_LCD_GetTextColor (void)</strong></td>
<td>Gets the LCD Text color.</td>
</tr>
<tr>
<td><strong>uint32_t BSP_LCD_GetBackColor (void)</strong></td>
<td>Gets the LCD Background color.</td>
</tr>
<tr>
<td>*<em>void BSP_LCD_SetFont (sFONT <em>pFonts)</em></em></td>
<td>Sets the Text Font.</td>
</tr>
<tr>
<td><strong>sFONT * BSP_LCD_GetFont (void)</strong></td>
<td>Gets the Text Font.</td>
</tr>
<tr>
<td><strong>uint32_t BSP_LCD_ReadPixel (uint16_t Xpos, uint16_t Ypos)</strong></td>
<td>Reads Pixel.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_DrawPixel (uint16_t Xpos, uint16_t Ypos, uint32_t pixel)</strong></td>
<td>Writes Pixel.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_Clear (uint32_t Color)</strong></td>
<td>Clears the hole LCD.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_ClearStringLine (uint32_t Line)</strong></td>
<td>Clears the selected line.</td>
</tr>
<tr>
<td>*<em>void BSP_LCD_DisplayStringAtLine (uint16_t Line, uint8_t <em>ptr)</em></em></td>
<td>Displays a maximum of 20 char on the LCD.</td>
</tr>
<tr>
<td>*<em>void BSP_LCD_DisplayStringAt (uint16_t X, uint16_t Y, uint8_t <em>pText, Text_AlignModeTypdef mode)</em></em></td>
<td>Displays a maximum of 60 char on the LCD.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_DisplayChar (uint16_t Xpos, uint16_t Ypos, uint8_t Ascii)</strong></td>
<td>Displays one character.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_DrawHLine (uint16_t Xpos, uint16_t Ypos, uint16_t Length)</strong></td>
<td>Displays an horizontal line.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_DrawVLine (uint16_t Xpos, uint16_t Ypos, uint16_t Length)</strong></td>
<td>Displays a vertical line.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawLine</code> (uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2)</td>
<td>Displays an uni-line (between two points).</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawRect</code> (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</td>
<td>Displays a rectangle.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawCircle</code> (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</td>
<td>Displays a circle.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawPolygon</code> (pPoint Points, uint16_t PointCount)</td>
<td>Displays an poly-line (between many points).</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawEllipse</code> (int Xpos, int Ypos, int XRadius, int YRadius)</td>
<td>Displays an Ellipse.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawBitmap</code> (uint32_t X, uint32_t Y, uint8_t *pBmp)</td>
<td>Displays a bitmap picture loaded in the internal Flash (32 bpp).</td>
</tr>
<tr>
<td><code>BSP_LCD_FillRect</code> (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</td>
<td>Displays a full rectangle.</td>
</tr>
<tr>
<td><code>BSP_LCD_FillCircle</code> (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</td>
<td>Displays a full circle.</td>
</tr>
<tr>
<td><code>BSP_LCD_FillTriangle</code> (uint16_t X1, uint16_t X2, uint16_t X3, uint16_t Y1, uint16_t Y2, uint16_t Y3)</td>
<td>Fill triangle.</td>
</tr>
<tr>
<td><code>BSP_LCD_FillPolygon</code> (pPoint Points, uint16_t PointCount)</td>
<td>Displays a full poly-line (between many points).</td>
</tr>
<tr>
<td><code>BSP_LCD_FillEllipse</code> (int Xpos, int Ypos, int XRadius, int YRadius)</td>
<td>Draw a full ellipse.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td><code>void BSP_LCD_DisplayOff (void)</code></td>
<td>Disables the Display.</td>
</tr>
<tr>
<td><code>void BSP_LCD_DisplayOn (void)</code></td>
<td>Enables the Display.</td>
</tr>
</tbody>
</table>
Detailed Description

This file contains all the functions prototypes for the 
stm32f429i_discovery_lcd.c driver.

Author:
MCD Application Team

Version:
V2.1.3

Date:
13-January-2016

| Attention: |
© COPYRIGHT(c) 2015 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 stm32f429i_discovery_lcd.h.

STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Directories

<table>
<thead>
<tr>
<th>File List</th>
<th>Globals</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Drivers > BSP > STM32F429I-Discovery

stm32f429i_discovery_sdram.c File Reference

This file provides a set of functions needed to drive the IS42S16400J SDRAM memory mounted on STM32F429I-Discovery Kit. More...

```
#include "stm32f429i_discovery_sdram.h"
```

Go to the source code of this file.
### Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static void <strong>MspInit</strong> (void)</td>
<td>Initializes SDRAM MSP.</td>
</tr>
<tr>
<td>void <strong>BSP_SDRAM_Init</strong> (void)</td>
<td>Initializes the SDRAM device.</td>
</tr>
<tr>
<td>void <strong>BSP_SDRAM_Initialization_sequence</strong></td>
<td>Programs the SDRAM device.</td>
</tr>
<tr>
<td>(uint32_t RefreshCount)</td>
<td></td>
</tr>
<tr>
<td>void <strong>BSP_SDRAM_ReadData</strong></td>
<td>Reads an amount of data from the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td>(uint32_t uwStartAddress, uint32_t *pData,</td>
<td></td>
</tr>
<tr>
<td>uint32_t uwDataSize)</td>
<td></td>
</tr>
<tr>
<td>void <strong>BSP_SDRAM_ReadData_DMA</strong></td>
<td>Reads an amount of data from the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td>(uint32_t uwStartAddress, uint32_t *pData,</td>
<td></td>
</tr>
<tr>
<td>uint32_t uwDataSize)</td>
<td></td>
</tr>
<tr>
<td>void <strong>BSP_SDRAM_WriteData</strong></td>
<td>Writes an amount of data to the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td>(uint32_t uwStartAddress, uint32_t *pData,</td>
<td></td>
</tr>
<tr>
<td>uint32_t uwDataSize)</td>
<td></td>
</tr>
<tr>
<td>void <strong>BSP_SDRAM_WriteData_DMA</strong></td>
<td>Writes an amount of data to the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td>(uint32_t uwStartAddress, uint32_t *pData,</td>
<td></td>
</tr>
<tr>
<td>uint32_t uwDataSize)</td>
<td></td>
</tr>
<tr>
<td>HAL_StatusTypeDef <strong>BSP_SDRAM_Sendcmd</strong></td>
<td>Sends command to the SDRAM bank.</td>
</tr>
<tr>
<td>(FMC_SDRAM_CommandTypeDef *SdramCmd)</td>
<td></td>
</tr>
<tr>
<td>void <strong>BSP_SDRAM_DMA_IRQHandler</strong></td>
<td></td>
</tr>
</tbody>
</table>
Handles SDRAM DMA transfer interrupt request.
Variables

<table>
<thead>
<tr>
<th>static SDRAM_HandleTypeDef</th>
<th>SdramHandle</th>
</tr>
</thead>
<tbody>
<tr>
<td>static FMC_SDRAM_TimingTypeDef</td>
<td>Timing</td>
</tr>
<tr>
<td>static FMC_SDRAM_CommandTypeDef</td>
<td>Command</td>
</tr>
</tbody>
</table>
Detailed Description

This file provides a set of functions needed to drive the IS42S16400J SDRAM memory mounted on STM32F429I-Discovery Kit.

Author:
MCD Application Team

Version:
V2.1.3

Date:
13-January-2016

Attention:
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 `stm32f429i_discovery_sdram.c`.

STM32F429I-Discovery BSP User Manual

stm32f429i_discovery_sdram.h File Reference

This file contains all the functions prototypes for the stm32f429i_discovery_sdram.c driver. More...

#include "stm32f429i_discovery.h"

Go to the source code of this file.
#define SDRAM_DEVICE_ADDR ((uint32_t)0xD0000000)  
FMC SDRAM Bank address.

#define SDRAM_DEVICE_SIZE ((uint32_t)0x800000) /* SDRAM device size in MBytes */

#define SDRAM_MEMORY_WIDTH FMC_SDRAM_MEM_BUS_WIDTH_16  
FMC SDRAM Memory Width.

#define SDRAM_CAS_LATENCY FMC_SDRAM_CAS_LATENCY_3  
FMC SDRAM CAS Latency.

#define SDCLOCK_PERIOD FMC_SDRAM_CLOCK_PERIOD_2 /* Default configuration used with LCD */
FMC SDRAM Memory clock period.

#define SDRAM_READBURST FMC_SDRAM_RBURST_DISABLE /* Default configuration used with LCD */
FMC SDRAM Memory Read Burst feature.

#define REFRESH_COUNT ((uint32_t)1386) /* SDRAM refresh count */
FMC SDRAM Bank Remap.

#define SDRAM_TIMEOUT ((uint32_t)0xFFFF)
#define __DMAx_CLK_ENABLE __DMA2_CLK_ENABLE
#define SDRAM_DMAx_CHANNEL DMA_CHANNEL_0
#define SDRAM_DMAx_STREAM DMA2_Stream0
#define SDRAM_DMAx_IRQHandler DMA2_Stream0_IRQHandler
#define SDRAM_DMAx_IRQn DMA2_Stream0_IRQn
#define SDRAM_MODEREG_BURST_LENGTH_1 ((uint16_t)0x0000) /* FMC SDRAM Mode definition register defines. */
#define SDRAM_MODEREG_BURST_LENGTH_2 ((uint16_t)0x0001)
#define SDRAM_MODEREG_BURST_LENGTH_4 ((uint16_t)0x0002)
#define SDRAM_MODEREG_BURST_LENGTH_8 ((uint16_t)0x0004)
#define SDRAM_MODEREG_BURST_TYPE_SEQUENTIAL ((uint16_t)0x0000)
#define SDRAM_MODEREG_BURST_TYPE_INTERLEAVED ((uint16_t)0x0008)
#define SDRAM_MODEREG_CAS_LATENCY_2 ((uint16_t)0x0020)
#define SDRAM_MODEREG_CAS_LATENCY_3 ((uint16_t)0x0030)
#define SDRAM_MODEREG_OPERATING_MODE_STANDARD ((uint16_t)0x0000)
#define SDRAM_MODEREG_WRITEBURST_MODE_PROGRAMMED
#define SDRAM_MODEREG_WRITEBURST_MODE_SINGLE ((uint'
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_Init</strong> (void)</td>
<td>Initializes the SDRAM device.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_Initialization_sequence</strong> (uint32_t RefreshCount)</td>
<td>Programs the SDRAM device.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_ReadData</strong> (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Reads an amount of data from the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_ReadData_DMA</strong> (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Reads an amount of data from the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_WriteData</strong> (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Writes an amount of data to the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_WriteData_DMA</strong> (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Writes an amount of data to the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td>HAL_StatusTypeDef</td>
<td><strong>BSP_SDRAM_Sendcmd</strong> (FMC_SDRAM_CommandTypeDef *SdramCmd)</td>
<td>Sends command to the SDRAM bank.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_DMA_IRQHandler</strong> (void)</td>
<td>Handles SDRAM DMA transfer interrupt request.</td>
</tr>
</tbody>
</table>
Detailed Description

This file contains all the functions prototypes for the `stm32f429i_discovery_sdram.c` driver.

**Author:**
  MCD Application Team

**Version:**
  V2.1.3

**Date:**
  13-January-2016

| Attention: |
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 stm32f429i_discovery_sdram.h.
This file provides a set of functions needed to manage Touch screen available with STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit. More...

#include "stm32f429i_discovery_ts.h" #include "stm32f429i_discovery_io.h"

Go to the source code of this file.
## Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t BSP_TS_Init (uint16_t XSize, uint16_t YSize)</code></td>
<td>Initializes and configures the touch screen functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td><code>uint8_t BSP_TS_ITConfig (void)</code></td>
<td>Configures and enables the touch screen interrupts.</td>
</tr>
<tr>
<td><code>uint8_t BSP_TS_ITGetStatus (void)</code></td>
<td>Gets the TS IT status.</td>
</tr>
<tr>
<td><code>void BSP_TS_GetState (TS_StateTypeDef *TsState)</code></td>
<td>Returns status and positions of the touch screen.</td>
</tr>
<tr>
<td><code>void BSP_TS_ITClear (void)</code></td>
<td>Clears all touch screen interrupts.</td>
</tr>
</tbody>
</table>
Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>static TS_DrvTypeDef *</td>
<td>TsDrv</td>
</tr>
<tr>
<td>static uint16_t</td>
<td>TsXBoundary</td>
</tr>
<tr>
<td>static uint16_t</td>
<td>TsYBoundary</td>
</tr>
</tbody>
</table>
Detailed Description

This file provides a set of functions needed to manage Touch screen available with STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit.

Author:
MCD Application Team

Version:
V2.1.3

Date:
13-January-2016

Attention:
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 stm32f429i_discovery_ts.c.
STM32F429I-Discovery BSP User Manual

stm32f429i_discovery_ts.h File Reference

This file contains all the functions prototypes for the stm32f429i_discovery_ts.c driver. More...

#include "stm32f429i_discovery.h"
#include "../Components/stmpe811/stmpe811.h"

Go to the source code of this file.
Data Structures

```c
struct TS_StateTypeDef
```
Defines

<table>
<thead>
<tr>
<th>Define</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>#define TS_SWAP_NONE</td>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>#define TS_SWAP_X</td>
<td>0x01</td>
<td></td>
</tr>
<tr>
<td>#define TS_SWAP_Y</td>
<td>0x02</td>
<td></td>
</tr>
<tr>
<td>#define TS_SWAP_XY</td>
<td>0x04</td>
<td></td>
</tr>
</tbody>
</table>
Enumerations

```c
enum TS_StatusTypeDef { TS_OK = 0x00, TS_ERROR = 0x01, TS_TIMEOUT = 0x02 }
```
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_TS_Init (uint16_t XSize, uint16_t YSize)</code></td>
<td>Initializes and configures the touch screen functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_TS_GetState (TS_StateTypeDef *TsState)</code></td>
<td>Returns status and positions of the touch screen.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_TS_ITConfig (void)</code></td>
<td>Configures and enables the touch screen interrupts.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_TS_ITGetStatus (void)</code></td>
<td>Gets the TS IT status.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_TS_ITClear (void)</code></td>
<td>Clears all touch screen interrupts.</td>
</tr>
</tbody>
</table>
Detailed Description

This file contains all the functions prototypes for the `stm32f429i_discovery_ts.c` driver.

**Author:**
MCD Application Team

**Version:**
V2.1.3

**Date:**
13-January-2016

**Attention:**
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 stm32f429i_discovery_ts.h.
STM32F429I-Discovery BSP User Manual

Here is a list of all modules:

- **BSP**
  - STM32F429I DISCOVERY
    - STM32F429I DISCOVERY LOW LEVEL
      - STM32F429I DISCOVERY LOW LEVEL Private TypesDefinitions
      - STM32F429I DISCOVERY LOW LEVEL Private Defines
      - STM32F429I DISCOVERY LOW LEVEL Private Macros
      - STM32F429I DISCOVERY LOW LEVEL Private Variables
      - STM32F429I DISCOVERY LOW LEVEL Private FunctionPrototypes
      - STM32F429I DISCOVERY LOW LEVEL Private Functions
      - STM32F429I DISCOVERY LOW LEVEL Exported Constants
        - STM32F429I DISCOVERY LOW LEVEL LED
        - STM32F429I DISCOVERY LOW LEVEL BUTTON
        - STM32F429I DISCOVERY LOW LEVEL BUS
      - STM32F429I DISCOVERY LOW LEVEL Exported Macros
      - STM32F429I DISCOVERY LOW LEVEL Exported Functions
- STM32F429I DISCOVERY LOW LEVEL Exported Types
- STM32F429I DISCOVERY GYROSCOPE
  - STM32F429I DISCOVERY GYROSCOPE Private Types Definitions
  - STM32F429I DISCOVERY GYROSCOPE Private Defines
  - STM32F429I DISCOVERY GYROSCOPE Private Macros
  - STM32F429I DISCOVERY GYROSCOPE Private Variables
  - STM32F429I DISCOVERY GYROSCOPE Private Function Prototypes
  - STM32F429I DISCOVERY GYROSCOPE Private Functions
  - STM32F429I DISCOVERY GYROSCOPE Exported Types
  - STM32F429I DISCOVERY GYROSCOPE Exported Constants
  - STM32F429I DISCOVERY GYROSCOPE Exported Macros
  - STM32F429I DISCOVERY GYROSCOPE Exported Functions
- STM32F429I DISCOVERY IO
  - STM32F429I DISCOVERY IO Private Types Definitions
  - STM32F429I DISCOVERY IO Private Defines
  - STM32F429I DISCOVERY IO Private Macros
  - STM32F429I DISCOVERY IO Private Variables
  - STM32F429I DISCOVERY IO Private Function Prototypes
  - STM32F429I DISCOVERY IO Private Functions
  - STM32F429I DISCOVERY IO Exported Types
  - STM32F429I DISCOVERY IO Exported Constants
  - STM32F429I DISCOVERY IO Exported Macros
  - STM32F429I DISCOVERY IO Exported Functions
- STM32F429I DISCOVERY LCD
- STM32F429I DISCOVERY LCD Private Types Definitions
- STM32F429I DISCOVERY LCD Private Defines
- STM32F429I DISCOVERY LCD Private Macros
- STM32F429I DISCOVERY LCD Private Variables
- STM32F429I DISCOVERY LCD Private Function Prototypes
- STM32F429I DISCOVERY LCD Private Functions
- STM32F429I DISCOVERY LCD Exported Types
- STM32F429I DISCOVERY LCD Exported Constants
- STM32F429I DISCOVERY LCD Exported Macros
- STM32F429I DISCOVERY LCD Exported Functions

- STM32F429I DISCOVERY SDRAM
  - STM32F429I DISCOVERY SDRAM Private Types Definitions
  - STM32F429I DISCOVERY SDRAM Private Defines
  - STM32F429I DISCOVERY SDRAM Private Macros
  - STM32F429I DISCOVERY SDRAM Private Variables
  - STM32F429I DISCOVERY SDRAM Private Function Prototypes
  - STM32F429I DISCOVERY SDRAM Private Functions
  - STM32F429I DISCOVERY SDRAM Exported Types
  - STM32F429I DISCOVERY SDRAM Exported Constants
  - STM32F429I DISCOVERY SDRAM Exported Macro
  - STM32F429I DISCOVERY SDRAM Exported Functions

- STM32F429I DISCOVERY TS
  - STM32F429I DISCOVERY TS Private Types Definitions
  - STM32F429I DISCOVERY TS Private Defines
  - STM32F429I DISCOVERY TS Private Macros
  - STM32F429I DISCOVERY TS Private Variables
- STM32F429I DISCOVERY TS Private Function Prototypes
- STM32F429I DISCOVERY TS Private Functions
- STM32F429I DISCOVERY TS Exported Types
- STM32F429I DISCOVERY TS Exported Constants
- STM32F429I DISCOVERY TS Exported Macros
- STM32F429I DISCOVERY TS Exported Functions

- STM32F429I DISCOVERY EEPROM
  - STM32F429I DISCOVERY EEPROM Exported Types
  - STM32F429I DISCOVERY EEPROM Exported Constants
  - STM32F429I DISCOVERY EEPROM Exported Macros
  - STM32F429I DISCOVERY EEPROM Exported Functions

BSP User Manual by doxygen 1.7.6.1
Here are the data structures with brief descriptions:

- **LCD_DrawPropTypeDef**
- **Point**
- **TS_StateTypeDef**
### File List

Here is a list of all files with brief descriptions:

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stm32f429i_discovery.c [code]</td>
<td>This file provides a set of firmware functions to manage LEDs and push-button available on STM32F429I-Discovery Kit from STMicroelectronics.</td>
</tr>
<tr>
<td>stm32f429i_discovery.h [code]</td>
<td>This file contains definitions for STM32F429I-Discovery Kit LEDs, push-buttons hardware resources.</td>
</tr>
<tr>
<td>stm32f429i_discovery_eeprom.c [code]</td>
<td>This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory. To be able to use this driver, the switch EE_M24LR64 must be defined in your compiler preprocessor.</td>
</tr>
<tr>
<td>stm32f429i_discovery_eeprom.h [code]</td>
<td>This file contains all the prototypes for the firmware driver.</td>
</tr>
<tr>
<td>stm32f429i_discovery_gyroscope.c [code]</td>
<td>This file provides a set of functions needed to manage the MEMS gyroscope available on STM32F429I-Discovery Kit.</td>
</tr>
<tr>
<td>File Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>stm32f429i_discovery_gyroscope.h</code></td>
<td>This file contains definitions for <code>stm32f429i_discovery_gyroscope.c</code> firmware driver</td>
</tr>
<tr>
<td><code>stm32f429i_discovery_io.c</code></td>
<td>This file provides a set of functions needed to manage the STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit</td>
</tr>
<tr>
<td><code>stm32f429i_discovery_io.h</code></td>
<td>This file contains all the function prototypes for the <code>stm32f429i_discovery_io.c</code></td>
</tr>
<tr>
<td><code>stm32f429i_discovery_lcd.c</code></td>
<td>This file includes the LCD driver for ILI9341 Liquid Crystal Display Modules of STM32F429I-Discovery kit (MB1075)</td>
</tr>
<tr>
<td><code>stm32f429i_discovery_lcd.h</code></td>
<td>This file contains all the function prototypes for the <code>stm32f429i_discovery_lcd.c</code></td>
</tr>
<tr>
<td><code>stm32f429i_discovery_sdram.c</code></td>
<td>This file provides a set of functions needed to drive the IS42S16400J SDRAM memory mounted on STM32F429I-Discovery Kit</td>
</tr>
<tr>
<td><code>stm32f429i_discovery_sdram.h</code></td>
<td>This file contains all the function prototypes for the <code>stm32f429i_discovery_sdram.c</code> driver</td>
</tr>
<tr>
<td><code>stm32f429i_discovery_ts.c</code></td>
<td>This file provides a set of functions needed to manage Touch screen available with STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit</td>
</tr>
<tr>
<td><code>stm32f429i_discovery_ts.h</code></td>
<td>This file contains all the function prototypes for the <code>stm32f429i_discovery_ts.c</code> driver</td>
</tr>
</tbody>
</table>
Directories

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

- Drivers
  - BSP
    - STM32F429I-Discovery
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LOW LEVEL

STM32F429I DISCOVERY

This file provides set of firmware functions to manage Leds and push-button available on STM32F429I-Discovery Kit from STMicroelectronics. More...
## Modules

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Types</td>
<td>Definitions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Defines</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Macros</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Variables</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Functions</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Exported Constants</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Exported Macros</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Exported Functions</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Exported Types</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

This file provides set of firmware functions to manage Leds and push-button available on STM32F429I-Discovery Kit from STMicroelectronics.
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td>Modules</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## STM32F429I DISCOVERY GYROSCOPE

STM32F429I DISCOVERY
## Modules

<table>
<thead>
<tr>
<th>Category</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Private Types</td>
<td>Definitions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Private Defines</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Private Macros</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Private Variables</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Private Function Prototypes</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Private Functions</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Exported Types</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Exported Constants</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Exported Macros</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE Exported Functions</td>
<td></td>
</tr>
</tbody>
</table>
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY
## Modules

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY IO Private</td>
<td>Types Definitions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Private</td>
<td>Defines</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Private</td>
<td>Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Private</td>
<td>Variables</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Private</td>
<td>Function Prototypes</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Private</td>
<td>Functions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Exported</td>
<td>Types</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Exported</td>
<td>Constants</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Exported</td>
<td>Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO Exported</td>
<td>Functions</td>
</tr>
</tbody>
</table>

STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LCD

STM32F429I DISCOVERY

This file includes the LCD driver for (ILI9341) More...
# Modules

<table>
<thead>
<tr>
<th>Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY LCD Private TypesDefinitions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Private Defines</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Private Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Private Variables</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Private FunctionPrototypes</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Private Functions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Exported Types</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Exported Constants</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Exported Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD Exported Functions</td>
</tr>
</tbody>
</table>
Detailed Description

This file includes the LCD driver for (ILI9341)
STM32F429I-Discovery BSP User Manual

Data Structure Index

L | P | T

L

LCD_DrawPropTypeDef

P

Point

T

TS_StateTypeDef

STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LCD Exported Types

STM32F429I DISCOVERY LCD
Data Structures

```c
struct LCD_DrawPropTypeDef
struct Point
```
Typedefs

typedef struct Point * pPoint
## Enumerations

<table>
<thead>
<tr>
<th>Enum Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_StatusTypeDef</td>
<td>LCD_OK = 0, LCD_ERROR = 1, LCD_TIMEOUT = 2</td>
</tr>
<tr>
<td>Text_AlignModeTypeDef</td>
<td>CENTER_MODE = 0x01, RIGHT_MODE = 0x02, LEFT_MODE = 0x03</td>
</tr>
</tbody>
</table>

Line mode structures definition. More...
typedef struct Point * pPoint
Enumeration Type Documentation

**enum LCD_StatusTypeDef**

**Enumerator:**
- LCD_OK
- LCD_ERROR
- LCD_TIMEOUT

Definition at line 70 of file `stm32f429i_discovery_lcd.h`.

**enum Text_AlignModeTypdef**

Line mode structures definition.

**Enumerator:**
- CENTER_MODE
- RIGHT_MODE
- LEFT_MODE

Definition at line 93 of file `stm32f429i_discovery_lcd.h`.

/**
 ******************************************
 ******************************************
 *
 *
 * @file stm32f429i_discovery_lcd.h
 * @author MCD Application Team
 * @version V2.1.3
 * @date 13-January-2016
 * @brief This file contains all the functions prototypes for the
 *       stm32f429i_discovery_lcd.c driver.
 * @attention
 *
 * <h2><center>&copy; COPYRIGHT(c) 2015 STM microelectronics</center></h2>
 *
 * Redistribution and use in source and binary forms, with or without modification,
 * are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
#ifndef __STM32F429I_DISCOVERY_LCD_H
#define __STM32F429I_DISCOVERY_LCD_H

#ifdef __cplusplus
extern "C" {
#endif

/* Includes ----------------------------------*/
#include "stm32f429i_discovery.h"
#include "stm32f429i_discovery_sdram.h"
#include "./../../Utilities/Fonts/fonts.h"
#include "../Components/ili9341/ili9341.h"

/** @addtogroup BSP */
* @{
*/
/** @addtogroup STM32F429I_DISCOVERY */
* @{
*/
/** @addtogroup STM32F429I_DISCOVERY_LCD */
* @{
*/
/** @defgroup STM32F429I_DISCOVERY_LCD_Export */
*/
typedef enum {
  LCD_OK = 0,
  LCD_ERROR = 1,
  LCD_TIMEOUT = 2
} LCD_StatusTypeDef;

typedef struct {
  uint32_t TextColor;
  uint32_t BackColor;
  sFONT *pFont;
} LCD_DrawPropTypeDef;

typedef struct {
  int16_t X;
  int16_t Y;
} Point, *pPoint;

/**
 * @brief  Line mode structures definition
 */

typedef enum {
  CENTER_MODE = 0x01, /* center mode */
  RIGHT_MODE = 0x02, /* right mode */
  LEFT_MODE = 0x03, /* left mode */
} Text_AlignModeTypdef;
/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Exported_Constants
 * STM32F429I DISCOVERY LCD Exported Constants
 *
 */

#define LCD_LayerCfgTypeDef LTDC_LayerCfgTypeDef

/**
 * @brief LCD status structure definition
 *
 */

#define MAX_LAYER_NUMBER 2
#define LCD_FRAME_BUFFER ((uint32_t)0xD0000000)
#define BUFFER_OFFSET ((uint32_t)0x50000)

/**
 * @brief LCD color
 *
 */

#define LCD_COLOR_BLUE 0xFF0000FF
#define LCD_COLOR_GREEN 0xFF00FF00
#define LCD_COLOR_RED 0xFFFF0000
#define LCD_COLOR_CYAN 0xFF00FFFF
#define LCD_COLOR_MAGENTA 0xFFFF00FF
#define LCD_COLOR_YELLOW 0xFFFFFF00
#define LCD_COLOR_LIGHTBLUE 0xFF8080FF
#define LCD_COLOR_LIGHTGREEN 0xFF80FF80
#define LCD_COLOR_LIGHTRED 0xFFFF8080
#define LCD_COLOR_LIGHTCYAN 0xFF80FFFF
#define LCD_COLOR_LIGHTMAGENTA 0xFFFF80FF
#define LCD_COLOR_LIGHTYELLOW 0xFFFFFF80
```c
#define LCD_COLOR_DARKBLUE       0xFF000080
#define LCD_COLOR_DARKGREEN      0xFF008000
#define LCD_COLOR_DARKRED        0xFF800000
#define LCD_COLOR_DARKCYAN       0xFF008080
#define LCD_COLOR_DARKMAGENTA    0xFF800080
#define LCD_COLOR_DARKYELLOW     0xFF808000
#define LCD_COLOR_WHITE          0xFFFFFFFF
#define LCD_COLOR_LIGHTGRAY      0xFFD3D3D3
#define LCD_COLOR_GRAY           0xFF808080
#define LCD_COLOR_DARKGRAY       0xFF404040
#define LCD_COLOR_BLACK          0xFF000000
#define LCD_COLOR_BROWN          0xFFA52A2A
#define LCD_COLOR_ORANGE         0xFFFFA500
#define LCD_COLOR_TRANSPARENT    0xFF000000

/**
 * @brief LCD default font
 */
#define LCD_DEFAULT_FONT         Font24

/**
 * @brief LCD Layer
 */
#define LCD_BACKGROUND_LAYER     0x0000
#define LCD_FOREGROUND_LAYER     0x0001

/** @defgroup STM32F429I_DISCOVERY_LCD_Exported_Macros STM32F429I DISCOVERY LCD Exported Macros

 * @{
 */

 * @brief LCD Pixel format
 */
```
#define LCD_PIXEL_FORMAT_ARGB8888     LT DC_PIXEL_FORMAT_ARGB8888
#define LCD_PIXEL_FORMAT_RGB888       LT DC_PIXEL_FORMAT_RGB888
#define LCD_PIXEL_FORMAT_RGB565       LT DC_PIXEL_FORMAT_RGB565
#define LCD_PIXEL_FORMAT_ARGB1555     LT DC_PIXEL_FORMAT_ARGB1555
#define LCD_PIXEL_FORMAT_ARGB4444     LT DC_PIXEL_FORMAT_ARGB4444
#define LCD_PIXEL_FORMAT_L8           LT DC_PIXEL_FORMAT_L8
#define LCD_PIXEL_FORMAT_AL44         LT DC_PIXEL_FORMAT_AL44
#define LCD_PIXEL_FORMAT_AL88         LT DC_PIXEL_FORMAT_AL88

/**
 * @}
 */
/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Exported_Functions
 * STM32F429I DISCOVERY LCD Exported Functions
 * @{
 */
uint8_t  BSP_LCD_Init(void);
uint32_t BSP_LCD_GetXSize(void);
uint32_t BSP_LCD_GetYSize(void);

void    BSP_LCD_LayerDefaultInit(uint16_t LayerIndex, uint32_t FrameBuffer);
void    BSP_LCD_SetTransparency(uint32_t LayerIndex, uint8_t Transparency);
void    BSP_LCD_SetLayerAddress(uint32_t LayerIndex, uint32_t Address);
void    BSP_LCD_SetColorKeying(uint32_t Lay
erIndex, uint32_t RGBValue);

00189 void BSP_LCD_ResetColorKeying(uint32_t LayerIndex);

00190 void BSP_LCD_SetLayerWindow(uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height);

00191 void BSP_LCD_SelectLayer(uint32_t LayerIndex);

00192 void BSP_LCD_SetLayerVisible(uint32_t LayerIndex, FunctionalState state);

00193

00194 void BSP_LCD_SetTextColor(uint32_t Color);

00195 void BSP_LCD_SetBackColor(uint32_t Color);

00196 uint32_t BSP_LCD_GetTextColor(void);

00197 uint32_t BSP_LCD_GetBackColor(void);

00198 void BSP_LCD_SetFont(sFONT *pFonts);

00199 sFONT BSP_LCD_GetFont(void);

00200

00201 uint32_t BSP_LCD_ReadPixel(uint16_t Xpos, uint16_t Ypos);

00202 void BSP_LCD_DrawPixel(uint16_t Xpos, uint16_t Ypos, uint32_t pixel);

00203 void BSP_LCD_Clear(uint32_t Color);

00204 void BSP_LCD_ClearStringLine(uint32_t Line);

00205 void BSP_LCD_DisplayStringAtLine(uint16_t Line, uint8_t *ptr);

00206 void BSP_LCD_DisplayStringAt(uint16_t X, uint16_t Y, uint8_t *pText, Text_AlignModeTypeDef mode);

00207 void BSP_LCD_DisplayChar(uint16_t Xpos, uint16_t Ypos, uint8_t Ascii);

00208

00209 void BSP_LCD_DrawHLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length);
void BSP_LCD_DrawVLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length);
void BSP_LCD_DrawLine(uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2);
void BSP_LCD_DrawRect(uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height);
void BSP_LCD_DrawCircle(uint16_t Xpos, uint16_t Ypos, uint16_t Radius);
void BSP_LCD_DrawPolygon(pPoint Points, uint16_t PointCount);
void BSP_LCD_DrawEllipse(int Xpos, int Ypos, int XRadius, int YRadius);
void BSP_LCD_DrawBitmap(uint32_t X, uint32_t Y, uint8_t *pBmp);
void BSP_LCD_FillRect(uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height);
void BSP_LCD_FillCircle(uint16_t Xpos, uint16_t Ypos, uint16_t Radius);
void BSP_LCD_FillTriangle(uint16_t X1, uint16_t X2, uint16_t X3, uint16_t Y1, uint16_t Y2, uint16_t Y3);
void BSP_LCD_FillPolygon(pPoint Points, uint16_t PointCount);
void BSP_LCD_FillEllipse(int Xpos, int Ypos, int XRadius, int YRadius);
void BSP_LCD_DisplayOff(void);
void BSP_LCD_DisplayOn(void);
/**
 * @}
 */
/**
 * @}
 */
/**
 * @}}
 */

#ifdef __cplusplus
} }
#endif
#endif /* __STM32F429I_DISCOVERY_LCD_H */

/****************************
(C) COPYRIGHT STMicroelectronics
*****END OF FILE****/
stm32f429i_discovery_lcd.c

Go to the documentation of this file.

/**
 * @file     stm32f429i_discovery_lcd.c
 * @author   MCD Application Team
 * @version  V2.1.3
 * @date     13-January-2016
 * @brief    This file includes the LCD driver for ILI9341 Liquid Crystal Display Modules of STM32F429I-Discovery kit (MB1075).
 */

 /**
 * @attention
 *
 * <h2><center>&copy; COPYRIGHT(c) 2015 STM microelectronics</center></h2>
 *
 * Redistribution and use in source and binary forms, with or without modification,
 * are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.

3. Neither the name of STMicroelectronics nor the names of its contributors
may be used to endorse or promote products derived from this software
without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
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.
1. How To use this driver:
- This driver is used to drive directly an LCD TFT using LTDC controller.
- This driver select dynamically the mounted LCD, ILI9341 240x320 LCD mounted on MB1075B discovery board, and use the adequate timing and setting for the specified LCD using device ID of the ILI9341 mounted on MB1075B discovery board.

2. Driver description:
+ Initialization steps:
- Initialize the LCD using the LCD_Init() function.
- Apply the Layer configuration using LCD_LayerDefaultInit() function.
- Select the LCD layer to be used using LCD_SelectLayer() function.
- Enable the LCD display using LCD_DispayOn() function.

+ Options
- Configure and enable the color keying functionality using LCD_SetColorKeying() function.
- Modify in the fly the transparency and/or the frame buffer address

- LCD_SetTransparency()
- LCD_SetLayerAddress()

+ Display on LCD
- Clear the hole LCD using LCD_Clear() function or only one specified string
- LCD_ClearStringLine() function.
- Display a character on the specified line and column using LCD_DisplayChar() function or a complete string line using LCD_DisplayStringAtLine() function.
- Display a string line on the specified position (x, y in pixel) and align mode
- LCD_DisplayStringAtLine() function.
- Draw and fill a basic shapes (dot, line, rectangle, circle, ellipse, .. bitmap)
- LCD_DisplayStringAtLine() function.

#include "stm32f429i_discovery_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"
/** @addtogroup BSP */

/** @addtogroup STM32F429I_DISCOVERY */

/**
 * @defgroup STM32F429I_DISCOVERY_LCD
 * @brief This file includes the LCD driver for (ILI9341)
 */

/** @defgroup STM32F429I_DISCOVERY_LCD_Private_TypesDefinitions */

/** @defgroup STM32F429I_DISCOVERY_LCD_Private_Defines */

#define POLY_X(Z) ((int32_t)(((Points + Z)->X))

#define POLY_Y(Z) ((int32_t)(((Points + Z)->Y))

/** @} */

/** @defgroup STM32F429I_DISCOVERY_LCD_Private_TypesDefinitions */

/** @defgroup STM32F429I_DISCOVERY_LCD_Private_Defines */

*/
/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Private_Macros
 * STM32F429I DISCOVERY LCD Private Macros
 *
 * #define ABS(X) ((X) > 0 ? (X) : -(X))
 */

/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Private_Variables
 * STM32F429I DISCOVERY LCD Private Variables
 *
 * static LTDC_HandleTypeDef LtdcHandler;
 * static DMA2D_HandleTypeDef Dma2dHandler;
 * static RCC_PercistCLKInitTypeDef PeriphClkInitStruct;
 * static uint32_t ActiveLayer = 0;
 * static LCD_DrawPropTypeDef DrawProp[MAX_LAYER_NUMBER];
 * static LCDDrvTypeDef *LcdDrv;
 */

/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Private_FunctionPrototypes
 * STM32F429I DISCOVERY LCD Private FunctionPrototypes
 *
 * static void MspInit(void);
 * static void DrawChar(uint16_t Xpos, uint16_t Ypos, const uint8_t *c);
 */

static void MspInit(void);
static void DrawChar(uint16_t Xpos, uint16_t Ypos, const uint8_t *c);
static void FillBuffer(uint32_t LayerIndex, void *pDst, uint32_t xSize, uint32_t ySize, uint32_t OffLine, uint32_t ColorIndex);

static void ConvertLineToARGB8888(void *pSrc, void *pDst, uint32_t xSize, uint32_t ColorMode);

/** @defgroup STM32F429I_DISCOVERY_LCD_Private_Functions STM32F429I DISCOVERY LCD Private Functions */

/**
@brief Initializes the LCD.
@retval LCD state */
uint8_t BSP_LCD_Init(void) {
  /* On STM32F429I-DISCO, it is not possible to read ILI9341 ID because */
  /* PIN EXTC is not connected to VDD and then LCD_READ_ID4 is not accessible. */
  /* In this case, ReadID function is bypassed. */
  /*if(ili9341_drv.ReadID() == ILI9341_ID)*/
  /* LTDC Configuration ------------------------------*/
  LtdcHandler.Instance = LTDC;
  /* Timing configuration (Typical configuration from ILI9341 datasheet)*/
  HSYNC=10 (9+1)
  HBP=20 (29-10+1)
/* Configure horizontal synchronization width */
LtdcHandler.Init.HorizontalSync = ILI9341_HSYNC;
/* Configure vertical synchronization height */
LtdcHandler.Init.VerticalSync = ILI9341_VSYNC;
/* Configure accumulated horizontal back porch */
LtdcHandler.Init.AccumulatedHBP = ILI9341_HBP;
/* Configure accumulated vertical back porch */
LtdcHandler.Init.AccumulatedVBP = ILI9341_VBP;
/* Configure accumulated active width */
LtdcHandler.Init.AccumulatedActiveW = 269;
/* Configure accumulated active height */
LtdcHandler.Init.AccumulatedActiveH = 323;
/* Configure total width */
LtdcHandler.Init.TotalWidth = 279;
/* Configure total height */
LtdcHandler.Init.TotalHeight = 327;
/* Configure R,G,B component values for LCD background color */
LtdcHandler.Init.Backcolor.Red = 0;
LtdcHandler.Init.Backcolor.Blue = 0;
LtdcHandler.Init.Backcolor.Green = 0;

/* LCD clock configuration */
/* PLLSAI_VCO Input = HSE_VALUE/PLL_M = 1 Mhz */
/* PLLSAI_VCO Output = PLLSAI_VCO Input * PLLSAIN = 192 Mhz */
/* PLLLCDCLK = PLLSAI_VCO Output/PLLSAIR = 192/4 = 48 Mhz */
/* LTDC clock frequency = PLLLCDCLK / LTDC_PLLSAI_DIVR_8 = 48/4 = 6Mhz */
PeriphClkInitStruct.PeriphClockSelection = RCC_PERIPHCLK_LTDC;
PeriphClkInitStruct.PLLSAI.PLLSAIN = 192;
PeriphClkInitStruct.PLLSAI.PLLSAIR = 4;
PeriphClkInitStruct.PLLSAIDivR = RCC_PLLSAIDIVR_8;
HAL_RCCEx_PeriphCLKConfig(&PeriphClkInitStruct);

/* Polarity */
LtdcHandler.Init.HSPolarity = LTDC_HSPOLARITY_AL;
LtdcHandler.Init.VSPolarity = LTDC_VSPOLARITY_AL;
LtdcHandler.Init.DEPolarity = LTDC_DEPOLARITY_AL;
LtdcHandler.Init.PCPolarity = LTDC_PCPOLARITY_IPC;

MspInit();
HAL_LTDC_Init(&LtdcHandler);
/* Select the device */
LcdDrv = &ili9341_drv;

/* LCD Init */
LcdDrv->Init();

/* Initialize the SDRAM */
BSP_SDRAM_Init();

/* Initialize the font */
BSP_LCD_SetFont(&LCD_DEFAULT_FONT);

return LCD_OK;

/** *
@brief Gets the LCD X size.
@retval The used LCD X size *

uint32_t BSP_LCD_GetXSize(void)
{
    return LcdDrv->GetLcdPixelWidth();
}

/** *
@brief Gets the LCD Y size.
@retval The used LCD Y size *

uint32_t BSP_LCD_GetYSize(void)
{
    return LcdDrv->GetLcdPixelHeight();
}

/** *
@brief Initializes the LCD layers.
@param LayerIndex: the layer foreground

or background.

