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 LOW LEVEL

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery 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>
</table>

### STM32F429I DISCOVERY LOW LEVEL Private Macros

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery 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>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY LOW LEVEL Exported Macros

STM32F429I DISCOVERY LOW LEVEL

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY GYROSCOPE Private TypesDefinitions

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
# STM32F429I-Discovery BSP User Manual

## Directories

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

## STM32F429I DISCOVERY GYROSCOPE Private Defines

STM32F429I DISCOVERY GYROSCOPE

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org/) 1.7.6.1
STM32F429I DISCOVERY GYROSCOPE Private Macros

STM32F429I DISCOVERY GYROSCOPE

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
STM32F429I-DISCOVERY GYROSCOPE Private Function Prototypes

STM32F429I-DISCOVERY GYROSCOPE

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY GYROSCOPE
Exported Constants

STM32F429I DISCOVERY GYROSCOPE

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
### STM32F429I DISCOVERY GYROSCOPE

Exported Macros

---

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.stackoverflow.com) 1.7.6.1
STM32F429I-DISCOVERY IO Private Defines

STM32F429I DISCOVERY IO

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery 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>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY IO Private Macros**

STM32F429I DISCOVERY IO

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org) 1.7.6.1
### STM32F429I DISCOVERY IO Private Function Prototypes

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by [doxygen](http://www.stackoverflow.com) 1.7.6.1
# STM32F429I DISCOVERY LCD Private Function Prototypes

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by [doxygen](http://www.stackoverflow.com) 1.7.6.1
#include <stm32f429i_discovery_lcd.h>
### Data Fields

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint32_t</td>
<td>TextColor</td>
</tr>
<tr>
<td>uint32_t</td>
<td>BackColor</td>
</tr>
<tr>
<td>sFONT *</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**

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

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

Definition at line 84 of file stm32f429i_discovery_lcm.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`

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery
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><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY SDRAM Private Types Definitions**

---

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](http://www.stackoverflow.com) 1.7.6.1
# STM32F429I-Discovery BSP User Manual

## STM32F429I DISCOVERY SDRAM Private Defines

**STM32F429I DISCOVERY SDRAM**

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org/) 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY SDRAM Private Macros

STM32F429I DISCOVERY SDRAM

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
### STM32F429I DISCOVERY SDRAM Private Function Prototypes

| STM32F429I DISCOVERY SDRAM |

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org) 1.7.6.1
STM32F429I DISCOVERY SDRAM Exported Macro

STM32F429I DISCOVERY SDRAM

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
# STM32F429I DISCOVERY TS Private Types Definitions

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org) 1.7.6.1
STM32F429I DISCOVERY TS Private Defines
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY TS Private Macros

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
# STM32F429I-DISCOVERY TS Private Function Prototypes

**STM32F429I DISCOVERY TS**

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](http://www.stackoverflow.com) 1.7.6.1
STM32F429I-Discovery BSP User Manual

TS_StateTypeDef
Struct Reference
STM32F429I DISCOVERY TS Exported Types

#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`
## STM32F429I DISCOVERY TS Exported Macros

### STM32F429I DISCOVERY TS

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](https://doxygen.nl) 1.7.6.1
# STM32F429I DISCOVERY EEPROM Exported Types

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.stackoverflow.com) 1.7.6.1
STM32F429I DISCOVERY EEPROM Exported Macros

STM32F429I DISCOVERY EEPROM

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
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`
STM32F429I-Discovery BSP User Manual

- BackColor : \texttt{LCD\_DrawPropTypeDef}
- pFont : \texttt{LCD\_DrawPropTypeDef}
- TextColor : \texttt{LCD\_DrawPropTypeDef}
- TouchDetected : \texttt{TS\_StateTypeDef}
- X : \texttt{Point , TS\_StateTypeDef}
- Y : \texttt{Point , TS\_StateTypeDef}
- Z : \texttt{TS\_StateTypeDef}

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery
BSP User Manual by \texttt{doxygen} 1.7.6.1
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:
V2.1.5