* @param FB_Address: the layer frame buffer.
 */

void BSP_LCD_LayerDefaultInit(uint16_t LayerIndex, uint32_t FB_Address)
{
    LCD_LayerCfgTypeDef Layercfg;

    /* Layer Init */
    Layercfg.WindowX0 = 0;
    Layercfg.WindowX1 = BSP_LCD_GetXSize();
    Layercfg.WindowY0 = 0;
    Layercfg.WindowY1 = BSP_LCD_GetYSize();
    Layercfg.PixelFormat = LTDC_PIXEL_FORMAT_ARGB8888;
    Layercfg.FBStartAdress = FB_Address;
    Layercfg.Alpha = 255;
    Layercfg.Alpha0 = 0;
    Layercfg.Backcolor.Blue = 0;
    Layercfg.Backcolor.Green = 0;
    Layercfg.Backcolor.Red = 0;
    Layercfg.BlendingFactor1 = LTDC_BLENDING_FACTOR1_PAxCA;
    Layercfg.BlendingFactor2 = LTDC_BLENDING_FACTOR2_PAxCA;
    Layercfg.ImageWidth = BSP_LCD_GetXSize();
    Layercfg.ImageHeight = BSP_LCD_GetYSize();
    HAL_LTDC_ConfigLayer(&LtdcHandler, &Layercfg, LayerIndex);

    DrawProp[LayerIndex].BackColor = LCD_COLOR_WHITE;
    DrawProp[LayerIndex].pFont = &Font24;
    DrawProp[LayerIndex].TextColor = LCD_COLOR_BLACK;
/* Dithering activation */

HAL_LTDC_EnableDither(&LtdcHandler);
}

/**
 * @brief Selects the LCD Layer.
 * @param LayerIndex: the Layer foreground or background.
 */

void BSP_LCD_SelectLayer(uint32_t LayerIndex)
{
    ActiveLayer = LayerIndex;
}

/**
 * @brief Sets a LCD Layer visible.
 * @param LayerIndex: the visible Layer.
 * @param state: new state of the specified layer.
 * This parameter can be: ENABLE or DISABLE.
 */

void BSP_LCD_SetLayerVisible(uint32_t LayerIndex, FunctionalState state)
{
    if(state == ENABLE)
    {
        __HAL_LTDC_LAYER_ENABLE(&LtdcHandler, LayerIndex);
    }
    else
    {
        __HAL_LTDC_LAYER_DISABLE(&LtdcHandler, LayerIndex);
    }
__HAL_LTDC_RELOAD_CONFIG(&LtdcHandler);

} /* @brief Configures the Transparency. */
/* @param LayerIndex: the Layer foreground or background. */
/* @param Transparency: the Transparency, 0x00 to 0xFF. */

void BSP_LCD_SetTransparency(uint32_t Index, uint8_t Transparency)
{
	HAL_LTDC_SetAlpha(&LtdcHandler, Transparency, Index);
}

/* @brief Sets a LCD layer frame buffer address. */
/* @param LayerIndex: specifies the Layer foreground or background */
/* @param Address: new LCD frame buffer value */

void BSP_LCD_SetLayerAddress(uint32_t Index, uint32_t Address)
{
	HAL_LTDC_SetAddress(&LtdcHandler, Address, Index);
}

/* @brief Sets the Display window. */
/* @param LayerIndex: layer index */
/* @param Xpos: LCD X position */
/* @param Ypos: LCD Y position */
/* @param Width: LCD window width */
/* @param Height: LCD window height */

void BSP_LCD_SetLayerWindow(uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
{
    /* reconfigure the layer size */
    HAL_LTDC_SetWindowSize(&LtdcHandler, Width, Height, LayerIndex);

    /* reconfigure the layer position */
    HAL_LTDC_SetWindowPosition(&LtdcHandler, Xpos, Ypos, LayerIndex);
}

/**
 * @brief Configures and sets the color Ke ying.
 * @param LayerIndex: the Layer foreground or background
 * @param RGBValue: the Color reference
 */

void BSP_LCD_SetColorKeying(uint32_t LayerIndex, uint32_t RGBValue)
{
    /* Configure and Enable the color Keying for LCD Layer */
    HAL_LTDC_ConfigColorKeying(&LtdcHandler, RGBValue, LayerIndex);
    HAL_LTDC_EnableColorKeying(&LtdcHandler, LayerIndex);
}

/**
 * @brief Disables the color Keying.
 */
void BSP_LCD_ResetColorKeying(uint32_t LayerIndex)
{
    /* Disable the color Keying for LCD Layer */
    HAL_LTDC_DisableColorKeying(&LtdcHandler, LayerIndex);
}

uint32_t BSP_LCD_GetTextColor(void)
{
    return DrawProp[ActiveLayer].TextColor;
}

uint32_t BSP_LCD_GetBackColor(void)
{
    return DrawProp[ActiveLayer].BackColor;
}

void BSP_LCD_SetTextColor(uint32_t Color)
{
DrawProp[ActiveLayer].TextColor = Color;
}

/**
 * @brief Sets the Background color.
 * @param Color: the layer Background color code ARGB(8-8-8-8)
 */
void BSP_LCD_SetBackColor(uint32_t Color)
{
    DrawProp[ActiveLayer].BackColor = Color;
}

/**
 * @brief Sets the Text Font.
 * @param pFonts: the layer font to be used
 */
void BSP_LCD_SetFont(sFONT *pFonts)
{
    DrawProp[ActiveLayer].pFont = pFonts;
}

/**
 * @brief Gets the Text Font.
 * @retval Layer font
 */
sFONT *BSP_LCD_GetFont(void)
{
    return DrawProp[ActiveLayer].pFont;
}

/**
 * @brief Reads Pixel.
 * @param Xpos: the X position
 * @param Ypos: the Y position
 * @retval RGB pixel color

uint32_t BSP_LCD_ReadPixel(uint16_t Xpos, uint16_t Ypos)
{
    uint32_t ret = 0;
    if(LtdcHandler.LayerCfg[ActiveLayer].PixelFormat == LTDC_PIXEL_FORMAT_ARGB8888)
    {
        /* Read data value from SDRAM memory */
        ret = *(__IO uint32_t*)(LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (2*(Ypos*BSP_LCD_GetXSize() + Xpos)));
    }
    else if(LtdcHandler.LayerCfg[ActiveLayer].PixelFormat == LTDC_PIXEL_FORMAT_RGB888)
    {
        /* Read data value from SDRAM memory */
        ret = *(__IO uint32_t*)(LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (2*(Ypos*BSP_LCD_GetXSize() + Xpos))) & 0x00FFFFFF;
    }
    else if(((LtdcHandler.LayerCfg[ActiveLayer].PixelFormat == LTDC_PIXEL_FORMAT_RGB565) ||
        (LtdcHandler.LayerCfg[ActiveLayer].PixelFormat == LTDC_PIXEL_FORMAT_ARGB4444) ||
        (LtdcHandler.LayerCfg[ActiveLayer].PixelFormat == LTDC_PIXEL_FORMAT_AL88))
    {
        /* Read data value from SDRAM memory */
        ret = *(__IO uint16_t*)(LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (2*(Ypos*BSP_LCD_GetXSize() + Xpos)));
    }
    else
    {
        /* Read data value from SDRAM memory */
    }
ret = *(__IO uint8_t*) (LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (2*(Ypos*BSP_LCD_GetXSize()) + Xpos));
}

return ret;

/**
 * @brief Clears the hole LCD.
 * @param Color: the color of the background
 */
void BSP_LCD_Clear(uint32_t Color) {
    /* Clear the LCD */
    FillBuffer(ActiveLayer, (uint32_t *)(LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress), BSP_LCD_GetXSize(), BSP_LCD_GetYSize(), 0, Color);
}

/**
 * @brief Clears the selected line.
 * @param Line: the line to be cleared
 */
void BSP_LCD_ClearStringLine(uint32_t Line) {
    uint32_t colorbackup = DrawProp[ActiveLayer].TextColor;
    DrawProp[ActiveLayer].TextColor = DrawProp[ActiveLayer].BackColor;
    /* Draw rectangle with background color */
    BSP_LCD_FillRect(0, (Line * DrawProp[ActiveLayer].pFont->Height), BSP_LCD_GetXSize(), DrawProp[ActiveLayer].pFont->Height);
}
DrawProp[ActiveLayer].TextColor = colorblack;
BSP_LCD_SetTextColor(DrawProp[ActiveLayer].TextColor);
}
/**
 * @brief Displays one character.
 * @param Xpos: start column address
 * @param Ypos: the line where to display the character shape
 * @param Ascii: character ascii code, must be between 0x20 and 0x7E
 */
void BSP_LCD_DisplayChar(uint16_t Xpos, uint16_t Ypos, uint8_t Ascii)
{
   DrawChar(Xpos, Ypos, &DrawProp[ActiveLayer].pFont->table[(Ascii-' ') *
      DrawProp[ActiveLayer].pFont->Height * ((DrawProp[ActiveLayer].pFont->Width + 7) / 8)));
}
/**
 * @brief Displays a maximum of 60 char on the LCD.
 * @param X: pointer to x position (in pixel)
 * @param Y: pointer to y position (in pixel)
 * @param pText: pointer to string to display on LCD
 * @param mode: The display mode
 * This parameter can be one of the following values:
 * @arg CENTER_MODE
void BSP_LCD_DisplayStringAt(uint16_t X, uint16_t Y, uint8_t *pText, Text_AlignModeTypeDef mode)
{
    uint16_t refcolumn = 1, i = 0;
    uint32_t size = 0, xsize = 0;
    uint8_t *ptr = pText;

    /* Get the text size */
    while (*ptr++) size ++;

    /* Characters number per line */
    xsize = (BSP_LCD_GetXSize() / DrawProp[ActiveLayer].pFont->Width);

    switch (mode)
    {
    case CENTER_MODE:
        refcolumn = X + ((xsize - size)* DrawProp[ActiveLayer].pFont->Width) / 2;
        break;
    case LEFT_MODE:
        refcolumn = X;
        break;
    case RIGHT_MODE:
        refcolumn = X + ((xsize - size)*DrawProp[ActiveLayer].pFont->Width);
        break;
    }
default:
{
  refcolumn = X;
  break;
}

/* Send the string character by character on LCD */
while ((*pText != 0) & (((BSP_LCD_GetXSize () - (i*DrawProp[ActiveLayer].pFont->Width)) & 0xFFF) >= DrawProp[ActiveLayer].pFont->Width))
{
  /* Display one character on LCD */
  BSP_LCD_DisplayChar(refcolumn, Y, *pText);
  /* Decrement the column position by 16 */
  refcolumn += DrawProp[ActiveLayer].pFont->Width;
  /* Point on the next character */
  pText++; i++;
}

/**
 * @brief Displays a maximum of 20 char on the LCD.
 * @param Line: the Line where to display the character shape
 * @param ptr: pointer to string to display on LCD
 */
void BSP_LCD_DisplayStringAtLine(uint16_t Line, uint8_t *ptr)
{
void BSP_LCD_DrawHLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length) {
    uint32_t xaddress = 0;
    /* Get the line address */
    xaddress = (LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress) + 4*(BSP_LCD_GetXSize() * Ypos + Xpos);
    /* Write line */
    FillBuffer(ActiveLayer, (uint32_t *)xaddress, Length, 1, 0, DrawProp[ActiveLayer].TextColor);
}

void BSP_LCD_DrawVLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length) {
    uint32_t xaddress = 0;
    /* Get the line address */
    xaddress = (LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress) + 4*(BSP_LCD_GetYSize() * Xpos + Ypos);
    /* Write line */
    FillBuffer(ActiveLayer, (uint32_t *)xaddress, Length, 0, 1, DrawProp[ActiveLayer].TextColor);
}
/* Get the line address */
xaddress = (LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress) + 4*(BSP_LCD_GetXSize() * Ypos + Xpos);

/* Write line */
FillBuffer(ActiveLayer, (uint32_t *)xaddress, 1, Length, (BSP_LCD_GetXSize() - 1), DrawProp[ActiveLayer].TextColor);

/**
 * @brief Displays an uni-line (between two points).
 * @param X1: the point 1 X position
 * @param Y1: the point 1 Y position
 * @param X2: the point 2 X position
 * @param Y2: the point 2 Y position
 */

void BSP_LCD_DrawLine(uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2)
{
    int16_t deltax = 0, deltay = 0, x = 0, y = 0, xinc1 = 0, xinc2 = 0,
    yinc1 = 0, yinc2 = 0, den = 0, num = 0, numadd = 0, numpixels = 0,
    curpixel = 0;

    deltax = ABS(X2 - X1); /* The difference between the x's */
    deltay = ABS(Y2 - Y1); /* The difference between the y's */
    x = X1; /* Start x off at the first pixel */
    y = Y1; /* Start y off at the first pixel */
if \((X_2 \geq X_1)\) /* The x-values are increasing */
{
    xinc1 = 1;
    xinc2 = 1;
}
else /* The x-values are decreasing */
{
    xinc1 = -1;
    xinc2 = -1;
}

if \((Y_2 \geq Y_1)\) /* The y-values are increasing */
{
    yinc1 = 1;
    yinc2 = 1;
}
else /* The y-values are decreasing */
{
    yinc1 = -1;
    yinc2 = -1;
}

if \((\text{deltax} \geq \text{deltay})\) /* There is at least one x-value for every y-value */
{
    xinc1 = 0; /* Don't change the x when numerator \(\geq\) denominator */
    yinc2 = 0; /* Don't change the y for every iteration */
    den = \text{deltax};
    num = \text{deltax} / 2;
    numadd = \text{deltay};
    numpixels = \text{deltax}; /* There are
more x-values than y-values */
00657  }
00658 else /* There is
00659 at least one y-value for every x-value */
00660 {
00661 xinc2 = 0; /* Don't cha
00662 change the x for every iteration */
00663 yinc1 = 0; /* Don't cha
00664 nte the y when numerator >= denominator */
00665 den = deltax;
00666 num = deltax / 2;
00667 numadd = deltax;
00668 numpixels = deltax; /* There are
00669 more y-values than x-values */
00670 }
00671 for (curpixel = 0; curpixel <= numpixels;
00672 curpixel++)
00673 {
00674 BSP_LCD_DrawPixel(x, y, DrawProp[ActiveL
00675 ayer].TextColor); /* Draw the current pixel */
00676 num += numadd;
00677 /* Increase the numerator by the top of the frac
00678 tion */
00679 if (num >= den)
00680 /* Check if numerator >= denominator */
00681 {
00682 num -= den;
00683 /* Calculate the new numerator value */
00684 x += xinc1;
00685 /* Change the x as appropriate */
00686 y += yinc1;
00687 /* Change the y as appropriate */
00688 }
00689 x += xinc2;
00690 /* Change the x as appropriate */
00691 y += yinc2;
/** Change the y as appropriate */
00680 }  
00681 }  
00682  
00683 /**
00684   * @brief Displays a rectangle.
00685   * @param Xpos: the X position
00686   * @param Ypos: the Y position
00687   * @param Height: display rectangle height
00688   * @param Width: display rectangle width
00689 */
00690 void BSP_LCD_DrawRect(uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)  
00691 {  
00692   /* Draw horizontal lines */
00693   BSP_LCD_DrawHLine(Xpos, Ypos, Width);
00694   BSP_LCD_DrawHLine(Xpos, (Ypos + Height), Width);
00695  
00696   /* Draw vertical lines */
00697   BSP_LCD_DrawVLine(Xpos, Ypos, Height);
00698   BSP_LCD_DrawVLine((Xpos + Width), Ypos, Height);
00699 }  
00700  
00701 /**
00702   * @brief Displays a circle.
00703   * @param Xpos: the X position
00704   * @param Ypos: the Y position
00705   * @param Radius: the circle radius
00706 */
00707 void BSP_LCD_DrawCircle(uint16_t Xpos, uint16_t Ypos, uint16_t Radius)  
00708 {  
00709   int32_t d; /* Decision Variable */
00710   uint32_t curx; /* Current X Value */
00711   uint32_t cury; /* Current Y Value */
d = 3 - (Radius << 1);
curx = 0;
cury = Radius;

while (curx <= cury)
{
    BSP_LCD_DrawPixel((Xpos + curx), (Ypos - cury), DrawProp[ActiveLayer].TextColor);
    BSP_LCD_DrawPixel((Xpos - curx), (Ypos - cury), DrawProp[ActiveLayer].TextColor);
    BSP_LCD_DrawPixel((Xpos + cury), (Ypos - curx), DrawProp[ActiveLayer].TextColor);
    BSP_LCD_DrawPixel((Xpos - cury), (Ypos - curx), DrawProp[ActiveLayer].TextColor);
    BSP_LCD_DrawPixel((Xpos + cury), (Ypos + curx), DrawProp[ActiveLayer].TextColor);
    BSP_LCD_DrawPixel((Xpos - cury), (Ypos + curx), DrawProp[ActiveLayer].TextColor);
    BSP_LCD_DrawPixel((Xpos + cury), (Ypos + curx), DrawProp[ActiveLayer].TextColor);
    BSP_LCD_DrawPixel((Xpos - cury), (Ypos + curx), DrawProp[ActiveLayer].TextColor);

    if (d < 0)
    {
        d += (curx << 2) + 6;
    }
    else
    {
        d += ((curx - cury) << 2) + 10;
        cury--;
    }
    curx++;
}

/**
 * @brief Displays an poly-line (between many points).
 * @param Points: pointer to the points array
 * @param PointCount: Number of points
 */

void BSP_LCD_DrawPolygon(pPoint Points, uint16_t PointCount)
{
    int16_t x = 0, y = 0;

    if(PointCount < 2)
    {
        return;
    }

    BSP_LCD_DrawLine(Points->X, Points->Y, (Points+PointCount-1)->X, (Points+PointCount-1)->Y);

    while(--PointCount)
    {
        x = Points->X;
        y = Points->Y;
        Points++;
        BSP_LCD_DrawLine(x, y, Points->X, Points->Y);
    }
}

/**
 * @brief Displays an Ellipse.
 * @param Xpos: the X position
 * @param Ypos: the Y position
 * @param XRadius: the X radius of ellipse
 * @param YRadius: the Y radius of ellipse
 */
```c
void BSP_LCD_DrawEllipse(int Xpos, int Ypos, int XRadius, int YRadius)
{
    int x = 0, y = -YRadius, err = 2-2*XRadius, e2;
    float k = 0, rad1 = 0, rad2 = 0;
    rad1 = XRadius;
    rad2 = YRadius;
    k = (float)(rad2/rad1);
    do {
        BSP_LCD_DrawPixel((Xpos-(uint16_t)(x/k)), (Ypos+y), DrawProp[ActiveLayer].TextColor);
        BSP_LCD_DrawPixel((Xpos+(uint16_t)(x/k)), (Ypos+y), DrawProp[ActiveLayer].TextColor);
        BSP_LCD_DrawPixel((Xpos+(uint16_t)(x/k)), (Ypos-y), DrawProp[ActiveLayer].TextColor);
        BSP_LCD_DrawPixel((Xpos-(uint16_t)(x/k)), (Ypos-y), DrawProp[ActiveLayer].TextColor);
        e2 = err;
        if (e2 <= x) {
            err += ++x*2+1;
            if (-y == x && e2 <= y) e2 = 0;
        }
        if (e2 > y) err += ++y*2+1;
    } while (y <= 0);
}
```

```markdown
/* @brief Displays a bitmap picture loaded in the internal Flash (32 bpp).
   * @param X: the bmp x position in the LCD
```
* @param Y: the bmp Y position in the LCD
* @param pBmp: Bmp picture address in the internal Flash
*/

void BSP_LCD_DrawBitmap(uint32_t X, uint32_t Y, uint8_t *pBmp)
{
    uint32_t index = 0, width = 0, height = 0, bitpixel = 0;
    uint32_t address;
    uint32_t inputcolormode = 0;

    /* Get bitmap data address offset */
    index = *__IO uint16_t (*) (pBmp + 10);
    index |= *__IO uint16_t (*) (pBmp + 12) << 16;

    /* Read bitmap width */
    width = *__IO uint16_t (*) (pBmp + 18);
    width |= *__IO uint16_t (*) (pBmp + 20) << 16;

    /* Read bitmap height */
    height = *__IO uint16_t (*) (pBmp + 22);
    height |= *__IO uint16_t (*) (pBmp + 24) << 16;

    /* Read bit/pixel */
    bitpixel = *__IO uint16_t (*) (pBmp + 28);

    /* Set Address */
    address = LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (((BSP_LCD_GetXSize())*Y) + X)*(4);

    /* Get the Layer pixel format */
    if ((bitpixel/8) == 4)
```c
else if ((bitpixel/8) == 2) {
    inputcolormode = CM_RGB565;
}
else {
    inputcolormode = CM_RGB888;
}

/* bypass the bitmap header */
pBmp += (index + (width * (height - 1) * (bitpixel/8)));

/* Convert picture to ARGB8888 pixel format */
for(index=0; index < height; index++) {
    /* Pixel format conversion */
    ConvertLineToARGB8888((uint32_t *)pBmp, (uint32_t *)address, width, inputcolormode);

    /* Increment the source and destination buffers */
    address+= ((BSP_LCD_GetXSize() - width + width)*4);
    pBmp -= width*(bitpixel/8);
}

/**
 * @brief Displays a full rectangle.
 * @param Xpos: the X position
 * @param Ypos: the Y position
 * @param Height: rectangle height
```
void BSP_LCD_FillRect(uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height) {
    uint32_t xaddress = 0;
    /* Set the text color */
    BSP_LCD_SetTextColor(DrawProp[ActiveLayer].TextColor);
    /* Get the rectangle start address */
    xaddress = (LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress) + 4*(BSP_LCD_GetXSize()*Ypos + Xpos);
    /* Fill the rectangle */
    FillBuffer(ActiveLayer, (uint32_t*)xaddress, Width, Height, (BSP_LCD_GetXSize() - Width), DrawProp[ActiveLayer].TextColor);
}

void BSP_LCD_FillCircle(uint16_t Xpos, uint16_t Ypos, uint16_t Radius) {
    int32_t d;    // Decision Variable */
    uint32_t curx;  // Current X Value */
    uint32_t cury;  // Current Y Value */
    d = 3 - (Radius << 1);
curx = 0;
cury = Radius;
BSP_LCD_SetTextColor(DrawProp[ActiveLayer].TextColor);

while (curx <= cury)
{
if (cury > 0)
{
BSP_LCD_DrawHLine(Xpos - cury, Ypos + curx, 2*cury);
BSP_LCD_DrawHLine(Xpos - cury, Ypos - curx, 2*cury);
}
if (curx > 0)
{
BSP_LCD_DrawHLine(Xpos - curx, Ypos - cury, 2*curx);
BSP_LCD_DrawHLine(Xpos - curx, Ypos + cury, 2*curx);
}
if (d < 0)
{
d += (curx << 2) + 6;
}
else
{
d += ((curx - cury) << 2) + 10;
cury--;
}
curx++;
}
BSP_LCD_SetTextColor(DrawProp[ActiveLayer].TextColor);
BSP_LCD_DrawCircle(Xpos, Ypos, Radius);

/**
 * @brief Fill triangle.
 * @param X1: the point 1 x position
 * @param Y1: the point 1 y position
 * @param X2: the point 2 x position
 * @param Y2: the point 2 y position
 * @param X3: the point 3 x position
 * @param Y3: the point 3 y position
 */
void BSP_LCD_FillTriangle(uint16_t X1, uint16_t X2, uint16_t X3, uint16_t Y1, uint16_t Y2, uint16_t Y3)
{
    int16_t deltax = 0, deltay = 0, x = 0, y = 0, xinc1 = 0, xinc2 = 0,
    yinc1 = 0, yinc2 = 0, den = 0, num = 0, numadd = 0, numpixels = 0,
    curpixel = 0;

deltax = ABS(X2 - X1); /* The difference between the x's */
deltay = ABS(Y2 - Y1); /* The difference between the y's */
    x = X1; /* Start x off at the first pixel */
y = Y1; /* Start y off at the first pixel */
    if (X2 >= X1) /* The x-values are increasing */
    {
        xinc1 = 1;
        xinc2 = 1;
    }
}
/* The x-values are decreasing */
{
  xinc1 = -1;
  xinc2 = -1;
}

if (Y2 >= Y1) /* The y-values are increasing */
{
  yinc1 = 1;
  yinc2 = 1;
}

else /* The y-values are decreasing */
{
  yinc1 = -1;
  yinc2 = -1;
}

if (deltax >= deltay) /* There is at least one x-value for every y-value */
{
  xinc1 = 0; /* Don't change the x when numerator >= denominator */
  yinc2 = 0; /* Don't change the y for every iteration */
  den = deltax;
  num = deltax / 2;
  numadd = deltay;
  numpixels = deltax; /* There are more x-values than y-values */
}

else /* There is at least one y-value for every x-value */
{
  xinc2 = 0; /* Don't cha
nge the x for every iteration */
00981   yinc1 = 0;       /* Don't change the y when numerator >= denominator */
00982   den = deltay;
00983   num = deltay / 2;
00984   numadd = deltax;
00985   numpixels = deltay;   /* There are more y-values than x-values */
00986   }
00987
00988   for (curpixel = 0; curpixel <= numpixels; curpixel++)
00989   {
00990       BSP_LCD_DrawLine(x, y, X3, Y3);
00991
00992       num += numadd;      /* Increase the numerator by the top of the fraction */
00993       if (num >= den)    /* Check if numerator >= denominator */
00994           {
00995               num -= den;    /* Calculate the new numerator value */
00996               x += xinc1;  /* Change the x as appropriate */
00997               y += yinc1;  /* Change the y as appropriate */
00998           }
00999       x += xinc2;       /* Change the x as appropriate */
01000       y += yinc2;       /* Change the y as appropriate */
01001   }
01002 }
01003
01004 /**
01005   * @brief Displays a full poly-line (between many points).
```c
void BSP_LCD_FillPolygon(pPoint Points, uint16_t PointCount)
{
    int16_t x = 0, y = 0, x2 = 0, y2 = 0, xcenter = 0, ycenter = 0, xfirst = 0, yfirst = 0, pixelx = 0, pixely = 0, counter = 0;
    uint16_t imageleft = 0, imageright = 0, imagetop = 0, imagebottom = 0;
    for(counter = 1; counter < PointCount; counter++)
    {
        pixelx = POLY_X(counter);
        if(pixelx < imageleft)
        {
            imageleft = pixelx;
        }
        if(pixelx > imageright)
        {
            imageright = pixelx;
        }
        pixely = POLY_Y(counter);
        if(pixely < imagetop)
        {
            imagetop = pixely;
        }
        if(pixely > imagebottom)
        {
```
imagebottom = pixely;
}

if(PointCount < 2) {
    return;
}
	xcenter = (imageleft + imageright)/2;
ycenter = (imagebottom + imagetop)/2;

xfirst = Points->X;
yfirst = Points->Y;

while(--PointCount) {
    x = Points->X;
y = Points->Y;
    Points++;
    x2 = Points->X;
y2 = Points->Y;
    BSP_LCD_FillTriangle(x, x2, xcenter, y, y2, ycenter);
    BSP_LCD_FillTriangle(x, xcenter, x2, y, ycenter, y2);
    BSP_LCD_FillTriangle(xcenter, x2, x, ycenter, y2, y);
}

BSP_LCD_FillTriangle(xfirst, x2, xcenter, y, y2, ycenter);
BSP_LCD_FillTriangle(xfirst, xcenter, x2, y, ycenter, y2);
BSP_LCD_FillTriangle(xcenter, x2, xfirst, ycenter, y2, yfirst);
/**
 * @brief Draw a full ellipse.
 * @param Xpos: the X position
 * @param Ypos: the Y position
 * @param XRadius: X radius of ellipse
 * @param YRadius: Y radius of ellipse.
 */

void BSP_LCD_FillEllipse(int Xpos, int Ypos, int XRadius, int YRadius) {
    int x = 0, y = -YRadius, err = 2-2*XRadius, e2;
    float K = 0, rad1 = 0, rad2 = 0;
    rad1 = XRadius;
    rad2 = YRadius;
    K = (float)(rad2/rad1);
    do {
        BSP_LCD_DrawHLine((Xpos-(uint16_t)(x/K)), (Ypos+y), (2*(uint16_t)(x/K) + 1));
        BSP_LCD_DrawHLine((Xpos-(uint16_t)(x/K)), (Ypos-y), (2*(uint16_t)(x/K) + 1));
        e2 = err;
        if (e2 <= x) {
            err += ++x*2+1;
            if (-y == x && e2 <= y) e2 = 0;
        }
        if (e2 > y) err += ++y*2+1;
    } while (y <= 0);
/**
 * @brief Enables the Display.
 */

void BSP_LCD_DisplayOn(void)
{
    if(LcdDrv->DisplayOn != NULL)
    {
        LcdDrv->DisplayOn();
    }
}

/**
 * @brief Disables the Display.
 */

void BSP_LCD_DisplayOff(void)
{
    if(LcdDrv->DisplayOff != NULL)
    {
        LcdDrv->DisplayOff();
    }
}

/*******************************************
************************************
LTDC and DMA2D BSP Routines
*******************************************
************************************/

/**
 * @brief Initializes the LTDC MSP.
 */

static void MspInit(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;

/* Enable the LTDC and DMA2D Clock */
__LTDC_CLK_ENABLE();
__DMA2D_CLK_ENABLE();

/* Enable GPIOs clock */
__GPIOA_CLK_ENABLE();
__GPIOB_CLK_ENABLE();
__GPIOC_CLK_ENABLE();
__GPIOD_CLK_ENABLE();
__GPIOF_CLK_ENABLE();
__GPIOG_CLK_ENABLE();

/* GPIOs Configuration */

+------------------------+---------------
|                      |                |
+------------------------+---------------
| LCD pins assignment    |                |
+------------------------+---------------
| LCD_TFT R2 <-> PC.10  | LCD_TFT G2 <-> PA.06 |
| LCD_TFT B2 <-> PD.06  |                |
| LCD_TFT R3 <-> PB.00  | LCD_TFT G3 <-> PG.10 |
| LCD_TFT B3 <-> PG.11  |                |
| LCD_TFT R4 <-> PA.11  | LCD_TFT G4 <-> PB.10 |
| LCD_TFT B4 <-> PG.12  |                |
| LCD_TFT R5 <-> PA.12  | LCD_TFT G5 <-> PB.11 |
| LCD_TFT B5 <-> PA.03  |                |
| LCD_TFT R6 <-> PB.01  | LCD_TFT G6 <-> PC.07 |
| LCD_TFT B6 <-> PB.08  |                |
| LCD_TFT R7 <-> PG.06  | LCD_TFT G7 <-> PD.03 |
| LCD_TFT B7 <-> PB.09  |                |
| LCD_TFT HSYNC <-> PC.06 | LCD_TFT CLK <-> PG.07 |
D_TFT DE  <->  PF.10 | 
01161           -------------------------------
                 ----------------------
01162           */
01163
01164  /* GPIOA configuration */
01165  GPIO_InitStructure.Pin = GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_6 | GPIO_PIN_11 | GPIO_PIN_12;
01166  GPIO_InitStructure.Mode = GPIO_MODE_AF_PP;
01167  GPIO_InitStructure.Pull = GPIO_NOPULL;
01168  GPIO_InitStructure.Speed = GPIO_SPEED_FAST;
01169  GPIO_InitStructure.Alternate = GPIO_AF14_LTC;
01170  HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
01171
01172  /* GPIOB configuration */
01173  GPIO_InitStructure.Pin = GPIO_PIN_8 | GPIO_PIN_9 | GPIO_PIN_10 | GPIO_PIN_11;
01174  HAL_GPIO_Init(GPIOB, &GPIO_InitStructure);
01175
01176  /* GPIOC configuration */
01177  GPIO_InitStructure.Pin = GPIO_PIN_6 | GPIO_PIN_7 | GPIO_PIN_10;
01178  HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);
01179
01180  /* GPIOD configuration */
01181  GPIO_InitStructure.Pin = GPIO_PIN_3 | GPIO_PIN_6;
01182  HAL_GPIO_Init(GPIOD, &GPIO_InitStructure);
01183
01184  /* GPIOF configuration */
01185  GPIO_InitStructure.Pin = GPIO_PIN_10;
01186  HAL_GPIO_Init(GPIOF, &GPIO_InitStructure);
01189 /* GPIOG configuration */
01190 GPIO_InitStructure.Pin = GPIO_PIN_6 | GPIO_PIN_7 | \ 
01191 GPIO_PIN_11;
01192 HAL_GPIO_Init(GPIOG, &GPIO_InitStructure);
01193 /* GPIOB configuration */
01194 GPIO_InitStructure.Pin = GPIO_PIN_0 | GPIO_PIN_1;
01195 GPIO_InitStructure.Alternate= GPIO_AF9_LTD;
01196 HAL_GPIO_Init(GPIOB, &GPIO_InitStructure);
01197 /* GPIOG configuration */
01198 GPIO_InitStructure.Pin = GPIO_PIN_10 | GPIO_PIN_12;
01199 HAL_GPIO_Init(GPIOG, &GPIO_InitStructure);
01200 }  
01201 }  
01202  
01203 /*******************************************
01204 ********************************************
01205 Static Functions
01206 ********************************************
01207 /*******************************************
01208 */
01209 /**
01210  * @brief  Writes Pixel.
01211  * @param  Xpos: the X position
01212  * @param  Ypos: the Y position
01213  * @param  RGB_Code: the pixel color in ARG
01214  *
01215 void BSP_LCD_DrawPixel(uint16_t Xpos, uint16_t Ypos, uint32_t RGB_Code)
01216  {
/* Write data value to all SDRAM memory */
*(__IO uint32_t*) (LtdcHandler.LayerCfg[ActiveLayer].FBStartAddress + (4*(Ypos*BSP_LCD_GetXSize() + Xpos))) = RGB_Code;

/**
 * @brief Draws a character on LCD.
 * @param Xpos: the Line where to display the character shape
 * @param Ypos: start column address
 * @param c: pointer to the character data
 */
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=0;
    height = DrawProp[ActiveLayer].pFont->Height;
    width = DrawProp[ActiveLayer].pFont->Width;
    offset = 8*(((width + 7)/8) - width);
    for(i = 0; i < height; i++)
    {
        pchar = ((uint8_t*)c + (width + 7)/8 * i);
        switch(((width + 7)/8))
        {
            case 1:
line = pchar[0];
break;

case 2:
    line = (pchar[0]<< 8) | pchar[1];
break;

case 3:
    default:
    line = (pchar[0]<< 16) | (pchar[1]<< 8) | pchar[2];
break;
}

for (j = 0; j < width; j++)
{
    if(line & (1 << (width- j + offset- 1)) )
    {
        BSP_LCD_DrawPixel((Xpos + j), Ypos, DrawProp[ActiveLayer].TextColor);
    }
    else
    {
        BSP_LCD_DrawPixel((Xpos + j), Ypos, DrawProp[ActiveLayer].BackColor);
    }
    Ypos++;
}

/**
 * @brief Fills buffer.
 * @param LayerIndex: layer index
 * @param pDst: output color
 * @param xSize: buffer width

static void FillBuffer(uint32_t LayerIndex, void * pDst, uint32_t xSize, uint32_t ySize, uint32_t OffLine, uint32_t ColorIndex)
{
    Dma2dHandler.Init.Mode = DMA2D_R2M;
    Dma2dHandler.Init.ColorMode = DMA2D_ARGB8888;
    Dma2dHandler.Init.OutputOffset = OffLine;

    Dma2dHandler.Instance = DMA2D;

    if(HAL_DMA2D_Init(&Dma2dHandler) == HAL_OK)
    {
        if(HAL_DMA2D_ConfigLayer(&Dma2dHandler, LayerIndex) == HAL_OK)
        {
            if (HAL_DMA2D_Start(&Dma2dHandler, ColorIndex, (uint32_t)pDst, xSize, ySize) == HAL_OK)
            {
                /* Polling For DMA transfer */
                HAL_DMA2D_PollForTransfer(&Dma2dHandler, 10);
            }
        }
    }
}
/**
 * @brief Converts Line to ARGB8888 pixel format.
 * @param pSrc: pointer to source buffer
 * @param pDst: output color
 * @param xSize: buffer width
 * @param ColorMode: input color mode
 */

static void ConvertLineToARGB8888(void *pSrc, void *pDst, uint32_t xSize, uint32_t ColorMode)
{
    /* Configure the DMA2D Mode, Color Mode and output offset */
    Dma2dHandler.Init.Mode = DMA2D_M2M_PFC;
    Dma2dHandler.Init.ColorMode = DMA2D_ARGB8888;
    Dma2dHandler.Init.OutputOffset = 0;

    /* Foreground Configuration */
    Dma2dHandler.LayerCfg[1].AlphaMode = DMA2D_NO_MODIF_ALPHA;
    Dma2dHandler.LayerCfg[1].InputAlpha = 0xFF;
    Dma2dHandler.LayerCfg[1].InputColorMode = ColorMode;
    Dma2dHandler.LayerCfg[1].InputOffset = 0;
    Dma2dHandler.Instance = DMA2D;

    /* DMA2D Initialization */
    if(HAL_DMA2D_Init(&Dma2dHandler) == HAL_OK)
    {
        if(HAL_DMA2D_ConfigLayer(&Dma2dHandler,}
if (HAL_DMA2D_Start(&Dma2dHandler, (uint32_t)pSrc, (uint32_t)pDst, xSize, 1) == HAL_OK)
{
  /* Polling For DMA transfer */
  HAL_DMA2D_PollForTransfer(&Dma2dHandler, 10);
}

/**************************** (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY SDRAM

STM32F429I DISCOVERY
## Modules

<table>
<thead>
<tr>
<th>Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY SDRAM Private Types Definitions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Private Defines</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Private Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Private Variables</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Private Function Prototypes</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Private Functions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Exported Types</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Exported Constants</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Exported Macro</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM Exported Functions</td>
</tr>
</tbody>
</table>

BSP User Manual by doxygen 1.7.6.1
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td><strong>Modules</strong></td>
<td>STM32F429I DISCOVERY TS</td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY

STM32F429I DISCOVERY
## Modules

<table>
<thead>
<tr>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY TS Private Types Definitions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Private Defines</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Private Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Private Variables</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Private Function Prototypes</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Private Functions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Exported Types</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Exported Constants</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Exported Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS Exported Functions</td>
</tr>
</tbody>
</table>

# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td><strong>Data Structures</strong></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY TS Exported Types**

**STM32F429I DISCOVERY TS**
Data Structures

struct TS_StateTypeDef

Go to the documentation of this file.

```c
/**
 * @file    stm32f429i_discovery_ts.h
 * @author  MCD Application Team
 * @version V2.1.3
 * @date    13-January-2016
 * @brief   This file contains all the functions prototypes for the
 *          stm32f429i_discovery_ts.c driver.
 * @attention
 * <h2><center>&copy; COPYRIGHT(c) 2015 STMicroelectronics</center></h2>
 * Redistribution and use in source and binary forms, with or without modification,
 * are permitted provided that the following conditions are met:
```
1. Redistributions of source code must retain the above copyright notice,
   this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions and the following disclaimer in the documentation
   and/or other materials provided with the distribution.
3. Neither the name of STMicroelectronics nor the names of its contributors
   may be used to endorse or promote products derived from this software
   without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
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.
* Define to prevent recursive inclusion  -------------------------------*/
#ifndef __STM32F429I_DISCOVERY_TS_H
#define __STM32F429I_DISCOVERY_TS_H

#ifdef __cplusplus
extern "C" {
#endif

/* Includes -----------------------------------------------*/
#include "stm32f429i_discovery.h"
/* Include TouchScreen component driver */
#include "../Components/stmpe811/stmpe811.h"

/** @addtogroup BSP
 * @{
 */

/** @addtogroup STM32F429I_DISCOVERY
 * @{
 */

/** @addtogroup STM32F429I_DISCOVERY_TS
 * @{
 */

/** @addtogroup STM32F429I_DISCOVERY_TS_Exported_Types STM32F429I DISCOVERY TS Exported Types
 * @{
 */

/** @defgroup STM32F429I_DISCOVERY_TS_Exported_Types STM32F429I DISCOVERY TS Exported Types
 * @{
 */

#ifndef __STM32F429I_DISCOVERY_TS_H
#define __STM32F429I_DISCOVERY_TS_H

#ifdef __cplusplus
extern "C" {
#endif

/* Includes -----------------------------------------------*/
#include "stm32f429i_discovery.h"
/* Include TouchScreen component driver */
#include "../Components/stmpe811/stmpe811.h"

/** @addtogroup BSP
 * @{
 */

/** @addtogroup STM32F429I_DISCOVERY
 * @{
 */

/** @addtogroup STM32F429I_DISCOVERY_TS
 * @{
 */

/** @addtogroup STM32F429I_DISCOVERY_TS_Exported_Types STM32F429I DISCOVERY TS Exported Types
 * @{
 */

/** @defgroup STM32F429I_DISCOVERY_TS_Exported_Types STM32F429I DISCOVERY TS Exported Types
 * @{
 */

*/
typedef struct {
    uint16_t TouchDetected;
    uint16_t X;
    uint16_t Y;
    uint16_t Z;
} TS_StateTypeDef;

/**
 * @}
 */

/**	@defgroup	STM32F429I_DISCOVERY_TS_Exported_Constants	STM32F429I DISCOVERY TS Exported Constants
 * @{
 */
#define TS_SWAP_NONE 0x00
#define TS_SWAP_X 0x01
#define TS_SWAP_Y 0x02
#define TS_SWAP_XY 0x04

typedef enum {
    TS_OK = 0x00,
    TS_ERROR = 0x01,
    TS_TIMEOUT = 0x02
} TS_StatusTypeDef;

/**
 * @}
 */

/**	@defgroup	STM32F429I_DISCOVERY_TS_Exported_Macros	STM32F429I DISCOVERY TS Exported Macros
 * @{
 */

/**
 * @}
/**	@defgroup STM32F429I_DISCOVERY_TS_Exported_Functions STM32F429I DISCOVERY TS Exported Functions
*/

/*@{
*/

uint8_t BSP_TS_Init(uint16_t XSize, uint16_t YSize);
void BSP_TS_GetState(TS_StateTypeDef *TsState);
uint8_t BSP_TS_ITConfig(void);
uint8_t BSP_TS_ITGetStatus(void);
void BSP_TS_ITClear(void);

/*@*/

#ifdef __cplusplus
}
#endif
#endif /*__STM32F429I_DISCOVERY_TS_H*/
stm32f429i_discovery_ts.c

Go to the documentation of this file.

00001 /**
00002 ******************************************
00003 * @file stm32f429i_discovery_ts.c
00004 * @author MCD Application Team
00005 * @version V2.1.3
00006 * @date 13-January-2016
00007 * @brief This file provides a set of functions needed to manage Touch screen available with STMPE811 IO Expander device mounted on
00008 * STM32F429I-Discovery Kit.
00009 ******************************************
00010 ******************************************
00011 * @attention
00012 *
00013  *
00014  *
00015  * Redistributions and use in source and binary forms, with or without modification,
00016  * are permitted provided that the following
If conditions are met:

1. Redistributions of source code must retain the above copyright notice,
2. Redistributions in binary form must reproduce the above copyright notice,
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,
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.

#include "stm32f429i_discovery_ts.h"
#include "stm32f429i_discovery_io.h"

/** @addtogroup BSP *
@{ */
/** @addtogroup STM32F429I_DISCOVERY *
@{ */
/** @defgroup STM32F429I_DISCOVERY_TS STM32F429I_DISCOVERY TS *
@{ */
/** @defgroup STM32F429I_DISCOVERY_TS_Private_Types_Definitions STM32F429I_DISCOVERY TS Private Types Definitions *
@{ */
/** @defgroup STM32F429I_DISCOVERY_TS_Private_Defines STM32F429I_DISCOVERY TS Private Defines *
@{ */
/**
 * @defgroup STM32F429I_DISCOVERY_TS_Private_Macros STM32F429I DISCOVERY TS Private Macros
 *
 */

/**
 * @defgroup STM32F429I_DISCOVERY_TS_Private_Variables STM32F429I DISCOVERY TS Private Variables
 *
 */

static TS_DrvTypeDef *TsDrv;
static uint16_t TsXBoundary, TsYBoundary;

/**
 * @defgroup STM32F429I_DISCOVERY_TS_Private_Function_Prototypes STM32F429I DISCOVERY TS Private Function Prototypes
 *
 */

/**
 * @defgroup STM32F429I_DISCOVERY_TS_Private_Functions STM32F429I DISCOVERY TS Private Functions
 *
 */

```c
static
```
/**
 * @brief Initializes and configures the touchscreen functionalities and configures all necessary hardware resources (GPIOs, clocks...).
 * @param XSize: The maximum X size of the TS area on LCD
 * @param YSize: The maximum Y size of the TS area on LCD
 * @retval TS_OK: if all initializations are OK. Other value if error.
 */

uint8_t BSP_TS_Init(uint16_t XSize, uint16_t YSize)
{
    uint8_t ret = TS_ERROR;
    TsXBoundary = XSize;
    TsYBoundary = YSize;
    if(ret == TS_OK)
    {
        /* Read ID and verify if the IO expander is ready */
        if(stmpe811_ts_drv.ReadID(TS_I2C_ADDRESS) == STMPE811_ID)
        {
            /* Initialize the TS driver structure */
            TsDrv = &stmpe811_ts_drv;
            ret = TS_OK;
        }
    }
    if(ret == TS_OK)
    {
        /* Initialize x and y positions boundaries */
        TsXBoundary = XSize;
        TsYBoundary = YSize;
    }
}
/* Initialize the LL TS Driver */
TsDrv->Init(TS_I2C_ADDRESS);
TsDrv->Start(TS_I2C_ADDRESS);
}
return ret;
}
/**
 * @brief Configures and enables the touch screen interrupts.
 * @retval TS_OK: if ITconfig is OK. Other value if error.
 */
uint8_t BSP_TS_ITConfig(void)
{
    /* Enable the TS ITs */
    TsDrv->EnableIT(TS_I2C_ADDRESS);
    return TS_OK;
}
/**
 * @brief Gets the TS IT status.
 * @retval Interrupt status.
 */
uint8_t BSP_TS_ITGetStatus(void)
{
    /* Return the TS IT status */
    return (TsDrv->GetITStatus(TS_I2C_ADDRESS));
}
/**
 * @brief Returns status and positions of the touch screen.
 * @param TsState: Pointer to touch screen

```c
/*
  current state structure
  */
void BSP_TS_GetState(TS_StateTypeDef* TsState)
{
  static uint32_t _x = 0, _y = 0;
  uint16_t xDiff, yDiff, x, y, xr, yr;
  TsState->TouchDetected = TsDrv->DetectTouch(TS_I2C_ADDRESS);

  if(TsState->TouchDetected)
  {
    TsDrv->GetXY(TS_I2C_ADDRESS, &x, &y);

    /* Y value first correction */
    y -= 360;

    /* Y value second correction */
    yr = y / 11;

    /* Return y position value */
    if(yr <= 0)
    {
      yr = 0;
    }
    else if (yr > TsYBoundary)
    {
      yr = TsYBoundary - 1;
    }
    else
    {
      y = yr;
    }

    /* X value first correction */
    if(x <= 3000)
    {
```
x = 3870 - x;
else
    x = 3800 - x;
/* X value second correction */
xr = x / 15;
/* Return X position value */
if (xr <= 0)
    { xr = 0; }
else if (xr > TsXBoundary)
    { xr = TsXBoundary - 1; }
else
    {} x = xr;
xDiff = x > _x? (x - _x): (_x - x);
yDiff = y > _y? (y - _y): (_y - y);
if (xDiff + yDiff > 5)
    { _x = x;
      _y = y; }
/* Update the X position */
TsState->X = _x;
/* Update the Y position */
TsState->Y = _y;
/**
 * @brief Clears all touch screen interrupts.
 */

void BSP_TS_ITClear(void)
{
    /* Clear TS IT pending bits */
    TsDrv->ClearIT(TS_I2C_ADDRESS);
}

/**
 * @}
 */

/**
 * @}
 */

/**
 * @}
 */

/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
### STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY EEPROM**

**STM32F429I DISCOVERY**
## Modules

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY EEPROM Exported Types</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY EEPROM Exported Constants</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY EEPROM Exported Macros</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY EEPROM Exported Functions</td>
<td></td>
</tr>
</tbody>
</table>

BSP User Manual by [doxygen](http://www.doxygen.org) 1.7.6.1
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Drivers</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Drivers Directory Reference
### Directories

<table>
<thead>
<tr>
<th>directory</th>
<th>BSP</th>
</tr>
</thead>
</table>

# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Drivers</strong></td>
<td><strong>BSP</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## BSP Directory Reference
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drivers</td>
<td>BSP</td>
<td>STM32F429I-Discovery</td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I-Discovery Directory Reference
### Files

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>file</strong></td>
<td><strong>stm32f429i_discovery.c</strong> [code]</td>
</tr>
<tr>
<td>This file provides set of firmware functions to manage Leds and push-button available on STM32F429I-Discovery Kit from STMicroelectronics.</td>
<td></td>
</tr>
<tr>
<td><strong>file</strong></td>
<td><strong>stm32f429i_discovery.h</strong> [code]</td>
</tr>
<tr>
<td>This file contains definitions for STM32F429I-Discovery Kit LEDs, push-buttons hardware resources.</td>
<td></td>
</tr>
<tr>
<td><strong>file</strong></td>
<td><strong>stm32f429i_discovery_eeprom.c</strong> [code]</td>
</tr>
<tr>
<td>This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory. To be able to use this driver, the switch EE_M24LR64 must be defined in your toolchain compiler preprocessor.</td>
<td></td>
</tr>
<tr>
<td><strong>file</strong></td>
<td><strong>stm32f429i_discovery_eeprom.h</strong> [code]</td>
</tr>
<tr>
<td>This file contains all the functions prototypes for the <strong>stm32f429i_discovery_eeprom.c</strong> firmware driver.</td>
<td></td>
</tr>
<tr>
<td><strong>file</strong></td>
<td><strong>stm32f429i_discovery_gyroscope.c</strong> [code]</td>
</tr>
<tr>
<td>This file provides a set of functions needed to manage the MEMS gyroscope available on STM32F429I-Discovery Kit.</td>
<td></td>
</tr>
<tr>
<td><strong>file</strong></td>
<td><strong>stm32f429i_discovery_gyroscope.h</strong> [code]</td>
</tr>
</tbody>
</table>
This file contains definitions for `stm32f429i_discovery_gyroscope.c` firmware driver.

**file** `stm32f429i_discovery_io.c` [code]

This file provides a set of functions needed to manage the STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit.

**file** `stm32f429i_discovery_io.h` [code]

This file contains all the functions prototypes for the `stm32f429i_discovery_io.c` driver.

**file** `stm32f429i_discovery_lcd.c` [code]

This file includes the LCD driver for ILI9341 Liquid Crystal Display Modules of STM32F429I-Discovery kit (MB1075).

**file** `stm32f429i_discovery_lcd.h` [code]

This file contains all the functions prototypes for the `stm32f429i_discovery_lcd.c` driver.

**file** `stm32f429i_discovery_sdram.c` [code]

This file provides a set of functions needed to drive the IS42S16400J SDRAM memory mounted on STM32F429I-Discovery Kit.

**file** `stm32f429i_discovery_sdram.h` [code]
This file contains all the functions prototypes for the 
stm32f429i_discovery_sdram.c driver.

file  stm32f429i_discovery_ts.c [code]

This file provides a set of functions needed to manage Touch screen available with STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit.

file  stm32f429i_discovery_ts.h [code]

This file contains all the functions prototypes for the stm32f429i_discovery_ts.c driver.
stm32f429i_discovery_eeprom.h

Go to the documentation of this file.

```c
/**
 * @file    stm32f429i_discovery_eeprom.h
 * @author  MCD Application Team
 * @version V2.1.3
 * @date    13-January-2016
 * @brief   This file contains all the function prototypes for
 *          the stm32f429i_discovery_eeprom.c firmware driver.
 */

* @attention

* <h2><center>&copy; COPYRIGHT(c) 2015 STMicroelectronics</center></h2>

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

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
 CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
/* Define to prevent recursive inclusion --------------------------------*/
#ifndef __STM32F429I_DISCOVERY_EEPROM_H
#define __STM32F429I_DISCOVERY_EEPROM_H
#endif

/* Includes ----------------------------------*/
#include "stm32f429i_discovery.h"

/** @addtogroup BSP */
/*@{*/
/** @addtogroup STM32F429I_DISCOVERY */
/*@{*/
/** @addtogroup STM32F429I_DISCOVERY_EEPROM */
STM32F429I_DISCOVERY_EEPROM
/** @} */

/** @defgroup STM32F429I_DISCOVERY_EEPROM_Exported_Types */
@{*/
/** @} */
/** @} */
/** @} */
/** @} */
/**
 *@defgroup STM32F429I_DISCOVERY_EEPROM_Exported_Constants STM32F429I DISCOVERY EEPROM Exported Constants
 *
 */

/* EEPROM hardware address and page size */
#define EEPROM_PAGESIZE  4
#define EEPROM_MAX_SIZE 0x2000 / * 64Kbit*/

/* Maximum Timeout values for flags and events waiting loops.
This timeout is based on systick set to 1ms*/

/* Timeout for read based if read all the EEPROM : EEPROM_MAX_SIZE * BSP_I2C_SPEED (640ms)*/
#define EEPROM_READ_TIMEOUT ((uint32_t)(1000))

/* Timeout for write based on max write which is EEPROM_PAGESIZE bytes: EEPROM_PAGESIZE * BSP_I2C_SPEED (320us)*/
#define EEPROM_WRITE_TIMEOUT ((uint32_t)(1))

/* Maximum number of trials for EEPROM_WaitEepromStandbyState() function */
#define EEPROM_MAX_TRIALS 300

#define EEPROM_OK 0
#define EEPROM_FAIL 1
#define EEPROM_TIMEOUT 2

/**
 */
*/
/** @defgroup STM32F429I_DISCOVERY_EEPROM_Exported_Macros STM32F429I DISCOVERY EEPROM Exported Macros */
* @{
*/

/** @defgroup STM32F429I_DISCOVERY_EEPROM_Exported_Functions STM32F429I DISCOVERY EEPROM Exported Functions */
* @{
*/

uint32_t BSP_EEPROM_Init(void);
uint32_t BSP_EEPROM_ReadBuffer(uint8_t *pBuffer, uint16_t ReadAddr, uint16_t *NumByteToRead);
uint32_t BSP_EEPROM_WritePage(uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite);
uint32_t BSP_EEPROM_WriteBuffer(uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite);
uint32_t BSP_EEPROM_WaitEepromStandbyState(void);

/* USER Callbacks: This function is declared as __weak in EEPROM driver and should be implemented into user application. */
void BSP_EEPROM_TIMEOUT_UserCallback(void);

/* USER Callbacks: This function is declared as __weak in EEPROM driver and should be implemented into user application. */
BSP_EEPROM_TIMEOUT_UserCallback();
00115 00116 00117 /* Link function for I2C EEPROM peripheral */

00118 void EEPROM_IO_Init(void);
00119 HAL_StatusTypeDef EEPROM_IO_WriteData(uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize);
00120 HAL_StatusTypeDef EEPROM_IO_ReadData(uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize);
00121 HAL_StatusTypeDef EEPROM_IO_IsDeviceReady(uint16_t DevAddress, uint32_t Trials);
00122
00123 #ifdef __cplusplus
00124 }
00125 #endif
00126
00127 #endif /* __STM32F429I_DISCOVERY_EEPROM_H */
00128
00129 /*
00130 * @}
00131 */
00132
00133 /*
00134 * @}
00135 */
00136
00137 /*
00138 * @}
00139 */
00140
00141 /*
00142 * @}
00143 */
00144
00145 /************************************** (C) COPYRIGHT STMi
/**
 * @file        stm32f429i_discovery_eeprom.c
 * @author      MCD Application Team
 * @version     V2.1.3
 * @date        13-January-2016
 * @brief       This file provides a set of functions needed to manage an I2C M24LR64 EEPROM memory.
 *              To be able to use this driver, the switch EE_M24LR64 must be defined in your toolchain compiler prep
 *              processor
 *              
 * Notes:
 * - This driver is intended for STM32F4xx families devices only.
 * - The I2C EEPROM memory (M24LR64) is available on separate daughter
board ANT7-M24LR-A, which is not provided with the STM32F429I DISCOVERY board.

To use this driver you have to connect the ANT7-M24LR-A to CN3 connector of STM32F429I DISCOVERY board.

It implements a high level communication layer for read and write from/to this memory. The needed STM32F4xx hardware resources (I2C and GPIO) are defined in stm32f429i_discovery.h file, and the initialization is performed in EEPROM_IO_Init() function declared in stm32f429i_discovery.c file.

You can easily tailor this driver to any other development board, by just adapting the defines for hardware resources and EEPROM_IO_Init() function.

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

@note Regarding BSP_EEPROM_WritePage(), it is an optimized function to perform small write (less than 1 page) BUT 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).

<table>
<thead>
<tr>
<th>Pin assignment for M24LR64 EEPROM</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F4xx I2C Pins</td>
<td>Pin</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>E0(GND)</td>
<td>1 (0V)</td>
</tr>
<tr>
<td>AC0</td>
<td>2</td>
</tr>
<tr>
<td>AC1</td>
<td>3</td>
</tr>
<tr>
<td>VSS</td>
<td>4 (0V)</td>
</tr>
<tr>
<td>SDA</td>
<td>5</td>
</tr>
<tr>
<td>SCL</td>
<td>6</td>
</tr>
<tr>
<td>E1(GND)</td>
<td>7 (0V)</td>
</tr>
</tbody>
</table>
\begin{verbatim}
  * | . 
  | VDD | 8 (3.3V) |
  *--------------------------------------------------------------------------
  *                                          
  *--------------------------------------------------------------------------
  *                                          
  *--------------------------------------------------------------------------
  *                                          
  *--------------------------------------------------------------------------

  * @attention

  * <h2><center>&copy; COPYRIGHT(c) 2015 STMicroelectronics</center></h2>

  *

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

  * 1. Redistributions of source code must retain the above copyright notice,

  * 2. Redistributions in binary form must reproduce the above copyright notice,

  * 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.

  * 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.

  * 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

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

******************************************
******************************************
/* Includes --------------------------------
----------------------------------*/
#include "stm32f429i_discovery_eeprom.h"

#ifdef EE_M24LR64
/** @addtogroup BSP *
 */

/** @addtogroup STM32F429I_DISCOVERY *
 */

/** @addtogroup STM32F429I_DISCOVERY_EEPROM STM32F429I_DISCOVERY_EEPROM *
 */

/** @brief This file includes the I2C E
EPROM driver of STM32F429I Discovery Kit.

```c
00103   * @{
00104   */
00105
00106   /** @defgroup STM32F429I_DISCOVERY_EEPROM_Private_Types STM32F429I DISCOVERY EEPROM Private Types
00107   * @{
00108   */
00109   /**
00110   * @}
00111   */
00112
00113   /** @defgroup STM32F429I_DISCOVERY_EEPROM_Private_Defines STM32F429I DISCOVERY EEPROM Private Defines
00114   * @{
00115   */
00116   /**
00117   * @}
00118   */
00119
00120   /** @defgroup STM32F429I_DISCOVERY_EEPROM_Private_Macros STM32F429I DISCOVERY EEPROM Private Macros
00121   * @{
00122   */
00123   /**
00124   * @}
00125   */
00126
00127   /** @defgroup STM32F429I_DISCOVERY_EEPROM_Private_Variables STM32F429I DISCOVERY EEPROM Private Variables
00128   * @{
00129   */
00130   __IO uint16_t EEPROMAddress = 0;
```
__IO uint32_t EEPROMTimeout = EEPROM_READ_TIMEOUT;
__IO uint16_t EEPROMDataRead;
__IO uint8_t EEPROMDataWrite;

/** @}
 */

/**	@defgroup STM32F429I_DISCOVERY_EEPROM_Private_Function_Prototypes STM32F429I DISCOVERY EEPROM Private Function Prototypes
* @}{
 */

/**	@defgroup STM32F429I_DISCOVERY_EEPROM_Private_Functions STM32F429I DISCOVERY EEPROM Private Functions
* @}{
 */

/**	@brief Initializes peripherals used by the I2C EEPROM driver.
* @note There are 2 different versions of M24LR64 (A01 & A02).
* Then try to connect on 1st one (EEPROM_I2C_ADDRESS_A01) and if problem, check the 2nd one (EEPROM_I2C_ADDRESS_A02)
* @retval EEPROM_OK (0) if operation is correctly performed, else return value different from EEPROM_OK (0)
 */
uint32_t BSP_EEPROM_Init(void)
{
    /* I2C Initialization */
    EEPROM_IO_Init();

    /*Select the EEPROM address for A01 and check if OK*/
    EEPROMAddress = EEPROM_I2C_ADDRESS_A01;
    if (EEPROM_IO_IsDeviceReady(EEPROMAddress, EEPROM_MAX_TRIALS) != HAL_OK)
    {
        /*Select the EEPROM address for A02 and check if OK*/
        EEPROMAddress = EEPROM_I2C_ADDRESS_A02;
        if (EEPROM_IO_IsDeviceReady(EEPROMAddress, EEPROM_MAX_TRIALS) != HAL_OK)
        {
            return EEPROM_FAIL;
        }
    }
    return EEPROM_OK;
}

/**
 * @brief Reads a block of data from the EEPROM.
 * @param pBuffer : pointer to the buffer that receives the data read from
 * @param ReadAddr : EEPROM's internal address to start reading from.
 * @param 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.

* @retval EEPROM_OK (0) if operation is correctly performed, else return value different from EEPROM_OK (0) or the timeout user callback.

*/

uint32_t BSP_EEPROM_ReadBuffer(uint8_t * pBuffer, uint16_t ReadAddr, uint16_t * NumByteToRead)
{
    uint32_t buffersize = *NumByteToRead;

    /* Set the pointer to the Number of data to be read. This pointer will be used by the DMA Transfer Completer interrupt Handler in order to reset the variable to 0. User should check on this variable in order to know if the DMA transfer has been complete or not. */
    EEPROMDataRead = *NumByteToRead;

    if (EEPROM_IO_ReadData(EEPROMAddress, ReadAddr, pBuffer, buffersize) != HAL_OK)
    {
        return EEPROM_FAIL;
    }

    /* Wait transfer through DMA to be complete */
    EEPROMTimeout = HAL_GetTick() + EEPROM_READ_TIMEOUT;

    while (EEPROMDataRead > 0)
    {
        if(HAL_GetTick() > EEPROMTimeout)


```c
00212  {
00213    BSP_EEPROM_TIMEOUT_UserCallback();
00214    return EEPROM_TIMEOUT;
00215  }
00216 }
00217 00218  /* If all operations OK, return EEPROM_OK (0) */
00219  return EEPROM_OK;
00220 }
00221 00222 /**
00223   * @brief  Writes more than one byte to the EEPROM with a single WRITE cycle.
00224   *
00225   * @note   The number of bytes (combined to write start address) must not
00226   *         cross the EEPROM page boundary. This function can only write into
00227   *         the boundaries of an EEPROM page.
00228   *         This function doesn't check on boundaries condition (in this driver
00229   *         the function BSP_EEPROM_WriteBuffer() which calls BSP_EEPROM_WritePage() is
00230   *         responsible of checking on Page boundaries).
00231   *
00232   * @param  pBuffer : pointer to the buffer containing the data to be written to
00233   *         the EEPROM.
00234   * @param  WriteAddr : EEPROM's internal address to write to.
00235   * @param  NumByteToWrite : pointer to the variable holding number of bytes to
00236   *         be written into the EEPROM.
00237   */
```
@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 to know when the transfer is complete.

@note  This function just configure the communication and enable the DMA channel to transfer data. Meanwhile, the user application may perform other tasks in parallel.

@retval EEPROM_OK (0) if operation is correctly performed, else return value different from EEPROM_OK (0) or the timeout user callback.

```c
uint32_t BSP_EEPROM_WritePage(uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite) {
  uint32_t buffersize = *NumByteToWrite;
  uint32_t status = EEPROM_OK;
  /* Set the pointer to the Number of data to be written. This pointer will be used by the DMA Transfer Completer interrupt Handler in order to reset the variable to 0. User should check on this variable in order to know if the DMA transfer has been complete or not. */
  EEPROMDataWrite = *NumByteToWrite;
  if (EEPROM_IO_WriteData(EEPROMAddress, WriteAddr, pBuffer, buffersize) != HAL_OK) {
    status = EEPROM_FAIL;
  }
}```
00262 }  
00263  
00264 /* Wait transfer through DMA to be complete */  
00265 EEPROMTimeout = HAL_GetTick() + EEPROM_WRITE_TIMEOUT;  
00266 while (EEPROMDataWrite > 0)  
00267 {  
00268 if(HAL_GetTick() > EEPROMTimeout)  
00269 {  
00270    BSP_EEPROM_TIMEOUT_UserCallback();  
00271    return EEPROM_TIMEOUT;  
00272  }  
00273 }  
00274  
00275 if (BSP_EEPROM_WaitEepromStandbyState() != EEPROM_OK)  
00276 {  
00277    return EEPROM_FAIL;  
00278 }  
00279  
00280 /* If all operations OK, return EEPROM_OK (0) */  
00281 return status;  
00282 }  
00283  
00284 /**  
00285 * @brief Writes buffer of data to the I2C EEPROM.  
00286 * @param pBuffer : pointer to the buffer containing the data to be written  
00287 * @param WriteAddr : EEPROM's internal address to write to.  
00288 * @param NumByteToWrite : number of bytes to write to the EEPROM.  
00289 * @retval EEPROM_OK (0) if operation is co
correctly performed, else return value different from EEPROM_OK (0) or the timeout user callback.

uint32_t BSP_EEPROM_WriteBuffer(uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite)
{
  uint16_t numofpage = 0, numofsing = 0, count = 0;
  uint16_t addr = 0;
  uint8_t dataindex = 0;
  uint32_t status = EEPROM_OK;

  addr = WriteAddr % EEPROM_PAGESIZE;
  count = EEPROM_PAGESIZE - addr;
  numofpage = NumByteToWrite / EEPROM_PAGESIZE;
  numofsing = NumByteToWrite % EEPROM_PAGESIZE;

  /* If WriteAddr is EEPROM_PAGESIZE aligned */
  if (addr == 0)
  {
    /* If NumByteToWrite < EEPROM_PAGESIZE */
    if (numofpage == 0)
    {
      /* Store the number of data to be written */
      dataindex = numofsing;
      /* Start writing data */
      status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
      if (status != EEPROM_OK)
      {
        return status;
      }
    }
  }
  return status;
If `NumByteToWrite > EEPROM_PAGESIZE` */

```c
else {
    while(numofpage--)
    {
        /* Store the number of data to be written */
        dataindex = EEPROM_PAGESIZE;
        status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
        if (status != EEPROM_OK)
            return status;
    }
    WriteAddr += EEPROM_PAGESIZE;
    pBuffer += EEPROM_PAGESIZE;
}
if(numofsingle!=0)
{
    /* Store the number of data to be written */
    dataindex = numofsingle;
    status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
    if (status != EEPROM_OK)
        return status;
}
```
gned */
00350    else
00351    {
00352        /* If NumByteToWrite < EEPROM_PAGESIZE */
00353
00354        if (numofpage == 0)
00355            {
00356                /* If the number of data to be written is more than the remaining space in the current page: */
00357                if (NumByteToWrite > count)
00358                    {
00359                        /* Store the number of data to be written */
00360                        dataindex = count;
00361                        /* Write the data contained in same page */
00362                        status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
00363                        if (status != EEPROM_OK)
00364                            {
00365                                return status;
00366                            }
00367                    }
00368                    /* Store the number of data to be written */
00369                    dataindex = (NumByteToWrite - count);
00370                    /* Write the remaining data in the following page */
00371                    status = BSP_EEPROM_WritePage((uint8_t*)(pBuffer + count), (WriteAddr + count), (uint8_t*)(&dataindex));
00372                    if (status != EEPROM_OK)
00373                        {
00374                            return status;
00375                        }
00376                }
/* Store the number of data to be written */
dataindex = numofsingle;
status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
if (status != EEPROM_OK)
{
    return status;
}

} /* If NumByteToWrite > EEPROM_PAGESIZE */

else
{
    NumByteToWrite -= count;
    numofpage = NumByteToWrite / EEPROM_PAGESIZE;
    numofsingle = NumByteToWrite % EEPROM_PAGESIZE;

    if(count != 0)
    {
        /* Store the number of data to be written */
dataindex = count;
status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
if (status != EEPROM_OK)
{
    return status;
}
WriteAddr += count;
pBuffer += count;
while(numofpage--)
{
    /* Store the number of data to be written */
    dataindex = EEPROM_PAGESIZE;

    status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
    if (status != EEPROM_OK)
        return status;

    WriteAddr += EEPROM_PAGESIZE;
    pBuffer += EEPROM_PAGESIZE;
}

if(numofsingle != 0)
{
    /* Store the number of data to be written */
    dataindex = numofsingle;
    status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
    if (status != EEPROM_OK)
        return status;
}

/* If all operations OK, return EEPROM_OK (0) */
return EEPROM_OK;
/**
 * @brief 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.
 *
 * @retval EEPROM_OK (0) if operation is correctly performed, else return value different from EEPROM_OK (0) or the timeout user callback.
 */

uint32_t BSP_EEPROM_WaitEepromStandbyState(void)
{
    /* Check if the maximum allowed number of trials has been reached */
    if (EEPROM_IO_IsDeviceReady(EEPROMAddress, EEPROM_MAX_TRIALS) != HAL_OK)
    {
        /* If the maximum number of trials has been reached, exit the function */
        BSP_EEPROM_TIMEOUT_UserCallback();
        return EEPROM_TIMEOUT;
    }
    return EEPROM_OK;
}
/**
 * @brief Memory Tx Transfer completed callbacks.
 * @param hi2c: I2C handle
 */
void HAL_I2C_MemTxCpltCallback(I2C_HandleTypeDef *hi2c)
{
    EEPROMDataWrite = 0;
}

/**
 * @brief Memory Rx Transfer completed callbacks.
 * @param hi2c: I2C handle
 */
void HAL_I2C_MemRxCpltCallback(I2C_HandleTypeDef *hi2c)
{
    EEPROMDataRead = 0;
}

/**
 * @brief Basic management of the timeout situation.
 */
__weak void BSP_EEPROM_TIMEOUT_UserCallback(void)
{
}

#error EE_M24LR64 /*
/**
 * @}
 */

/**
 * @}
 */

/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
### STM32F429I DISCOVERY EEPROM Exported Functions

STM32F429I DISCOVERY EEPROM
### Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint32_t</td>
<td><strong>BSP_EEPROM_Init</strong></td>
<td>(void)</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_EEPROM_ReadBuffer</strong></td>
<td>(uint8_t *pBuffer, uint16_t ReadAddr, uint16_t *NumByteToRead)</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_EEPROM_WritePage</strong></td>
<td>(uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite)</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_EEPROM_WriteBuffer</strong></td>
<td>(uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite)</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_EEPROM_WaitEepromStandbyState</strong></td>
<td>(void)</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_EEPROM_TIMEOUT_UserCallback</strong></td>
<td>(void)</td>
</tr>
<tr>
<td>void</td>
<td><strong>EEPROM_IO_Init</strong></td>
<td>(void)</td>
</tr>
<tr>
<td>HAL_StatusTypeDef</td>
<td><strong>EEPROM_IO_WriteData</strong></td>
<td>(uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize)</td>
</tr>
<tr>
<td>HAL_StatusTypeDef</td>
<td><strong>EEPROM_IO_ReadData</strong></td>
<td>(uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize)</td>
</tr>
<tr>
<td>HAL_StatusTypeDef</td>
<td><strong>EEPROM_IO_IsDeviceReady</strong></td>
<td>(uint16_t DevAddress, uint32_t Trials)</td>
</tr>
</tbody>
</table>
Function Documentation

uint32_t BSP_EEPROM_Init ( void )

uint32_t BSP_EEPROM_ReadBuffer ( uint8_t * pBuffer, uint16_t ReadAddr, uint16_t * NumByteToRead )

void BSP_EEPROM_TIMEOUT_UserCallback ( void )

uint32_t BSP_EEPROM_WaitEepromStandbyState ( void )

uint32_t BSP_EEPROM_WriteBuffer ( uint8_t * pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite )

uint32_t BSP_EEPROM_WritePage ( uint8_t * pBuffer, uint16_t WriteAddr, uint8_t * NumByteToWrite )

void EEPROM_IO_Init ( void )

HAL_StatusTypeDef EEPROM_IO_IsDeviceReady ( uint16_t DevAddr, uint32_t Trials )

HAL_StatusTypeDef EEPROM_IO_ReadData ( uint16_t DevAddr, ... )
HAL_StatusTypeDef EEPROM_IO_WriteData (uint16_t DevAddress,
uint16_t MemAddress,
uint8_t * pBuffer,
uint32_t BufferSize)
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY SDRAM Exported Constants**

**STM32F429I DISCOVERY SDRAM**
**Defines**

```c
#define SDRAMDEVICEADDR ((uint32_t)0xD0000000)  
FMC SDRAM Bank address.
#define SDRAM_DEVICESIZE ((uint32_t)0x800000)  
/* SDRAM device size in MBytes */
#define SDRAMMEMORYWIDTH FMC_SDRAM_MEM_BUS_WIDTH_16  
FMC SDRAM Memory Width.
#define SDRAMCASLATENCY FMC_SDRAM_CAS_LATENCY_3  
FMC SDRAM CAS Latency.
#define SDCLOCK_PERIOD FMC_SDRAM_CLOCK_PERIOD_2  
/* Default configuration used with LCD */
FMC SDRAM Memory clock period.
#define SDRAMREADBURST FMC_SDRAM_RBURST_DISABLE  
/* Default configuration used with LCD */
FMC SDRAM Memory Read Burst feature.
#define REFRESH_COUNT ((uint32_t)1386)  
/* SDRAM refresh count */
FMC SDRAM Bank Remap.
#define SDRAM_TIMEOUT ((uint32_t)0xFFFF)
#define __DMAx_CLK_ENABLE __DMA2_CLK_ENABLE
#define SDRAM_DMAx_CHANNEL DMA_CHANNEL_0
#define SDRAM_DMAx_STREAM DMA2_Stream0
#define SDRAM_DMAx_IRQn DMA2_Stream0_IRQn
#define SDRAM_DMAx_IRQHandler DMA2_Stream0_IRQHandler
#define SDRAM_MODEREG_BURST_LENGTH_1 ((uint16_t)0x0000)  
FMC SDRAM Mode definition register defines.
#define SDRAM_MODEREG_BURST_LENGTH_2 ((uint16_t)0x0001)
#define SDRAM_MODEREG_BURST_LENGTH_4 ((uint16_t)0x0002)
#define SDRAM_MODEREG_BURST_LENGTH_8 ((uint16_t)0x0004)
#define SDRAM_MODEREG_BURST_TYPE_SEQUENTIAL ((uint16_t)0x0000)
#define SDRAM_MODEREG_BURST_TYPE_INTERLEAVED ((uint16_t)0x0008)
#define SDRAM_MODEREG_CAS_LATENCY_2 ((uint16_t)0x0020)
#define SDRAM_MODEREG_CAS_LATENCY_3 ((uint16_t)0x0030)
#define SDRAM_MODEREG_OPERATING_MODE_STANDARD ((uint16_t)0x0000)
```


#define SDRAM_MODEREG_WRITEBURST_MODE_PROGRAMMED
#define SDRAM_MODEREG_WRITEBURST_MODE_SINGLE ((uint16_t)0x0200)
Define Documentation

```c
#define __DMAx_CLK_ENABLE __DMA2_CLK_ENABLE
```
Definition at line 113 of file `stm32f429i_discovery_sdram.h`.
Referenced by `MspInit()`.

```c
#define REFRESH_COUNT ((uint32_t)1386) /* SDRAM refresh count */
```
FMC SDRAM Bank Remap.
Definition at line 109 of file `stm32f429i_discovery_sdram.h`.
Referenced by `BSP_SDRAM_Init()`.

```c
#define SDCLOCK_PERIOD FMC_SDRAM_CLOCK_PERIOD_2 /* FMC SDRAM Memory clock period. */
```
Definition at line 93 of file `stm32f429i_discovery_sdram.h`.
Referenced by `BSP_SDRAM_Init()`.

```c
#define SDRAM_CAS_LATENCY FMC_SDRAM_CAS_LATENCY_3 /* FMC SDRAM CAS Latency. */
```
Definition at line 88 of file `stm32f429i_discovery_sdram.h`.
Referenced by `BSP_SDRAM_Init()`.

```c
#define SDRAM_DEVICE_ADDR ((uint32_t)0xD0000000)
```

FMC SDRAM Bank address.
Definition at line 75 of file `stm32f429i_discovery_sdram.h`.

```
#define SDRAM_DEVICE_SIZE ((uint32_t)0x800000) /* SDRAM device size in MBytes */
```
Definition at line 76 of file `stm32f429i_discovery_sdram.h`.

Referenced by `MspInit()`.

```
#define SDRAM_DMAx_CHANNEL DMA_CHANNEL_0
```
Definition at line 114 of file `stm32f429i_discovery_sdram.h`.

Referenced by `MspInit()`.

```
#define SDRAM_DMAx_IRQHandler DMA2_Stream0_IRQHandler
```
Definition at line 117 of file `stm32f429i_discovery_sdram.h`.

Referenced by `MspInit()`.

```
#define SDRAM_DMAx_IRQn DMA2_Stream0_IRQn
```
Definition at line 116 of file `stm32f429i_discovery_sdram.h`.

Referenced by `MspInit()`.

```
#define SDRAM_DMAx_STREAM DMA2_Stream0
```
Definition at line 115 of file `stm32f429i_discovery_sdram.h`.

Referenced by `MspInit()`.
#define SDRAM_MEMORY_WIDTH FMC_SDRAM_MEM_BUS_WIDTH

FMC SDRAM Memory Width.
Definition at line 82 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Init().

#define SDRAM_MODEREG_BURST_LENGTH_1 ((uint16_t)0x0000)

FMC SDRAM Mode definition register defines.
Definition at line 122 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_BURST_LENGTH_2 ((uint16_t)0x0001)

Definition at line 123 of file stm32f429i_discovery_sdram.h.

#define SDRAM_MODEREG_BURST_LENGTH_4 ((uint16_t)0x0002)

Definition at line 124 of file stm32f429i_discovery_sdram.h.

#define SDRAM_MODEREG_BURST_LENGTH_8 ((uint16_t)0x0004)

Definition at line 125 of file stm32f429i_discovery_sdram.h.

#define SDRAM_MODEREG_BURST_TYPE_INTERLEAVED ((uint16_t)0x0008)

Definition at line 127 of file stm32f429i_discovery_sdram.h.
```c
#define SDRAM_MODEREG_BURST_TYPE_SEQUENTIAL  ((uint16_t)0x0000)

Definition at line 126 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_CAS_LATENCY_2  ((uint16_t)0x0020)

Definition at line 128 of file stm32f429i_discovery_sdram.h.

#define SDRAM_MODEREG_CAS_LATENCY_3  ((uint16_t)0x0030)

Definition at line 129 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_OPERATING_MODE_STANDARD  ((uint16_t)0x0000)

Definition at line 130 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_WRITEBURST_MODE_PROGRAMMED

Definition at line 131 of file stm32f429i_discovery_sdram.h.

#define SDRAM_MODEREG_WRITEBURST_MODE_SINGLE  ((uint16_t)0x0200)

Definition at line 132 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Initialization_sequence().
```
#define SDRAM_READBURST FMC_SDRAM_RBURST_DISABLE

FMC SDRAM Memory Read Burst feature.

Definition at line 99 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Init().

#define SDRAM_TIMEOUT ((uint32_t)0xFFFF)

Definition at line 110 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Initialization_sequence(), and BSP_SDRAM_Sendcmd().

BSP User Manual by doxygen 1.7.6.1
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td>STM32F429I DISCOVERY LOW LEVEL Private Defines</td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY LOW LEVEL

STM32F429I DISCOVERY LOW LEVEL
Defines

<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>__STM32F429I_DISCO_BSP_VERSION_MAIN</code></td>
<td>0x02</td>
</tr>
<tr>
<td>STM32F429I DISCO BSP Driver version number V2.1.3.</td>
<td></td>
</tr>
<tr>
<td><code>__STM32F429I_DISCO_BSP_VERSION_SUB1</code></td>
<td>0x01</td>
</tr>
<tr>
<td><code>__STM32F429I_DISCO_BSP_VERSION_SUB2</code></td>
<td>0x03</td>
</tr>
<tr>
<td><code>__STM32F429I_DISCO_BSP_VERSION_RC</code></td>
<td>0x00</td>
</tr>
<tr>
<td><code>__STM32F429I_DISCO_BSP_VERSION</code></td>
<td></td>
</tr>
</tbody>
</table>
Define Documentation

#define __STM32F429I_DISCO_BSP_VERSION

Value:

```
((__STM32F429I_DISCO_BSP_VERSION_MAIN << 24) | (___
STM32F429I_DISCO_BSP_VERSION_SUB1 << 16) | (___
STM32F429I_DISCO_BSP_VERSION_SUB2 << 8 ) | (___
STM32F429I_DISCO_BSP_VERSION_RC))
```

Definition at line 74 of file stm32f429i_discovery.c.

Referenced by BSP_GetVersion().

#define __STM32F429I_DISCO_BSP_VERSION_MAIN (0x02)

STM32F429I DISCO BSP Driver version number V2.1.3.

[31:24] main version

Definition at line 70 of file stm32f429i_discovery.c.

#define __STM32F429I_DISCO_BSP_VERSION_RC (0x00)

[7:0] release candidate

Definition at line 73 of file stm32f429i_discovery.c.

#define __STM32F429I_DISCO_BSP_VERSION_SUB1 (0x01)
[23:16] sub1 version

Definition at line 71 of file stm32f429i_discovery.c.

```
define __STM32F429I_DISCO_BSP_VERSION_SUB2 (0x03)
```

[15:8] sub2 version

Definition at line 72 of file stm32f429i_discovery.c.

BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
<th>Defines</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Defines

```c
#define ABS(X) ((X) > 0 ? (X) : -(X))
```
Define Documentation

```c
#define ABS ( X ) ((X) > 0 ? (X) : -(X))
```

Definition at line 117 of file `stm32f429i_discovery_lcd.c`.

Referenced by `BSP_LCD_DrawLine()`, and `BSP_LCD_FillTriangle()`.
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LCD Private Variables
Variables

<table>
<thead>
<tr>
<th>Static Type</th>
<th>Variable Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTDC_HandleTypeDef</td>
<td>LttcHandler</td>
</tr>
<tr>
<td>DMA2D_HandleTypeDef</td>
<td>Dma2dHandler</td>
</tr>
<tr>
<td>RCC_PeriphCLKInitTypeDef</td>
<td>PeriphClkInitStruct</td>
</tr>
<tr>
<td>static uint32_t ActiveLayer</td>
<td>ActiveLayer = 0</td>
</tr>
<tr>
<td>LCD_DrawPropTypeDef</td>
<td>DrawProp [MAX_LAYER_NUMBER]</td>
</tr>
<tr>
<td>LCD_DrvTypeDef * LcdDrv</td>
<td>LcdDrv</td>
</tr>
</tbody>
</table>
Variable Documentation

**uint32_t** `ActiveLayer = 0` [static]

Definition at line 130 of file `stm32f429i_discovery_lcd.c`.


**DMA2D_HandleTypeDef** `Dma2dHandler` [static]

Definition at line 126 of file `stm32f429i_discovery_lcd.c`.

Referenced by `ConvertLineToARGB8888()` and `FillBuffer()`.

**LCD_DrawPropTypeDef** `DrawProp[ ]` [static]

Definition at line 131 of file `stm32f429i_discovery_lcd.c`.

**LCD_DrvTypeDef** `LcdDrv`

Definition at line 132 of file `stm32f429i_discovery_lcd.c`.

Referenced by `BSP_LCD_DisplayOff()`, `BSP_LCD_DisplayOn()`,
BSP_LCDGetXSize(), BSP_LCDGetYSize(), and BSP_LCD_Init.

LTDC_HandleTypeDef LdctHandler [static]

Definition at line 125 of file stm32f429i_discovery_lcd.c.

Referenced by BSP_LCD_Clear(), BSP_LCD_DrawBitmap(), BSP_LCD_DrawHLine(), BSP_LCD_DrawPixel(), BSP_LCD_DrawVLine(), BSP_LCD_FillRect(), BSP_LCD_Init(), BSP_LCD_LayerDefaultInit(), BSP_LCD_ReadPixel(), BSP_LCD_ResetColorKeying(), BSP_LCD_SetColorKeying(), BSP_LCD_SetLayerAddress(), BSP_LCD_SetLayerVisible(), BSP_LCD_SetLayerWindow(), and BSP_LCD_SetTransparency().

RCC_PeriphCLKInitTypeDef PeriphClkInitStruct [static]

Definition at line 127 of file stm32f429i_discovery_lcd.c.

Referenced by BSP_LCD_Init().
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LOW LEVEL Exported Functions

STM32F429I DISCOVERY LOW LEVEL
### Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>uint32_t</strong> BSP_GetVersion (void)</td>
<td>This method returns the STM32F429I DISCO BSP Driver revision.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_LED_Init (Led_TypeDef Led)</td>
<td>Configures LED GPIO.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_LED_On (Led_TypeDef Led)</td>
<td>Turns selected LED On.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_LED_Off (Led_TypeDef Led)</td>
<td>Turns selected LED Off.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_LED_Toggle (Led_TypeDef Led)</td>
<td>Toggles the selected LED.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_PB_Init (Button_TypeDef Button, ButtonMode_TypeDef ButtonMode)</td>
<td>Configures Button GPIO and EXTI Line.</td>
</tr>
<tr>
<td><strong>uint32_t</strong> BSP_PB_GetState (Button_TypeDef Button)</td>
<td>Returns the selected Button state.</td>
</tr>
</tbody>
</table>
Function Documentation

uint32_t BSP_GetVersion ( void )

This method returns the STM32F429I DISCO BSP Driver revision.

Return values:

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

Definition at line 180 of file stm32f429i_discovery.c.

References __STM32F429I_DISCO_BSP_VERSION.

void BSP_LED_Init ( Led_TypeDef Led )

Configures LED GPIO.

Parameters:

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

- LED3
- LED4

Definition at line 192 of file stm32f429i_discovery.c.

References GPIO_PIN, GPIO_PORT, and LEDx_GPIO_CLK_ENABLE.

void BSP_LED_Off ( Led_TypeDef Led )

Turns selected LED Off.

Parameters:

Led, Specifies the Led to be set off. This parameter can be
void BSP_LED_On ( Led_TypeDef Led )

Turns selected LED On.

Parameters:
   Led,: Specifies the Led to be set on. This parameter can be one of following parameters:
   • LED3
   • LED4

Definition at line 229 of file stm32f429i_discovery.c.
References GPIO_PIN, and GPIO_PORT.

void BSP_LED_Toggle ( Led_TypeDef Led )

Toggles the selected LED.

Parameters:
   Led,: Specifies the Led to be toggled. This parameter can be one of following parameters:
   • LED3
   • LED4

Definition at line 241 of file stm32f429i_discovery.c.
References GPIO_PIN, and GPIO_PORT.
uint32_t BSP_PB_GetState ( Button_TypeDef Button )

Returns the selected Button state.

**Parameters:**

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

**Return values:**

- *The* Button GPIO pin value.

Definition at line 293 of file stm32f429i_discovery.c.

References BUTTON_PIN, and BUTTON_PORT.

void BSP_PB_Init ( Button_TypeDef Button,
                  ButtonMode_TypeDef ButtonMode )

Configures Button GPIO and EXTI Line.

**Parameters:**

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

- **ButtonMode:** Specifies Button mode. This parameter can be one of following parameters:
  - BUTTON_MODE_GPIO: Button will be used as simple IO
  - BUTTON_MODE_EXTI: Button will be connected to EXTI line with interrupt generation capability

Definition at line 256 of file stm32f429i_discovery.c.

References BUTTON_IRQHandler, BUTTON_MODE_EXTI,
BUTTON_MODE_GPIO, BUTTON_PIN, BUTTON_PORT, and BUTTONx_GPIO_CLK_ENABLE.
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STM32F429I DISCOVERY LOW LEVEL Private Functions</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_GetVersion (void)</code></td>
<td>This method returns the STM32F429I DISCO BSP Driver revision.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LED_Init (Led_TypeDef Led)</code></td>
<td>Configures LED GPIO.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LED_On (Led_TypeDef Led)</code></td>
<td>Turns selected LED On.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LED_Off (Led_TypeDef Led)</code></td>
<td>Turns selected LED Off.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LED_Toggle (Led_TypeDef Led)</code></td>
<td>Toggles the selected LED.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>void BSP_PB_Init (Button_TypeDef Button, ButtonMode_TypeDef ButtonMode)</code></td>
<td>Configures Button GPIO and EXTI Line.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_PB_GetState (Button_TypeDef Button)</code></td>
<td>Returns the selected Button state.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>static void I2Cx_MspInit (I2C_HandleTypeDef *hi2c)</code></td>
<td>I2Cx MSP Initialization.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>static void I2Cx_WriteData (uint8_t Addr, uint8_t Reg, uint8_t Value)</code></td>
<td>Writes a value in a register of the device through BUS.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>static void I2Cx_WriteBuffer (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)</code></td>
<td>Writes a value in a register of the device through BUS.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>static uint8_t I2Cx_ReadData (uint8_t Addr, uint8_t Reg)</code></td>
<td>Reads a register of the device through BUS.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>static uint8_t I2Cx_ReadBuffer (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)</code></td>
<td>Reads multiple data on the BUS.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>static uint32_t SPIx_Read (uint8_t ReadSize)</code></td>
<td>Reads 4 bytes from device.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
static void **SPIx_Write** (uint16_t Value)
Writes a byte to device.

static uint8_t **SPIx_WriteRead** (uint8_t Byte)
Sends a Byte through the SPI interface and return the Byte received from the SPI bus.

static void **SPIx_MspInit** (SPI_HandleTypeDef *hspi)
SPI MSP Init.

uint32_t **LCD_IO_ReadData** (uint16_t RegValue, uint8_t ReadSize)
Reads register value.

void **LCD_Delay** (uint32_t Delay)
Wait for loop in ms.

void **IOE_Write** (uint8_t Addr, uint8_t Reg, uint8_t Value)
IOE Writes single data operation.

uint8_t **IOE_Read** (uint8_t Addr, uint8_t Reg)
IOE Reads single data.

void **IOE_WriteMultiple** (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)
IOE Writes multiple data.

uint16_t **IOE_ReadMultiple** (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)
IOE Reads multiple data.

void **IOE_Delay** (uint32_t Delay)
IOE Delay.

void **GYRO_IO_Write** (uint8_t *pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite)
Writes one byte to the Gyroscope.

void **GYRO_IO_Read** (uint8_t *pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead)
Reads a block of data from the Gyroscope.
Function Documentation

`uint32_t BSP_GetVersion ( void )`

This method returns the STM32F429I DISCO BSP Driver revision.

**Return values:**

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

Definition at line 180 of file `stm32f429i_discovery.c`.

References `__STM32F429I_DISCO_BSP_VERSION`.

`void BSP_LED_Init ( Led_TypeDef Led )`

Configures LED GPIO.

**Parameters:**

- `Led`: Specifies the Led to be configured. This parameter can be one of following parameters:
  - LED3
  - LED4

Definition at line 192 of file `stm32f429i_discovery.c`.

References `GPIO_PIN`, `GPIO_PORT`, and `LEDx_GPIO_CLK_ENABLE`.

`void BSP_LED_Off ( Led_TypeDef Led )`

Turns selected LED Off.

**Parameters:**

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

- LED3
- LED4

Definition at line 229 of file stm32f429i_discovery.c.

References GPIO_PIN, and GPIO_PORT.

```c
void BSP_LED_On (Led_TypeDef Led)
```

Turns selected LED On.

**Parameters:**

- **Led:** Specifies the Led to be set on. This parameter can be one of following parameters:
  
  - LED3
  
  - LED4

Definition at line 217 of file stm32f429i_discovery.c.

References GPIO_PIN, and GPIO_PORT.

```c
void BSP_LED_Toggle (Led_TypeDef Led)
```

Toggles the selected LED.

**Parameters:**

- **Led:** Specifies the Led to be toggled. This parameter can be one of following parameters:
  
  - LED3
  
  - LED4

Definition at line 241 of file stm32f429i_discovery.c.

References GPIO_PIN, and GPIO_PORT.
uint32_t BSP_PB_GetState ( Button_TypeDef Button )

Returns the selected Button state.

**Parameters:**

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

**Return values:**

- The Button GPIO pin value.

Definition at line 293 of file stm32f429i_discovery.c.

References BUTTON_PIN, and BUTTON_PORT.

void BSP_PB_Init ( Button_TypeDef Button,
                   ButtonMode_TypeDef ButtonMode )

Configures Button GPIO and EXTI Line.

**Parameters:**

- **Button**: Specifies the Button to be configured. This parameter should be: BUTTON_KEY
- **ButtonMode**: Specifies Button mode. This parameter can be one of following parameters:
  - BUTTON_MODE_GPIO: Button will be used as simple IO
  - BUTTON_MODE_EXTI: Button will be connected to EXTI line with interrupt generation capability

Definition at line 256 of file stm32f429i_discovery.c.

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

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

Reads a block of data from the Gyroscope.

**Parameters:**
- `pBuffer`: Pointer to the buffer that receives the data read from the Gyroscope.
- `ReadAddr`: Gyroscope's internal address to read from.
- `NumByteToRead`: Number of bytes to read from the Gyroscope.

Definition at line 1028 of file `stm32f429i_discovery.c`.

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

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

Writes one byte to the Gyroscope.