Date:
27-January-2017

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 of 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 | 8 (3.3V) | +---------------------------------------+-----------+------|  

**Attention:**
© COPYRIGHT(c) 2017 STMicroelectronics

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

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

Definition in file 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)
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](#)
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.c, stm32f429i_discovery_io.h
- 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()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_DrawHLine()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_DrawLine()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_DrawPixel()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_DrawPolygon()` in `stm32f429i_discovery_lcd.h`, `stm32f429i_discovery_lcd.c`
- `BSP_LCD_DrawRect()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_DrawVLine()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_FillCircle()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_FillEllipse()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_FillPolygon()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_FillRect()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_FillTriangle()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_GetBackColor()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_GetFont()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_GetTextColor()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_GetXSize()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_GetYSize()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_Init()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LCD_LayerDefaultInit()` in `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
stm32f429i_discovery_lcd.h

- BSP_LCD_MspInit() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_ReadPixel() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_Relaod() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_ResetColorKeying() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_ResetColorKeying_NoReload() : 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_SetColorKeying_NoReload() : 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_SetLayerAddress_NoReload() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerVisible() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerVisible_NoReload() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerWindow() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerWindow_NoReload() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetTextColor() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetTransparency() : stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- `BSP_LCD_SetTransparency_NoReload()`: `stm32f429i_discovery_lcd.c`, `stm32f429i_discovery_lcd.h`
- `BSP_LED_Init()`: `stm32f429i_discovery.h`, `stm32f429i_discovery.c`
- `BSP_LED_Off()`: `stm32f429i_discovery.h`, `stm32f429i_discovery.c`
- `BSP_LED_On()`: `stm32f429i_discovery.h`, `stm32f429i_discovery.c`
- `BSP_LED_Toggle()`: `stm32f429i_discovery.c`, `stm32f429i_discovery.h`
- `BSP_PB_GetState()`: `stm32f429i_discovery.h`, `stm32f429i_discovery.c`
- `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_MspDeInit()`: `stm32f429i_discovery_sdram.h`, `stm32f429i_discovery_sdram.c`
- `BSP_SDRAM_MspInit()`: `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.h`, `stm32f429i_discovery_sdram.c`
- `BSP_SDRAM_Sendcmd()`: `stm32f429i_discovery_sdram.h`, `stm32f429i_discovery_sdram.c`
- `BSP_SDRAM_WriteData()`: `stm32f429i_discovery_sdram.h`, `stm32f429i_discovery_sdram.c`
- `BSP_SDRAM_WriteData_DMA()`: `stm32f429i_discovery_sdram.h`, `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.h*, *stm32f429i_discovery_ts.c*
  - BSP_TS_ITGetStatus() : *stm32f429i_discovery_ts.h*, *stm32f429i_discovery_ts.c*
  - BUFFER_OFFSET : *stm32f429i_discovery_lcd.h*
  - BUTTON_IRQn : *stm32f429i_discovery.c*
  - BUTTON_KEY : *stm32f429i_discovery.h*
  - BUTTON_MODE_EXTI : *stm32f429i_discovery.h*
  - BUTTON_MODE_GPIO : *stm32f429i_discovery.h*
  - BUTTON_PIN : *stm32f429i_discovery.c*
  - BUTTON_PORT : *stm32f429i_discovery.c*
  - Button_TypeDef : *stm32f429i_discovery.h*
  - 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_IRQHandler : [stm32f429i_discovery.h](#)
- DISCOVERY_I2Cx_EV_IRQHandler : [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_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](stm32f429i_discovery_eeprom.h)
- EEPROM_IO_Init() : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_IO_IsDeviceReady() : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_IO_ReadData() : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_IO_WriteData() : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_MAX_SIZE : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_MAX_TRIALS : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_OK : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_PAGESIZE : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_READ_TIMEOUT : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_TIMEOUT : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
- EEPROM_WRITE_TIMEOUT : [stm32f429i_discovery_eeprom.h](stm32f429i_discovery_eeprom.h)
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- 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** -