**Parameters:**
- `pBuffer`: Pointer to the buffer containing the data to be written to the Gyroscope.
- `WriteAddr`: Gyroscope's internal address to write to.
**NumByteToWrite:** Number of bytes to write.

Definition at line 994 of file stm32f429i_discovery.c.

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

```c
static void I2Cx_MspInit ( I2C_HandleTypeDef * hi2c )
```

I2Cx MSP Initialization.

**Parameters:**

- **hi2c:** I2C handle

Definition at line 308 of file stm32f429i_discovery.c.

References **DISCOVERY_I2Cx**, **DISCOVERY_I2Cx_CLOCK_ENABLE**, **DISCOVERY_I2Cx_ER_IRQn**, **DISCOVERY_I2Cx_EV_IRQn**, **DISCOVERY_I2Cx_FORCE_RESET**, **DISCOVERY_I2Cx_RELEASE_RESET**, **DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE**, **DISCOVERY_I2Cx_SCL_GPIO_PORT**, **DISCOVERY_I2Cx_SCL_PIN**, **DISCOVERY_I2Cx_SCL_SDA_AF**, **DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE**, **DISCOVERY_I2Cx_SDA_GPIO_PORT**, **DISCOVERY_I2Cx_SDA_PIN**, and **I2cHandle**.

Referenced by **I2Cx_Init()**.

```c
static uint8_t I2Cx_ReadBuffer ( uint8_t Addr,
                                 uint8_t Reg,
                                 uint8_t * pBuffer,
                                 uint16_t Length
                               )
```

[static]
Reads multiple data on the BUS.

**Parameters:**
- **Addr:** I2C Address
- **Reg:** Reg Address
- **pBuffer:** pointer to read data buffer
- **Length:** length of the data

**Return values:**
- **0** if no problems to read multiple data

Definition at line **535** of file **stm32f429i_discovery.c**.

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

Referenced by **IOE_ReadMultiple()**.

```c
static uint8_t I2Cx_ReadData ( uint8_t Addr,
                               uint8_t Reg            
                           )
```

Reads a register of the device through BUS.

**Parameters:**
- **Addr:** Device address on BUS Bus.
- **Reg:** The target register address to write

**Return values:**
- **Data** read at register address

Definition at line **510** of file **stm32f429i_discovery.c**.

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

Referenced by **IOE_Read()**.
static void I2Cx_WriteBuffer ( uint8_t Addr,
   uint8_t Reg,
   uint8_t * pBuffer,
   uint16_t Length
)

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

Parameters:
   Addr,: Device address on BUS Bus.
   Reg,: The target register address to write
   pBuffer,: The target register value to be written
   Length,: buffer size to be written

Definition at line 490 of file stm32f429i_discovery.c.

References I2cHandle, I2Cx_Error(), and I2cxTimeout.

Referenced by IOE_WriteMultiple().

static void I2Cx_WriteData ( uint8_t Addr,
   uint8_t Reg,
   uint8_t Value
)

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

Parameters:
   Addr,: Device address on BUS Bus.
   Reg,: The target register address to write
   Value,: The target register value to be written

Definition at line 469 of file stm32f429i_discovery.c.
References **I2cHandle**, **I2Cx_Error()**, and **I2cxTimeout**.

Referenced by **IOE_Write()**.

```c
void IOE_Delay ( uint32_t Delay )
```

IOE Delay.

**Parameters:**

- **Delay** in ms

Definition at line 951 of file *stm32f429i_discovery.c*.

```c
uint8_t IOE_Read ( uint8_t Addr,
                  uint8_t Reg)
```

IOE Reads single data.

**Parameters:**

- **Addr:** I2C Address
- **Reg:** Reg Address

**Return values:**

- The read data

Definition at line 917 of file *stm32f429i_discovery.c*.

References **I2Cx_ReadData()**.

```c
uint16_t IOE_ReadMultiple ( uint8_t Addr,
                           uint8_t Reg,
                           uint8_t * pBuffer,
```

...
IOE Reads multiple data.

**Parameters:**
- **Addr:** I2C Address
- **Reg:** Reg Address
- **pBuffer:** pointer to data buffer
- **Length:** length of the data

**Return values:**
- **0** if no problems to read multiple data

Definition at line 942 of file `stm32f429i_discovery.c`.

References `I2Cx_ReadBuffer()`.

```c
void IOE_Write ( uint8_t Addr,
    uint8_t Reg,
    uint8_t Value
    )
```

IOE Writes single data operation.

**Parameters:**
- **Addr:** I2C Address
- **Reg:** Reg Address
- **Value:** Data to be written

Definition at line 906 of file `stm32f429i_discovery.c`.

References `I2Cx_WriteData()`.
void IOE_WriteMultiple ( uint8_t Addr,
                      uint8_t Reg,
                      uint8_t * pBuffer,
                      uint16_t Length
                    )

IOE Writes multiple data.

**Parameters:**
- **Addr:** I2C Address
- **Reg:** Reg Address
- **pBuffer:** pointer to data buffer
- **Length:** length of the data

Definition at line 929 of file stm32f429i_discovery.c.

References `I2Cx_WriteBuffer()`.

void LCD_Delay ( uint32_t Delay )

Wait for loop in ms.

**Parameters:**
- **Delay** in ms.

Definition at line 873 of file stm32f429i_discovery.c.

uint32_t LCD_IO_ReadData ( uint16_t RegValue,
                           uint8_t ReadSize
                         )

Reads register value.

**Parameters:**
RegValue Address of the register to read
ReadSize Number of bytes to read

Return values:
Content of the register value

Definition at line 846 of file stm32f429i_discovery.c.

References LCD_CS_HIGH, LCD_CS_LOW, LCD_WRX_HIGH, LCD_WRX_LOW, SPIx_Read(), and SPIx_Write().

static void SPIx_MspInit (SPI_HandleTypeDef * hspi) [static]

SPI MSP Init.

Parameters:
  hspi: SPI handle

Definition at line 743 of file stm32f429i_discovery.c.

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

Referenced by SPIx_Init().

static uint32_t SPIx_Read (uint8_t ReadSize) [static]

Reads 4 bytes from device.

Parameters:
  ReadSize: Number of bytes to read (max 4 bytes)
Return values:

Value read on the SPI

Definition at line 672 of file stm32f429i_discovery.c.

References SpiHandle, SPIx_Error(), and SpixTimeout.

Referenced by LCD_IO_ReadData().

static void SPIx_Write ( uint16_t Value ) [static]

Writes a byte to device.

Parameters:

Value,: value to be written

Definition at line 693 of file stm32f429i_discovery.c.

References SpiHandle, SPIx_Error(), and SpixTimeout.

Referenced by LCD_IO_ReadData(), LCD_IO_WriteData(), and LCD_IO_WriteReg().

static uint8_t SPIx_WriteRead ( uint8_t Byte ) [static]

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

Parameters:

Byte,: Byte send.

Return values:

The received byte value

Definition at line 713 of file stm32f429i_discovery.c.
References `SpiHandle`, `SPIx_Error()`, and `SpixTimeout`.

Referenced by `GYRO_IO_Read()`, and `GYRO_IO_Write()`.
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td><strong>Functions</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY GYROSCOPE Private Functions**

STM32F429I DISCOVERY GYROSCOPE
## Functions

<table>
<thead>
<tr>
<th>uint8_t</th>
<th><strong>BSP_GYRO_Init</strong> (void)</th>
<th>Set Gyroscope Initialization.</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><strong>BSP_GYRO_ReadID</strong> (void)</td>
<td>Read ID of Gyroscope component.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_Reset</strong> (void)</td>
<td>Reboot memory content of Gyroscope.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_ITConfig</strong> (GYRO_InterruptConfigTypeDef *pIntConfig)</td>
<td>Configures INT1 interrupt.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_EnableIT</strong> (uint8_t IntPin)</td>
<td>Enables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_DisableIT</strong> (uint8_t IntPin)</td>
<td>Disables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_GYRO_GetXYZ</strong> (float *pfData)</td>
<td>Gets XYZ angular acceleration/.</td>
</tr>
</tbody>
</table>
Function Documentation

```c
void BSP_GYRO_DisableIT ( uint8_t IntPin )
```
Disables INT1 or INT2 interrupt.

**Parameters:**
- **IntPin:** Interrupt pin This parameter can be:
  - L3GD20_INT1
  - L3GD20_INT2

Definition at line 219 of file stm32f429i_discovery_gyroscope.c.

References GyroscopeDrv.

```c
void BSP_GYRO_EnableIT ( uint8_t IntPin )
```
Enables INT1 or INT2 interrupt.

**Parameters:**
- **IntPin:** Interrupt pin This parameter can be:
  - L3GD20_INT1
  - L3GD20_INT2

Definition at line 204 of file stm32f429i_discovery_gyroscope.c.

References GyroscopeDrv.

```c
void BSP_GYRO_GetXYZ ( float * pfData )
```
Gets XYZ angular acceleration/.

**Parameters:**
- **pfData:** pointer on floating array
uint8_t BSP_GYRO_Init ( void )

Set Gyroscope Initialization.

**Return values:**
- **GYRO_OK** if no problem during initialization

Definition at line 231 of file stm32f429i_discovery_gyroscope.c.

References GyroscopeDrv.

void BSP_GYRO_ITConfig ( GYRO_InterruptConfigTypeDef * pIntConfig)

Configures INT1 interrupt.

**Parameters:**
- **pIntConfig**: pointer to a L3GD20_InterruptConfig_TypeDef structure that contains the configuration setting for the L3GD20 Interrupt.

Definition at line 98 of file stm32f429i_discovery_gyroscope.c.

References GYRO_ERROR, GYRO_OK, and GyroscopeDrv.

uint8_t BSP_GYRO_ReadID ( void )

Read ID of Gyroscope component.

**Return values:**
- **ID**
Definition at line 154 of file `stm32f429i_discovery_gyroscope.c`. References `GyroscopeDrv`.

```c
void BSP_GYRO_Reset ( void )
```

Reboot memory content of Gyroscope.

Definition at line 168 of file `stm32f429i_discovery_gyroscope.c`. References `GyroscopeDrv`.

# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### STM32F429I DISCOVERY GYROSCOPE Exported Functions

**STM32F429I DISCOVERY GYROSCOPE**
## Functions

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t</code></td>
<td><strong>BSP_GYRO_Init</strong> (void)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_GYRO_Reset</strong> (void)</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><strong>BSP_GYRO_ReadID</strong> (void)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_GYRO_ITConfig</strong> (GYRO_InterruptConfigTypeDef *pIntConfigStruct)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_GYRO_EnableIT</strong> (uint8_t IntPin)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_GYRO_DisableIT</strong> (uint8_t IntPin)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_GYRO_GetXYZ</strong> (float *pfData)</td>
</tr>
</tbody>
</table>
Function Documentation

void **BSP_GYRO_DisableIT** ( uint8_t **IntPin** )

Disables INT1 or INT2 interrupt.

**Parameters:**

**IntPin**; Interrupt pin This parameter can be:

- L3GD20_INT1
- L3GD20_INT2

Definition at line 219 of file `stm32f429i_discovery_gyroscope.c`.

References **GyroscopeDrv**.

---

void **BSP_GYRO_EnableIT** ( uint8_t **IntPin** )

Enables INT1 or INT2 interrupt.

**Parameters:**

**IntPin**; Interrupt pin This parameter can be:

- L3GD20_INT1
- L3GD20_INT2

Definition at line 204 of file `stm32f429i_discovery_gyroscope.c`.

References **GyroscopeDrv**.

---

void **BSP_GYRO_GetXYZ** ( float * **pfData** )

Gets XYZ angular acceleration/.

**Parameters:**

**pfData**; pointer on floating array
Definition at line 231 of file stm32f429i_discovery_gyroscope.c.

References GyroscopeDrv.

```c
uint8_t BSP_GYRO_Init ( void )
```

Set Gyroscope Initialization.

**Return values:**
- **GYRO_OK** if no problem during initialization

Definition at line 98 of file stm32f429i_discovery_gyroscope.c.

References GYRO_ERROR, GYRO_OK, and GyroscopeDrv.

```c
void BSP_GYRO_ITConfig ( GYRO_InterruptConfigTypeDef * pIntConfig
```

Configures INT1 interrupt.

**Parameters:**
- **pIntConfig:** pointer to a L3GD20_InterruptConfig_TypeDef structure that contains the configuration setting for the L3GD20 Interrupt.

Definition at line 181 of file stm32f429i_discovery_gyroscope.c.

References GyroscopeDrv.

```c
uint8_t BSP_GYRO_ReadID ( void )
```

Read ID of Gyroscope component.

**Return values:**
- **ID**
Definition at line 154 of file stm32f429i_discovery_gyroscope.c.
References GyroscopeDrv.

```c
void BSP_GYRO_Reset ( void )
```

Reboot memory content of Gyroscope.

Definition at line 168 of file stm32f429i_discovery_gyroscope.c.
References GyroscopeDrv.

BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY IO Private Functions

STM32F429I DISCOVERY IO
# Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td>BSP_IO_Init (void)</td>
<td>Initializes and configures the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td>uint8_t</td>
<td>BSP_IO_ITGetStatus (uint16_t IoPin)</td>
<td>Gets the selected pins IT status.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_IO_ITClear (void)</td>
<td>Clears all the IO IT pending bits.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_IO_ConfigPin (uint16_t IoPin, IO_ModeTypedef IoMode)</td>
<td>Configures the IO pin(s) according to IO mode structure value.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_IO_WritePin (uint16_t IoPin, uint8_t PinState)</td>
<td>Sets the selected pins state.</td>
</tr>
<tr>
<td>uint16_t</td>
<td>BSP_IO_ReadPin (uint16_t IoPin)</td>
<td>Gets the selected pins current state.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_IO_TogglePin (uint16_t IoPin)</td>
<td>Toggles the selected pins state.</td>
</tr>
</tbody>
</table>
Function Documentation

```c
void BSP_IO_ConfigPin ( uint16_t IoPin, IO_ModeTypedef IoMode )
```

Configures the IO pin(s) according to IO mode structure value.

**Parameters:**

- **IoPin:** IO pin(s) to be configured. This parameter could be any combination of the following values:
  - STMPE811_PIN_x: where x can be from 0 to 7.
- **IoMode:** The IO pin mode to configure, could 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

Definition at line 162 of file stm32f429i_discovery_io.c.

References IO_I2C_ADDRESS, and IoDrv.

```c
uint8_t BSP_IO_Init ( void )
```

Initializes and configures 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 \texttt{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:**
\texttt{IO\_OK} if all initializations done correctly. Other value if error.

Definition at line 108 of file \texttt{stm32f429i\_discovery\_io.c}.

References \texttt{IO\_ERROR}, \texttt{IO\_I2C\_ADDRESS}, \texttt{IO\_OK}, \texttt{IO\_PIN\_ALL}, and \texttt{IoDrv}.

\begin{verbatim}
void BSP\_IO\_ITClear ( void )
\end{verbatim}

Clears all the IO IT pending bits.

Definition at line 143 of file \texttt{stm32f429i\_discovery\_io.c}.

References \texttt{IO\_I2C\_ADDRESS}, \texttt{IO\_PIN\_ALL}, and \texttt{IoDrv}.

\begin{verbatim}
uint8\_t BSP\_IO\_ITGetStatus ( uint16\_t IoPin )
\end{verbatim}

Gets the selected pins IT status.

**Parameters:**
\textit{IoPin,}: The selected pins to check the status. This parameter could be any combination of the IO pins.

**Return values:**
\textit{Status} of IO Pin checked.

Definition at line 134 of file \texttt{stm32f429i\_discovery\_io.c}.

References \texttt{IO\_I2C\_ADDRESS}, and \texttt{IoDrv}. 
uint16_t BSP_IO_ReadPin ( uint16_t IoPin )

Gets the selected pins current state.

**Parameters:**

- **IoPin:** The selected pins to read. This parameter could be any combination of the IO pins.

**Return values:**

- The current pins state

Definition at line 186 of file **stm32f429i_discovery_io.c**.

References **IO_I2C_ADDRESS**, and **IoDrv**.

---

void BSP_IO_TogglePin ( uint16_t IoPin )

Toggles the selected pins state.

**Parameters:**

- **IoPin:** The selected pins to toggle. This parameter could be any combination of the IO pins.

Definition at line 196 of file **stm32f429i_discovery_io.c**.

References **IO_I2C_ADDRESS**, and **IoDrv**.

---

void BSP_IO_WritePin ( uint16_t IoPin, uint8_t PinState )

Sets the selected pins state.

**Parameters:**
**IoPin,** The selected pins to write. This parameter could be any combination of the IO pins.

**PinState,** the new pins state to write.

Definition at line 174 of file `stm32f429i_discovery_io.c`.

References `IO_I2C_ADDRESS`, and `IoDrv`. 
## STM32F429I-Discovery BSP User Manual

### Directories

<table>
<thead>
<tr>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
</table>

### STM32F429I DISCOVERY IO Exported Functions

STM32F429I DISCOVERY IO
## Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_IO_Init</code> (void)</td>
<td>Initializes and configures the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_IO_ITGetStatus</code> (uint16_t IoPin)</td>
<td>Gets the selected pins IT status.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_IO_ITClear</code> (void)</td>
<td>Clears all the IO IT pending bits.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_IO_ConfigPin</code> (uint16_t IoPin, IO_ModeTypedef IoMode)</td>
<td>Configures the IO pin(s) according to IO mode structure value.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_IO_WritePin</code> (uint16_t IoPin, uint8_t PinState)</td>
<td>Sets the selected pins state.</td>
</tr>
<tr>
<td><code>uint16_t</code></td>
<td><code>BSP_IO_ReadPin</code> (uint16_t IoPin)</td>
<td>Gets the selected pins current state.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_IO_TogglePin</code> (uint16_t IoPin)</td>
<td>Toggles the selected pins state.</td>
</tr>
</tbody>
</table>
Function Documentation

void BSP_IO_ConfigPin ( uint16_t IoPin, IO_ModeTypedef IoMode )

Configures the IO pin(s) according to IO mode structure value.

**Parameters:**

- **IoPin:** IO pin(s) to be configured. This parameter could be any combination of the following values:
  - STMPE811_PIN_x: where x can be from 0 to 7.

- **IoMode:** The IO pin mode to configure, could 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

Definition at line 162 of file stm32f429i_discovery_io.c.

References IO_I2C_ADDRESS, and IoDrv.

uint8_t BSP_IO_Init ( void )

Initializes and configures 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 done correctly. Other value if error.

Definition at line 108 of file `stm32f429i_discovery_io.c`.

References `IO_ERROR`, `IO_I2C_ADDRESS`, `IO_OK`, `IO_PIN_ALL`, and `IoDrv`.

```c
void BSP_IO_ITClear ( void )
```

Clears all the IO IT pending bits.

Definition at line 143 of file `stm32f429i_discovery_io.c`.

References `IO_I2C_ADDRESS`, `IO_PIN_ALL`, and `IoDrv`.

```c
uint8_t BSP_IO_ITGetStatus ( uint16_t IoPin )
```

Gets the selected pins IT status.

**Parameters:**

- **IoPin,** The selected pins to check the status. This parameter could be any combination of the IO pins.

**Return values:**

- **Status** of IO Pin checked.

Definition at line 134 of file `stm32f429i_discovery_io.c`.

References `IO_I2C_ADDRESS`, and `IoDrv`. 
# BSP IO Functions

**uint16_t BSP_IO_ReadPin (uint16_t IoPin)**

Gets the selected pins current state.

**Parameters:**
- **IoPin:** The selected pins to read. This parameter could be any combination of the IO pins.

**Return values:**
- The current pins state

Definition at line 186 of file `stm32f429i_discovery_io.c`.

References `IO_I2C_ADDRESS`, and `IoDrv`.

**void BSP_IO_TogglePin (uint16_t IoPin)**

Toggles the selected pins state.

**Parameters:**
- **IoPin:** The selected pins to toggle. This parameter could be any combination of the IO pins.

Definition at line 196 of file `stm32f429i_discovery_io.c`.

References `IO_I2C_ADDRESS`, and `IoDrv`.

**void BSP_IO_WritePin (uint16_t IoPin, uint8_t PinState)**

Sets the selected pins state.

**Parameters:**
IoPin,: The selected pins to write. This parameter could be any combination of the IO pins.

PinState,: the new pins state to write

Definition at line 174 of file stm32f429i_discovery_io.c.

References IO_I2C_ADDRESS, and IoDrv.
## STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STM32F429I</strong></td>
<td><strong>DISCOVERY LCD Private Functions</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STM32F429I DISCOVERY LCD</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><strong>BSP_LCD_Init</strong> (void)</td>
<td>Initializes the LCD.</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_LCD_GetXSize</strong> (void)</td>
<td>Gets the LCD X size.</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_LCD_GetYSize</strong> (void)</td>
<td>Gets the LCD Y size.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_LayerDefaultInit</strong> (uint16_t LayerIndex, uint32_t FB_Address)</td>
<td>Initializes the LCD layers.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SelectLayer</strong> (uint32_t LayerIndex)</td>
<td>Selects the LCD Layer.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetLayerVisible</strong> (uint32_t LayerIndex, FunctionalState state)</td>
<td>Sets a LCD Layer visible.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetTransparency</strong> (uint32_t LayerIndex, uint8_t Transparency)</td>
<td>Configures the Transparency.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetLayerAddress</strong> (uint32_t LayerIndex, uint32_t Address)</td>
<td>Sets a LCD layer frame buffer address.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetLayerWindow</strong> (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</td>
<td>Sets the Display window.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_SetColorKeying</strong> (uint32_t LayerIndex, uint32_t RGBValue)</td>
<td>Configures and sets the color Keying.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_ResetColorKeying</strong> (uint32_t LayerIndex)</td>
<td>Disables the color Keying.</td>
</tr>
<tr>
<td>uint32_t</td>
<td><strong>BSP_LCD_GetTextColor</strong> (void)</td>
<td>Gets the LCD Text color.</td>
</tr>
<tr>
<td>Function Type</td>
<td>Function Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>uint32_t</td>
<td><code>BSP_LCD_GetBackColor</code></td>
<td>(void) Gets the LCD Background color.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SetTextColor</code></td>
<td>(uint32_t Color) Sets the Text color.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SetBackColor</code></td>
<td>(uint32_t Color) Sets the Background color.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SetFont</code></td>
<td>(sFONT *pFonts) Sets the Text Font.</td>
</tr>
<tr>
<td>sFONT *</td>
<td><code>BSP_LCD_GetFont</code></td>
<td>(void) Gets the Text Font.</td>
</tr>
<tr>
<td>uint32_t</td>
<td><code>BSP_LCD_ReadPixel</code></td>
<td>(uint16_t Xpos, uint16_t Ypos) Reads Pixel.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_Clear</code></td>
<td>(uint32_t Color) Clears the hole LCD.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_ClearStringLine</code></td>
<td>(uint32_t Line) Clears the selected line.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_DisplayChar</code></td>
<td>(uint16_t Xpos, uint16_t Ypos, uint8_t Ascii) Displays one character.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_DisplayStringAt</code></td>
<td>(uint16_t X, uint16_t Y, uint8_t *pText, <code>Text_AlignModeTypdef</code> mode) Displays a maximum of 60 char on the LCD.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_DisplayStringAtLine</code></td>
<td>(uint16_t Line, uint8_t *ptr) Displays a maximum of 20 char on the LCD.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_DrawHLine</code></td>
<td>(uint16_t Xpos, uint16_t Ypos, uint16_t Length) Displays an horizontal line.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_DrawVLine</code></td>
<td>(uint16_t Xpos, uint16_t Ypos, uint16_t Length) Displays a vertical line.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_DrawLine</code></td>
<td>(uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2) Displays an uni-line (between two points).</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_DrawRect</code></td>
<td>Draws a rectangle.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_DrawCircle</code></td>
<td>Draws a circle.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_DrawPolygon</code></td>
<td>Draws a poly-line (between many points).</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_DrawEllipse</code></td>
<td>Draws an ellipse.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_DrawBitmap</code></td>
<td>Draws a bitmap picture loaded in the internal Flash (32 bpp).</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_FillRect</code></td>
<td>Draws a full rectangle.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_FillCircle</code></td>
<td>Draws a full circle.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_FillTriangle</code></td>
<td>Draws a filled triangle.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_FillPolygon</code></td>
<td>Draws a full poly-line (between many points).</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_FillEllipse</code></td>
<td>Draws a full ellipse.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_DisplayOn</code></td>
<td>Enables the display.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_DisplayOff</code></td>
<td>Disables the display.</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Parameters</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>void <strong>BSP_LCD_DrawPixel</strong></td>
<td>(uint16_t Xpos, uint16_t Ypos, uint32_t RGB_Code)</td>
<td>Writes Pixel</td>
</tr>
<tr>
<td>static void <strong>DrawChar</strong></td>
<td>(uint16_t Xpos, uint16_t Ypos, const uint8_t *c)</td>
<td>Draws a character on LCD</td>
</tr>
<tr>
<td>static void <strong>FillBuffer</strong></td>
<td>(uint32_t LayerIndex, void *pDst, uint32_t xSize, uint32_t ySize, uint32_t OffLine, uint32_t ColorIndex)</td>
<td>Fills buffer</td>
</tr>
<tr>
<td>static void <strong>ConvertLineToARGB8888</strong></td>
<td>(void *pSrc, void *pDst, uint32_t xSize, uint32_t ColorMode)</td>
<td>Converts Line to ARGB8888 pixel format.</td>
</tr>
</tbody>
</table>
Function Documentation

void BSP_LCD_Clear ( uint32_t Color )

Clears the hole LCD.

Parameters:
   Color,: the color of the background

Definition at line 471 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, BSP_LCD_GetXSize(), BSP_LCD_GetYSize(), FillBuffer(), and LtdcHandler.

void BSP_LCD_ClearStringLine ( uint32_t Line )

Clears the selected line.

Parameters:
   Line,: the line to be cleared

Definition at line 481 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, LCD_DrawPropTypeDef::BackColor, BSP_LCD_FillRect(), BSP_LCD_GetXSize(), BSP_LCD_SetTextColor(), and LCD_DrawPropTypeDef::TextColor.

void BSP_LCD_DisplayChar ( uint16_t Xpos, uint16_t Ypos, uint8_t Ascii )
Displays one character.

**Parameters:**
- **Xpos:** start column address
- **Ypos:** the Line where to display the character shape
- **Ascii:** character ascii code, must be between 0x20 and 0x7E

Definition at line 499 of file *stm32f429i_discovery_lcd.c*.

References *ActiveLayer*, *DrawChar()*, and *LCD_DrawPropTypeDef::pFont*.

Referenced by *BSP_LCD_DisplayStringAt()*.

```c
void BSP_LCD_DisplayOff ( void )
```

Disables the Display.

Definition at line 1116 of file *stm32f429i_discovery_lcd.c*.

References *LcdDrv*.

```c
void BSP_LCD_DisplayOn ( void )
```

Enables the Display.

Definition at line 1105 of file *stm32f429i_discovery_lcd.c*.

References *LcdDrv*.

```c
void BSP_LCD_DisplayStringAt ( uint16_t X, uint16_t Y, uint8_t * pText,
```
Displays a maximum of 60 char on the LCD.

**Parameters:**

- **X:** pointer to x position (in pixel)
- **Y:** pointer to y position (in pixel)
- **pText:** pointer to string to display on LCD
- **mode:** The display mode. This parameter can be one of the following values:
  - CENTER_MODE
  - RIGHT_MODE
  - LEFT_MODE

Definition at line 516 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, `BSP_LCD_DisplayChar()`, `BSP_LCD_GetXSize()`, `CENTER_MODE`, `LEFT_MODE`, `LCD_DrawPropTypeDef::pFont`, and `RIGHT_MODE`.

Referenced by `BSP_LCD_DisplayStringAtLine()`.

```c
void BSP_LCD_DisplayStringAtLine (uint16_t Line,
                                  uint8_t * ptr
                               )
```

Displays a maximum of 20 char on the LCD.

**Parameters:**

- **Line:** the Line where to display the character shape
- **ptr:** pointer to string to display on LCD

Definition at line 570 of file `stm32f429i_discovery_lcd.c`. 

References `BSP_LCD_DisplayStringAt()`, and `LEFT_MODE`.

```c
void BSP_LCD_DrawBitmap ( uint32_t X,
                          uint32_t Y,
                          uint8_t * pBmp
                    )
```

Displays a bitmap picture loaded in the internal Flash (32 bpp).

**Parameters:**
- `X`: the bmp x position in the LCD
- `Y`: the bmp Y position in the LCD
- `pBmp`: Bmp picture address in the internal Flash

Definition at line 805 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, `BSP_LCD_GetXSize()`, `ConvertLineToARGB8888()`, and `LtdcHandler`.

```c
void BSP_LCD_DrawCircle ( uint16_t Xpos,
                           uint16_t Ypos,
                           uint16_t Radius
                     )
```

Displays a circle.

**Parameters:**
- `Xpos`: the X position
- `Ypos`: the Y position
- `Radius`: the circle radius

Definition at line 707 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, and `BSP_LCD_DrawPixel()`.
Referenced by `BSP_LCD_FillCircle()`. 

```c
void BSP_LCD_DrawEllipse ( int Xpos,
                          int Ypos,
                          int XRadius,
                          int YRadius
                     )
```

Displays an Ellipse.

**Parameters:**

- `Xpos:` the X position
- `Ypos:` the Y position
- `XRadius:` the X radius of ellipse
- `YRadius:` the Y radius of ellipse

Definition at line 773 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, and `BSP_LCD_DrawPixel()`. 

```c
void BSP_LCD_DrawHLine ( uint16_t Xpos,
                         uint16_t Ypos,
                         uint16_t Length
                    )
```

Displays an horizontal line.

**Parameters:**

- `Xpos:` the X position
- `Ypos:` the Y position
- `Length:` line length

Definition at line 581 of file `stm32f429i_discovery_lcd.c`. 
References \textit{ActiveLayer}, \textit{BSP\_LCD\_GetXSize()}, \textit{FillBuffer()}, and \textit{LtdcHandler}.

Referenced by \textit{BSP\_LCD\_DrawRect()}, \textit{BSP\_LCD\_FillCircle()}, and \textit{BSP\_LCD\_FillEllipse}.

\begin{verbatim}
void BSP\_LCD\_DrawLine ( uint16_t X1,  
                       uint16_t Y1,  
                       uint16_t X2,  
                       uint16_t Y2  
                   )

Displays an uni-line (between two points).

\textbf{Parameters:}
\begin{itemize}
  \item \textbf{X1,:} the point 1 X position
  \item \textbf{Y1,:} the point 1 Y position
  \item \textbf{X2,:} the point 2 X position
  \item \textbf{Y2,:} the point 2 Y position
\end{itemize}

Definition at line 616 of file \textit{stm32f429i\_discovery\_lcd.c}.

References \textit{ABS}, \textit{ActiveLayer}, and \textit{BSP\_LCD\_DrawPixel}.

Referenced by \textit{BSP\_LCD\_DrawPolygon()}, and \textit{BSP\_LCD\_FillTriangle}.

\begin{verbatim}
void BSP\_LCD\_DrawPixel ( uint16_t Xpos,  
                         uint16_t Ypos,  
                         uint32_t RGB\_Code  
                     )

Writes Pixel.
\end{verbatim}
Parameters:

- **Xpos**: the X position
- **Ypos**: the Y position
- **RGB_Code**: the pixel color in ARGB mode (8-8-8-8)

Definition at line 1215 of file `stm32f429i_discovery_lcd.c`.

References **ActiveLayer**, **BSP_LCD_GetXSize()**, and **LtdcHandler**.

Referenced by **BSP_LCD_DrawCircle()**, **BSP_LCD_DrawEllipse()**, **BSP_LCD_DrawLine()**, and **DrawChar()**.

```c
void BSP_LCD_DrawPolygon ( pPoint Points, uint16_t PointCount )
```

Displays an poly-line (between many points).

**Parameters:**

- **Points**: pointer to the points array
- **PointCount**: Number of points

Definition at line 746 of file `stm32f429i_discovery_lcd.c`.

References **BSP_LCD_DrawLine()**, **Point::X**, and **Point::Y**.

```c
void BSP_LCD_DrawRect ( uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height )
```

Displays a rectangle.
Parameters:
- **Xpos:** the X position
- **Ypos:** the Y position
- **Height:** display rectangle height
- **Width:** display rectangle width

Definition at line 690 of file `stm32f429i_discovery_lcd.c`.

References `BSP_LCD_DrawHLine()`, and `BSP_LCD_DrawVLine()`.

```c
void BSP_LCD_DrawVLine ( uint16_t Xpos,
                         uint16_t Ypos,
                         uint16_t Length )
```

Displays a vertical line.

Parameters:
- **Xpos:** the X position
- **Ypos:** the Y position
- **Length:** line length

Definition at line 598 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, `BSP_LCD_GetXSize()`, `FillBuffer()`, and `LtdcHandler`.

Referenced by `BSP_LCD_DrawRect()`.

```c
void BSP_LCD_FillCircle ( uint16_t Xpos,
                          uint16_t Ypos,
                          uint16_t Radius )
```
Displays a full circle.

**Parameters:**
- **Xpos:** the X position
- **Ypos:** the Y position
- **Radius:** the circle radius

Definition at line 885 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, `BSP_LCD_DrawCircle()`, `BSP_LCD_DrawHLine()`, and `BSP_LCD_SetTextColor()`.

```c
void BSP_LCD_FillEllipse ( int Xpos,
                           int Ypos,
                           int XRRadius,
                           int YRadius
                     )
```

Draw a full ellipse.

**Parameters:**
- **Xpos:** the X position
- **Ypos:** the Y position
- **XRadius:** X radius of ellipse
- **YRadius:** Y radius of ellipse.

Definition at line 1077 of file `stm32f429i_discovery_lcd.c`.

References `BSP_LCD_DrawHLine()`.

```c
void BSP_LCD_FillPolygon ( pPoint Points,
                           uint16_t PointCount
                     )
```
Displays a full poly-line (between many points).

**Parameters:**
- **Points,:** pointer to the points array
- **PointCount,:** Number of points

Definition at line 1009 of file `stm32f429i_discovery_lcd.c`.

References `BSP_LCD_FillTriangle()`, `POLY_X`, `POLY_Y`, `Point::X`, and `Point::Y`.

```c
void BSP_LCD_FillRect ( uint16_t Xpos,
                       uint16_t Ypos,
                       uint16_t Width,
                       uint16_t Height )
```

Displays a full rectangle.

**Parameters:**
- **Xpos,:** the X position
- **Ypos,:** the Y position
- **Height,:** rectangle height
- **Width,:** rectangle width

Definition at line 865 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, `BSP_LCD_GetXSize()`, `BSP_LCD_SetTextColor()`, `FillBuffer()`, and `LtdcHandler`.

Referenced by `BSP_LCD_ClearStringLine()`.

```c
void BSP_LCD_FillTriangle ( uint16_t X1,
                           uint16_t X2,
                           ... )
```
uint16_t X3,
uint16_t Y1,
uint16_t Y2,
uint16_t Y3
)

Fill triangle.

**Parameters:**
- \(X_1,:\) the point 1 x position
- \(Y_1,:\) the point 1 y position
- \(X_2,:\) the point 2 x position
- \(Y_2,:\) the point 2 y position
- \(X_3,:\) the point 3 x position
- \(Y_3,:\) the point 3 y position

Definition at line 936 of file `stm32f429i_discovery_lcd.c`.

References ABS, and BSP_LCD_DrawLine().

Referenced by BSP_LCD_FillPolygon().

```c
uint32_t BSP_LCD_GetBackColor ( void )
```

Gets the LCD Background color.

**Return values:**
- **Background** color

Definition at line 390 of file `stm32f429i_discovery_lcd.c`.

References ActiveLayer, and LCD_DrawPropTypeDef::BackColor.

```c
sFONT* BSP_LCD_GetFont ( void )
```
Gets the Text Font.

**Return values:**
- **Layer** font

Definition at line 426 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, and `LCD_DrawPropTypeDef::pFont`.

```c
uint32_t BSP_LCD_GetTextColor ( void )
```

Gets the LCD Text color.

**Return values:**
- **Text** color

Definition at line 381 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, and `LCD_DrawPropTypeDef::TextColor`.

```c
uint32_t BSP_LCD_GetXSize ( void )
```

Gets the LCD X size.

**Return values:**
- **The** used LCD X size

Definition at line 239 of file `stm32f429i_discovery_lcd.c`.

References `LcdDrv`.

uint32_t BSP_LCD_GetYSize ( void )

Gets the LCD Y size.

Return values:
The used LCD Y size

Definition at line 248 of file stm32f429i_discovery_lcd.c.

References LcdDrv.

Referenced by BSP_LCD_Clear(), and
BSP_LCD_LayerDefaultInit().

uint8_t BSP_LCD_Init ( void )

Initializes the LCD.

Return values:
LCD state

Definition at line 156 of file stm32f429i_discovery_lcd.c.

References BSP_LCD_SetFont(), BSP_SDRAM_Init(),
LCD_DEFAULT_FONT, LCD_OK, LcdDrv, LtdcHandler, MspInit(),
andPeriphClkInitStruct.

void BSP_LCD_LayerDefaultInit ( uint16_t LayerIndex,
                                  uint32_t FB_Address
                                )

Initializes the LCD layers.

Parameters:
LayerIndex,:

- the layer foreground or background.

FB_Address,:

- the layer frame buffer.

Definition at line 258 of file stm32f429i_discovery_lcd.c.

References LCD_DrawPropTypeDef::BackColor,
BSP_LCD_GetXSize(), BSP_LCD_GetYSize(),
LCD_COLOR_BLACK, LCD_COLOR_WHITE,
LCD_LayerCfgTypeDef, LtdcHandler,
LCD_DrawPropTypeDef::pFont, and
LCD_DrawPropTypeDef::TextColor.

```
uint32_t BSP_LCD_ReadPixel ( uint16_t Xpos,
                           uint16_t Ypos )
```

Reads Pixel.

**Parameters:**

- **Xpos,**: the X position
- **Ypos,**: the Y position

**Return values:**

- **RGB** pixel color

Definition at line 437 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, BSP_LCD_GetXSize(), and LtdcHandler.

```
void BSP_LCD_ResetColorKeying ( uint32_t LayerIndex )
```

Disables the color Keying.

**Parameters:**
LayerIndex,: the Layer foreground or background

Definition at line 371 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.

```c
void BSP_LCD_SelectLayer ( uint32_t LayerIndex )
```

Selects the LCD Layer.

**Parameters:**

- **LayerIndex,:** the Layer foreground or background.

Definition at line 293 of file stm32f429i_discovery_lcd.c.

References ActiveLayer.

```c
void BSP_LCD_SetBackColor ( uint32_t Color )
```

Sets the Background color.

**Parameters:**

- **Color,:** the layer Background color code ARGB(8-8-8-8)

Definition at line 408 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, and LCD_DrawPropTypeDef::BackColor.

```c
void BSP_LCD_SetColorKeying ( uint32_t LayerIndex, uint32_t RGBValue )
```

Configures and sets the color Keying.
Parameters:

LayerIndex,: the Layer foreground or background
RGBValue,: the Color reference

Definition at line 360 of file stm32f429i_discovery_lcd.c.

ReferencesLtdcHandler.

void BSP_LCD_SetFont ( sFONT * pFonts )

Sets the Text Font.

Parameters:

pFonts,: the layer font to be used

Definition at line 417 of file stm32f429i_discovery_lcd.c.

ReferencesActiveLayer, and LCD_DrawPropTypeDef::pFont.

Referenced by BSP_LCD_Init().

void BSP_LCD_SetLayerAddress ( uint32_t LayerIndex, uint32_t Address )

Sets a LCD layer frame buffer address.

Parameters:

LayerIndex,: specifies the Layer foreground or background
Address,: new LCD frame buffer value

Definition at line 333 of file stm32f429i_discovery_lcd.c.

ReferencesLtdcHandler.
void BSP_LCD_SetLayerVisible (uint32_t LayerIndex, FunctionalState state)

Sets a LCD Layer visible.

**Parameters:**
- **LayerIndex:** the visible Layer.
- **state:** new state of the specified layer. This parameter can be: ENABLE or DISABLE.

Definition at line 304 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.

---

void BSP_LCD_SetLayerWindow (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)

Sets the Display window.

**Parameters:**
- **LayerIndex:** layer index
- **Xpos:** LCD X position
- **Ypos:** LCD Y position
- **Width:** LCD window width
- **Height:** LCD window height

Definition at line 346 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.
void BSP_LCD_SetTextColor ( uint32_t Color )
Sets the Text color.

**Parameters:**
- **Color:** the Text color code ARGB(8-8-8-8)

Definition at line 399 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, and LCD_DrawPropTypeDef::TextColor.

Referenced by BSP_LCD_ClearStringLine(), BSP_LCD_FillCircle(), and BSP_LCD_FillRect().

void BSP_LCD_SetTransparency ( uint32_t LayerIndex,
                             uint8_t Transparency )
Configures the Transparency.

**Parameters:**
- **LayerIndex:** the Layer foreground or background.
- **Transparency:** the Transparency, This parameter must range from 0x00 to 0xFF.

Definition at line 323 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.

static void ConvertLineToARGB8888 ( void * pSrc,
                                    void * pDst,
                                    uint32_t xSize,
                                    uint32_t ColorMode)
Converts Line to ARGB8888 pixel format.

**Parameters:**
- `pSrc`: pointer to source buffer
- `pDst`: output color
- `xSize`: buffer width
- `ColorMode`: input color mode

Definition at line 1315 of file `stm32f429i_discovery_lcd.c`.

References Dma2dHandler.

Referenced by `BSP_LCD_DrawBitmap()`.

```c
static void DrawChar ( uint16_t Xpos, uint16_t Ypos, const uint8_t * c )
```

Draws a character on LCD.

**Parameters:**
- `Xpos`: the Line where to display the character shape
- `Ypos`: start column address
- `c`: pointer to the character data

Definition at line 1227 of file `stm32f429i_discovery_lcd.c`.

References ActiveLayer, `BSP_LCD_DrawPixel()`, and `LCD_DrawlPropTypeDef::pFont`.

Referenced by `BSP_LCD_DisplayChar()`.
static void FillBuffer ( uint32_t LayerIndex, 
  void * pDst, 
  uint32_t xSize, 
  uint32_t ySize, 
  uint32_t OffLine, 
  uint32_t ColorIndex )

Fills buffer.

**Parameters:**
- **LayerIndex:** layer index
- **pDst:** output color
- **xSize:** buffer width
- **ySize:** buffer height
- **OffLine:** offset
- **ColorIndex:** color Index

Definition at line 1284 of file stm32f429i_discovery_lcd.c.

References **Dma2dHandler**.

Referenced by **BSP_LCD_Clear()**, **BSP_LCD_DrawHLine()**, **BSP_LCD_DrawVLine()**, and **BSP_LCD_FillRect()**.

# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LCD Exported Functions**

STM32F429I DISCOVERY LCD
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>BSP_LCD_Init</code></td>
<td>Initializes the LCD.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetXSize</code></td>
<td>Gets the LCD X size.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetYSize</code></td>
<td>Gets the LCD Y size.</td>
</tr>
<tr>
<td><code>BSP_LCD_LayerDefaultInit</code></td>
<td>Initializes the LCD layers.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetTransparency</code></td>
<td>Configures the Transparency.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetLayerAddress</code></td>
<td>Sets a LCD layer frame buffer address.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetColorKeying</code></td>
<td>Configures and sets the color Keying.</td>
</tr>
<tr>
<td><code>BSP_LCD_ResetColorKeying</code></td>
<td>Disables the color Keying.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetLayerWindow</code></td>
<td>Sets the Display window.</td>
</tr>
<tr>
<td><code>BSP_LCD_SelectLayer</code></td>
<td>Selects the LCD Layer.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetLayerVisible</code></td>
<td>Sets a LCD Layer visible.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetTextColor</code></td>
<td>Sets the Text color.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>BSP_LCD_SetBackColor(uint32_t Color)</code></td>
<td>Sets the background color.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetTextColor(void)</code></td>
<td>Gets the LCD text color.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetBackColor(void)</code></td>
<td>Gets the LCD background color.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetFont(sFONT *pFonts)</code></td>
<td>Sets the text font.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetFont(void)</code></td>
<td>Gets the text font.</td>
</tr>
<tr>
<td><code>BSP_LCD_ReadPixel(uint16_t Xpos, uint16_t Ypos)</code></td>
<td>Reads pixel.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawPixel(uint16_t Xpos, uint16_t Ypos, uint32_t pixel)</code></td>
<td>Writes pixel.</td>
</tr>
<tr>
<td><code>BSP_LCD_Clear(uint32_t Color)</code></td>
<td>Clears the whole LCD.</td>
</tr>
<tr>
<td><code>BSP_LCD_ClearStringLine(uint32_t Line)</code></td>
<td>Clears the selected line.</td>
</tr>
<tr>
<td><code>BSP_LCD_DisplayStringAtLine(uint16_t Line, uint8_t *ptr)</code></td>
<td>Displays a maximum of 20 char on the LCD.</td>
</tr>
<tr>
<td><code>BSP_LCD_DisplayStringAt(uint16_t X, uint16_t Y, uint8_t *pText, Text_AlignModeTypdef mode)</code></td>
<td>Displays a maximum of 60 char on the LCD.</td>
</tr>
<tr>
<td><code>BSP_LCD_DisplayChar(uint16_t Xpos, uint16_t Ypos, uint8_t Ascii)</code></td>
<td>Displays one character.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawHLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length)</code></td>
<td>Displays an horizontal line.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawVLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length)</code></td>
<td>Displays a vertical line.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DrawLine</strong> (uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2)</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Displays an uni-line (between two points).</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DrawRect</strong> (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</td>
</tr>
<tr>
<td></td>
<td>Displays a rectangle.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DrawCircle</strong> (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</td>
</tr>
<tr>
<td></td>
<td>Displays a circle.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DrawPolygon</strong> (pPoint Points, uint16_t PointCount)</td>
</tr>
<tr>
<td></td>
<td>Displays an poly-line (between many points).</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DrawEllipse</strong> (int Xpos, int Ypos, int XRadius, int YRadius)</td>
</tr>
<tr>
<td></td>
<td>Displays an Ellipse.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DrawBitmap</strong> (uint32_t X, uint32_t Y, uint8_t *pBmp)</td>
</tr>
<tr>
<td></td>
<td>Displays a bitmap picture loaded in the internal Flash (32 bpp).</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_FillRect</strong> (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</td>
</tr>
<tr>
<td></td>
<td>Displays a full rectangle.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_FillCircle</strong> (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</td>
</tr>
<tr>
<td></td>
<td>Displays a full circle.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_FillTriangle</strong> (uint16_t X1, uint16_t X2, uint16_t X3, uint16_t Y1, uint16_t Y2, uint16_t Y3)</td>
</tr>
<tr>
<td></td>
<td>Fill triangle.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_FillPolygon</strong> (pPoint Points, uint16_t PointCount)</td>
</tr>
<tr>
<td></td>
<td>Displays a full poly-line (between many points).</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_FillEllipse</strong> (int Xpos, int Ypos, int XRadius, int YRadius)</td>
</tr>
<tr>
<td></td>
<td>Draw a full ellipse.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DisplayOff</strong> (void)</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td></td>
<td>Disables the Display.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>BSP_LCD_DisplayOn</strong> (void)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables the Display.</td>
</tr>
</tbody>
</table>
Function Documentation

void **BSP_LCD_Clear** ( uint32_t **Color** )

Clears the hole LCD.

**Parameters:**

- **Color:** the color of the background

Definition at line 471 of file *stm32f429i_discovery_lcd.c*.

References **ActiveLayer**, **BSP_LCD_GetXSize()**, **BSP_LCD_GetYSize()**, **FillBuffer()**, and **LtdcHandler**.

void **BSP_LCD_ClearStringLine** ( uint32_t **Line** )

Clears the selected line.

**Parameters:**

- **Line:** the line to be cleared

Definition at line 481 of file *stm32f429i_discovery_lcd.c*.

References **ActiveLayer**, **LCD_DrawPropTypeDef::BackColor**, **BSP_LCD_FillRect()**, **BSP_LCD_GetXSize()**, **BSP_LCD_SetTextColor()**, and **LCD_DrawPropTypeDef::TextColor**.

void **BSP_LCD_DisplayChar** ( uint16_t **Xpos**, uint16_t **Ypos**, uint8_t **Ascii** )
Displays one character.

**Parameters:**
- **Xpos,** start column address
- **Ypos,** the Line where to display the character shape
- **Ascii,** character ascii code, must be between 0x20 and 0x7E

Definition at line 499 of file *stm32f429i_discovery_lcd.c.*

References *ActiveLayer*, *DrawChar()*, and *LCD_DrawPropTypeDef::pFont*.

Referenced by *BSP_LCD_DisplayStringAt()*.

```c
void BSP_LCD_DisplayOff ( void )
```

Disables the Display.

Definition at line 1116 of file *stm32f429i_discovery_lcd.c.*

References *LcdDrv*.

```c
void BSP_LCD_DisplayOn ( void )
```

Enables the Display.

Definition at line 1105 of file *stm32f429i_discovery_lcd.c.*

References *LcdDrv*.

```c
void BSP_LCD_DisplayStringAt ( uint16_t X, uint16_t Y, uint8_t * pText,
```
Text_AlignModeTypeDef mode

Displays a maximum of 60 char on the LCD.

**Parameters:**

\( X, \): pointer to x position (in pixel)

\( Y, \): pointer to y position (in pixel)

\( pText, \): pointer to string to display on LCD

**mode,:** The display mode This parameter can be one of the following values:

- CENTER_MODE
- RIGHT_MODE
- LEFT_MODE

Definition at line 516 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, BSP_LCD_DisplayChar(), BSP_LCD_GetXSize(), CENTER_MODE, LEFT_MODE, LCD_DrawPropTypeDef::pFont, and RIGHT_MODE.

Referenced by BSP_LCD_DisplayStringAtLine().

```c
void BSP_LCD_DisplayStringAtLine ( uint16_t Line,
uint8_t * ptr
)
```

Displays a maximum of 20 char on the LCD.

**Parameters:**

**Line,**: the Line where to display the character shape

**ptr,**: pointer to string to display on LCD

Definition at line 570 of file stm32f429i_discovery_lcd.c.
void BSP_LCD_DrawBitmap ( uint32_t X,
            uint32_t Y,
            uint8_t * pBmp
        )

Displays a bitmap picture loaded in the internal Flash (32 bpp).

Parameters:
  X,:    the bmp x position in the LCD
  Y,:    the bmp Y position in the LCD
  pBmp,: Bmp picture address in the internal Flash

Definition at line 805 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, BSP_LCD_GetXSize(), ConvertLineToARGB8888(), and LtdcHandler.

void BSP_LCD_DrawCircle ( uint16_t Xpos,
                      uint16_t Ypos,
                      uint16_t Radius
                  )

Displays a circle.

Parameters:
  Xpos,:    the X position
  Ypos,:    the Y position
  Radius,:  the circle radius

Definition at line 707 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, and BSP_LCD_DrawPixel().
void **BSP_LCD_D**raw**E**llipse ( int Xpos,
   int Ypos,
   int XRadius,
   int YRadius
 )

Displays an Ellipse.

**Parameters:**

- **Xpos,** the X position
- **Ypos,** the Y position
- **XRadius,** the X radius of ellipse
- **YRadius,** the Y radius of ellipse

Definition at line 773 of file *stm32f429i_discovery_lcd.c*.

References **ActiveLayer**, and **BSP_LCD_DrawPixel()**.

void **BSP_LCD_D**raw**H**line ( uint16_t Xpos,
   uint16_t Ypos,
   uint16_t Length
 )

Displays an horizontal line.

**Parameters:**

- **Xpos,** the X position
- **Ypos,** the Y position
- **Length,** line length

Definition at line 581 of file *stm32f429i_discovery_lcd.c*.
References *ActiveLayer*, *BSP_LCD_GetXSize()*, *FillBuffer()*, and *LtdcHandler*.

Referenced by *BSP_LCD_DrawRect()*, *BSP_LCD_FillCircle()*, and *BSP_LCD_FillEllipse()*. 

```c
void BSP_LCD_DrawLine ( uint16_t X1,
                          uint16_t Y1,
                          uint16_t X2,
                          uint16_t Y2
                    )
```

Displays an uni-line (between two points).

**Parameters:**
- **X1**: the point 1 X position
- **Y1**: the point 1 Y position
- **X2**: the point 2 X position
- **Y2**: the point 2 Y position

Definition at line 616 of file *stm32f429i_discovery_lcd.c*.

References *ABS*, *ActiveLayer*, and *BSP_LCD_DrawPixel()*.

Referenced by *BSP_LCD_DrawPolygon()*, and *BSP_LCD_FillTriangle()*.

```c
void BSP_LCD_DrawPixel ( uint16_t Xpos,
                             uint16_t Ypos,
                             uint32_t RGB_Code
                      )
```

Writes Pixel.
Parameters:

**Xpos:** the X position

**Ypos:** the Y position

**RGB_Code:** the pixel color in ARGB mode (8-8-8-8)

Definition at line 1215 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, `BSP_LCD_GetXSize()`, and `LtdcHandler`.

Referenced by `BSP_LCD_DrawCircle()`, `BSP_LCD_DrawEllipse()`, `BSP_LCD_DrawLine()`, and `DrawChar()`.

```c
void BSP_LCD_DrawPolygon ( pPoint Points,
                          uint16_t PointCount
                      )
```

Displays an poly-line (between many points).

Parameters:

**Points:** pointer to the points array

**PointCount:** Number of points

Definition at line 746 of file `stm32f429i_discovery_lcd.c`.

References `BSP_LCD_DrawLine()`, `Point::X`, and `Point::Y`.

```c
void BSP_LCD_DrawRect ( uint16_t Xpos,
                        uint16_t Ypos,
                        uint16_t Width,
                        uint16_t Height
                    )
```

Displays a rectangle.
Parameters:

- **Xpos**: the X position
- **Ypos**: the Y position
- **Height**: display rectangle height
- **Width**: display rectangle width

Definition at line 690 of file *stm32f429i_discovery_lcd.c*.

References **BSP_LCD_DrawHLine()**, and **BSP_LCD_DrawVLine()**.

```c
void BSP_LCD_DrawVLine ( uint16_t Xpos,
                        uint16_t Ypos,
                        uint16_t Length )
```

Displays a vertical line.

Parameters:

- **Xpos**: the X position
- **Ypos**: the Y position
- **Length**: line length

Definition at line 598 of file *stm32f429i_discovery_lcd.c*.

References **ActiveLayer**, **BSP_LCD_GetXSize()**, **FillBuffer()**, and **LtdcHandler**.

Referenced by **BSP_LCD_DrawRect()**.

```c
void BSP_LCD_FillCircle ( uint16_t Xpos,
                          uint16_t Ypos,
                          uint16_t Radius )
```
Displays a full circle.

**Parameters:**
- **Xpos,** the X position
- **Ypos,** the Y position
- **Radius,** the circle radius

Definition at line 885 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, `BSP_LCD_DrawCircle()`, `BSP_LCD_DrawHLine()`, and `BSP_LCD_SetTextColor()`.

```c
void BSP_LCD_FillEllipse ( int Xpos,
                        int Ypos,
                        int XRadius,
                        int YRadius
                    )
```

Draw a full ellipse.

**Parameters:**
- **Xpos,** the X position
- **Ypos,** the Y position
- **XRadius,** X radius of ellipse
- **YRadius,** Y radius of ellipse.

Definition at line 1077 of file `stm32f429i_discovery_lcd.c`.

References `BSP_LCD_DrawHLine()`.

```c
void BSP_LCD_FillPolygon ( pPoint Points,
                        uint16_t PointCount
                    )
```
Displays a full poly-line (between many points).

**Parameters:**
- **Points**: pointer to the points array
- **PointCount**: Number of points

Definition at line 1009 of file `stm32f429i_discovery_lcd.c`.

References `BSP_LCD_FillTriangle()`, `POLY_X`, `POLY_Y`, `Point::X`, and `Point::Y`.

```c
void BSP_LCD_FillRect ( uint16_t Xpos,
                         uint16_t Ypos,
                         uint16_t Width,
                         uint16_t Height )
```

Displays a full rectangle.

**Parameters:**
- **Xpos**: the X position
- **Ypos**: the Y position
- **Height**: rectangle height
- **Width**: rectangle width

Definition at line 865 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, `BSP_LCD_GetXSize()`, `BSP_LCD_SetTextColor()`, `FillBuffer()`, and `LtdcHandler`.

Referenced by `BSP_LCD_ClearStringLine()`.

```c
void BSP_LCD_FillTriangle ( uint16_t X1,
                            uint16_t X2,
```
uint16_t X3,
uint16_t Y1,
uint16_t Y2,
uint16_t Y3
)

Fill triangle.

**Parameters:**
- **X1:** the point 1 x position
- **Y1:** the point 1 y position
- **X2:** the point 2 x position
- **Y2:** the point 2 y position
- **X3:** the point 3 x position
- **Y3:** the point 3 y position

Definition at line 936 of file stm32f429i_discovery_lcd.c.

References ABS, and BSP_LCD_DrawLine().

Referenced by BSP_LCD_FillPolygon().

```c
uint32_t BSP_LCD_GetBackColor ( void )
```

Gets the LCD Background color.

**Return values:**
- **Background** color

Definition at line 390 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, and LCD_DrawPropTypeDef::BackColor.

```c
sFONT* BSP_LCD_GetFont ( void )
```
Gets the Text Font.

**Return values:**

Layer font

Definition at line 426 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, and `LCD_DrawPropTypeDef::pFont`.

```c
uint32_t BSP_LCD_GetTextColor ( void )
```

Gets the LCD Text color.

**Return values:**

Text color

Definition at line 381 of file `stm32f429i_discovery_lcd.c`.

References `ActiveLayer`, and `LCD_DrawPropTypeDef::TextColor`.

```c
uint32_t BSP_LCD_GetXSize ( void )
```

Gets the LCD X size.

**Return values:**

The used LCD X size

Definition at line 239 of file `stm32f429i_discovery_lcd.c`.

References `LcdDrv`.

**uint32_t** `BSP_LCD_GetYSize ( void )`

Gets the LCD Y size.

**Return values:**
- The used LCD Y size

Definition at line 248 of file `stm32f429i_discovery_lcd.c`.

References **LcdDrv**.

Referenced by `BSP_LCD_Clear()`, and `BSP_LCD_LayerDefaultInit()`.

---

**uint8_t** `BSP_LCD_Init ( void )`

Initializes the LCD.

**Return values:**
- **LCD** state

Definition at line 156 of file `stm32f429i_discovery_lcd.c`.

References `BSP_LCD_SetFont()`, `BSP_SDRAM_Init()`, `LCD_DEFAULT_FONT`, `LCD_OK`, `LcdDrv`, `LtdcHandler`, `MspInit()`, and `PeriphClkInitStruct`.

---

**void** `BSP_LCD_LayerDefaultInit ( uint16_t LayerIndex, uint32_t FB_Address )`

Initializes the LCD layers.

**Parameters:**
LayerIndex: the layer foreground or background.

FB_Address: the layer frame buffer.

Definition at line 258 of file stm32f429i_discovery_lcd.c.

References LCD_DrawPropTypeDef::BackColor, BSP_LCD_GetXSize(), BSP_LCD_GetYSize(), LCD_COLOR_BLACK, LCD_COLOR_WHITE, LCD_LayerCfgTypeDef, LtdcHandler, LCD_DrawPropTypeDef::pFont, and LCD_DrawPropTypeDef::TextColor.

```c
uint32_t BSP_LCD_ReadPixel ( uint16_t Xpos, uint16_t Ypos )
```

Reads Pixel.

**Parameters:**

Xpos: the X position

Ypos: the Y position

**Return values:**

RGB pixel color

Definition at line 437 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, BSP_LCD_GetXSize(), and LtdcHandler.

```c
void BSP_LCD_ResetColorKeying ( uint32_t LayerIndex )
```

Disables the color Keying.

**Parameters:**
LayerIndex, : the Layer foreground or background

Definition at line 371 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.

```c
void BSP_LCD_SelectLayer ( uint32_t LayerIndex )
```

Selects the LCD Layer.

**Parameters:**

- **LayerIndex, :** the Layer foreground or background.

Definition at line 293 of file stm32f429i_discovery_lcd.c.

References ActiveLayer.

```c
void BSP_LCD_SetBackColor ( uint32_t Color )
```

Sets the Background color.

**Parameters:**

- **Color, :** the layer Background color code ARGB(8-8-8-8)

Definition at line 408 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, and LCD_DrawPropTypeDef::BackColor.

```c
void BSP_LCD_SetColorKeying ( uint32_t LayerIndex,
                                uint32_t RGBValue
)
```

Configures and sets the color Keying.
**Parameters:**
- **LayerIndex,**: the Layer foreground or background
- **RGBValue,**: the Color reference

Definition at line 360 of file *stm32f429i_discovery_lcd.c.*

References *LtdcHandler.*

```c
void BSP_LCD_SetFont ( sFONT * pFonts )
```

Sets the Text Font.

**Parameters:**
- **pFonts,**: the layer font to be used

Definition at line 417 of file *stm32f429i_discovery_lcd.c.*

References *ActiveLayer,* and *LCD_DrawPropTypeDef::pFont.*

Referenced by *BSP_LCD_Init().*

```c
void BSP_LCD_SetLayerAddress ( uint32_t LayerIndex,
                               uint32_t Address
                               )
```

Sets a LCD layer frame buffer address.

**Parameters:**
- **LayerIndex,**: specifies the Layer foreground or background
- **Address,**: new LCD frame buffer value

Definition at line 333 of file *stm32f429i_discovery_lcd.c.*

References *LtdcHandler.*
void **BSP_LCD_SetLayerVisible** (uint32_t **LayerIndex**, FunctionalState **state**)

Sets a LCD Layer visible.

**Parameters:**
- **LayerIndex:** the visible Layer.
- **state:** new state of the specified layer. This parameter can be: ENABLE or DISABLE.

Definition at line 304 of file stm32f429i_discovery_lcd.c.

References **LtdcHandler**.

void **BSP_LCD_SetLayerWindow** (uint16_t **LayerIndex**, uint16_t **Xpos**, uint16_t **Ypos**, uint16_t **Width**, uint16_t **Height**)

Sets the Display window.

**Parameters:**
- **LayerIndex:** layer index
- **Xpos:** LCD X position
- **Ypos:** LCD Y position
- **Width:** LCD window width
- **Height:** LCD window height

Definition at line 346 of file stm32f429i_discovery_lcd.c.

References **LtdcHandler**.
void **BSP_LCD_SetTextColor** ( uint32_t **Color** )

Sets the Text color.

**Parameters:**

**Color,**: the Text color code ARGB(8-8-8-8)

Definition at line 399 of file stm32f429i_discovery_lcd.c.

References **ActiveLayer**, and **LCD_DrawPropTypeDef::TextColor**.

Referenced by **BSP_LCD_ClearStringLine()**, **BSP_LCD_FillCircle()**, and **BSP_LCD_FillRect()**.

---

void **BSP_LCD_SetTransparency** ( uint32_t **LayerIndex,**
                                  uint8_t **Transparency** )

Configures the Transparency.

**Parameters:**

**LayerIndex,**: the Layer foreground or background.

**Transparency,**: the Transparency, This parameter must range from 0x00 to 0xFF.

Definition at line 323 of file stm32f429i_discovery_lcd.c.

References **LtdcHandler**.
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY SDRAM Exported Functions

STM32F429I DISCOVERY SDRAM
### Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void</strong> BSP_SDRAM_Init (void)</td>
<td>Initializes the SDRAM device.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_SDRAM_Initialization_sequence (uint32_t RefreshCount)</td>
<td>Programs the SDRAM device.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_SDRAM_ReadData (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Reads an mount of data from the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_SDRAM_ReadData_DMA (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Reads an mount of data from the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_SDRAM_WriteData (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Writes an mount of data to the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_SDRAM_WriteData_DMA (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Writes an mount of data to the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td><strong>HAL_StatusTypeDef</strong> BSP_SDRAM_Sendcmd (FMC_SDRAM_CommandTypeDef *SdramCmd)</td>
<td>Sends command to the SDRAM bank.</td>
</tr>
<tr>
<td><strong>void</strong> BSP_SDRAM_DMA_IRQHandler (void)</td>
<td>Handles SDRAM DMA transfer interrupt request.</td>
</tr>
</tbody>
</table>
Function Documentation

```c
void BSP_SDRAM_DMA_IRQHandler ( void )
```
Handles SDRAM DMA transfer interrupt request.
Definition at line 265 of file stm32f429i_discovery_sdram.c.
References SdramHandle.

```c
void BSP_SDRAM_Init ( void )
```
Initializes the SDRAM device.
Definition at line 100 of file stm32f429i_discovery_sdram.c.
References BSP_SDRAM_Initialization_sequence(), MspInit(), REFRESH_COUNT, SDCLOCK_PERIOD, SDRAM_CAS_LATENCY, SDRAM_MEMORY_WIDTH, SDRAM_READBURST, SdramHandle, and Timing.
Referenced by BSP_LCD_Init().

```c
void BSP_SDRAM_Initialization_sequence ( uint32_t RefreshCount )
```
Programs the SDRAM device.

**Parameters:**

- `RefreshCount`: SDRAM refresh counter value

Definition at line 149 of file stm32f429i_discovery_sdram.c.
References Command, SDRAM_MODEREG_BURST_LENGTH_1, SDRAM_MODEREG_BURST_TYPESEQUENTIAL, SDRAM_MODEREG_CAS_LATENCY_3,
SDRAM_MODEREG_OPERATING_MODE_STANDARD, SDRAM_MODEREG_WRITEBURST_MODE_SINGLE, SDRAM_TIMEOUT, and SdramHandle.

Referenced by BSP_SDRAM_Init().

```c
void BSP_SDRAM_ReadData ( uint32_t uwStartAddress, uint32_t * pData, uint32_t uwDataSize )
```

Reads an amount of data from the SDRAM memory in polling mode.

**Parameters:**
- **uwStartAddress**: Read start address
- **pData**: Pointer to data to be read
- **uwDataSize**: Size of read data from the memory

Definition at line 210 of file stm32f429i_discovery_sdram.c.

References SdramHandle.

```c
void BSP_SDRAM_ReadData_DMA ( uint32_t uwStartAddress, uint32_t * pData, uint32_t uwDataSize )
```

Reads an amount of data from the SDRAM memory in DMA mode.

**Parameters:**
- **uwStartAddress**: Read start address
- **pData**: Pointer to data to be read
- **uwDataSize**: Size of read data from the memory
Definition at line 221 of file stm32f429i_discovery_sdram.c.

References SdramHandle.

HAL_StatusTypeDef BSP_SDRAM_Sendcmd ( FMC_SDRAM_CommandTypeDef * SdramCmd,:

Sends command to the SDRAM bank.

Parameters:

SdramCmd,: Pointer to SDRAM command structure

Return values:

HAL status

Definition at line 257 of file stm32f429i_discovery_sdram.c.

References SDRAM_TIMEOUT, and SdramHandle.

void BSP_SDRAM_WriteData ( uint32_t uwStartAddress, uint32_t * pData, uint32_t uwDataSize )

Writes an mount of data to the SDRAM memory in polling mode.

Parameters:

uwStartAddress : Write start address
pData : Pointer to data to be written
uwDataSize,: Size of written data from the memory

Definition at line 232 of file stm32f429i_discovery_sdram.c.

References SdramHandle.
void BSP_SDRAM_WriteData_DMA ( uint32_t uwStartAddress, uint32_t * pData, uint32_t uwDataSize )

Writes an mount of data to the SDRAM memory in DMA mode.

Parameters:

uwStartAddress : Write start address
pData : Pointer to data to be written
uwDataSize : Size of written data from the memory

Definition at line 247 of file stm32f429i_discovery_sdram.c.

References SdramHandle.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td><strong>STM32F429I DISCOVERY SDRAM</strong></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY SDRAM Private Functions**

STM32F429I DISCOVERY SDRAM
### Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_Init</strong> (void)</td>
<td>Initializes the SDRAM device.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_Initialization_sequence</strong> (uint32_t RefreshCount)</td>
<td>Programs the SDRAM device.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_ReadData</strong> (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Reads a mount of data from the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_ReadData_DMA</strong> (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Reads a mount of data from the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_WriteData</strong> (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Writes a mount of data to the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_WriteData_DMA</strong> (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</td>
<td>Writes a mount of data to the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td>HAL_StatusTypeDef</td>
<td><strong>BSP_SDRAM_Sendcmd</strong> (FMC_SDRAM_CommandTypeDef *SdramCmd)</td>
<td>Sends command to the SDRAM bank.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_SDRAM_DMA_IRQHandler</strong> (void)</td>
<td>Handles SDRAM DMA transfer interrupt request.</td>
</tr>
</tbody>
</table>
Function Documentation

**void** BSP_SDRAM_DMA_IRQHandler ( void )

Handles SDRAM DMA transfer interrupt request.

Definition at line **265** of file stm32f429i_discovery_sdram.c.

References **SdramHandle**.

**void** BSP_SDRAM_Init ( void )

Initializes the SDRAM device.

Definition at line **100** of file stm32f429i_discovery_sdram.c.

References **BSP_SDRAM_Initialization_sequence()**, **MspInit()**, **REFRESH_COUNT**, **SDCLOCK_PERIOD**, **SDRAM_CAS_LATENCY**, **SDRAM_MEMORY_WIDTH**, **SDRAM_READBURST**, **SdramHandle**, and **Timing**.

Referenced by **BSP_LCD_Init()**.

**void** BSP_SDRAM_Initialization_sequence ( uint32_t RefreshCount )

Programs the SDRAM device.

**Parameters:**

- **RefreshCount:** SDRAM refresh counter value

Definition at line **149** of file stm32f429i_discovery_sdram.c.

References **Command**, **SDRAM_MODEREG_BURST_LENGTH_1**, **SDRAM_MODEREG_BURST_TYPE_SEQUENTIAL**, **SDRAM_MODEREG_CAS_LATENCY_3**, **...**
SDRAM_MODEREG_OPERATING_MODE_STANDARD,  
SDRAM_MODEREG_WRITEBURST_MODE_SINGLE,  
SDRAM_TIMEOUT, and SdramHandle.

Referenced by **BSP_SDRAM_Init()**.

```c
void BSP_SDRAM_ReadData ( uint32_t uwStartAddress,  
        uint32_t * pData,  
        uint32_t uwDataSize
 )
```

Reads an mount of data from the SDRAM memory in polling mode.

**Parameters:**
- **uwStartAddress**: Read start address
- **pData**: Pointer to data to be read
- **uwDataSize**: Size of read data from the memory

Definition at line **210** of file **stm32f429i_discovery_sdram.c**.

References **SdramHandle**.

```c
void BSP_SDRAM_ReadData_DMA ( uint32_t uwStartAddress,  
        uint32_t * pData,  
        uint32_t uwDataSize
 )
```

Reads an mount of data from the SDRAM memory in DMA mode.

**Parameters:**
- **uwStartAddress**: Read start address
- **pData**: Pointer to data to be read
- **uwDataSize**: Size of read data from the memory
Definition at line 221 of file stm32f429i_discovery_sdram.c.
References SdramHandle.

HAL_StatusTypeDef BSP_SDRAM_Sendcmd ( FMC_SDRAM_CommandTypeDef * SdramCmd)

Sends command to the SDRAM bank.

Parameters:
  SdramCmd,: Pointer to SDRAM command structure

Return values:
  HAL status

Definition at line 257 of file stm32f429i_discovery_sdram.c.
References SDRAM_TIMEOUT, and SdramHandle.

void BSP_SDRAM_WriteData ( uint32_t uwStartAddress, uint32_t * pData, uint32_t uwDataSize )

Writes an amount of data to the SDRAM memory in polling mode.

Parameters:
  uwStartAddress : Write start address
  pData : Pointer to data to be written
  uwDataSize,: Size of written data from the memory

Definition at line 232 of file stm32f429i_discovery_sdram.c.
References SdramHandle.
void BSP_SDRAM_WriteData_DMA ( uint32_t uwStartAddress,  
   uint32_t * pData,  
   uint32_t uwDataSize  
  )  

Writes an amount of data to the SDRAM memory in DMA mode.

**Parameters:**

- **uwStartAddress**: Write start address
- **pData**: Pointer to data to be written
- **uwDataSize**: Size of written data from the memory

Definition at line 247 of file **stm32f429i_discovery_sdram.c**.

References **SdramHandle**.
## STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STM32F429I DISCOVERY TS Exported Functions</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY TS**
### Functions

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Function Name</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><strong>BSP_TS_Init</strong> (uint16_t XSize, uint16_t YSize)</td>
<td></td>
<td>Initializes and configures the touch screen functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_TS_GetState</strong> (TS_StateTypeDef *TsState)</td>
<td></td>
<td>Returns status and positions of the touch screen.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><strong>BSP_TS_ITConfig</strong> (void)</td>
<td></td>
<td>Configures and enables the touch screen interrupts.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><strong>BSP_TS_ITGetStatus</strong> (void)</td>
<td></td>
<td>Gets the TS IT status.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_TS_ITClear</strong> (void)</td>
<td></td>
<td>Clears all touch screen interrupts.</td>
</tr>
</tbody>
</table>
Function Documentation

```c
void BSP_TS_GetState ( TS_StateTypeDef * TsState )
```

Returns status and positions of the touch screen.

**Parameters:**
- **TsState:** Pointer to touch screen current state structure

Definition at line 157 of file stm32f429i_discovery_ts.c.

References **TS_StateTypeDef::TouchDetected**, **TS_I2C_ADDRESS**, **TsDrv**, **TsXBoundary**, **TsYBoundary**, **TS_StateTypeDef::X**, and **TS_StateTypeDef::Y**.

```c
uint8_t BSP_TS_Init ( uint16_t XSize, uint16_t YSize )
```

Initializes and configures the touch screen functionalities and configures all necessary hardware resources (GPIOs, clocks..).

**Parameters:**
- **XSize:** The maximum X size of the TS area on LCD
- **YSize:** The maximum Y size of the TS area on LCD

**Return values:**
- **TS_OK:** if all initializations are OK. Other value if error.

Definition at line 104 of file stm32f429i_discovery_ts.c.

References **TS_ERROR**, **TS_I2C_ADDRESS**, **TS_OK**, **TsDrv**, **TsXBoundary**, and **TsYBoundary**.
void BSP_TS_ITClear (void)

Clears all touch screen interrupts.

Definition at line 233 of file stm32f429i_discovery_ts.c.

References TS_I2C_ADDRESS, and TsDrv.

uint8_t BSP_TS_ITConfig (void)

Configures and enables the touch screen interrupts.

Return values:

TS_OK,: if ITconfig is OK. Other value if error.

Definition at line 135 of file stm32f429i_discovery_ts.c.

References TS_I2C_ADDRESS, TS_OK, and TsDrv.

uint8_t BSP_TS_ITGetStatus (void)

Gets the TS IT status.

Return values:

Interrupt status.

Definition at line 147 of file stm32f429i_discovery_ts.c.

References TS_I2C_ADDRESS, and TsDrv.
### STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY TS Private Functions**

STM32F429I DISCOVERY TS
## Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_TS_Init</code> (uint16_t XSize, uint16_t YSize)</td>
<td>Initializes and configures the touch screen functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_TS_ITConfig</code> (void)</td>
<td>Configures and enables the touch screen interrupts.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_TS_ITGetStatus</code> (void)</td>
<td>Gets the TS IT status.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_TS_GetState</code> (TS_StateTypeDef *TsState)</td>
<td>Returns status and positions of the touch screen.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_TS_ITClear</code> (void)</td>
<td>Clears all touch screen interrupts.</td>
</tr>
</tbody>
</table>
Function Documentation

void BSP_TS_GetState ( TS_StateTypeDef * TsState )

Returns status and positions of the touch screen.

Parameters:
    TsState,: Pointer to touch screen current state structure

Definition at line 157 of file stm32f429i_discovery_ts.c.

References TS_StateTypeDef::TouchDetected, TS_I2C_ADDRESS, TsDrv, TsXBoundary, TsYBoundary, TS_StateTypeDef::X, and TS_StateTypeDef::Y.

uint8_t BSP_TS_Init ( uint16_t XSize,
                      uint16_t YSize
                   )

Initializes and configures the touch screen functionalities and configures all necessary hardware resources (GPIOs, clocks..).

Parameters:
    XSize,: The maximum X size of the TS area on LCD
    YSize,: The maximum Y size of the TS area on LCD

Return values:
    TS_OK,: if all initializations are OK. Other value if error.

Definition at line 104 of file stm32f429i_discovery_ts.c.

References TS_ERROR, TS_I2C_ADDRESS, TS_OK, TsDrv, TsXBoundary, and TsYBoundary.
void BSP_TS_ITClear (void)

Clears all touch screen interrupts.

Definition at line 233 of file stm32f429i_discovery_ts.c.

References TS_I2C_ADDRESS, and TsDrv.

uint8_t BSP_TS_ITConfig (void)

Configures and enables the touch screen interrupts.

Return values:

    TS_OK,: if ITconfig is OK. Other value if error.

Definition at line 135 of file stm32f429i_discovery_ts.c.

References TS_I2C_ADDRESS, TS_OK, and TsDrv.

uint8_t BSP_TS_ITGetStatus (void)

Gets the TS IT status.

Return values:

    Interrupt status.

Definition at line 147 of file stm32f429i_discovery_ts.c.

References TS_I2C_ADDRESS, and TsDrv.
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## STM32F429I DISCOVERY LCD Exported Constants

STM32F429I DISCOVERY LCD
Defines

#define LCD_LayerCfgTypeDef  LTDC_LayerCfgTypeDef
#define MAX_LAYER_NUMBER  2
LCD status structure definition.
#define LCD_FRAME_BUFFER  ((uint32_t)0xD0000000)
#define BUFFER_OFFSET  ((uint32_t)0x50000)
#define LCD_COLOR_BLUE  0xFF0000FF
LCD color.
#define LCD_COLOR_GREEN  0xFF00FF00
#define LCD_COLOR_RED  0xFFFF0000
#define LCD_COLOR_CYAN  0xFF00FFFF
#define LCD_COLOR_MAGENTA  0xFFFF00FF
#define LCD_COLOR_YELLOW  0xFFFFFF00
#define LCD_COLOR_LIGHTBLUE  0xFF8080FF
#define LCD_COLOR_LIGHTGREEN  0xFF80FF80
#define LCD_COLOR_LIGHTRED  0xFFFF8080
#define LCD_COLOR_LIGHTCYAN  0xFF80FFFF
#define LCD_COLOR_LIGHTMAGENTA  0xFFFF80FF
#define LCD_COLOR_LIGHTYELLOW  0xFFFFFF80
#define LCD_COLOR_DARKBLUE  0xFF000080
#define LCD_COLOR_DARKGREEN  0xFF008000
#define LCD_COLOR_DARKRED  0xFF800000
#define LCD_COLOR_DARKCYAN  0xFF008080
#define LCD_COLOR_DARKMAGENTA  0xFF800080
#define LCD_COLOR_DARKYELLOW  0xFF808000
#define LCD_COLOR_WHITE  0xFFFFFFFF
#define LCD_COLOR_LIGHTGRAY  0xFFD3D3D3
#define LCD_COLOR_GRAY  0xFF808080
#define LCD_COLOR_DARKGRAY  0xFF404040
#define LCD_COLOR_BLACK  0xFF000000
#define LCD_COLOR_BROWN  0xFFA52A2A
#define LCD_COLOR_ORANGE  0xFFFFA500
<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_COLOR_TRANSPARENT</td>
<td>0xFF000000</td>
</tr>
<tr>
<td>LCD_DEFAULT_FONT</td>
<td>Font24</td>
</tr>
<tr>
<td>LCD_DEFAULT_FONT</td>
<td>Font24</td>
</tr>
<tr>
<td>LCD_BACKGROUND_LAYER</td>
<td>0x0000</td>
</tr>
<tr>
<td>LCD_FOREGROUND_LAYER</td>
<td>0x0001</td>
</tr>
</tbody>
</table>

LCD default font.
#Define Documentation

```c
#define BUFFER_OFFSET ((uint32_t)0x50000)
```
Definition at line 113 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCDBACKGROUND_LAYER 0x0000
```
LCD Layer.
Definition at line 152 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_BLACK 0xFF000000
```
Definition at line 140 of file `stm32f429i_discovery_lcd.h`.
Referenced by `BSP_LCD_LayerDefaultInit()`.

```c
#define LCD_COLOR_BLUE 0xFF0000FF
```
LCD color.
Definition at line 118 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_BROWN 0xFFA52A2A
```
Definition at line 141 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_CYAN 0xFF00FFFF
```
Definition at line 121 of file `stm32f429i_discovery_lcd.h`.
#define LCD_COLOR_DARKBLUE  0xFF000080
Definition at line 130 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_DARKCYAN  0xFF008080
Definition at line 133 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_DARKGRAY  0xFF404040
Definition at line 139 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_DARKGREEN  0xFF008000
Definition at line 131 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_DARKMAGENTA  0xFF800080
Definition at line 134 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_DARKRED  0xFF800000
Definition at line 132 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_DARKYELLOW  0xFF808000
Definition at line 135 of file stm32f429i_discovery_lcd.h.
<table>
<thead>
<tr>
<th>Definition</th>
<th>Value</th>
<th>Line</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>#define LCD_COLOR_GRAY</td>
<td>0xFF808080</td>
<td>138</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>#define LCD_COLOR_GREEN</td>
<td>0xFF00FF00</td>
<td>119</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>#define LCD_COLOR_LIGHTBLUE</td>
<td>0xFF8080FF</td>
<td>124</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>#define LCD_COLOR_LIGHTCYAN</td>
<td>0xFF80FFFF</td>
<td>127</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>#define LCD_COLOR_LIGHTGRAY</td>
<td>0xFFD3D3D3</td>
<td>137</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>#define LCD_COLOR_LIGHTGREEN</td>
<td>0xFF80FF80</td>
<td>125</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>#define LCD_COLOR_LIGHTMAGENTA</td>
<td>0xFFFF80FF</td>
<td>128</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>#define LCD_COLOR_LIGHTRED</td>
<td>0xFFFF8080</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Definition at line 126 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_LIGHTYELLOW 0xFFFF800
```

Definition at line 129 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_MAGENTA 0xFFFF00FF
```

Definition at line 122 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_ORANGE 0xFFFFA500
```

Definition at line 142 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_RED 0xFFFF0000
```

Definition at line 120 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_TRANSPARENT 0xFF000000
```

Definition at line 143 of file `stm32f429i_discovery_lcd.h`.

```c
#define LCD_COLOR_WHITE 0xFFFFFFFF
```

Definition at line 136 of file `stm32f429i_discovery_lcd.h`.

Referenced by `BSP_LCD_LayerDefaultInit()`.

```c
#define LCD_COLOR_YELLOW 0xFFFFFF00
```
Definition at line 123 of file stm32f429i_discovery_lcd.h.

#define LCD_DEFAULT_FONT  Font24

LCD default font.

Definition at line 147 of file stm32f429i_discovery_lcd.h.

Referenced by BSP_LCD_Init().

#define LCD_FOREGROUND_LAYER  0x0001

Definition at line 153 of file stm32f429i_discovery_lcd.h.

#define LCD_FRAME_BUFFER  ((uint32_t)0xD0000000)

Definition at line 112 of file stm32f429i_discovery_lcd.h.

#define LCD_LayerCfgTypeDef  LTDC_LayerCfgTypeDef

Definition at line 106 of file stm32f429i_discovery_lcd.h.

Referenced by BSP_LCD_LayerDefaultInit().

#define MAX_LAYER_NUMBER  2

LCD status structure definition.

Definition at line 111 of file stm32f429i_discovery_lcd.h.
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LOW LEVEL Private Variables**

STM32F429I DISCOVERY LOW LEVEL
### Variables

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIO_TypeDef *</td>
<td>GPIO_PORT [LEDn]</td>
</tr>
<tr>
<td>const uint16_t</td>
<td>GPIO_PIN [LEDn]</td>
</tr>
<tr>
<td>GPIO_TypeDef *</td>
<td>BUTTON_PORT [BUTTONn] = {KEY_BUTTON_GPIO_PORT}</td>
</tr>
<tr>
<td>const uint16_t</td>
<td>BUTTON_PIN [BUTTONn] = {KEY_BUTTON_PIN}</td>
</tr>
<tr>
<td>const uint8_t</td>
<td>BUTTON_IRQn [BUTTONn] = {KEY_BUTTON_EXTI_IRQn}</td>
</tr>
<tr>
<td>uint32_t</td>
<td>I2cxTimeout = I2Cx_TIMEOUT_MAX</td>
</tr>
<tr>
<td>uint32_t</td>
<td>SpixTimeout = SPIx_TIMEOUT_MAX</td>
</tr>
<tr>
<td>I2C_HandleTypeDef</td>
<td>I2cHandle</td>
</tr>
<tr>
<td>static SPI_HandleTypeDef</td>
<td>SpiHandle</td>
</tr>
<tr>
<td>static uint8_t</td>
<td>Is_LCD_IO_Initialized = 0</td>
</tr>
</tbody>
</table>
Variable Documentation

const uint8_t BUTTON_IRQn[BUTTONn] = \{KEY_BUTTON_EXTI_IR

Definition at line 100 of file stm32f429i_discovery.c.

Referenced by BSP_PB_Init().

const uint16_t BUTTON_PIN[BUTTONn] = \{KEY_BUTTON_PIN

Definition at line 99 of file stm32f429i_discovery.c.

Referenced by BSP_PB_GetState(), and BSP_PB_Init().

GPIO_TypeDef* BUTTON_PORT[BUTTONn] = \{KEY_BUTTON_GPIO_PORT

Definition at line 98 of file stm32f429i_discovery.c.

Referenced by BSP_PB_GetState(), and BSP_PB_Init().

const uint16_t GPIO_PIN[LEDn]

Initial value:

\{LED3_PIN,\n  LED4_PIN\}

Definition at line 95 of file stm32f429i_discovery.c.

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

GPIO_TypeDef* GPIO_PORT[LEDn]
Initial value:

\[
\{ \text{LED3_GPIO_PORT}, \text{LED4_GPIO_PORT} \}
\]

Definition at line 92 of file `stm32f429i_discovery.c`.

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

**I2C_HandleTypeDef I2cHandle**

Definition at line 105 of file `stm32f429i_discovery.c`.

Referenced by `I2Cx_Error()`, `I2Cx_Init()`, `I2Cx_MspInit()`, `I2Cx_ReadBuffer()`, `I2Cx_ReadData()`, `I2Cx_WriteBuffer()`, and `I2Cx_WriteData()`.

**uint32_t I2cxTimeout = I2Cx_TIMEOUT_MAX**

Definition at line 102 of file `stm32f429i_discovery.c`.

Referenced by `I2Cx_ReadBuffer()`, `I2Cx_ReadData()`, `I2Cx_WriteBuffer()`, and `I2Cx_WriteData()`.

**uint8_t Is_LCD_IO_Initialized = 0 [static]**

Definition at line 107 of file `stm32f429i_discovery.c`.

Referenced by `LCD_IO_Init()`.

**SPI_HandleTypeDef SpiHandle [static]**

Definition at line 106 of file `stm32f429i_discovery.c`.
uint32_t SpixTimeout = SPIx_TIMEOUT_MAX

Definition at line 103 of file stm32f429i_discovery.c.

Referenced by SPIx_Read(), SPIx_Write(), and SPIx_WriteRead().
STM32F429I-Discovery BSP User Manual

Directories

STM32F429I DISCOVERY LOW LEVEL Exported Types

STM32F429I DISCOVERY LOW LEVEL

Enumerations
## Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Led_TypeDef</td>
<td>enum Led_TypeDef { LED3 = 0, LED4 = 1 }</td>
</tr>
<tr>
<td>Button_TypeDef</td>
<td>enum Button_TypeDef { BUTTON_KEY = 0 }</td>
</tr>
<tr>
<td>ButtonMode_TypeDef</td>
<td>enum ButtonMode_TypeDef { BUTTON_MODE_GPIO = 0, BUTTON_MODE_EXTI = 1 }</td>
</tr>
</tbody>
</table>
## Enumeration Type Documentation

### enum Button_TypeDef

**Enumerator:**

- `BUTTON_KEY`

Definition at line 71 of file `stm32f429i_discovery.h`.

### enum ButtonMode_TypeDef

**Enumerator:**

- `BUTTON_MODE_GPIO`
- `BUTTON_MODE_EXTI`

Definition at line 76 of file `stm32f429i_discovery.h`.

### enum Led_TypeDef

**Enumerator:**

- `LED3`
- `LED4`

Definition at line 65 of file `stm32f429i_discovery.h`.

---

STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LOW LEVEL BUTTON**

STM32F429I DISCOVERY LOW LEVEL Exported Constants
#define BUTTONn 1
#define KEY_BUTTON_PIN GPIO_PIN_0
Wakeup push-button.

#define KEY_BUTTON_GPIO_PORT GPIOA
#define KEY_BUTTON_GPIO_CLK_ENABLE() __GPIOA_CLK_ENABLE()
#define KEY_BUTTON_GPIO_CLK_DISABLE() __GPIOA_CLK_DISABLE()
#define KEY_BUTTON_EXTI_IRQn EXTI0_IRQn
#define BUTTONx_GPIO_CLK_ENABLE(__INDEX__) __GPIOA_CLK_ENABLE(__INDEX__)
#define BUTTONx_GPIO_CLK_DISABLE(__INDEX__)
Define Documentation

#define BUTTONn 1

Definition at line 125 of file stm32f429i_discovery.h.

#define BUTTONx_GPIO_CLK_DISABLE (__INDEX__)

Value:

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

Definition at line 138 of file stm32f429i_discovery.h.

#define BUTTONx_GPIO_CLK_ENABLE (__INDEX__)

Value:

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

Definition at line 136 of file stm32f429i_discovery.h.

Referenced by BSP_PB_Init().

#define KEY_BUTTON EXTI IRQn EXTI0 IRQn

Definition at line 134 of file stm32f429i_discovery.h.
\#define KEY_BUTTON_GPIO_CLK_DISABLE () __GPIOA_CLK_DISABLE()

Definition at line 133 of file stm32f429i_discovery.h.

\#define KEY_BUTTON_GPIO_CLK_ENABLE () __GPIOA_CLK_ENABLE()

Definition at line 132 of file stm32f429i_discovery.h.

\#define KEY_BUTTON_GPIO_PORT GPIOA

Definition at line 131 of file stm32f429i_discovery.h.

\#define KEY_BUTTON_PIN GPIO_PIN_0

Wakeup push-button.

Definition at line 130 of file stm32f429i_discovery.h.
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY SDRAM Private Variables**

STM32F429I DISCOVERY SDRAM
### Variables

<table>
<thead>
<tr>
<th>Type Definition</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>static SDRAM_HandleTypeDef</td>
<td>SdramHandle</td>
</tr>
<tr>
<td>static FMC_SDRAM_TimingTypeDef</td>
<td>Timing</td>
</tr>
<tr>
<td>static FMC_SDRAM_CommandTypeDef</td>
<td>Command</td>
</tr>
</tbody>
</table>
## Variable Documentation

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Variable Name</th>
<th>Definition Line</th>
<th>Referenced by</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>FMC_SDRAM_CommandTypeDef</code></td>
<td><code>Command</code></td>
<td>80</td>
<td><code>BSP_SDRAM_Initialization_sequence()</code></td>
</tr>
<tr>
<td><code>SDRAM_HandleTypeDef</code></td>
<td><code>SdramHandle</code></td>
<td>78</td>
<td><code>BSP_SDRAM_DMA_IRQHandler()</code>, <code>BSP_SDRAM_Init()</code>, <code>BSP_SDRAM_Initialization_sequence()</code>, <code>BSP_SDRAM_ReadData()</code>, <code>BSP_SDRAM_ReadData_DMA()</code>, <code>BSP_SDRAM_Sendcmd()</code>, <code>BSP_SDRAM_WriteData()</code>, <code>BSP_SDRAM_WriteData_DMA()</code>, and <code>MspInit()</code></td>
</tr>
<tr>
<td><code>FMC_SDRAM_TimingTypeDef</code></td>
<td><code>Timing</code></td>
<td>79</td>
<td><code>BSP_SDRAM_Init()</code></td>
</tr>
</tbody>
</table>

STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LOW LEVEL BUS**

STM32F429I DISCOVERY LOW LEVEL Exported Constants

Defines
Defines

```c
#define IO_I2C_ADDRESS 0x82
#define TS_I2C_ADDRESS 0x82
#define DISCOVERY_I2Cx I2C3
#define DISCOVERY_I2Cx_CLOCK_ENABLE() __I2C3_CLK_ENABLE()
#define DISCOVERY_I2Cx_FORCE_RESET() __I2C3_FORCE_RESET()
#define DISCOVERY_I2Cx_RELEASE_RESET() __I2C3_RELEASE_RESET()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE() __GPIOC_CLK_ENABLE()
#define DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE() __GPIOA_CLK_ENABLE()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE() __GPIOC_CLK_DISABLE()
#define DISCOVERY_I2Cx_SCL_GPIO_PORT GPIOA
#define DISCOVERY_I2Cx_SCL_PIN GPIO_PIN_8
#define DISCOVERY_I2Cx_SCL_GPIO_PORT GPIOA
#define DISCOVERY_I2Cx_SDA_GPIO_PORT GPIOC
#define DISCOVERY_I2Cx_SDA_PIN GPIO_PIN_9
#define DISCOVERY_I2Cx_EV_IRQn I2C3_EV_IRQn
#define DISCOVERY_I2Cx_ER_IRQn I2C3_ER_IRQn
#define I2Cx_TIMEOUT_MAX 0x3000 /*<! The value of the maximal timeout for I2C waiting loops */
#define DISCOVERY_SPIx SPI5
#define DISCOVERY_SPIx_CLK_ENABLE() __SPI5_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_PORT GPIOF /* GPIOF */
#define DISCOVERY_SPIx_GPIO_CLK_ENABLE() __GPIOF_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_AF SPI_AF5_SPI5
#define DISCOVERY_SPIx_SCK_PIN GPIO_PIN_7 /* PF.07 */
#define DISCOVERY_SPIx_MISO_PIN GPIO_PIN_8 /* PF.08 */
#define DISCOVERY_SPIx_MOSI_PIN GPIO_PIN_9 /* PF.09 */
#define SPIx_TIMEOUT_MAX ((uint32_t)0x1000)
#define STMPE811_INT_PIN GPIO_PIN_15
#define STMPE811_INT_GPIO_PORT GPIOA
```
#define STMPE811_INT_CLK_ENABLE() __GPIOA_CLK_ENABLE()
#define STMPE811_INT_CLK_DISABLE() __GPIOA_CLK_DISABLE()
#define STMPE811_INT_EXTI EXTI15_10_IRQHandler
#define STMPE811_INT_EXTIHandler EXTI15_10_IRQHandler
#define LCD_CS_LOW()  HAL_GPIO_WritePin(LCDC_NCS_GPIO_PORT, LCD_NCS_PIN, GPIO_PIN_RESET)
#define LCD_CS_HIGH() HAL_GPIO_WritePin(LCDC_NCS_GPIO_PORT, LCD_NCS_PIN, GPIO_PIN_SET)
#define LCD_WRX_LOW()  HAL_GPIO_WritePin(LCDC_WRX_GPIO_PORT, LCD_WRX_PIN, GPIO_PIN_RESET)
#define LCD_WRX_HIGH() HAL_GPIO_WritePin(LCDC_WRX_GPIO_PORT, LCD_WRX_PIN, GPIO_PIN_SET)
#define LCD_RDX_LOW()  HAL_GPIO_WritePin(LCDC_RDX_GPIO_PORT, LCD_RDX_PIN, GPIO_PIN_RESET)
#define LCD_RDX_HIGH() HAL_GPIO_WritePin(LCDC_RDX_GPIO_PORT, LCD_RDX_PIN, GPIO_PIN_SET)
#define LCD_NCS_PIN GPIO_PIN_2
LCD Control pin.
#define LCD_NCS_GPIO_PORT GPIOC
#define LCD_NCS_GPIO_CLK_ENABLE() __GPIOC_CLK_ENABLE()
#define LCD_NCS_GPIO_CLK_DISABLE() __GPIOC_CLK_DISABLE()
Define Documentation

```
#define DISCOVERY_I2Cx I2C3
Definition at line 159 of file stm32f429i_discovery.h.
Referenced by I2Cx_Init(), and I2Cx_MspInit().
```

```
#define DISCOVERY_I2Cx_CLOCK_ENABLE () __I2C3_CLK_ENABLE()
Definition at line 160 of file stm32f429i_discovery.h.
Referenced by I2Cx_MspInit().
```

```
#define DISCOVERY_I2Cx_ER_IRQn I2C3_ER_IRQn
Definition at line 176 of file stm32f429i_discovery.h.
Referenced by I2Cx_MspInit().
```

```
#define DISCOVERY_I2Cx_EV_IRQn I2C3_EV_IRQn
Definition at line 175 of file stm32f429i_discovery.h.
Referenced by I2Cx_MspInit().
```

```
#define DISCOVERY_I2Cx_FORCE_RESET () __I2C3_FORCE_RESET()
Definition at line 161 of file stm32f429i_discovery.h.
Referenced by I2Cx_MspInit().
```


```c
#define DISCOVERY_I2Cx_RELEASE_RESET () __I2C3_RELEASE_RESET()
```

Definition at line 162 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE () __GPIOA_CLK_ENABLE()
```

Definition at line 164 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_SCL_GPIO_PORT GPIOA
```

Definition at line 169 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_SCL_PIN GPIO_PIN_8
```

Definition at line 168 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_SCL_SDA_AF GPIO_AF4_I2C3
```

Definition at line 170 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE () __GPIOA_CLK_DISABLE()
```

Definition at line 164 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.
Definition at line 165 of file `stm32f429i_discovery.h`.

```c
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE ( ) __GPIOC
```

Definition at line 163 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_SDA_GPIO_PORT GPIOC
```

Definition at line 172 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_SDA_PIN GPIO_PIN_9
```

Definition at line 171 of file `stm32f429i_discovery.h`.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_SPIx SPI5
```

Definition at line 189 of file `stm32f429i_discovery.h`.

Referenced by `SPIx_Init()`.

```c
#define DISCOVERY_SPIx_AF GPIO_AF5_SPI5
```

Definition at line 192 of file `stm32f429i_discovery.h`.

Referenced by `SPIx_MspInit()`.
#define DISCOVERY_SPIx_CLK_ENABLE ( ) __SPI5_CLK_ENABLE()
Definition at line 190 of file stm32f429i_discovery.h.
Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_GPIO_CLK_DISABLE ( ) __GPIOF_CLK_DISABLE()
Definition at line 194 of file stm32f429i_discovery.h.

#define DISCOVERY_SPIx_GPIO_CLK_ENABLE ( ) __GPIOF_CLK_ENABLE()
Definition at line 193 of file stm32f429i_discovery.h.
Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_GPIO_PORT GPIOF /* GPIOF */
Definition at line 191 of file stm32f429i_discovery.h.
Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_MISO_PIN GPIO_PIN_8 /* PF.08 */
Definition at line 196 of file stm32f429i_discovery.h.
Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_MOSI_PIN GPIO_PIN_9 /* PF.09 */
Definition at line 197 of file stm32f429i_discovery.h.
Referenced by SPIx_MspInit().

#define DISCOVERY_SPIx_SCK_PIN GPIO_PIN_7 /* PF.07 */
Definition at line 195 of file stm32f429i_discovery.h.
Referenced by SPIx_MspInit().

#define I2Cx_TIMEOUT_MAX 0x3000 /*<! The value of the maxima
Definition at line 186 of file stm32f429i_discovery.h.

#define IO_I2C_ADDRESS 0x82
Definition at line 148 of file stm32f429i_discovery.h.
Referenced by BSP_IO_ConfigPin(), BSP_IO_Init(),
BSP_IO_ITClear(), BSP_IO_ITGetStatus(), BSP_IO_ReadPin(),
BSP_IO_TogglePin(), and BSP_IO_WritePin().

#define LCD_CS_HIGH ( ) HAL_GPIO_WritePin(LCD_NCS_GPIO_}
Definition at line 221 of file stm32f429i_discovery.h.
Referenced by LCD_IO_Init(), LCD_IO_ReadData(),
LCD_IO_WriteData(), and LCD_IO_WriteReg().

#define LCD_CS_LOW ( ) HAL_GPIO_WritePin(LCD_NCS_GPIO_}
Definition at line 220 of file stm32f429i_discovery.h.
Referenced by LCD_IO_Init(), LCD_IO_ReadData(),
LCD_IO_WriteData(), and LCD_IO_WriteReg().

```c
#define LCD_NCS_GPIO_CLK_DISABLE () __GPIOC_CLK_DISABLE()
```
Definition at line 237 of file stm32f429i_discovery.h.

```c
#define LCD_NCS_GPIO_CLK_ENABLE () __GPIOC_CLK_ENABLE()
```
Definition at line 236 of file stm32f429i_discovery.h.

Referenced by LCD_IO_Init().

```c
#define LCD_NCS_GPIO_PORT GPIOC
```
Definition at line 235 of file stm32f429i_discovery.h.

Referenced by LCD_IO_Init().

```c
#define LCD_NCS_PIN GPIO_PIN_2
```
Definition at line 234 of file stm32f429i_discovery.h.

Referenced by LCD_IO_Init().

```c
#define LCD_RDX_HIGH() HAL_GPIO_WritePin(LCD_RDX_GPIO
```
Definition at line 229 of file stm32f429i_discovery.h.

```c
#define LCD_RDX_LOW() HAL_GPIO_WritePin(LCD_RDX_GPIO
```


```c
#define LCD_WRX_HIGH()  HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, LCD_WRX_PIN,

Definition at line 228 of file stm32f429i_discovery.h.

Referenced by LCD_IO_ReadData(), and LCD_IO_WriteData().

#define LCD_WRX_LOW()   HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, LCD_WRX_PIN,

Definition at line 224 of file stm32f429i_discovery.h.

Referenced by LCD_IO_ReadData(), and LCD_IO_WriteReg().

#define SPIx_TIMEOUT_MAX ((uint32_t)0x1000)

Definition at line 203 of file stm32f429i_discovery.h.

#define STMPE811_INT_CLK_DISABLE()  __GPIOA_CLK_DISABLE()

Definition at line 214 of file stm32f429i_discovery.h.

Referenced by I2Cx_ITConfig().

#define STMPE811_INT_CLK_ENABLE()   __GPIOA_CLK_ENABLE()

Definition at line 213 of file stm32f429i_discovery.h.

#define STMPE811_INT_EXTI   EXTI15_10_IRQHandler

Definition at line 215 of file stm32f429i_discovery.h.
```
Referenced by `I2Cx_ITConfig()`.

```c
#define STMPE811_INT_EXTIHandler EXTI15_10_IRQHandler
```
Definition at line 216 of file `stm32f429i_discovery.h`.

```c
#define STMPE811_INT_GPIO_PORT GPIOA
```
Definition at line 212 of file `stm32f429i_discovery.h`.
Referenced by `I2Cx_ITConfig()`.

```c
#define STMPE811_INT_PIN GPIO_PIN_15
```
IOE Control pin.
Definition at line 211 of file `stm32f429i_discovery.h`.
Referenced by `I2Cx_ITConfig()`.

```c
#define TS_I2C_ADDRESS 0x82
```
Definition at line 149 of file `stm32f429i_discovery.h`.
Referenced by `BSP_TS_GetState()`, `BSP_TS_Init()`, `BSP_TS_ITClear()`, `BSP_TS_ITConfig()`, and `BSP_TS_ITGetStatus()`.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td>Modules</td>
<td>Defines</td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LOW LEVEL Exported Constants**

STM32F429I DISCOVERY LOW LEVEL
## Modules

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL LED</td>
<td>Define for STM32F429I_DISCO board.</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL BUTTON</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL BUS</td>
<td></td>
</tr>
</tbody>
</table>
**Defines**

```c
#define LCD_WRX_PIN    GPIO_PIN_13
     LCD Command/data pin.
#define LCD_WRX_GPIO_PORT    GPIOD
#define LCD_WRX_GPIO_CLK_ENABLE()  __GPIOD_CLK_ENABLE()  
#define LCD_WRX_GPIO_CLK_DISABLE() __GPIOD_CLK_DISABLE()  
#define LCD_RDX_PIN    GPIO_PIN_12
#define LCD_RDX_GPIO_PORT    GPIOD
#define LCD_RDX_GPIO_CLK_ENABLE()  __GPIOD_CLK_ENABLE()  
#define LCD_RDX_GPIO_CLK_DISABLE() __GPIOD_CLK_DISABLE()  
#define READWRITE_CMD    ((uint8_t)0x80)
#define MULTIPLEBYTE_CMD    ((uint8_t)0x40)
#define DUMMY_BYTE    ((uint8_t)0x00)
#define GYRO_CS_LOW()      HAL_GPIO_WritePin(GYRO_CS_GPIO_PORT, GYRO_CS_PIN, GPIO_PIN_RESET)
#define GYRO_CS_HIGH()     HAL_GPIO_WritePin(GYRO_CS_GPIO_PORT, GYRO_CS_PIN, GPIO_PIN_SET)
#define GYRO_CS_PIN    GPIO_PIN_1 /* PC.01 */  
     GYROSCOPE SPI Interface pins.
#define GYRO_CS_GPIO_PORT    GPIOC /* GPIOC */
#define GYRO_CS_GPIO_CLK_ENABLE()  __GPIOC_CLK_ENABLE()  
#define GYRO_CS_GPIO_CLK_DISABLE() __GPIOC_CLK_DISABLE()  
#define GYRO_INT_GPIO_CLK_ENABLE() __GPIOA_CLK_ENABLE()  
#define GYRO_INT_GPIO_CLK_DISABLE() __GPIOA_CLK_DISABLE()  
#define GYRO_INT_GPIO_PORT    GPIOA /* GPIOA */
#define GYRO_INT1_PIN    GPIO_PIN_1 /* PA.01 */
#define GYRO_INT1_EXTIIRQn  EXTI1_IRQn
#define GYRO_INT2_PIN    GPIO_PIN_2 /* PA.02 */
#define GYRO_INT2_EXTIIRQn  EXTI2_IRQn
```
Define Documentation

```c
#define DUMMY_BYTE  ((uint8_t)0x00)
```
Definition at line 260 of file `stm32f429i_discovery.h`.
Referenced by `GYRO_IO_Read()`.

```c
#define GYRO_CS_GPIO_CLK_DISABLE( ) __GPIOC_CLK_DISABLE()
```
Definition at line 272 of file `stm32f429i_discovery.h`.

```c
#define GYRO_CS_GPIO_CLK_ENABLE( ) __GPIOC_CLK_ENABLE()
```
Definition at line 271 of file `stm32f429i_discovery.h`.
Referenced by `GYRO_IO_Init()`.

```c
#define GYRO_CS_GPIO_PORT  GPIOC /* GPIOC */
```
Definition at line 270 of file `stm32f429i_discovery.h`.
Referenced by `GYRO_IO_Init()`.

```c
#define GYRO_CS_HIGH( )  HAL_GPIO_WritePin(GYRO_CS_GPIO_PORT, }
                        GYRO_CS_PIN, GPIO_PIN_SET)
```
Definition at line 264 of file `stm32f429i_discovery.h`.
Referenced by `GYRO_IO_Init()`, `GYRO_IO_Read()`, and `GYRO_IO_Write()`.
#define GYRO_CS_LOW( ) HAL_GPIO_WritePin(GYRO_CS_GPIO_PORT

Definition at line 263 of file stm32f429i_discovery.h.

Referenced by GYRO_IO_Read(), and GYRO_IO_Write().

#define GYRO_CS_PIN GPIO_PIN_1 /* PC.01 */

GYROSCOPE SPI Interface pins.

Definition at line 269 of file stm32f429i_discovery.h.

Referenced by GYRO_IO_Init().

#define GYRO_INT1_EXTI_IRQn EXTI1_IRQn

Definition at line 278 of file stm32f429i_discovery.h.

#define GYRO_INT1_PIN GPIO_PIN_1 /* PA.01 */

Definition at line 277 of file stm32f429i_discovery.h.

Referenced by GYRO_IO_Init().

#define GYRO_INT2_EXTI_IRQn EXTI2_IRQn

Definition at line 280 of file stm32f429i_discovery.h.

#define GYRO_INT2_PIN GPIO_PIN_2 /* PA.02 */

Definition at line 279 of file stm32f429i_discovery.h.
Referenced by `GYRO_IO_Init()`.

```c
#define GYRO_INT_GPIO_CLK_DISABLE ( ) __GPIOA_CLK_DISABLE()
```
Definition at line 275 of file `stm32f429i_discovery.h`.

```c
#define GYRO_INT_GPIO_CLK_ENABLE ( ) __GPIOA_CLK_ENABLE()
```
Definition at line 274 of file `stm32f429i_discovery.h`.

Referenced by `GYRO_IO_Init()`.

```c
#define GYRO_INT_GPIO_PORT GPIOA /* GPIOA */
```
Definition at line 276 of file `stm32f429i_discovery.h`.

Referenced by `GYRO_IO_Init()`.

```c
#define LCD_RDX_GPIO_CLK_DISABLE ( ) __GPIOD_CLK_DISABLE()
```
Definition at line 252 of file `stm32f429i_discovery.h`.

```c
#define LCD_RDX_GPIO_CLK_ENABLE ( ) __GPIOD_CLK_ENABLE()
```
Definition at line 251 of file `stm32f429i_discovery.h`.

Referenced by `LCD_IO_Init()`.

```c
#define LCD_RDX_GPIO_PORT GPIOD
```
Definition at line 250 of file `stm32f429i_discovery.h`.
Referenced by `LCD_IO_Init()`.

```c
#define LCD_RDX_PIN GPIO_PIN_12
```
Definition at line 249 of file `stm32f429i_discovery.h`.
Referenced by `LCD_IO_Init()`.

```c
#define LCD_WRX_GPIO_CLK_DISABLE ( ) __GPIOD_CLK_DISABLE()
```
Definition at line 247 of file `stm32f429i_discovery.h`.

```c
#define LCD_WRX_GPIO_CLK_ENABLE ( ) __GPIOD_CLK_ENABLE()
```
Definition at line 246 of file `stm32f429i_discovery.h`.
Referenced by `LCD_IO_Init()`.

```c
#define LCD_WRX_GPIO_PORT GPIOD
```
Definition at line 245 of file `stm32f429i_discovery.h`.
Referenced by `LCD_IO_Init()`.

```c
#define LCD_WRX_PIN GPIO_PIN_13
```
LCD Command/data pin.
Definition at line 244 of file `stm32f429i_discovery.h`.
Referenced by `LCD_IO_Init()`.
#define MULTIPLEBYTE_CMD ((uint8_t)0x40)

Definition at line 258 of file stm32f429i_discovery.h.

Referenced by GYRO_IO_Read(), and GYRO_IO_Write().

#define READWRITE_CMD ((uint8_t)0x80)

Definition at line 256 of file stm32f429i_discovery.h.

Referenced by GYRO_IO_Read().
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I**  
DISCOVERY EEPROM Exported Constants

STM32F429I DISCOVERY EEPROM
<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>EEPROM_PAGESIZE</code></td>
<td>4</td>
</tr>
<tr>
<td><code>EEPROM_MAX_SIZE</code></td>
<td>0x2000 /* 64Kbit*/</td>
</tr>
<tr>
<td><code>EEPROM_READ_TIMEOUT</code></td>
<td>((uint32_t)(1000))</td>
</tr>
<tr>
<td><code>EEPROM_WRITE_TIMEOUT</code></td>
<td>((uint32_t)(1))</td>
</tr>
<tr>
<td><code>EEPROM_MAX_TRIALS</code></td>
<td>300</td>
</tr>
<tr>
<td><code>EEPROM_OK</code></td>
<td>0</td>
</tr>
<tr>
<td><code>EEPROM_FAIL</code></td>
<td>1</td>
</tr>
<tr>
<td><code>EEPROM_TIMEOUT</code></td>
<td>2</td>
</tr>
</tbody>
</table>
Define Documentation

#define EEPROM_FAIL 1
Definition at line 87 of file stm32f429i_discovery_eeprom.h.

#define EEPROM_MAX_SIZE 0x2000 /* 64Kbit*/
Definition at line 74 of file stm32f429i_discovery_eeprom.h.

#define EEPROM_MAX_TRIALS 300
Definition at line 84 of file stm32f429i_discovery_eeprom.h.

#define EEPROM_OK 0
Definition at line 86 of file stm32f429i_discovery_eeprom.h.

#define EEPROM_PAGESIZE 4
Definition at line 73 of file stm32f429i_discovery_eeprom.h.

#define EEPROM_READ_TIMEOUT ((uint32_t)(1000))
Definition at line 79 of file stm32f429i_discovery_eeprom.h.

#define EEPROM_TIMEOUT 2
Definition at line 88 of file stm32f429i_discovery_eeprom.h.
#define EEPROM_WRITE_TIMEOUT ((uint32_t)(1))

Definition at line 81 of file stm32f429i_discovery_eeprom.h.
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY GYROSCOPE Exported Types**

STM32F429I DISCOVERY GYROSCOPE
Enumerations

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


**Enumeration Type Documentation**

```
enum GYRO_StatusTypeDef
```

**Enumerator:**

- `GYRO_OK`
- `GYRO_ERROR`
- `GYRO_TIMEOUT`

Definition at line 67 of file `stm32f429i_discovery_gyroscope.h`.

STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LOW LEVEL Private Function Prototypes

STM32F429I DISCOVERY LOW LEVEL
### Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static void</td>
<td><strong>I2Cx_Init</strong> (void)</td>
<td>I2Cx Bus initialization.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>I2Cx_ITConfig</strong> (void)</td>
<td>Configures Interruption pin for I2C communication.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>I2Cx_Error</strong> (void)</td>
<td>I2Cx error treatment function.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>SPIx_Init</strong> (void)</td>
<td>SPIx Bus initialization.</td>
</tr>
<tr>
<td>static void</td>
<td><strong>SPIx_Error</strong> (void)</td>
<td>SPIx error treatment function.</td>
</tr>
<tr>
<td>void</td>
<td><strong>LCD_IO_Init</strong> (void)</td>
<td>Configures the LCD_SPI interface.</td>
</tr>
<tr>
<td>void</td>
<td><strong>LCD_IO_WriteData</strong> (uint16_t RegValue)</td>
<td>Writes register value.</td>
</tr>
<tr>
<td>void</td>
<td><strong>LCD_IO_WriteReg</strong> (uint8_t Reg)</td>
<td>Writes register address.</td>
</tr>
<tr>
<td>void</td>
<td><strong>IOE_Init</strong> (void)</td>
<td>IOE Low Level Initialization.</td>
</tr>
<tr>
<td>void</td>
<td><strong>IOE_ITConfig</strong> (void)</td>
<td>IOE Low Level Interrupt configuration.</td>
</tr>
<tr>
<td>void</td>
<td><strong>GYRO_IO_Init</strong> (void)</td>
<td>Configures the Gyroscope SPI interface.</td>
</tr>
</tbody>
</table>
Function Documentation

void GYRO_IO_Init ( void )

Configures the Gyroscope SPI interface.

Definition at line 961 of file stm32f429i_discovery.c.

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

static void I2Cx_Error ( void ) [static]

I2Cx error treatment function.

Definition at line 620 of file stm32f429i_discovery.c.

References I2cHandle, and I2Cx_Init().

Referenced by I2Cx_ReadBuffer(), I2Cx_ReadData(), I2Cx_WriteBuffer(), and I2Cx_WriteData().

static void I2Cx_Init ( void ) [static]

I2Cx Bus initialization.

Definition at line 422 of file stm32f429i_discovery.c.

References DISCOVERY_I2Cx, I2cHandle, and I2Cx_MspInit().

Referenced by I2Cx_Error(), and IOE_Init().
static void I2Cx_ITConfig ( void ) [static]

Configures Interruption pin for I2C communication.

Definition at line 445 of file stm32f429i_discovery.c.

References STMPE811_INT_CLK_ENABLE, STMPE811_INT_EXTI, STMPE811_INT_GPIO_PORT, and STMPE811_INT_PIN.

Referenced by IOE_ITConfig().

void IOE_Init ( void )

IOE Low Level Initialization.

Definition at line 887 of file stm32f429i_discovery.c.

References I2Cx_Init().

void IOE_ITConfig ( void )

IOE Low Level Interrupt configuration.

Definition at line 895 of file stm32f429i_discovery.c.

References I2Cx_ITConfig().

void LCD_IO_Init ( void )

Configures the LCD_SPI interface.

Definition at line 767 of file stm32f429i_discovery.c.

References Is_LCD_IO_INITIALIZED, LCD_CS_HIGH, LCD_CS_LOW, LCD_NCS_GPIO_CLK_ENABLE, LCD_NCS_GPIO_PORT,
void LCD_IO_WriteData (uint16_t RegValue)

Writes register value.

Definition at line 811 of file stm32f429i_discovery.c.

References LCD_CS_HIGH, LCD_CS_LOW, LCD_WRX_HIGH, and SPIx_Write().

void LCD_IO_WriteReg (uint8_t Reg)

Writes register address.

Definition at line 827 of file stm32f429i_discovery.c.

References LCD_CS_HIGH, LCD_CS_LOW, LCD_WRX_LOW, and SPIx_Write().

static void SPIx_Error (void) [static]

SPIx error treatment function.

Definition at line 730 of file stm32f429i_discovery.c.

References SpiHandle, and SPIx_Init().

Referenced by SPIx_Read(), SPIx_Write(), and SPIx_WriteRead().

static void SPIx_Init (void) [static]
SPIx Bus initialization.

Definition at line 634 of file stm32f429i_discovery.c.

References DISCOVERY_SPIx, SpiHandle, and SPIx_MspInit().

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

BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STM32F429I</strong> DISCOVERY GYROSCOPE Private Variables</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY GYROSCOPE
## Variables

```c
static GYRO_DrvTypeDef * GyroscopeDrv
```
GYRO_DrvTypeDef* GyroscopeDrv [static]

Definition at line 77 of file stm32f429i_discovery_gyroscope.c.

Referenced by BSP_GYRO_DisableIT(), BSP_GYRO_EnableIT(), BSP_GYRO_GetXYZ(), BSP_GYRO_Init(), BSP_GYRO_ITConfig(), BSP_GYRO_ReadID(), and BSP_GYRO_Reset.
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Directories</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
<th>Enumerations</th>
</tr>
</thead>
</table>

## STM32F429I Discovery IO Exported Types

STM32F429I Discovery IO
Enumerations

```c
enum IO_StatusTypeDef { IO_OK = 0, IO_ERROR = 1, IO_TIMEOUT = 2 }```

Enumeration Type Documentation

```c
enum IO_StatusTypeDef
```

**Enumerator:**

- `IO_OK`
- `IO_ERROR`
- `IO_TIMEOUT`

Definition at line 67 of file `stm32f429i_discovery_io.h`.
STM32F429I-Discovery BSP User Manual

Directories

STM32F429I DISCOVERY IO Exported Constants

STM32F429I DISCOVERY IO
## Defines

<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IO_PIN_0</td>
<td>0x01</td>
</tr>
<tr>
<td>IO_PIN_1</td>
<td>0x02</td>
</tr>
<tr>
<td>IO_PIN_2</td>
<td>0x04</td>
</tr>
<tr>
<td>IO_PIN_3</td>
<td>0x08</td>
</tr>
<tr>
<td>IO_PIN_4</td>
<td>0x10</td>
</tr>
<tr>
<td>IO_PIN_5</td>
<td>0x20</td>
</tr>
<tr>
<td>IO_PIN_6</td>
<td>0x40</td>
</tr>
<tr>
<td>IO_PIN_7</td>
<td>0x80</td>
</tr>
<tr>
<td>IO_PIN_ALL</td>
<td>0xFF</td>
</tr>
</tbody>
</table>
Define Documentation

#define IO_PIN_0 0x01
Definition at line 80 of file stm32f429i_discovery_io.h.

#define IO_PIN_1 0x02
Definition at line 81 of file stm32f429i_discovery_io.h.

#define IO_PIN_2 0x04
Definition at line 82 of file stm32f429i_discovery_io.h.

#define IO_PIN_3 0x08
Definition at line 83 of file stm32f429i_discovery_io.h.

#define IO_PIN_4 0x10
Definition at line 84 of file stm32f429i_discovery_io.h.

#define IO_PIN_5 0x20
Definition at line 85 of file stm32f429i_discovery_io.h.

#define IO_PIN_6 0x40
Definition at line 86 of file stm32f429i_discovery_io.h.


```c
#define IO_PIN_7 0x80

Definition at line 87 of file stm32f429i_discovery_io.h.
```

```c
#define IO_PIN_ALL 0xFF

Definition at line 88 of file stm32f429i_discovery_io.h.
```

Referenced by `BSP_IO_Init()`, and `BSP_IO_ITClear()`.

---

BSP User Manual by `doxygen` 1.7.6.1
## STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### STM32F429I DISCOVERY IO Private Variables

**STM32F429I DISCOVERY IO**
Variables

| static IO_DrvTypeDef * ioDrv |
Variable Documentation

**IO_DrvTypeDef* IoDrv** [static]

Definition at line 79 of file *stm32f429i_discovery_io.c*.

Referenced by **BSP_IO_ConfigPin()**, **BSP_IO_Init()**, **BSP_IO_ITClear()**, **BSP_IO_ITGetStatus()**, **BSP_IO_ReadPin()**, **BSP_IO_TogglePin()**, and **BSP_IO_WritePin()**.

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LCD Exported Macros**

STM32F429I DISCOVERY LCD
Defines

#define LCD_PIXEL_FORMAT_ARGB8888 LTDC_PIXEL_FORMAT_ARGB8888
LCD Pixel format.
#define LCD_PIXEL_FORMAT_RGB888 LTDC_PIXEL_FORMAT_RGB888
#define LCD_PIXEL_FORMAT_RGB565 LTDC_PIXEL_FORMAT_RGB565
#define LCD_PIXEL_FORMAT_ARGB1555 LTDC_PIXEL_FORMAT_ARGB1555
#define LCD_PIXEL_FORMAT_ARGB4444 LTDC_PIXEL_FORMAT_ARGB4444
#define LCD_PIXEL_FORMAT_L8 LTDC_PIXEL_FORMAT_L8
#define LCD_PIXEL_FORMAT_AL44 LTDC_PIXEL_FORMAT_AL44
#define LCD_PIXEL_FORMAT_AL88 LTDC_PIXEL_FORMAT_AL88
Define Documentation

#define LCD_PIXEL_FORMAT_AL44  LTDC_PIXEL_FORMAT_AL44

Definition at line 171 of file stm32f429i_discovery_lcd.h.

#define LCD_PIXEL_FORMAT_AL88  LTDC_PIXEL_FORMAT_AL88

Definition at line 172 of file stm32f429i_discovery_lcd.h.

#define LCD_PIXEL_FORMAT_ARGB1555  LTDC_PIXEL_FORMAT_ARGB1555

Definition at line 168 of file stm32f429i_discovery_lcd.h.

#define LCD_PIXEL_FORMAT_ARGB4444  LTDC_PIXEL_FORMAT_ARGB4444

Definition at line 169 of file stm32f429i_discovery_lcd.h.

#define LCD_PIXEL_FORMAT_ARGB8888  LTDC_PIXEL_FORMAT_ARGB8888

Definition at line 165 of file stm32f429i_discovery_lcd.h.

#define LCD_PIXEL_FORMAT_L8  LTDC_PIXEL_FORMAT_L8

Definition at line 170 of file stm32f429i_discovery_lcd.h.
Definition at line 167 of file \texttt{stm32f429i\_discovery\_lcd.h}.

\begin{verbatim}
#define LCD_PIXEL_FORMAT_RGB888  LTDC_PIXEL_FORMAT_RGB888
\end{verbatim}

Definition at line 166 of file \texttt{stm32f429i\_discovery\_lcd.h}.

BSP User Manual by \texttt{doxygen} 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LOW LEVEL LED

STM32F429I DISCOVERY LOW LEVEL Exported Constants

Define for STM32F429I_DISCO board. More...
<table>
<thead>
<tr>
<th>Defines</th>
</tr>
</thead>
<tbody>
<tr>
<td>#define LEDn 2</td>
</tr>
<tr>
<td>#define LED3_PIN GPIO_PIN_13</td>
</tr>
<tr>
<td>#define LED3_GPIO_PORT GPIOG</td>
</tr>
<tr>
<td>#define LED3_GPIO_CLK_ENABLE() __GPIOG_CLK_ENABLE()</td>
</tr>
<tr>
<td>#define LED3_GPIO_CLK_DISABLE() __GPIOG_CLK_DISABLE()</td>
</tr>
<tr>
<td>#define LED4_PIN GPIO_PIN_14</td>
</tr>
<tr>
<td>#define LED4_GPIO_PORT GPIOG</td>
</tr>
<tr>
<td>#define LED4_GPIO_CLK_ENABLE() __GPIOG_CLK_ENABLE()</td>
</tr>
<tr>
<td>#define LED4_GPIO_CLK_DISABLE() __GPIOG_CLK_DISABLE()</td>
</tr>
<tr>
<td>#define LEDx_GPIO_CLK_ENABLE(<strong>INDEX</strong>)</td>
</tr>
<tr>
<td>#define LEDx_GPIO_CLK_DISABLE(<strong>INDEX</strong>)</td>
</tr>
</tbody>
</table>
Detailed Description

Define for STM32F429I_DISCO board.
Define Documentation

#define LED3_GPIO_CLK_DISABLE ( ) __GPIOG_CLK_DISABLE()
Definition at line 105 of file stm32f429i_discovery.h.

#define LED3_GPIO_CLK_ENABLE ( ) __GPIOG_CLK_ENABLE()
Definition at line 104 of file stm32f429i_discovery.h.

#define LED3_GPIO_PORT GPIOG
Definition at line 103 of file stm32f429i_discovery.h.

#define LED3_PIN GPIO_PIN_13
Definition at line 102 of file stm32f429i_discovery.h.

#define LED4_GPIO_CLK_DISABLE ( ) __GPIOG_CLK_DISABLE()
Definition at line 110 of file stm32f429i_discovery.h.

#define LED4_GPIO_CLK_ENABLE ( ) __GPIOG_CLK_ENABLE()
Definition at line 109 of file stm32f429i_discovery.h.

#define LED4_GPIO_PORT GPIOG
Definition at line 108 of file stm32f429i_discovery.h.
```c
#define LED4_PIN GPIO_PIN_14

Definition at line 107 of file stm32f429i_discovery.h.

#define LEDn 2

Definition at line 100 of file stm32f429i_discovery.h.

#define LEDx_GPIO_CLK_DISABLE(__INDEX__) 
Value:
  do{if((__INDEX__) == 0) LED3_GPIO_CLK_DISABLE();
     else 
        if((__INDEX__) == 1) LED4_GPIO_CLK_DISABLE(); 
     }while(0)

Definition at line 115 of file stm32f429i_discovery.h.

#define LEDx_GPIO_CLK_ENABLE(__INDEX__) 
Value:
  do{if((__INDEX__) == 0) LED3_GPIO_CLK_ENABLE();
     else 
        if((__INDEX__) == 1) LED4_GPIO_CLK_ENABLE(); 
     }while(0)

Definition at line 112 of file stm32f429i_discovery.h.

Referenced by BSP_LED_Init().
```
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LCD Private Function Prototypes**

STM32F429I DISCOVERY LCD
## Functions

<table>
<thead>
<tr>
<th>static void</th>
<th><strong>MspInit</strong> (void)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Initializes the LTDC MSP.</td>
</tr>
</tbody>
</table>


```c
static void MspInit ( void ) [static]
```

Initializes the LTDC MSP.

Definition at line 1131 of file `stm32f429i_discovery_lcd.c`.

Referenced by `BSP_LCD_Init()`.

---

BSP User Manual by [doxygen](http://www.doxygen.org) 1.7.6.1
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY SDRAM Private Function Prototypes**

**STM32F429I DISCOVERY SDRAM**
## Functions

<table>
<thead>
<tr>
<th>static void MspInit (void)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initializes SDRAM MSP.</td>
</tr>
</tbody>
</table>
Function Documentation

```c
static void MspInit ( void ) [static]
```

Initializes SDRAM MSP.

Definition at line 273 of file stm32f429i_discovery_sdram.c.

References __DMAx_CLK_ENABLE, SDRAM_DMAx_CHANNEL, SDRAM_DMAx_IRQn, SDRAM_DMAx_STREAM, and SdramHandle.

Referenced by BSP_SDRAM_Init().

# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## STM32F429I DISCOVERY LCD Private Defines

STM32F429I DISCOVERY LCD
Defines

```c
#define POLY_X(Z) ((int32_t)((Points + Z)->X))
#define POLY_Y(Z) ((int32_t)((Points + Z)->Y))
```
Define Documentation

#define POLY_X (Z) ((int32_t)((Points + Z)->X))

Definition at line 108 of file stm32f429i_discovery_lcd.c.
Referenced by BSP_LCD_FillPolygon().

#define POLY_Y (Z) ((int32_t)((Points + Z)->Y))

Definition at line 109 of file stm32f429i_discovery_lcd.c.
Referenced by BSP_LCD_FillPolygon().

BSP User Manual by doxygen 1.7.6.1
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
<th>Directories</th>
</tr>
</thead>
</table>

STM32F429I DISCOVERY TS Exported Constants

STM32F429I DISCOVERY TS
<table>
<thead>
<tr>
<th>Defines</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#define TS_SWAP_NONE</td>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>#define TS_SWAP_X</td>
<td>0x01</td>
<td></td>
</tr>
<tr>
<td>#define TS_SWAP_Y</td>
<td>0x02</td>
<td></td>
</tr>
<tr>
<td>#define TS_SWAP_XY</td>
<td>0x04</td>
<td></td>
</tr>
</tbody>
</table>
Enumerations

```c
enum TS_StatusTypeDef { TS_OK = 0x00, TS_ERROR = 0x01, TS_TIMEOUT = 0x02 }
```
Define Documentation

```c
#define TS_SWAP_NONE  0x00
```
Definition at line 81 of file `stm32f429i_discovery_ts.h`.

```c
#define TS_SWAP_X   0x01
```
Definition at line 82 of file `stm32f429i_discovery_ts.h`.

```c
#define TS_SWAP_XY  0x04
```
Definition at line 84 of file `stm32f429i_discovery_ts.h`.

```c
#define TS_SWAP_Y   0x02
```
Definition at line 83 of file `stm32f429i_discovery_ts.h`. 
Enumeration Type Documentation

```c
enum TS_StatusTypeDef
```

**Enumerator:**

- `TS_OK`
- `TS_ERROR`
- `TS_TIMEOUT`

Definition at line 86 of file `stm32f429i_discovery_ts.h`.

BSP User Manual by [doxygen](http://www.doxygen.org) 1.7.6.1
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STM32F429I</strong></td>
<td><strong>DISCOVERY TS Private Variables</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STM32F429I DISCOVERY TS</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>static TS_DrvTypeDef *</td>
<td>TsDrv</td>
</tr>
<tr>
<td>static uint16_t</td>
<td>TsXBoundary</td>
</tr>
<tr>
<td>static uint16_t</td>
<td>TsYBoundary</td>
</tr>
</tbody>
</table>
Variable Documentation

**TS_DrvTypeDef**  **TsDrv**  [static]

Definition at line 80 of file *stm32f429i_discovery_ts.c*.

Referenced by **BSP_TS_GetState()**, **BSP_TS_Init()**, **BSP_TS_ITClear()**, **BSP_TS_ITConfig()**, and **BSP_TS_ITGetStatus()**.

**uint16_t**  **TsXBoundary**  [static]

Definition at line 81 of file *stm32f429i_discovery_ts.c*.

Referenced by **BSP_TS_GetState()**, and **BSP_TS_Init()**.

**uint16_t**  **TsYBoundary**  [static]

Definition at line 81 of file *stm32f429i_discovery_ts.c*.

Referenced by **BSP_TS_GetState()**, and **BSP_TS_Init()**.
stm32f429i_discovery.h

Go to the documentation of this file.

/

**************************
**************************

* @file stm32f429i_discovery.h
* @author MCD Application Team
* @version V2.1.3
* @date 13-January-2016
* @brief This file contains definitions for STM32F429I-Discovery Kit LEDs,
  push-buttons hardware resources.

**************************
**************************

* @attention

* <h2><center>&copy; COPYRIGHT(c) 2015 STM microelectronics</center></h2>

* Redistribution and use in source and binary forms, with or without modification,
  are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```c
/* Define to prevent recursive inclusion -------------------------------*/
#ifndef __STM32F429I_DISCOVERY_H
#define __STM32F429I_DISCOVERY_H

#ifdef __cplusplus
extern "C" {
#endif

/* Includes ----------------------------------
#include "stm32f4xx_hal.h"

/** @addtogroup BSP *
 */

/** @addtogroup STM32F429I_DISCOVERY *
 */

/** @addtogroup STM32F429I_DISCOVERY_LOW_LEVEL *
 */

/** @addtogroup STM32F429I_DISCOVERY_LOW_LEVEL_Exported_Types *
*/
```

typedef enum
{
    LED3 = 0,
    LED4 = 1
} Led_TypeDef;

typedef enum
{
    BUTTON_KEY = 0,
} Button_TypeDef;

typedef enum
{
    BUTTON_MODE_GPIO = 0,
    BUTTON_MODE_EXTI = 1
} ButtonMode_TypeDef;

/**
 * @}
 */
/**
 * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL_EXPORTED_CONSTANTS
 STM32F429I DISCOVERY LOW LEVEL Exported Constants
 *
 */
/**
 * @brief Define for STM32F429I_DISCO board
 *
 */
#if !defined(USE_STM32F429I_DISCO)
#define USE_STM32F429I_DISCO
#endif
/**
 * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL_LED
 STM32F429I DISCOVERY LOW LEVEL LED
 */
#define LEDn 2

#define LED3_PIN GPIO_PIN_13
#define LED3_GPIO_PORT GPIOG
#define LED3_GPIO_CLK_ENABLE() __GPIOG_CLK_ENABLE()
#define LED3_GPIO_CLK_DISABLE() __GPIOG_CLK_DISABLE()

#define LED4_PIN GPIO_PIN_14
#define LED4_GPIO_PORT GPIOG
#define LED4_GPIO_CLK_ENABLE() __GPIOG_CLK_ENABLE()
#define LED4_GPIO_CLK_DISABLE() __GPIOG_CLK_DISABLE()

#define LEDx_GPIO_CLK_ENABLE(__INDEX__) do{
    if((__INDEX__) == 0) LED3_GPIO_CLK_ENABLE(); else 
    if((__INDEX__) == 1) LED4_GPIO_CLK_ENABLE(); \
}while(0)

#define LEDx_GPIO_CLK_DISABLE(__INDEX__) do{
    if((__INDEX__) == 0) LED3_GPIO_CLK_DISABLE(); else 
    if((__INDEX__) == 1) LED4_GPIO_CLK_DISABLE(); \
}while(0)
/**
   * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL _BUTTON STM32F429I DISCOVERY LOW LEVEL BUTTON
   * @{
   */
#define BUTTONn 1
/**
   * @brief Wakeup push-button
   */
#define KEY_BUTTON_PIN GPIO_PIN_0
#define KEY_BUTTON_GPIO_PORT GPIOA
#define KEY_BUTTON_GPIO_CLK_ENABLE() __GPIOA_CLK_ENABLE()
#define KEY_BUTTON_GPIO_CLK_DISABLE() __GPIOA_CLK_DISABLE()
#define KEY_BUTTON_EXTI_IRQn EXTI0_IRQn
#define BUTTONx_GPIO_CLK_ENABLE(__INDEX__) do{if((__INDEX__) == 0) KEY_BUTTON_GPIO_CLK_ENABLE(); \
} while(0)
#define BUTTONx_GPIO_CLK_DISABLE(__INDEX__) do{if((__INDEX__) == 0) KEY_BUTTON_GPIO_CLK_DISABLE(); \
} while(0)
/**
   * @}
*/
/**
 * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL
 _BUS STM32F429I DISCOVERY LOW LEVEL BUS
 */

/* Exported constanIO ----------------------
----------------------------------*/
#define IO_I2C_ADDRESS 0x82
#define TS_I2C_ADDRESS 0x82

#ifdef EE_M24LR64
#define EEPROM_I2C_ADDRESS_A01 0xA0
#define EEPROM_I2C_ADDRESS_A02 0xA6
#endif /* EE_M24LR64 */
/*###############################
##################################*/
/*
User can use this section to tailor I2Cx instance used and associated
resources */
#define DISCOVERY_I2Cx I2C3
#define DISCOVERY_I2Cx_CLOCK_ENABLE() __I2C3_CLK_ENABLE()
#define DISCOVERY_I2Cx_FORCE_RESET() __I2C3_FORCE_RESET()
#define DISCOVERY_I2Cx_RELEASE_RESET() __I2C3_RELEASE_RESET()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE() __GPIOC_CLK_ENABLE()
#define DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE() __GPIOA_CLK_ENABLE()
```c
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE() __GPIOC_CLK_DISABLE()

/* Definition for DISCO I2Cx Pins */
#define DISCOVERY_I2Cx_SCL_PIN GPIO_PIN_8
#define DISCOVERY_I2Cx_SCL_GPIO_PORT GPIOA
#define DISCOVERY_I2Cx_SCL_SDA_AF GPIO_AF4_I2C3
#define DISCOVERY_I2Cx_SDA_PIN GPIO_PIN_9
#define DISCOVERY_I2Cx_SDA_GPIO_PORT GPIOC

/* Definition for IOE I2Cx's NVIC */
#define DISCOVERY_I2Cx_EV_IRQn I2C3_EV_IRQn
#define DISCOVERY_I2Cx_ER_IRQn I2C3_ER_IRQn

/* I2C clock speed configuration (in Hz)
WARNING:
Make sure that this define is not already declared in other files.
It can be used in parallel by other modules. */
#ifndef BSP_I2C_SPEED
#define BSP_I2C_SPEED
#define BSP_I2C_SPEED 100000
#endif /* BSP_I2C_SPEED */

#define I2Cx_TIMEOUT_MAX 0x3000 /*<! The value of the maximal timeout for I2C waiting loops */

/* `spix` */
00189 #define DISCOVERY_SPIx
    SPI5
00190 #define DISCOVERY_SPIx_CLK_ENABLE()
    __SPI5_CLK_ENABLE()
00191 #define DISCOVERY_SPIx_GPIO_PORT
    GPIOF      /* GPIOF */
00192 #define DISCOVERY_SPIx_AF
    GPIO_AF5_SPI5
00193 #define DISCOVERY_SPIx_GPIO_CLK_ENABLE()
    __GPIOF_CLK_ENABLE()
00194 #define DISCOVERY_SPIx_GPIO_CLK_DISABLE()
    __GPIOF_CLK_DISABLE()
00195 #define DISCOVERY_SPIx_SCK_PIN
    GPIO_PIN_7      /* PF.07 */
00196 #define DISCOVERY_SPIx_MISO_PIN
    GPIO_PIN_8      /* PF.08 */
00197 #define DISCOVERY_SPIx_MOSI_PIN
    GPIO_PIN_9      /* PF.09 */
00198 /* Maximum Timeout values for flags waiting loops. These timeouts are not based
    on accurate values, they just guarantee that the application will not remain
    stuck if the SPI communication is corrupted.
    You may modify these timeout values depending on CPU frequency and application
    conditions (interrupts routines ...). */
00203 #define SPIx_TIMEOUT_MAX      ((uint32_t)0x1000)
00204
00205
00206 /*############################################################################### IOE ####
   ###############################################################################*/
00207 /**
00208    * @brief IOE Control pin
/* Definition for external IT for STMPE811 */

#define STMPE811_INT_PIN GPIO_PIN_15
#define STMPE811_INT_GPIO_PORT GPIOA
#define STMPE811_INT_CLK_ENABLE() __GPIOA_CLK_ENABLE()
#define STMPE811_INT_CLK_DISABLE() __GPIOA_CLK_DISABLE()
#define STMPE811_INT_EXTI EXTI15_10_IRQn
#define STMPE811_INT_EXTIHandler EXTI15_10_IRQHandler

/**************************** LCD ################
Chip Select macro definition */
#define LCD_CS_LOW() HAL_GPIO_WritePin (LCD_NCS_GPIO_PORT, LCD_NCS_PIN, GPIO_PIN_RESET)
#define LCD_CS_HIGH() HAL_GPIO_WritePin (LCD_NCS_GPIO_PORT, LCD_NCS_PIN, GPIO_PIN_SET)

/**************************** LCD ################
Set WRX High to send data */
#define LCD_WRX_LOW() HAL_GPIO_WritePin (LCD_WRX_GPIO_PORT, LCD_WRX_PIN, GPIO_PIN_RESET)
#define LCD_WRX_HIGH() HAL_GPIO_WritePin (LCD_WRX_GPIO_PORT, LCD_WRX_PIN, GPIO_PIN_SET)

/**************************** LCD ################
Set WRX High to send data */
#define LCD_RDX_LOW() HAL_GPIO_WritePin (LCD_RDX_GPIO_PORT, LCD_RDX_PIN, GPIO_PIN_RESET)
#define LCD_RDX_HIGH() HAL_GPIO_WritePin (LCD_RDX_GPIO_PORT, LCD_RDX_PIN, GPIO_PIN_SET)

/**
/*
@brief LCD Control pin
*/
#define LCD_NCS_PIN GPIO_PIN_2
#define LCD_NCS_GPIO_PORT GPIOC
#define LCD_NCS_GPIO_CLK_ENABLE() __GPIOC_CLK_ENABLE()
#define LCD_NCS_GPIO_CLK_DISABLE() __GPIOC_CLK_DISABLE()

/** *
@brief LCD Command/data pin
*/
#define LCD_WRX_PIN GPIO_PIN_13
#define LCD_WRX_GPIO_PORT GPIOD
#define LCD_WRX_GPIO_CLK_ENABLE() __GPIOD_CLK_ENABLE()
#define LCD_WRX_GPIO_CLK_DISABLE() __GPIOD_CLK_DISABLE()

#define LCD_RDX_PIN GPIO_PIN_12
#define LCD_RDX_GPIO_PORT GPIOD
#define LCD_RDX_GPIO_CLK_ENABLE() __GPIOD_CLK_ENABLE()
#define LCD_RDX_GPIO_CLK_DISABLE() __GPIOD_CLK_DISABLE()

/*################################	GYROSCOPE
#################################*/
/*	Read/Write command */
#define READWRITE_CMD ((uint8_t)0x80)

/* Multiple byte read/write command */
#define MULTIPLEBYTE_CMD ((uint8_t)0x40)

/* Dummy Byte Send by the SPI Master device in order to generate the Clock to the Slave device */
#define DUMMY_BYTE ((uint8_t)0x00)

/* Chip Select macro definition */
#define GYRO_CS_LOW() HAL_GPIO_WritePin(GYRO_CS_GPIO_PORT, GYRO_CS_PIN, GPIO_PIN_RESET)
#define GYRO_CS_HIGH() HAL_GPIO_WritePin(GYRO_CS_GPIO_PORT, GYRO_CS_PIN, GPIO_PIN_SET)

/**
 * @brief GYROSCOPE SPI Interface pins
 */
#define GYRO_CS_PIN GPIO_PIN_1 /* PC.01 */
#define GYRO_CS_GPIO_PORT GPIOC /* GPIOC */
#define GYRO_CS_GPIO_CLK_ENABLE() __GPIOC_CLK_ENABLE()
#define GYRO_CS_GPIO_CLK_DISABLE() __GPIOC_CLK_DISABLE()

#define GYRO_INT_GPIO_CLK_ENABLE() __GPIOA_CLK_ENABLE()
#define GYRO_INT_GPIO_CLK_DISABLE() __GPIOA_CLK_DISABLE()
#define GYRO_INT_GPIO_PORT GPIOA /* GPIOA */
#define GYRO_INT1_PIN GPIO_PIN_1 /* PA.01 */
#define GYRO_INT1_EXTI_IRQn EXTI1_IRQHandler

#define GYRO_INT2_PIN GPIO_PIN_2 /* PA.02 */

#define GYRO_INT2_EXTI_IRQn EXTI2_IRQHandler

/**
 * @}
 */

#define EE_M24LR64

/**
 * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL_I2C_EEPROM
STM32F429I DISCOVERY LOW LEVEL I2C EEPROM
*/

* @{
*/
/**
 * @brief I2C EEPROM Interface pins
*/

#define EEPROM_I2C_DMA_DMA1

#define EEPROM_I2C_DMA_CHANNEL DMA_CHANNEL_3

#define EEPROM_I2C_DMA_STREAM_TX DMA1_Stream4

#define EEPROM_I2C_DMA_STREAM_RX DMA1_Stream2

#define EEPROM_I2C_DMA_CLK_ENABLE() __DMA1_CLK_ENABLE()

#define EEPROM_I2C_DMA_TX_IRQn DMA1_Stream4_IRQn

#define EEPROM_I2C_DMA_RX_IRQn DMA1_Stream2_IRQn

#define EEPROM_I2C_DMA_TX_IRQHandler DMA1_Stream4_IRQHandler

#define EEPROM_I2C_DMA_RX_IRQHandler

#define EEPROM_I2C_DMA_TX_IRQHandler DMA1_Stream4_IRQHandler

#define EEPROM_I2C_DMA_RX_IRQHandler

#define EEPROM_I2C_DMA_TX_IRQHandler DMA1_Stream4_IRQHandler

#define EEPROM_I2C_DMA_RX_IRQHandler

#define EEPROM_I2C_DMA_TX_IRQHandler DMA1_Stream4_IRQHandler

#define EEPROM_I2C_DMA_RX_IRQHandler

#define EEPROM_I2C_DMA_TX_IRQHandler DMA1_Stream4_IRQHandler

#define EEPROM_I2C_DMA_RX_IRQHandler
DMA1_Stream2_IRQHandler

#define EEPROM_I2C_DMA_PREPRIO 0

/**
 * @}
 */

#ifdef EE_M24LR64

/**
 * @} STM32F429I_DISCOVERY_LOW_LEVEL_Exported_Macros STM32F429I DISCOVERY LOW LEVEL Exported Macros
 * @{ */
 */
/**
 * @} STM32F429I_DISCOVERY_LOW_LEVEL_Exported_Functions STM32F429I DISCOVERY LOW LEVEL Exported Functions
 * @{ */
 */
uint32_t BSP_GetVersion(void);
void BSP_LED_Init(Led_TypeDef Led);
void BSP_LED_On(Led_TypeDef Led);
void BSP_LED_Off(Led_TypeDef Led);
void BSP_LED_Toggle(Led_TypeDef Led);
void BSP_PB_Init(Button_TypeDef Button, ButtonMode_TypeDef ButtonMode);
uint32_t BSP_PB_GetState(Button_TypeDef Button);
*/
/**
 * @}
 */

*/
stm32f429i_discovery.c

Go to the documentation of this file.

```c
/**
   ******************************************
   ************************************
   * @file stm32f429i_discovery.c
   * @author MCD Application Team
   * @version V2.1.3
   * @date 13-January-2016
   * @brief This file provides set of firmware
   *       functions to manage Leds and
   *       push-button available on STM32F429I-Discovery Kit from STMicroelectronics.
   ******************************************
   ************************************
   * @attention
   *
   * <h2><center>&copy; COPYRIGHT(c) 2015 STMicroelectronics</center></h2>
   * Redistributio
   * 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.
```c
#include "stm32f429i_discovery.h"

/**
 * @defgroup BSP BSP
 */

/**
 * @defgroup STM32F429I_DISCOVERY STM32F429I DISCOVERY
 */

/**
 * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL STM32F429I DISCOVERY LOW LEVEL
 * @brief This file provides set of firmware functions to manage Leds and push-button available on STM32F429I-Discovery Kit from STMicroelectronics.
 */

/**
 * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL_Private_TypesDefinitions STM32F429I DISCOVERY LOW LEVEL Private TypesDefinitions
 */

/**
 * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL
 */
```
_Private_Defines STM32F429I DISCOVERY LOW LEVEL Private Defines

00064  * @{
00065  */
00066
00067  /**
00068  * @brief STM32F429I DISCO BSP Driver version number V2.1.3
00069  */
00070  #define __STM32F429I_DISCO_BSP_VERSION_MAIN (0x02) /*!< [31:24] main version */
00071  #define __STM32F429I_DISCO_BSP_VERSION_SUB1 (0x01) /*!< [23:16] sub1 version */
00072  #define __STM32F429I_DISCO_BSP_VERSION_SUB2 (0x03) /*!< [15:8] sub2 version */
00073  #define __STM32F429I_DISCO_BSP_VERSION_RC (0x00) /*!< [7:0] release candidate */
00074  #define __STM32F429I_DISCO_BSP_VERSION
((__STM32F429I_DISCO_BSP_VERSION_MAIN << 24)\n00075  |(__STM32F429I_DISCO_BSP_VERSION_SUB1 << 16)\n00076  |(__STM32F429I_DISCO_BSP_VERSION_SUB2 << 8 )\n00077  |(__STM32F429I_DISCO_BSP_VERSION_RC))
00078  /**
00079  * @}}
00080  */
00081
00082  /** @defgroup STM32F429I_DISCOVERY_LOW_LEVEL_Private_Macros STM32F429I DISCOVERY LOW LEVEL Private Macros
00083  * @{
00084  */
00085  /**
00086  * @}}
00087  */
/**
@defgroup STM32F429I_DISCOVERY_LOW_LEVEL
_Private_Variables STM32F429I DISCOVERY LOW LEVEL Private Variables
*/

GPIO_TypeDef* GPIO_PORT[LEDn] = {LED3_GPIO_PORT, LED4_GPIO_PORT};

const uint16_t GPIO_PIN[LEDn] = {LED3_PIN, LED4_PIN};

GPIO_TypeDef* BUTTON_PORT[BUTTONn] = {KEY_BUTTON_GPIO_PORT};

const uint16_t BUTTON_PIN[BUTTONn] = {KEY_BUTTON_PIN};

const uint8_t BUTTON_IRQn[BUTTONn] = {KEY_BUTTON_EXTI_IRQn};

uint32_t I2cxTimeout = I2Cx_TIMEOUT_MAX; /**<
! Value of Timeout when I2C communication fails */

uint32_t SpixTimeout = SPIx_TIMEOUT_MAX; /**<
! Value of Timeout when SPI communication fails */

I2C_HandleTypeDef I2cHandle;
static SPI_HandleTypeDef SpiHandle;
static uint8_t Is_LCD_IO_Initialized = 0;

/**
* @}*/
/* I2Cx bus function */
static void I2Cx_Init(void);
static void I2Cx_ITConfig(void);
static void I2Cx_WriteData(uint8_t Addr, uint8_t Reg, uint8_t Value);
static void I2Cx_WriteBuffer(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length);
static uint8_t I2Cx_ReadData(uint8_t Addr, uint8_t Reg);
static uint8_t I2Cx_ReadBuffer(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length);
static void I2Cx_Error(void);
static void I2Cx_MspInit(I2C_HandleTypeDef *hi2c);
#ifdef EE_M24LR64
static HAL_StatusTypeDef I2Cx_WriteBufferDMA(uint8_t Addr, uint16_t Reg, uint8_t *pBuffer, uint16_t Length);
static HAL_StatusTypeDef I2Cx_ReadBufferDMA(uint8_t Addr, uint16_t Reg, uint8_t *pBuffer, uint16_t Length);
static HAL_StatusTypeDef I2Cx_IsDeviceReady(uint16_t DevAddress, uint32_t Trials);
#endif /* EE_M24LR64 */
/* SPIx bus function */
static void SPIx_Init(void);
static void SPIx_Write(uint16_t Value);
static uint32_t SPIx_Read(uint8_t...
00135 static uint8_t SPIx_WriteRead(uint8_t Byte);
00136 static void SPIx_Error(void);
00137 static void SPIx_MspInit(SPI_HandleTypeDef *hspi);
00138
00139 /* Link function for LCD peripheral */
00140 void LCD_IO_Init(void);
00141 void LCD_IO_WriteData(int16_t RegValue);
00142 void LCD_IO_WriteReg(uint8_t Reg);
00143 uint32_t LCD_IO_ReadData(int16_t RegValue, uint8_t ReadSize);
00144 void LCD_Delay(uint32_t delay);
00145
00146 /* IOExpander IO functions */
00147 void IOE_Init(void);
00148 void IOE_ITConfig(void);
00149 void IOE_Delay(uint32_t Delay);
00150 void IOE_Write(uint8_t Addr, uint8_t Reg, uint8_t Value);
00151 uint8_t IOE_Read(uint8_t Addr, uint8_t Reg);
00152 uint16_t IOE_ReadMultiple(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length);
00153 void IOE_WriteMultiple(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length);
00154
00155 /* Link function for GYRO peripheral */
00156 void GYRO_IO_Init(void)
void GYRO_IO_Write(uint8_t* pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite);

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

#ifdef EE_M24LR64
/* Link function for I2C EEPROM peripheral */

void EEPROM_IO_Init(void);

HAL_StatusTypeDef EEPROM_IO_WriteData(uint16_t DevAddress, uint16_t MemAddress, uint8_t* pBuffer, uint32_t BufferSize);

HAL_StatusTypeDef EEPROM_IO_ReadData(uint16_t DevAddress, uint16_t MemAddress, uint8_t* pBuffer, uint32_t BufferSize);

HAL_StatusTypeDef EEPROM_IO_IsDeviceReady(uint16_t DevAddress, uint32_t Trials);
#endif /* EE_M24LR64 */

/** @defgroup STM32F429I_DISCOVERY_LOW_LEVEL_Private_Functions STM32F429I DISCOVERY LOW LEVEL Private Functions */

/* @brief This method returns the STM32F429I DISCO BSP Driver revision */

@retval version: 0xXYZR (8bits for each
decimal, R for RC)
00179   */
00180        uint32_t BSP_GetVersion(void)
00181        {
00182            return __STM32F429I_DISCO_BSP_VERSION;
00183        }
00184
00185        /**<
00186        * @brief Configures LED GPIO.
00187        * @param Led: Specifies the Led to be configured.
00188        * This parameter can be one of following parameters:
00189        *   @arg LED3
00190        *   @arg LED4
00191        */
00192        void BSP_LED_Init(Led_TypeDef Led)
00193        {
00194            GPIO_InitTypeDef    GPIO_InitStruct;
00195        */
00196        /* Enable the GPIO_LED Clock */
00197            LEDx_GPIO_CLK_ENABLE(Led);
00198        /* Configure the GPIO_LED pin */
00199            GPIO_InitStruct.Pin = GPIO_PIN[Led];
00200            GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
00201            GPIO_InitStruct.Pull = GPIO_PULLUP;
00202            GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
00203            HAL_GPIO_Init(GPIO_PORT[Led], &GPIO_InitStruct);
00204            HAL_GPIO_WritePin(GPIO_PORT[Led], GPIO_PIN[Led], GPIO_PIN_RESET);
00205    }
00206
/**
 * @brief Turns selected LED On.
 * @param Led: Specifies the Led to be set on.
 * This parameter can be one of following parameters:
 * @arg LED3
 * @arg LED4
 */

void BSP_LED_On(Led_TypeDef Led)
{
    HAL_GPIO_WritePin(GPIO_PORT[Led], GPIO_PIN[Led], GPIO_PIN_SET);
}

/**
 * @brief Turns selected LED Off.
 * @param Led: Specifies the Led to be set off.
 * This parameter can be one of following parameters:
 * @arg LED3
 * @arg LED4
 */

void BSP_LED_Off(Led_TypeDef Led)
{
    HAL_GPIO_WritePin(GPIO_PORT[Led], GPIO_PIN[Led], GPIO_PIN_RESET);
}

/**
 * @brief Toggles the selected LED.
 * @param Led: Specifies the Led to be toggled.
 * This parameter can be one of following parameters:
 * @arg LED3
 */
void BSP_LED_Toggle(Led_TypeDef Led) {
    HAL_GPIO_TogglePin(GPIO_PORT[Led], GPIO_PIN[Led]);
}

/**
 * @brief Configures Button GPIO and EXTI Line.
 * @param Button: Specifies the Button to be configured.
 * This parameter should be: BUTTON_KEY
 * @param ButtonMode: Specifies Button mode.
 * This parameter can be one of following parameters:
 * @arg BUTTON_MODE_GPIO: Button will be used as simple IO
 * @arg BUTTON_MODE_EXTI: Button will be connected to EXTI line with interrupt generation capability
 */
void BSP_PB_Init(Button_TypeDef Button, ButtonMode_TypeDef ButtonMode) {
    GPIO_InitTypeDef GPIO_InitStruct;

    /* Enable the BUTTON Clock */
    BUTTONx_GPIO_CLK_ENABLE(Button);

    if (ButtonMode == BUTTON_MODE_GPIO) {
        /* Configure Button pin as input */
        GPIO_InitStruct.Pin = BUTTON_PIN[Button]
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_PULLDOWN;
GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
HAL_GPIO_Init(BUTTON_PORT[Button], &GPIO_InitStruct);

if (ButtonMode == BUTTON_MODE_EXTI)
{
    /* Configure Button pin as input with External interrupt */
    GPIO_InitStruct.Pin = BUTTON_PIN[Button];
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    GPIO_InitStruct.Mode = GPIO_MODE_IT_RISING;
    HAL_GPIO_Init(BUTTON_PORT[Button], &GPIO_InitStruct);

    /* Enable and set Button EXTI Interrupt to the lowest priority */
    HAL_NVIC_SetPriority((IRQn_Type)(BUTTON_IRQHandler[Button]), 0x0F, 0x00);
    HAL_NVIC_EnableIRQ((IRQn_Type)(BUTTON_IRQHandler[Button]));
}

/**
 * @brief Returns the selected Button state.
 * @param Button: Specifies the Button to be checked.
 * This parameter should be: BUTTON_KEY
 * @retval The Button GPIO pin value.
 */
`uint32_t BSP_PB_GetState(Button_TypeDef Button)`

```c
return HAL_GPIO_ReadPin(BUTTON_PORT[Button], BUTTON_PIN[Button]);
```

```c
/********************
*********

BUS OPERATIONS
*********

/****************************
I2C Routine
**************************//*

/**
@brief I2Cx MSP Initialization
@param hi2c: I2C handle
*/

static void I2Cx_MspInit(I2C_HandleTypeDef *hi2c)
{
    GPIO_InitTypeDef GPIO_InitStruct;
    #ifdef EE_M24LR64
    static DMA_HandleTypeDef hdma_tx;
    static DMA_HandleTypeDef hdma_rx;
    I2C_HandleTypeDef* pI2cHandle;
    pI2cHandle = &I2cHandle;
    #endif /* EE_M24LR64 */
    if (hi2c->Instance == DISCOVERY_I2Cx) {
        /* Configure the GPIOs -------------------------------*/
        /* Enable GPIO clock */
```
/* Configure I2C Tx as alternate function */
GPIO_InitStruct.Pin = DISCOVERY_I2Cx_SCL_PIN;
GPIO_InitStruct.Mode = GPIO_MODE_AF_OD;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_FAST;
GPIO_InitStruct.Alternate = DISCOVERY_I2Cx_SCL_SDA_AF;
HAL_GPIO_Init(DISCOVERY_I2Cx_SCL_GPIO_PORT, &GPIO_InitStruct);

/* Configure I2C Rx as alternate function */
GPIO_InitStruct.Pin = DISCOVERY_I2Cx_SDA_PIN;
HAL_GPIO_Init(DISCOVERY_I2Cx_SDA_GPIO_PORT, &GPIO_InitStruct);

/* Configure the Discovery I2Cx peripheral */
/* Enable I2C3 clock */
DISCOVERY_I2Cx_CLOCK_ENABLE();
/* Force the I2C Peripheral Clock Reset */
DISCOVERY_I2Cx_FORCE_RESET();
/* Release the I2C Peripheral Clock Reset */
DISCOVERY_I2Cx_RELEASE_RESET();
00348 /* Enable and set Discovery I2Cx Interrupt to the highest priority */
00349 HAL_NVIC_SetPriority(DISCOVERY_I2Cx_EV_IRQn, 0x00, 0);
00350 HAL_NVIC_EnableIRQ(DISCOVERY_I2Cx_EV_IRQn);
00351
00352 /* Enable and set Discovery I2Cx Interrupt to the highest priority */
00353 HAL_NVIC_SetPriority(DISCOVERY_I2Cx_ER_IRQn, 0x00, 0);
00354 HAL_NVIC_EnableIRQ(DISCOVERY_I2Cx_ER_IRQn);
00355
00356 #ifdef EE_M24LR64
00357 /* I2C DMA TX and RX channels configuration */
00358 /* Enable the DMA clock */
00359 EEPROM_I2C_DMA_CLK_ENABLE();
00360
00361 /* Configure the DMA stream for the EE I2C peripheral TX direction */
00362 /* Configure the DMA Stream */
00363 hdma_tx.Instance = EEPROM_I2C_DMA_STREAM_TX;
00364 /* Set the parameters to be configured */
00365
00366 hdma_tx.Init.Channel = EEPROM_I2C_DMA_CHANNEL;
00367 hdma_tx.Init.Direction = DMA_MEMORY_TO_PERIPH;
00368 hdma_tx.InitPeriphInc = DMA_PINC_DISABLE;
00369 hdma_tx.Init.MemInc = DMA_MINC_ENABLE;
00370 hdma_tx.Init.PeriphDataAlignment = DMA_
DATAALIGN_BYTE;
00371  hdma_tx.Init.MemDataAlignment   = DMA_{
MDATAALIGN_BYTE;
00372  hdma_tx.Init.Mode               = DMA_{
NORMAL;
00373  hdma_tx.Init.Priority           = DMA_{
PRIORITY_VERY_HIGH;
00374  hdma_tx.Init.FIFOMode          = DMA_{
FIFOMODE_ENABLE;
00375  hdma_tx.Init.FIFOThreshold     = DMA_{
FIFO_THRESHOLD_FULL;
00376  hdma_tx.Init.MemBurst          = DMA_{
MBURST_SINGLE;
00377  hdma_tx.Init.PeriphBurst       = DMA_{
PBURST_SINGLE;
00378
00379  /* Associate the initialized hdma_tx handle to the the pI2cHandle handle */
00380   __HAL_LINKDMA(pI2cHandle, hdmatx, hdma_t
00381
00382  /* Configure the DMA Stream */
00383   HAL_DMA_Init(&hdma_tx);
00384
00385  /* Configure and enable I2C DMA TX Channel interrupt */
00386   HAL_NVIC_SetPriority((IRQn_Type)(EEPROM_I2C_DMA_TX_IRQn), EEPROM_I2C_DMA_PREPRIOR, 0);
00387   HAL_NVIC_EnableIRQ((IRQn_Type)(EEPROM_I2
00388
00389  /* Configure the DMA stream for the EE I
00390  /* Configure the DMA Stream */
00391   hdma_rx.Instance               = EEPROM
00392  /* Set the parameters to be configured */
00393    hdma_rx.Init.Channel    = EEPROM_I2C_DMA_CHANNEL;
00394    hdma_rx.Init.Direction  = DMA_PERIPH_TO_MEMORY;
00395    hdma_rx.Init.PeriphInc  = DMA_PINC_DISABLE;
00396    hdma_rx.Init.MemInc    = DMA_MINC_ENABLE;
00397    hdma_rx.Init.PeriphDataAlignment = DMA_PDATAALIGN_BYTE;
00398    hdma_rx.Init.MemDataAlignment  = DMA_MDATAALIGN_BYTE;
00399    hdma_rx.Init.Mode      = DMA_NORMAL;
00400    hdma_rx.Init.Priority  = DMA_PRIORITY_VERY_HIGH;
00401    hdma_rx.Init.FIFOMode  = DMA_FIFOMODE_ENABLE;
00402    hdma_rx.Init.FIFOThreshold = DMA_FIFO_THRESHOLD_FULL;
00403    hdma_rx.Init.MemBurst  = DMA_MBURST_SINGLE;
00404    hdma_rx.Init.PeriphBurst = DMA_PBURST_SINGLE;
00405
00406    /* Associate the initialized hdma_rx handle to the the pI2cHandle handle*/
00407    __HAL_LINKDMA(pI2cHandle, hdmarx, hdma_rx);
00408
00409    /* Configure the DMA Stream */
00410    HAL_DMA_Init(&hdma_rx);
00411
00412    /* Configure and enable I2C DMA RX Channel interrupt */
00413    HAL_NVIC_SetPriority((IRQn_Type)(EEPROM_
I2C_DMA_RX_IRQHandler, EEPROM_I2C_DMA_PREPrio, 0);

HAL_NVIC_EnableIRQ((IRQn_Type)(EEPROM_I2C_DMA_RX_IRQn));

/*	EE_M24LR64 */
#endif

/**
 * @brief I2Cx Bus initialization.
 */

static void I2Cx_Init(void)
{
    if(HAL_I2C_GetState(&I2cHandle) == HAL_I2C_STATE_RESET)
    {
        I2cHandle.Instance = DISCOVERY_I2Cx;
        I2cHandle.Init.ClockSpeed = BSP_I2C_SPEED;
        I2cHandle.Init.DutyCycle = I2C_DUTYCYCLE_2;
        I2cHandle.Init.OwnAddress1 = 0;
        I2cHandle.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
        I2cHandle.Init.DualAddressMode = I2C_DUALADDRESS_DISABLED;
        I2cHandle.Init.OwnAddress2 = 0;
        I2cHandle.Init.GeneralCallMode = I2C_GENERALCALL_DISABLED;
        I2cHandle.Init.NoStretchMode = I2C_NO_STRETCH_DISABLED;
        /* Init the I2C */
        I2Cx_MspInit(&I2cHandle);
        HAL_I2C_Init(&I2cHandle);
    }
}
/**
 * @brief Configures Interruption pin for I2C communication.
 */
static void I2Cx_ITConfig(void)
{
    GPIO_InitTypeDef GPIO_InitStruct;
    /* Enable the GPIO EXTI Clock */
    STMPE811_INT_CLK_ENABLE();
    GPIO_InitStruct.Pin = STMPE811_INT_PIN;
    GPIO_InitStruct.Pull = GPIO_PULLUP;
    GPIO_InitStruct.Speed = GPIO_SPEED_LOW;
    GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
    HAL_GPIO_Init(STMPE811_INT_GPIO_PORT, &GPIO_InitStruct);
    /* Enable and set GPIO EXTI Interrupt to the highest priority */
    HAL_NVIC_SetPriority((IRQn_Type)(STMPE811_INT_EXTI), 0x00, 0x00);
    HAL_NVIC_EnableIRQ((IRQn_Type)(STMPE811_INT_EXTI));
}

/**
 * @brief Writes a value in a register of the device through BUS.
 * @param Addr: Device address on BUS Bus.
 * @param Reg: The target register address to write
 * @param Value: The target register value to be written
static void I2Cx_WriteData(uint8_t Addr, uint8_t Reg, uint8_t Value)
{
    HAL_StatusTypeDef status = HAL_OK;
    status = HAL_I2C_Mem_Write(&I2cHandle, Addr, (uint16_t)Reg, I2C_MEMADD_SIZE_8BIT, &Value, 1, I2cxTimeout);
    /* Check the communication status */
    if (status != HAL_OK)
    {
        /* Re-Initialize the BUS */
        I2Cx_Error();
    }
}

/**
 * @brief Writes a value in a register of the device through BUS.
 * @param Addr: Device address on BUS Bus.
 * @param Reg: The target register address to write
 * @param pBuffer: The target register value to be written
 * @param Length: buffer size to be written
 */
static void I2Cx_WriteBuffer(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)
/**
 * @brief Reads a register of the device through BUS.
 *
 * @param Addr: Device address on BUS Bus.
 *
 * @param Reg: The target register address to write
 *
 * @retval Data read at register address
 */

static uint8_t I2Cx_ReadData(uint8_t Addr, uint8_t Reg)
{
    HAL_StatusTypeDef status = HAL_OK;
    uint8_t value = 0;

    status = HAL_I2C_Mem_Read(&I2cHandle, Addr, Reg, I2C_MEMADD_SIZE_8BIT, &value, 1, I2cxTimeout);

    if(status != HAL_OK)
    {
        /* Re-Initialize the BUS */
        I2Cx_Error();
    }

    return value;

static uint8_t I2Cx_ReadBuffer(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length){
    HAL_StatusTypeDef status = HAL_OK;
    status = HAL_I2C_Mem_Read(&I2cHandle, Addr, (uint16_t)Reg, I2C_MEMADD_SIZE_8BIT, pBuffer, Length, I2cxTimeout);
    /* Check the communication status */
    if(status == HAL_OK){
        return 0;
    } else {
        /* Re-Initialize the BUS */
        I2Cx_Error();
        return 1;
    }
}

#ifdef EE_M24LR64

#endif
* @brief Writes a value in a register of the device through BUS in using DMA mode.
* @param Addr: Device address on BUS Bus.
* @param Reg: The target register address to write
* @param pBuffer: The target register value to be written
* @param Length: buffer size to be written
* @retval HAL status

static HAL_StatusTypeDef I2Cx_WriteBufferDMA(uint8_t Addr, uint16_t Reg, uint8_t *pBuffer, uint16_t Length)
{
    HAL_StatusTypeDef status = HAL_OK;
    status = HAL_I2C_Mem_Write_DMA(&I2cHandle, Addr, Reg, I2C_MEMADD_SIZE_16BIT, pBuffer, Length);
    /* Check the communication status */
    if(status != HAL_OK)
    {
        /* Re-Initialize the BUS */
        I2Cx_Error();
    }
    return status;
}
static HAL_StatusTypeDef I2Cx_ReadBufferDMA(uint8_t Addr, uint16_t Reg, uint8_t *pBuffer, uint16_t Length) {

    HAL_StatusTypeDef status = HAL_OK;

    status = HAL_I2C_Mem_Read_DMA(&I2cHandle, Addr, Reg, I2C_MEMADD_SIZE_16BIT, pBuffer, Length);

    /* Check the communication status */
    if (status != HAL_OK) {
        /* Re-Initialize the BUS */
        I2Cx_Error();
    }

    return status;
}

/**
 * @brief Checks if target device is ready for communication.
 * @param DevAddress: Target device address
 * @param Trials: Number of trials
 * @retval HAL status
 */
static HAL_StatusTypeDef I2Cx_IsDeviceReady(uint16_t DevAddress, uint32_t Trials) {

    */
return (HAL_I2C_IsDeviceReady(&I2cHandle, DevAddress, Trials, I2cxTimeout));

#endif /* EE_M24LR64 */

/**
 * @brief I2Cx error treatment function
 */
static void I2Cx_Error(void)
{
    /* De-initialize the SPI communication BUS */
    HAL_I2C_DeInit(&I2cHandle);
    /* Re-Initialize the SPI communication BUS */
    I2Cx_Init();
}

/******************************* SPI Routine s ********************************/

/**
 * @brief SPIx Bus initialization
 */
static void SPIx_Init(void)
{
    if(HAL_SPI_GetState(&SpiHandle) == HAL_SPI_STATE_RESET)
    {
        /* SPI configuration ----------------------------
        -----------------------------------------------*/
        SpiHandle.Instance = DISCOVERY_SPIx;
        /* SPI baudrate is set to 5.6 MHz (PCLK2 /SPI_BaudRatePrescaler = 90/16 = 5.625 MHz)
        to verify these constraints:
        - ILI9341 LCD SPI interface max baudr
ate is 10MHz for write and 6.66MHz for read

- l3gd20 SPI interface max baudrate is 10MHz for write/read
- PCLK2 frequency is set to 90 MHz

*/

SpiHandle.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_16;

/* On STM32F429I-Discovery, LCD ID cannot be read then keep a common configuration */

/* for LCD and GYRO (SPI_DIRECTION_2LINES) */

/* Note: To read a register a LCD, SPI_DIRECTION_1LINE should be set */

SpiHandle.Init.Direction = SPI_DIRECTION_2LINES;
SpiHandle.Init.CLKPhase = SPI_PHASE_1EDGE;
SpiHandle.Init.CLKPolarity = SPI_POLARITY_LOW;
SpiHandle.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLED;
SpiHandle.Init.CRCPolynomial = 7;
SpiHandle.Init.DataSize = SPI_DATASIZE_8BIT;
SpiHandle.Init.FirstBit = SPI_FIRSTBIT_MSB;
SpiHandle.Init.NSS = SPI_NSS_SOFT;
SpiHandle.Init.TIMode = SPI_TIMODE_DISABLED;
SpiHandle.Init.Mode = SPI_MODE_MASTER;

SPIx_MspInit(&SpiHandle);
HAL_SPI_Init(&SpiHandle);
}
/**
 * @brief Reads 4 bytes from device.
 * @param ReadSize: Number of bytes to read (max 4 bytes)
 * @retval Value read on the SPI
 */
static uint32_t SPIx_Read(uint8_t ReadSize) {
    HAL_StatusTypeDef status = HAL_OK;
    uint32_t readvalue;
    status = HAL_SPI_Receive(&SpiHandle, (uint8_t*) &readvalue, ReadSize, SpixTimeout);
    /* Check the communication status */
    if (status != HAL_OK) {
        /* Re-Initialize the BUS */
        SPIx_Error();
    }
    return readvalue;
}

/**
 * @brief Writes a byte to device.
 * @param Value: value to be written
 */
static void SPIx_Write(uint16_t Value) {
    HAL_StatusTypeDef status = HAL_OK;
    status = HAL_SPI_Transmit(&SpiHandle, (uint8_t*) &Value, 1, SpixTimeout);
}
/* Check the communication status */
if (status != HAL_OK) {
    /* Re-Initialize the BUS */
    SPIx_Error();
}

/** *
@brief Sends a Byte through the SPI interface and return the Byte received from the SPI bus.
@param Byte: Byte send.
@retval The received byte value
*/
static uint8_t SPIx_WriteRead(uint8_t Byte) {
    uint8_t receivedbyte = 0;
    /* Send a Byte through the SPI peripheral */
    /* Read byte from the SPI bus */
    if (HAL_SPI_TransmitReceive(&SpiHandle, (uint8_t*) &Byte, (uint8_t*) &receivedbyte, 1, SpiTimeout) != HAL_OK) {
        SPIx_Error();
    }
    return receivedbyte;
}

/** *
@brief SPIx error treatment function.
*/
static void SPIx_Error(void) {

/* De-initialize the SPI communication BUS */
HAL_SPI_DeInit(&SpiHandle);
/* Re- Initialize the SPI communication BUS */
SPIx_Init();
}
/**
 * @brief SPI MSP Init.
 * @param hspi: SPI handle
 */
static void SPIx_MspInit(SPI_HandleTypeDef *hspi)
{
    GPIO_InitTypeDef GPIO_InitStructure;
    /* Enable SPIx clock */
    DISCOVERY_SPIx_CLK_ENABLE();
    /* Enable DISCOVERY_SPI GPIO clock */
    DISCOVERY_SPIx_GPIO_CLK_ENABLE();
    /* configure SPI SCK, MOSI and MISO */
    GPIO_InitStructure.Pin   = (DISCOVERY_SPIx_SCK_PIN | DISCOVERY_SPIx_MOSI_PIN | DISCOVERY_SPIx_MISO_PIN);
    GPIO_InitStructure.Mode  = GPIO_MODE_AF_PP;
    GPIO_InitStructure.Pull  = GPIO_PULLDOWN;
    GPIO_InitStructure.Speed = GPIO_SPEED_MEDIUM;
    GPIO_InitStructure.Alternate = DISCOVERY_SPIx_AF;
    HAL_GPIO_Init(DISCOVERY_SPIx_GPIO_PORT, &GPIO_InitStructure);
void LCD_IO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStructure;

  if(Is_LCD_IO_Initialized == 0)
  {
    Is_LCD_IO_Initialized = 1;

    /* Configure NCS in Output Push-Pull mode */
    LCD_WRX_GPIO_CLK_ENABLE();
    GPIO_InitStructure.Pin      = LCD_WRX_PIN;
    GPIO_InitStructure.Mode     = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStructure.Pull     = GPIO_NOPULL;
    GPIO_InitStructure.Speed   = GPIO_SPEED_FAST;
    HAL_GPIO_Init(LCD_WRX_GPIO_PORT, &GPIO_InitStructure);

    LCD_RDX_GPIO_CLK_ENABLE();
    GPIO_InitStructure.Pin      = LCD_RDX_PIN;
    GPIO_InitStructure.Mode     = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStructure.Pull     = GPIO_NOPULL;
  }
}
GPIO_InitStructure.Speed = GPIO_SPEED_FAST;
HAL_GPIO_Init(LCD_RDX_GPIO_PORT, &GPIO_InitStructure);

/* Configure the LCD Control pins -------
----------------------------------*/
LCD_NCS_GPIO_CLK_ENABLE();

/* Configure NCS in Output Push-Pull mode */
GPIO_InitStructure.Pin = LCD_NCS_PIN;
GPIO_InitStructure.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStructure.Pull = GPIO_NOPULL;
GPIO_InitStructure.Speed = GPIO_SPEED_FAST;
HAL_GPIO_Init(LCD_NCS_GPIO_PORT, &GPIO_InitStructure);

/* Set or Reset the control line */
LCD_CS_LOW();
LCD_CS_HIGH();
SPIx_Init();
}
}

/**
@brief Writes register value.
*/
void LCD_IO_WriteData(uint16_t RegValue) {
/* Set WRX to send data */


00814  LCD_WRX_HIGH();
00815  /* Reset LCD control line(/CS) and Send data */
00816  LCD_CS_LOW();
00817  SPIx_Write(RegValue);
00818  }
00819  /* Deselect: Chip Select high */
00820  LCD_CS_HIGH();
00821  }
00822  */
00823  /**
00824  * @brief Writes register address.
00825  */
00826  void LCD_IO_WriteReg(uint8_t Reg)
00827  {  
00828     /* Reset WRX to send command */
00829     LCD_WRX_LOW();
00830     /* Reset LCD control line(/CS) and Send command */
00831     LCD_CS_LOW();
00832     SPIx_Write(Reg);
00833     /* Deselect: Chip Select high */
00834     LCD_CS_HIGH();
00835  }
00836  }
00837  */
00838  /**
00839  * @brief Reads register value.
00840  * @param RegValue Address of the register to read
00841  * @param ReadSize Number of bytes to read
00842  * @retval Content of the register value
00843  */
00844  uint32_t LCD_IO_ReadData(uint16_t RegValue, uint8_t ReadSize)
uint32_t readvalue = 0;

/* Select: Chip Select low */
LCD_CS_LOW();

/* Reset WRX to send command */
LCD_WRX_LOW();

SPIx_Write(RegValue);

readvalue = SPIx_Read(ReadSize);

/* Set WRX to send data */
LCD_WRX_HIGH();

/* Deselect: Chip Select high */
LCD_CS_HIGH();

return readvalue;

/** *
 * @brief Wait for loop in ms.
 * @param Delay in ms.
 */
void LCD_Delay(uint32_t Delay)
{
    HAL_Delay(Delay);
}

/***************************************************************************/

LINK OPERATIONS
***************************************************************************/
/**
 * @brief IOE Low Level Initialization.
 */

void IOE_Init(void)
{
    I2Cx_Init();
}

/**
 * @brief IOE Low Level Interrupt configuration.
 */

void IOE_ITConfig(void)
{
    I2Cx_ITConfig();
}

/**
 * @brief IOE Writes single data operation.
 * @param Addr: I2C Address
 * @param Reg: Reg Address
 * @param Value: Data to be written
 */

void IOE_Write(uint8_t Addr, uint8_t Reg, uint8_t Value)
{
    I2Cx_WriteData(Addr, Reg, Value);
}

/**
 * @brief IOE Reads single data.
 * @param Addr: I2C Address
 * @param Reg: Reg Address
 */
uint8_t IOE_Read(uint8_t Addr, uint8_t Reg) {
    return I2Cx_ReadData(Addr, Reg);
}

void IOE_WriteMultiple(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length) {
    I2Cx_WriteBuffer(Addr, Reg, pBuffer, Length);
}

uint16_t IOE_ReadMultiple(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length) {
    return I2Cx_ReadBuffer(Addr, Reg, pBuffer, Length);
}
/**
 * @brief IOE Delay.
 * @param Delay in ms
 */

void IOE_Delay(uint32_t Delay) {
    HAL_Delay(Delay);
}

/*********************************	LINK	GYRO
SCOPE	****************************/

/**
 * @brief Configures the Gyroscope SPI interface.
 */

void GYRO_IO_Init(void) {
    GPIO_InitTypeDef GPIO_InitStructure;

    /* Configure the Gyroscope Control pins --
    ----------------------------------*/
    /* Enable CS GPIO clock and Configure GPIO PIN for Gyroscope Chip select */
    GYRO_CS_GPIO_CLK_ENABLE();
    GPIO_InitStructure.Pin = GYRO_CS_PIN;
    GPIO_InitStructure.Mode = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStructure.Pull = GPIO_NOPULL;
    GPIO_InitStructure.Speed = GPIO_SPEED_MEDIUM;
    HAL_GPIO_Init(GYRO_CS_GPIO_PORT, &GPIO_InitStructure);

    /* Deselect: Chip Select high */
    GYRO_CS_HIGH();
}
Enable INT1, INT2 GPIO clock and configure GPIO PINs to detect Interrupts */

GYRO_INT_GPIO_CLK_ENABLE();

GPIO_InitStructure.Pin = GYRO_INT1_PIN | GYRO_INT2_PIN;

GPIO_InitStructure.Mode = GPIO_MODE_INPUT;
GPIO_InitStructure.Speed = GPIO_SPEED_FAST;
GPIO_InitStructure.Pull = GPIO_NOPULL;
HAL_GPIO_Init(GYRO_INT_GPIO_PORT, &GPIO_InitStructure);

SPIx_Init();

/**
 * @brief Writes one byte to the Gyroscope.
 *
 * @param pBuffer: Pointer to the buffer containing the data to be written to the Gyroscope.
 * @param WriteAddr: Gyroscope's internal address to write to.
 * @param NumByteToWrite: Number of bytes to write.
 *
 * @param MULTIPLEBYTE_CMD
 *
 * void GYRO_IO_Write(uint8_t* pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite)
 *
 * Configure the MS bit:
 * - When 0, the address will remain unchanged in multiple read/write commands.
 * - When 1, the address will be auto incremented in multiple read/write commands.
 *
 * if(NumByteToWrite > 0x01)
 * {
 *     WriteAddr |= (uint8_t)MULTIPLEBYTE_CMD;
 * }
/* Set chip select Low at the start of the transmission */
GYRO_CS_LOW();

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

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

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

/* @brief Reads a block of data from the Gyroscope.
@param pBuffer: Pointer to the buffer that receives the data read from the Gyroscope.
@param ReadAddr: Gyroscope's internal address to read from.
@param NumByteToRead: Number of bytes to read from the Gyroscope. */

void GYRO_IO_Read(uint8_t* pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead) {
    if(NumByteToRead > 0x01)
{ 
  ReadAddr |= (uint8_t)(READWRITE_CMD | MULTIPLEBYTE_CMD);
}

else
{
  ReadAddr |= (uint8_t)READWRITE_CMD;
}

/* Set chip select Low at the start of the transmission */
GYRO_CS_LOW();

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

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

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

#ifdef EE_M24LR64
/* *********************************************** LINK I2C E EPROM **********************************************/
/**
 * @brief Initializes peripherals used by the I2C EEPROM driver.
 */

void EEPROM_IO_Init(void)
{
    I2Cx_Init();
}

/**
 * @brief Writes data to I2C EEPROM driver in using DMA channel.
 * @param DevAddress: Target device address
 * @param MemAddress: Internal memory address
 * @param pBuffer: Pointer to data buffer
 * @param BufferSize: Amount of data to be sent
 * @retval HAL status
 */

HAL_StatusTypeDef EEPROM_IO_WriteData(uint16_t DevAddress, uint16_t MemAddress, uint8_t* pBuffer, uint32_t BufferSize)
{
    return (I2Cx_WriteBufferDMA(DevAddress, MemAddress, pBuffer, BufferSize));
}

/**
 * @brief Reads data from I2C EEPROM driver in using DMA channel.
 * @param DevAddress: Target device address
 * @param MemAddress: Internal memory address
*/
HAL_StatusTypeDef EEPROM_IO_ReadData(uint16_t DevAddress, uint16_t MemAddress, uint8_t* pBuffer, uint32_t BufferSize)
{
    return (I2Cx_ReadBufferDMA(DevAddress, MemAddress, pBuffer, BufferSize));
}

HAL_StatusTypeDef EEPROM_IO_IsDeviceReady(uint16_t DevAddress, uint32_t Trials)
{
    return (I2Cx_IsDeviceReady(DevAddress, Trials));
}
stm32f429i_discovery_gyroscope.h

Go to the documentation of this file.

```c
/**
 * @file stm32f429i_discovery_gyroscope.h
 *
 * @author MCD Application Team
 * @version V2.1.3
 * @date 13-January-2016
 * @brief This file contains definitions for stm32f429i_discovery_gyroscope.c
 *        firmware driver.

 * @attention
 * Redistibution 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,
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.
/*
 * Define to prevent recursive inclusion ----------------------------------*/

#ifndef __STM32F429I_DISCOVERY_GYROSCOPE_H
#define __STM32F429I_DISCOVERY_GYROSCOPE_H

#ifdef __cplusplus
extern "C" {
#endif

/*
 * Includes ----------------------------------
 *------------------------------------------*/

#include "stm32f429i_discovery.h"

/* Include Gyroscope component driver */
#include "../Components/l3gd20/l3gd20.h"

/**
 * @addtogroup BSP
 * @{
 *
 * @addtogroup STM32F429I_DISCOVERY
 * @{
 *
 * @addtogroup STM32F429I_DISCOVERY_GYROSCOPE
 */

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Exported_Types
 * STM32F429I DISCOVERY GYROSCOPE Exported Types */

/** @} */
typedef enum {
    GYRO_OK = 0,
    GYRO_ERROR = 1,
    GYRO_TIMEOUT = 2
} GYRO_StatusTypeDef;

/**
 * @}
 * @}
 */

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE
 _Exported_Constants
 * @group STM32F429I_DISCOVERY_GYROSCOPE
 _Exported_Constants
 */

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE
 _Exported_Macros
 * @group STM32F429I_DISCOVERY_GYROSCOPE
 _Exported_Macros
 */

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE
 _Exported_Functions
 * @group STM32F429I_DISCOVERY_GYROSCOPE
 _Exported_Functions
 */

/* Gyroscope Functions */

uint8_t BSP_GYRO_Init(void);

void BSP_GYRO_Reset(void);
uint8_t BSP_GYRO_ReadID(void);

void BSP_GYRO_ITConfig(GYRO_InterruptConfigTypeDef *pIntConfigStruct);

void BSP_GYRO_EnableIT(uint8_t IntPin);

void BSP_GYRO_DisableIT(uint8_t IntPin);

void BSP_GYRO_GetXYZ(float *pfData);

/*__STM32F429I_DISCOVERY_GYROSCOPE_H*/

/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
stm32f429i_discovery_gyroscope.c

Go to the documentation of this file.

```c
00001 /*
00002 **********************************************************************************************
00003 * @file stm32f429i_discovery_gyroscope.c
00004 *
00005 * @author MCD Application Team
00006 * @version V2.1.3
00007 * @date 13-January-2016
00008 * @brief This file provides a set of functions needed to manage the
00009 * MEMS gyroscope available on STM32F429I-Discovery Kit.
00010 **********************************************************************************************
00011 * @attention
00012 * <h2><center>&copy; COPYRIGHT(c) 2015 STM microelectronics</center></h2>
00013 *
00014 * Redistribution and use in source and binary forms, with or without modification,
00015 * are permitted provided that the followin
g 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,
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.

00035 *
00036 *****************************************************************************
*****************************************************************************
00037 */
00038 /* Includes ****************************
---------------------------------------*
00039 #include "stm32f429i_discovery_gyroscope.h"
00040
00041 /** @addtogroup BSP
00042  *
00043  */
00044
00045 /** @addtogroup STM32F429I_DISCOVERY
00046  *
00047  */
00048
00049 /** @defgroup STM32F429I_DISCOVERY_GYROSCOPE
STM32F429I DISCOVERY GYROSCOPE
00050  *
00051  */
00052
00053 /** @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_TypesDefinitions STM32F429I DISCOVERY GYROSCOPE Private TypesDefinitions
00054  *
00055  */
00056 /**
00057  *
00058  */
00059
00060 /** @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_Defines STM32F429I DISCOVERY GYROSCOPE Private Defines
00061  *
00062  */
00063 /**
/**	@defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_Macros STM32F429I DISCOVERY GYROSCOPE Private Macros

*/

/**	@defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_Variables STM32F429I DISCOVERY GYROSCOPE Private Variables

*/

/**	@defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_FunctionPrototypes STM32F429I DISCOVERY GYROSCOPE Private FunctionPrototypes

*/

/**	@defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_Functions STM32F429I DISCOVERY GYROSCOPE Private Functions

*/

static GYRO_DrvTypeDef *GyroscopeDrv;
/**
 * @brief Set Gyroscope Initialization.
 * @retval GYRO_OK if no problem during initialization
 */

uint8_t BSP_GYRO_Init(void)
{
    uint8_t ret = GYRO_ERROR;
    uint16_t ctrl = 0x0000;
    GYRO_InitTypeDef L3GD20_InitStructure;
    GYRO_FilterConfigTypeDef L3GD20_FilterStructure={0,0};
    if((L3gd20Drv.ReadID() == I_AM_L3GD20) || (L3gd20Drv.ReadID() == I_AM_L3GD20_TR))
    {
        /* Initialize the Gyroscope driver structure */
        GyroscopeDrv = &L3gd20Drv;
        /* MEMS configuration ------------------------------*/
        /* Fill the Gyroscope structure */
        L3GD20_InitStructure.Power_Mode = L3GD20_MODE_ACTIVE;
        L3GD20_InitStructure.Output_DataRate = L3GD20_OUTPUT_DATARATE_1;
        L3GD20_InitStructure.Axes_Enable = L3GD20_AXES_ENABLE;
        L3GD20_InitStructure.Band_Width = L3GD20_BANDWIDTH_4;
        L3GD20_InitStructure.BlockData_Update = L3GD20_BlockDataUpdate_Continous;
        L3GD20_InitStructure.Endianness = L3GD20_BLE_LSB;
        L3GD20_InitStructure.Full_Scale = L3GD20
/* Configure MEMS: data rate, power mode, full scale and axes */
ctrl = (uint16_t) (L3GD20_InitStructure.Power_Mode | L3GD20_InitStructure.Output_DataRate | 
L3GD20_InitStructure.Axes_Enable | L3GD20_InitStructure.Band_Width);
ctrl |= (uint16_t) ((L3GD20_InitStructure.BlockData_Update | L3GD20_InitStructure.Endianness | 
L3GD20_InitStructure.Full_Scale) << 8);
/* Configure the Gyroscope main parameters */
GyroscopeDrv->Init(ctrl);
L3GD20_FilterStructure.HighPassFilter_Mode_Selection = L3GD20_HPM_NORMAL_MODE_RES;
L3GD20_FilterStructure.HighPassFilter_CutOff_Frequency = L3GD20_HPFCF_0;
ctrl = (uint8_t) ((L3GD20_FilterStructure.HighPassFilter_Mode_Selection | 
L3GD20_FilterStructure.HighPassFilter_CutOff_Frequency));
/* Configure the Gyroscope main parameters */
GyroscopeDrv->FilterConfig(ctrl);
GyroscopeDrv->FilterCmd(L3GD20_HIGHPASSFILTER_ENABLE);
ret = GYRO_OK;
}
else {
  ret = GYRO_ERROR;
}
return ret;
}

/**
 * @brief Read ID of Gyroscope component.
 * @retval ID
 */
uint8_t BSP_GYRO_ReadID(void)
{
  uint8_t id = 0x00;
  if(GyroscopeDrv->ReadID != NULL)
  {
    id = GyroscopeDrv->ReadID();
  }
  return id;
}

/**
 * @brief Reboot memory content of Gyroscope.
 */
void BSP_GYRO_Reset(void)
{
  if(GyroscopeDrv->Reset != NULL)
  {
    GyroscopeDrv->Reset();
  }
}
void BSP_GYRO_ITConfig(GYRO_InterruptConfigTypeDef *pIntConfig)
{
    uint16_t interruptconfig = 0x0000;
    if (GyroscopeDrv->ConfigIT != NULL)
        /* Configure latch Interrupt request and axe interrupts */
        interruptconfig |= ((uint8_t)(pIntConfig->Latch_Request) \ 
            pIntConfig->Interrupt_Axes) << 8);
    interruptconfig |= (uint8_t)(pIntConfig->Interrupt_ActiveEdge);
    GyroscopeDrv->ConfigIT(interruptconfig);
}

/* @brief Enables INT1 or INT2 interrupt. */
/* @param  IntPin: Interrupt pin This parameter can be: */
/* @arg  L3GD20_INT1 */
/* @arg  L3GD20_INT2 */
void BSP_GYRO_EnableIT(uint8_t IntPin)
{
    if (GyroscopeDrv->EnableIT != NULL)
GyroscopeDrv->EnableIT(IntPin);

/**
 * @brief Disables INT1 or INT2 interrupt.
 * @param IntPin: Interrupt pin
 * This parameter can be:
 * @arg L3GD20_INT1
 * @arg L3GD20_INT2
 */
void BSP_GYRO_DisableIT(uint8_t IntPin)
{
    if(GyroscopeDrv->DisableIT != NULL)
    {
        GyroscopeDrv->DisableIT(IntPin);
    }
}

/**
 * @brief Gets XYZ angular acceleration/
 * @param pfData: pointer on floating array
 */
void BSP_GYRO_GetXYZ(float *pfData)
{
    if(GyroscopeDrv->GetXYZ != NULL)
    {
        GyroscopeDrv->GetXYZ(pfData);
    }
}
stm32f429i_discovery_io.h

Go to the documentation of this file.

```c
/**
 * @file    stm32f429i_discovery_io.h
 * @author  MCD Application Team
 * @version V2.1.3
 * @date    13-January-2016
 * @brief   This file contains all the functions prototypes for the
 *          stm32f429i_discovery_io.c driver.
 * @attention
 * <h2><center>&copy; COPYRIGHT(c) 2015 STMicroelectronics</center></h2>
 * Redistribution and use in source and binary forms, with or without modification,
 * are permitted provided that the following conditions are met:
```
1. Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.
3. Neither the name of STMicroelectronics nor the names of its contributors
may be used to endorse or promote products derived from this software
without specific prior written permission.

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

#ifndef __STM32F429I_DISCOVERY_IO_H
#define __STM32F429I_DISCOVERY_IO_H

#ifdef __cplusplus
extern "C"
{
#endif

/* Includes -----------------------------------------------

#include "stm32f429i_discovery.h"
/* Include IO component driver */
#include "../Components/stmpe811/stmpe811.h"

/** @addtogroup BSP */
* @{
*/

/** @addtogroup STM32F429I_DISCOVERY */
* @{
*/

/** @addtogroup STM32F429I_DISCOVERY IO */
* @{
*/

/** @defgroup STM32F429I_DISCOVERY_IO_Exported_Types STM32F429I_DISCOVERY IO Exported Types */
* @{
*/
typedef enum {
    IO_OK = 0,
    IO_ERROR = 1,
    IO_TIMEOUT = 2
} IO_StatusTypeDef;

/**
 * @}*/

/**
 * @defgroup STM32F429I_DISCOVERY_IO_Exported_Constants STM32F429I DISCOVERY IO Exported Constants
 * @}*/

#define IO_PIN_0 0x01
#define IO_PIN_1 0x02
#define IO_PIN_2 0x04
#define IO_PIN_3 0x08
#define IO_PIN_4 0x10
#define IO_PIN_5 0x20
#define IO_PIN_6 0x40
#define IO_PIN_7 0x80
#define IO_PIN_ALL 0xFF

/**
 * @}*/

/**
 * @defgroup STM32F429I_DISCOVERY_IO_Exported_Macros STM32F429I DISCOVERY IO Exported Macros
 * @}*/

/**
 * @}*/

/**
 * @defgroup STM32F429I_DISCOVERY_IO_Exported_Definitions STM32F429I DISCOVERY IO Exported Definitions
 * @}*/
ed_Functions STM32F429I DISCOVERY IO Exported Functions

00101 * @{
00102 */
00103 uint8_t BSP_IO_Init(void);
00104 uint8_t BSP_IO_ITGetStatus(uint16_t IoPin);
00105 void BSP_IO_ITClear(void);
00106 void BSP_IO_ConfigPin(uint16_t IoPin, IO_ModeTypeDef IoMode);
00107 void BSP_IO_WritePin(uint16_t IoPin, uint8_t PinState);
00108 uint16_t BSP_IO_ReadPin(uint16_t IoPin);
00109 void BSP_IO_TogglePin(uint16_t IoPin);
00110
00111 /**
00112 * @} 00113 */
00114
00115 /**
00116 * @} 00117 */
00118
00119 /**
00120 * @} 00121 */
00122
00123 /**
00124 * @} 00125 */
00126
00127 #ifdef __cplusplus
00128 }
00129 #endif
00130
00131 #endif /* __STM32F429I_DISCOVERY_IO_H */
00132
00133 ************************ (C) COPYRIGHT STMicroelectronics
stm32f429i_discovery_io.c

Go to the documentation of this file.

```c
/**
 * @file stm32f429i_discovery_io.c
 * @author MCD Application Team
 * @version V2.1.3
 * @date 13-January-2016
 * @brief This file provides a set of functions needed to manage the STMPE811
 * IO Expander device mounted on STM32F429I-Discovery Kit.
 * @attention
 * <h2><center>&copy; COPYRIGHT(c) 2015 STMicroelectronics</center></h2>
 * Redistribution and use in source and binary forms, with or without modification,
 * are permitted provided that the following conditions are met:
```
1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of STMicroelectronics nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*  
* **************************************************************************************************  
* **************************************************************************************************  
*  
* */  
*  
* #include "stm32f429i_discovery_io.h"  
*  
* /** @addtogroup BSP  
*  
* */  
*  
* /** @addtogroup STM32F429I_DISCOVERY  
*  
* */  
*  
* /** @defgroup STM32F429I_DISCOVERY_IO STM32F429I_DISCOVERY IO  
*  
* */  
*  
* /** @defgroup STM32F429I_DISCOVERY_IO_Private_Types_Definitions STM32F429I DISCOVERY IO Private Types Definitions  
*  
* */  
*  
* /** @defgroup STM32F429I_DISCOVERY_IO_Private_Defines STM32F429I DISCOVERY IO Private Defines  
*  
* */  
*  
* */
/**
 * @defgroup STM32F429I_DISCOVERY_IO_Private_Macros STM32F429I DISCOVERY IO Private Macros
 */

/**
 * @}
 */

/**
 * @defgroup STM32F429I_DISCOVERY_IO_Private_Variables STM32F429I DISCOVERY IO Private Variables
 */

static IO_DrvTypeDef *IoDrv;

/**
 * @}
 */

/**
 * @defgroup STM32F429I_DISCOVERY_IO_Private_Function_Prototypes STM32F429I DISCOVERY IO Private Function Prototypes
 */

/**
 * @}
 */

/**
 * @defgroup STM32F429I_DISCOVERY_IO_Private_Functions STM32F429I DISCOVERY IO Private Functions
 */
/**
 * @brief Initializes and configures 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.
 * @retval IO_OK if all initializations done correctly. Other value if error.
 */

uint8_t BSP_IO_Init(void)
{
    uint8_t ret = IO_ERROR;

    /* Read ID and verify the IO expander is ready */
    if (stmpe811_io_drv.ReadID(IO_I2C_ADDRESS) == STMPE811_ID)
    {
        /* Initialize the IO driver structure */
        IoDrv = &stmpe811_io_drv;
        ret = IO_OK;
    }

    /* Check if the initialization was successful */
    if (ret == IO_OK)
IoDrv->Init(IO_I2C_ADDRESS);

IoDrv->Start(IO_I2C_ADDRESS, IO_PIN_ALL);

return ret;

/**
 * @brief Gets the selected pins IT status.
 *
 */

uint8_t BSP_IO_ITGetStatus(uint16_t IoPin)
{
    /* Return the IO Pin IT status */
    return (IoDrv->ITStatus(IO_I2C_ADDRESS, IoPin));
}

/**
 * @brief Clears all the IO IT pending bits
 *
 */

void BSP_IO_ITClear(void)
{
    /* Clear all IO IT pending bits */
    IoDrv->ClearIT(IO_I2C_ADDRESS, IO_PIN_ALL);
}

/**
 * @brief Configures the IO pin(s) according to IO mode structure value.
 */
* @param IoPin: IO pin(s) to be configured.

This parameter could be any combination of the following values:

- @arg STMPE811_PIN_x: where x can be from 0 to 7.

* @param IoMode: The IO pin mode to configure, could be one of the following values:

- @arg IO_MODE_INPUT
- @arg IO_MODE_OUTPUT
- @arg IO_MODE_IT_RISING_EDGE
- @arg IO_MODE_IT_FALLING_EDGE
- @arg IO_MODE_IT_LOW_LEVEL
- @arg IO_MODE_IT_HIGH_LEVEL

*/

void BSP_IO_ConfigPin(uint16_t IoPin, IO_ModeTypeDef IoMode)
{
    /* Configure the selected IO pin(s) mode */

    IoDrv->Config(IO_I2C_ADDRESS, IoPin, IoMode);
}

/**
 * @brief Sets the selected pins state.
 * @param IoPin: The selected pins to write.

This parameter could be any combination of the IO pins.

* @param PinState: the new pins state to write

*/

void BSP_IO_WritePin(uint16_t IoPin, uint8_t PinState)
{
    /* Set the Pin state */
IoDrv->WritePin(IO_I2C_ADDRESS, IoPin, Pin State);
}

/**
 * @brief Gets the selected pins current state.
 */
uint16_t BSP_IO_ReadPin(uint16_t IoPin)
{
    return(IoDrv->ReadPin(IO_I2C_ADDRESS, IoPin));
}

/**
 * @brief Toggles the selected pins state.
 */
void BSP_IO_TogglePin(uint16_t IoPin)
{
    /* Toggle the current pin state */
    if(IoDrv->ReadPin(IO_I2C_ADDRESS, IoPin) = = 1 /* Set */)
    {
        IoDrv->WritePin(IO_I2C_ADDRESS, IoPin, 0 /* Reset */);
    }
    else
    {
    }
IoDrv->WritePin(IO_I2C_ADDRESS, IoPin, 1 /* Set */);
}
}

/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
stm32f429i_discovery_sdram.h

Go to the documentation of this file.

```c
/**
 * @file    stm32f429i_discovery_sdram.h
 * @author  MCD Application Team
 * @version V2.1.3
 * @date    13-January-2016
 * @brief   This file contains all the functions prototypes for the
 *          stm32f429i_discovery_sdram.c driver.
 */

/* @attention

<h2><center>&copy; COPYRIGHT(c) 2015 STMicroelectronics</center></h2>

* Redistribution and use in source and binary forms, with or without modification,
* are permitted provided that the following conditions are met:
*/
```
1. Redistributions of source code must retain the above copyright notice,
   this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions and the following disclaimer in the documentation
   and/or other materials provided with the distribution.

3. Neither the name of STMicroelectronics nor the names of its contributors
   may be used to endorse or promote products derived from this software
   without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
 CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
/* Define to prevent recursive inclusion -------------------------------*/
#ifndef __STM32F429I_DISCOVERY_SDRAM_H
#define __STM32F429I_DISCOVERY_SDRAM_H

#ifdef __cplusplus
extern "C" {
#endif

/* Includes ---------------------------------- ----------------------------------*/
#include "stm32f429i_discovery.h"

/** @addtogroup BSP */
*@{*

/** @addtogroup STM32F429I_DISCOVERY */
*@{*

/** @addtogroup STM32F429I_DISCOVERY_SDRAM */
*@{*

/** @defgroup STM32F429I_DISCOVERY_SDRAM_Expported_Types STM32F429I DISCOVERY_SDRAM Exported Types */
*@{*

/** @defgroup STM32F429I_DISCOVERY_SDRAM_Expported_Types STM32F429I DISCOVERY SDRAM Exported Types */
*@{*

```c
```
/**
 * @defgroup STM32F429I_DISCOVERY_SDRAM_Exported_Constants STM32F429I DISCOVERY SDRAM Exported Constants
 *
 * @brief FMC SDRAM Bank address
 */
#define SDRAM_DEVICE_ADDR ((uint32_t)0xD0000000)

/**
 * @brief FMC SDRAM Memory Width
 */
#define SDRAM_MEMORY_WIDTH FMC_SDRAM_MEM_BUS_WIDTH_16

/**
 * @brief FMC SDRAM CAS Latency
 */
#define SDRAM_CAS_LATENCY FMC_SDRAM_CAS_LATENCY_3

/**
 * @brief FMC SDRAM Memory clock period
 */
#define SDCLOCK_PERIOD FMC_SDRAM_CLOCK_PERIOD_2
/* Default configuration used with LCD */
/*
#define SDCLOCK_PERIOD  FMC_SDRAM_CLOCK_PERIOD_3 */

/**
@brief FMC SDRAM Memory Read Burst feature
*/

#define SDRAM_READBURST  FMC_SDRAM_READBURST_DISABLE /* Default configuration used with LCD */

/*
#define SDRAM_READBURST  FMC_SDRAM_READBURST_ENABLE */

/**
@brief FMC SDRAM Bank Remap
*/

/*
Set the refresh rate counter
(15.62 us x Freq) - 20 */

#define REFRESH_COUNT  ((uint32_t)1386) /* SDRAM refresh counter */

#define SDRAM_TIMEOUT  ((uint32_t)0xFFFF)

/**
DMA definitions for SDRAM DMA transfer */

#define __DMAx_CLK_ENABLE  __DMA2_CLK_ENABLE

#define SDRAM_DMAx_CHANNEL  DMA_CHANNEL_0

#define SDRAM_DMAxSTREAM  DMA2_Stream0

#define SDRAM_DMAx_IRQn  DMA2_Stream0_IRQn

#define SDRAM_DMAx_IRQHandler  DMA2_Stream0_IRQHandler

/**
* @brief FMC SDRAM Mode definition register defines

#define SDRAM_MODEREG_BURST_LENGTH_1 ((uint16_t)0x0000)
#define SDRAM_MODEREG_BURST_LENGTH_2 ((uint16_t)0x0001)
#define SDRAM_MODEREG_BURST_LENGTH_4 ((uint16_t)0x0002)
#define SDRAM_MODEREG_BURST_LENGTH_8 ((uint16_t)0x0004)
#define SDRAM_MODEREG_BURST_TYPE_SEQUENTIAL ((uint16_t)0x0000)
#define SDRAM_MODEREG_BURST_TYPE_INTERLEAVED ((uint16_t)0x0008)
#define SDRAM_MODEREG_CAS_LATENCY_2 ((uint16_t)0x0020)
#define SDRAM_MODEREG_CAS_LATENCY_3 ((uint16_t)0x0030)
#define SDRAM_MODEREG_OPERATING_MODE_STANDARD ((uint16_t)0x0000)
#define SDRAM_MODEREG_WRITEBURST_MODE_PROGRAMMED ((uint16_t)0x0000)
#define SDRAM_MODEREG_WRITEBURST_MODE_SINGLE ((uint16_t)0x0020)

/**
 * @}
 */

/**
 * @defgroup STM32F429I_DISCOVERY_SDRAM_Exported_Macro	STM32F429I	DISCOVERY	SDRAM	Exported	Macro
 * @}
 */
/**
@defgroup STM32F429I_DISCOVERY_SDRAM_Exported_Functions STM32F429I DISCOVERY SDRAM Exported Functions
*/

void BSP_SDRAM_Init(void);
void BSP_SDRAM_Initialization_sequence(uint32_t RefreshCount);
void BSP_SDRAM_ReadData(uint32_t uwStartAddress, uint32_t* pData, uint32_t uwDataSize);
void BSP_SDRAM_ReadData_DMA(uint32_t uwStartAddress, uint32_t* pData, uint32_t uwDataSize);
void BSP_SDRAM_WriteData(uint32_t uwStartAddress, uint32_t* pData, uint32_t uwDataSize);
void BSP_SDRAM_WriteData_DMA(uint32_t uwStartAddress, uint32_t* pData, uint32_t uwDataSize);
HAL_StatusTypeDef BSP_SDRAM_Sendcmd(FMC_SDRAM_CommandTypeDef *SdramCmd);
void BSP_SDRAM_DMA_IRQHandler(void);

/**
*/

*/
#ifdef __cplusplus
}
#endif

#endif /* __STM32F429I_DISCOVERY_SDRAM_H */

="/*************** (C) COPYRIGHT STMicroelectronics *****END OF FILE****/"
stm32f429i_discovery_sdram.c

Go to the documentation of this file.

```c
/**
 * @file   stm32f429i_discovery_sdram.c
 * @author MCD Application Team
 * @version V2.1.3
 * @date   13-January-2016
 * @brief  This file provides a set of functions needed to drive the IS42S16400J SDRAM memory mounted on STM32F429I-Discovery Kit.
 * @attention
 * <h2><center>&copy; COPYRIGHT(c) 2015 STMicroelectronics</center></h2>
 * Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
```
1. Redistributions of source code must retain the above copyright notice,
   this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions and the following disclaimer in the documentation
   and/or other materials provided with the distribution.
3. Neither the name of STMicroelectronics nor the names of its contributors
   may be used to endorse or promote products derived from this software
   without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
 CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
#include "stm32f429i_discovery_sdram.h"

/** 
 * @addtogroup BSP 
 * @{ 
 */

/** 
 * @addtogroup STM32F429I_DISCOVERY 
 * @{ 
 */

/** 
 * @defgroup STM32F429I_DISCOVERY_SDRAM STM32F429I DISCOVERY SDRAM 
 * @{ 
 */

/** 
 * @defgroup STM32F429I_DISCOVERY_SDRAM_Private_Types_Definitions STM32F429I DISCOVERY SDRAM Private Types Definitions 
 * @{ 
 */

/** 
 * @defgroup STM32F429I_DISCOVERY_SDRAM_Private_Defines STM32F429I DISCOVERY SDRAM Private Defines 
 * @{ 
 */
/**	@defgroup STM32F429I_DISCOVERY_SDRAM_Private_Macros
STM32F429I DISCOVERY SDRAM Private Macros
*/

/**	@defgroup STM32F429I_DISCOVERY_SDRAM_Private_Variables
STM32F429I DISCOVERY SDRAM Private Variables
*/

/**	@defgroup STM32F429I_DISCOVERY_SDRAM_Private_Function_Prototypes
STM32F429I DISCOVERY SDRAM Private Function Prototypes
*/

/**	@defgroup STM32F429I_DISCOVERY_SDRAM_Private_Functions
STM32F429I DISCOVERY SDRAM Private Functions
*/

static SDRAM_HandleTypeDef SdramHandle;
static FMC_SDRAM_TimingTypeDef Timing;
static FMC_SDRAM_CommandTypeDef Command;

static void MspInit(void);

/**
*/

/**
*/

/**
*/

/**
*/
void BSP_SDRAM_Init(void)
{
    /* SDRAM device configuration */
    SdramHandle.Instance = FMC_SDRAM_DEVICE;

    /* FMC Configuration */
    /* FMC SDRAM Bank configuration */
    /* Timing configuration for 90 Mhz of SDClock frequency (180Mhz/2) */
    /* TMRD: 2 Clock cycles */
    Timing.LoadToActiveDelay = 2;
    /* TXSR: min=70ns (7x11.11ns) */
    Timing.ExitSelfRefreshDelay = 7;
    /* TRAS: min=42ns (4x11.11ns) max=120k (ns ) */
    Timing.SelfRefreshTime = 4;
    /* TRC:  min=70 (7x11.11ns) */
    Timing.RowCycleDelay = 7;
    /* TWR:  min=1+ 7ns (1+1x11.11ns) */
    Timing.WriteRecoveryTime = 2;
    /* TRP:  20ns => 2x11.11ns*/
    Timing.RPDelay = 2;
    /* TRCD: 20ns => 2x11.11ns */
    Timing.RCDDelay = 2;

    /* FMC SDRAM control configuration */
    SdramHandle.Init.SDBank = FMC_SDRAM_BANK2;
    /* Row addressing: [7:0] */
    SdramHandle.Init.ColumnBitsNumber = FMC_
SDRAM_COLUMN_BITS_NUM_8;
00127    /* Column addressing: [11:0] */
00128    SdramHandle.Init.RowBitsNumber = FMC_SDRAM_ROW_BITS_NUM_12;
00129    SdramHandle.Init.MemoryDataWidth = SDRAM_MEMORY_WIDTH;
00130    SdramHandle.Init.InternalBankNumber = FMC_SDRAM_INTERN_BANKS_NUM_4;
00131    SdramHandle.Init.CASLatency = SDRAM_CAS_LATENCY;
00132    SdramHandle.Init.WriteProtection = FMC_SDRAM_WRITE_PROTECTION_DISABLE;
00133    SdramHandle.Init.SDClockPeriod = SDCLK_PERIOD;
00134    SdramHandle.Init.ReadBurst = SDRAM_READBURST;
00135    SdramHandle.Init.ReadPipeDelay = FMC_SDRAM_RPIPE_DELAY_1;
00136
00137    /* SDRAM controller initialization */
00138    MspInit();
00139    HAL_SDRAM_Init(&SdramHandle, &Timing);
00140
00141    /* SDRAM initialization sequence */
00142    BSP_SDRAM_Initialization_sequence(REFRESH_COUNT);
00143 }
00144
00145 /**
00146    * @brief Programs the SDRAM device.
00147    * @param RefreshCount: SDRAM refresh counter value
00148    */
00149 void BSP_SDRAM_Initialization_sequence(uint32_t RefreshCount) {
00150    __IO uint32_t tmpmrd =0;
/* Step 1: Configure a clock configuration enable command */
Command.CommandMode = FMC_SDRAM_M_CMD_CLK_ENABLE;
Command.CommandTarget = FMC_SDRAM_M_CMD_TARGET_BANK2;
Command.AutoRefreshNumber = 1;
Command.ModeRegisterDefinition = 0;

/* Send the command */
HAL_SDRAM_SendCommand(&SdramHandle, &Command, SDRAM_TIMEOUT);

/* Step 2: Insert 100 us minimum delay */
/* Inserted delay is equal to 1 ms due to systick time base unit (ms) */
HAL_Delay(1);

/* Step 3: Configure a PALL (precharge all) command */
Command.CommandMode = FMC_SDRAM_M_CMD_PALL;
Command.CommandTarget = FMC_SDRAM_M_CMD_TARGET_BANK2;
Command.AutoRefreshNumber = 1;
Command.ModeRegisterDefinition = 0;

/* Send the command */
HAL_SDRAM_SendCommand(&SdramHandle, &Command, SDRAM_TIMEOUT);

/* Step 4: Configure an Auto Refresh command */
Command.CommandMode = FMC_SDRAM_M_CMD_AUTOREFRESH_MODE;
Command.CommandTarget = FMC_SDRAM
M_CMD_TARGET_BANK2;
00178 Command.AutoRefreshNumber = 4;
00179 Command.ModeRegisterDefinition = 0;
00180
00181 /* Send the command */
00182 HAL_SDRAM_SendCommand(&SdramHandle, &Command, SDRAM_TIMEOUT);
00183
00184 /* Step 5: Program the external memory mode register */
00185 tmpmrd = (uint32_t)SDRAM_MODEREG_BURST_LEN_GTH_1 | SDRAM_MODEREG_BURST_TYP_E_SEQUENTIAL | SDRAM_MODEREG_CAS_LATEN_CY_3 | SDRAM_MODEREG_OPERATING_MODE_STANDARD | SDRAM_MODEREG_WRITEBURST_MODE_SINGLE;
00190
00191 Command.CommandMode = FMC_SDRAM_M_CMD_LOAD_MODE;
00192 Command.CommandTarget = FMC_SDRAM_M_CMD_TARGET_BANK2;
00193 Command.AutoRefreshNumber = 1;
00194 Command.ModeRegisterDefinition = tmpmrd;
00195
00196 /* Send the command */
00197 HAL_SDRAM_SendCommand(&SdramHandle, &Command, SDRAM_TIMEOUT);
00198
00199 /* Step 6: Set the refresh rate counter */
00200 /* Set the device refresh rate */
00201 HAL_SDRAM_ProgramRefreshRate(&SdramHandle, RefreshCount);
00202 }
/**
* @brief Reads an mount of data from the SDRAM memory in polling mode.
* @param uwStartAddress : Read start address
* @param pData : Pointer to data to be read
* @param uwDataSize: Size of read data from the memory
*/
void BSP_SDRAM_ReadData(uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)
{
    HAL_SDRAM_Read_32b(&SdramHandle, (uint32_t *)uwStartAddress, pData, uwDataSize);
}

/**
* @brief Reads an mount of data from the SDRAM memory in DMA mode.
* @param uwStartAddress : Read start address
* @param pData : Pointer to data to be read
* @param uwDataSize: Size of read data from the memory
*/
void BSP_SDRAM_ReadData_DMA(uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)
{
    HAL_SDRAM_Read_DMA(&SdramHandle, (uint32_t *)uwStartAddress, pData, uwDataSize);
}

/**
* @brief Writes an mount of data to the S

DRAM memory in polling mode.

```c
00228 * @param uwStartAddress : Write start address
00229 * @parampData : Pointer to data to be written
00230 * @param uwDataSize: Size of written data from the memory
00231 */
00232 void BSP_SDRAM_WriteData(uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)
00233 {
00234  /* Disable write protection */
00235  HAL_SDRAM_WriteProtection_Disable(&SdramHandle);
00236  /*Write 32-bit data buffer to SDRAM memory */
00237  HAL_SDRAM_Write_32b(&SdramHandle, (uint32_t *)uwStartAddress, pData, uwDataSize);
00239 }
00240
00241 /**
00242 * @brief Writes an amount of data to the SDRAM memory in DMA mode.
00243 * @param uwStartAddress : Write start address
00244 * @param pData : Pointer to data to be written
00245 * @param uwDataSize: Size of written data from the memory
00246 */
00247 void BSP_SDRAM_WriteData_DMA(uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)
00248 {
00249  HAL_SDRAM_Write_DMA(&SdramHandle, (uint32_t *)uwStartAddress, pData, uwDataSize);
00250 }
```
/**
 * @brief Sends command to the SDRAM bank.
 * @param SdramCmd: Pointer to SDRAM command structure
 * @retval HAL status
 */

HAL_StatusTypeDef BSP_SDRAM_Sendcmd(FMC_SDRAM_CommandTypeDef *SdramCmd)
{
    return (HAL_SDRAM_SendCommand(&SdramHandle, SdramCmd, SDRAM_TIMEOUT));
}

/**
 * @brief Handles SDRAM DMA transfer interrupt request.
 */

void BSP_SDRAM_DMA_IRQHandler(void)
{
    HAL_DMA_IRQHandler(SdramHandle.hdma);
}

/**
 * @brief Initializes SDRAM MSP.
 */

static void MspInit(void)
{
    static DMA_HandleTypeDef dmaHandle;
    GPIO_InitTypeDef GPIO_InitStructure;
    SDRAM_HandleTypeDef *hsdram = &SdramHandle;

    /* Enable FMC clock */
    __FMC_CLK_ENABLE();

    /* Enable chosen DMAx clock */
__DMAX_CLK_ENABLE();

/* Enable GPIOs clock */
__GPIOB_CLK_ENABLE();
__GPIOC_CLK_ENABLE();
__GPIOD_CLK_ENABLE();
__GPIOE_CLK_ENABLE();
__GPIOF_CLK_ENABLE();
__GPIOG_CLK_ENABLE();

/*--- GPIOs Configuration -------------------
----------------------------------------*/

+-------------------+--------------------+---+
+ PD0 <-> FMC_D2   | PE0 <-> FMC_NBL0  |
PF0 <-> FMC_A0    | PG0 <-> FMC_A10   |
+ PD1 <-> FMC_D3   | PE1 <-> FMC_NBL1  |
PF1 <-> FMC_A1    | PG1 <-> FMC_A11   |
+ PD8 <-> FMC_D13  | PE7 <-> FMC_D4    |
PF2 <-> FMC_A2    | PG8 <-> FMC_SDCLK |
+ PD9 <-> FMC_D14  | PE8 <-> FMC_D5    |
PF3 <-> FMC_A3    | PG15 <-> FMC_NCAS |
+ PD10 <-> FMC_D15 | PE9 <-> FMC_D6    |
PF4 <-> FMC_A4    |-------------------+
+ PD14 <-> FMC_D0  | PE10 <-> FMC_D7   |
PF5 <-> FMC_A5    |---
+ PD15 <-> FMC_D1  | PE11 <-> FMC_D8   |
P11 <-> FMC_NRAS  |---
+-------------------|
PF12 <-> FMC_A6   |
+ PE13 <-> FMC_D10 |
P13 <-> FMC_A7    |
<table>
<thead>
<tr>
<th>PE14 &lt;-&gt; FMC_D11</th>
<th>PF14 &lt;-&gt; FMC_A8</th>
</tr>
</thead>
<tbody>
<tr>
<td>PE15 &lt;-&gt; FMC_D12</td>
<td>PF15 &lt;-&gt; FMC_A9</td>
</tr>
</tbody>
</table>

+-------------------+--------------------+
-------------------+

| PB5 <-> FMC_SDCKE1| PB6 <-> FMC_SDNE1 |
| PC0 <-> FMC_SDNWE |

+-------------------+

/* Common GPIO configuration */
GPIO_InitStructure.Mode = GPIO_MODE_AF_PP;
GPIO_InitStructure.Speed = GPIO_SPEED_FAST;
GPIO_InitStructure.Pull = GPIO_NOPULL;
GPIO_InitStructure.Alternate = GPIO_AF12_FMC;

/* GPIOB configuration */
GPIO_InitStructure.Pin = GPIO_PIN_5 | GPIO_PIN_6;
HAL_GPIO_Init(GPIOB, &GPIO_InitStructure);

/* GPIOC configuration */
GPIO_InitStructure.Pin = GPIO_PIN_0;
HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);

/* GPIOD configuration */
GPIO_InitStructure.Pin = GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_8 | GPIO_PIN_9 | GPIO
GPIO_PIN_10 | GPIO_PIN_14 |
00334 GPIO_PIN_15;
00335 HAL_GPIO_Init(GPIOD, &GPIO_InitStructure);
00336
00337 /* GPIOE configuration */
00338 GPIO_InitStructure.Pin = GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_7 |
00339 GPIO_PIN_8 | GPIO_PIN_9 |
00340 GPIO_PIN_10 | GPIO_PIN_11 |
00341 GPIO_PIN_12 | GPIO_PIN_13 |
00342 GPIO_PIN_14 | GPIO_PIN_15;
00343 HAL_GPIO_Init(GPIOE, &GPIO_InitStructure);
00344
00345 /* GPIOF configuration */
00346 GPIO_InitStructure.Pin = GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 |
00347 GPIO_PIN_3 | GPIO_PIN_4 |
00348 GPIO_PIN_5 | GPIO_PIN_11 |
00349 GPIO_PIN_12 | GPIO_PIN_13 |
00350 GPIO_PIN_14 | GPIO_PIN_15;
00351 HAL_GPIO_Init(GPIOF, &GPIO_InitStructure);
00352
00353 /* GPIOG configuration */
00354 GPIO_InitStructure.Pin = GPIO_PIN_0 | GPIO_PIN_1 |
00355 GPIO_PIN_4 | GPIO_PIN_5 |
00356 GPIO_PIN_8 | GPIO_PIN_15;
00357 HAL_GPIO_Init(GPIOG, &GPIO_InitStructure);
00358
00359 /* Configure common DMA parameters */
00360 dmaHandle.Init.Channel = SDRAM_DMAx_CHANNEL;
00361 dmaHandle.Init.Direction = DMA_M
EMORY_TO_MEMORY;
00359    dmaHandle.Init.PeriphInc   = DMA_P
INC_ENABLE;
00360    dmaHandle.Init.MemInc     = DMA_M
INC_ENABLE;
00361    dmaHandle.Init.PeriphDataAlignment = DMA_P
DATAALIGN_WORD;
00362    dmaHandle.Init.MemDataAlignment = DMA_M
DATAALIGN_WORD;
00363    dmaHandle.Init.Mode       = DMA_N
ORMAL;
00364    dmaHandle.Init.Priority   = DMA_P
RIORITY_HIGH;
00365    dmaHandle.Init.FIFOMode   = DMA_F
IFOMODE_DISABLE;
00366    dmaHandle.Init.FIFOThreshold = DMA_F
IFO_THRESHOLD_FULL;
00367    dmaHandle.Init.MemBurst   = DMA_M
BURST_SINGLE;
00368    dmaHandle.Init.PeriphBurst = DMA_P
BURST_SINGLE;
00369
00370    dmaHandle.Instance = SDRAM_DMAx_STREAM;
00371
00372    /* Associate the DMA handle */
00373    ___HAL_LINKDMA(hsdram, hdma, dmaHandle);
00374
00375    /* Deinitialize the stream for new transfer */
00376    HAL_DMA_DeInit(&dmaHandle);
00377
00378    /* Configure the DMA stream */
00379    HAL_DMA_Init(&dmaHandle);
00380
00381    /* NVIC configuration for DMA transfer complete interrupt */
00382    HAL_NVIC_SetPriority(SDRAM_DMAx_IRQn, 0, 0
HAL_NVIC_EnableIRQ(SDRAM_DMAx_IRQn);
}

/** *
 */
*/
/** *
 */
*/
/** *
 */
/****************************
(C) COPYRIGHT STMicroelectronics *****END OF FILE****/
STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Directories</th>
<th>Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSP</td>
<td></td>
</tr>
<tr>
<td>Modules</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY</td>
<td></td>
</tr>
</tbody>
</table>

## STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Data Structures</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directories</strong></td>
<td><strong>STM32F429I DISCOVERY</strong></td>
<td></td>
<td><strong>BSP</strong></td>
</tr>
</tbody>
</table>

Modules
### Modules

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL</td>
<td>This file provides set of firmware functions to manage Leds and push-button available on STM32F429I-Discovery Kit from STMicroelectronics.</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY GYROSCOPE</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY IO</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LCD</td>
<td>This file includes the LCD driver for (ILI9341)</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS</td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY EEPROM</td>
<td></td>
</tr>
</tbody>
</table>