<table>
<thead>
<tr>
<th>Function/Variable</th>
<th>File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIO_PIN</td>
<td>stm32f429i_discovery.c</td>
</tr>
<tr>
<td>GPIO_PORT</td>
<td>stm32f429i_discovery.c</td>
</tr>
<tr>
<td>GYRO_CS_GPIO_CLK_DISABLE</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_CS_GPIO_CLK_ENABLE</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_CS_GPIO_PORT</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_CS_HIGH</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_CS_LOW</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_CS_PIN</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_ERROR</td>
<td>stm32f429i_discovery_gyroscope.h</td>
</tr>
<tr>
<td>GYRO_INT1 EXTI IRQn</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_INT1_PIN</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_INT2 EXTI IRQn</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_INT2_PIN</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_INT_GPIO_CLK_DISABLE</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_INT_GPIO_CLK_ENABLE</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_INT_GPIO_PORT</td>
<td>stm32f429i_discovery.h</td>
</tr>
<tr>
<td>GYRO_IO_Init()</td>
<td>stm32f429i_discovery.c</td>
</tr>
<tr>
<td>GYRO_IO_Read()</td>
<td>stm32f429i_discovery.c</td>
</tr>
<tr>
<td>GYRO_IO_Write()</td>
<td>stm32f429i_discovery.c</td>
</tr>
</tbody>
</table>
- GYRO_OK: stm32f429i_discovery_gyroscope.h
- GYRO_StatusTypeDef: stm32f429i_discovery_gyroscope.h
- GYRO_TIMEOUT: stm32f429i_discovery_gyroscope.h
- GyroscopeDrv: stm32f429i_discovery_gyroscope.c

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery
BSP User Manual by doxygen 1.7.6.1
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:

- i -

- **I2cHandle**: [stm32f429i_discovery.c](#)
- **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_TIMEOUT_MAX**: [stm32f429i_discovery.h](#)
- **I2Cx_WriteBuffer**: [stm32f429i_discovery.c](#)
- **I2Cx_WriteData**: [stm32f429i_discovery.c](#)
- **I2cxTimeout**: [stm32f429i_discovery.c](#)
- **IO_ERROR**: [stm32f429i_discovery_io.h](#)
- **IO_I2C_ADDRESS**: [stm32f429i_discovery.h](#)
- **IO_OK**: [stm32f429i_discovery_io.h](#)
- **IO_PIN_0**: [stm32f429i_discovery_io.h](#)
- **IO_PIN_1**: [stm32f429i_discovery_io.h](#)
- **IO_PIN_2**: [stm32f429i_discovery_io.h](#)
- **IO_PIN_3**: [stm32f429i_discovery_io.h](#)
- **IO_PIN_4**: [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](#)
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 : \texttt{stm32f429i\_discovery.h}
- LCD_RDX_GPIO_PORT : \texttt{stm32f429i\_discovery.h}
- LCD_RDX_HIGH : \texttt{stm32f429i\_discovery.h}
- LCD_RDX_LOW : \texttt{stm32f429i\_discovery.h}
- LCD_RDX_PIN : \texttt{stm32f429i\_discovery.h}
- LCD_RELOAD_IMMEDIATE : \texttt{stm32f429i\_discovery\_lcd.h}
- LCD_RELOAD_VERTICAL_BLANKING : \texttt{stm32f429i\_discovery\_lcd.h}
- LCD_StatusTypeDef : \texttt{stm32f429i\_discovery\_lcd.h}
- LCD_TIMEOUT : \texttt{stm32f429i\_discovery\_lcd.h}
- LCD_WRX_GPIO_CLK_DISABLE : \texttt{stm32f429i\_discovery.h}
- LCD_WRX_GPIO_CLK_ENABLE : \texttt{stm32f429i\_discovery.h}
- LCD_WRX_GPIO_PORT : \texttt{stm32f429i\_discovery.h}
- LCD_WRX_HIGH : \texttt{stm32f429i\_discovery.h}
- LCD_WRX_LOW : \texttt{stm32f429i\_discovery.h}
- LCD_WRX_PIN : \texttt{stm32f429i\_discovery.h}
- LcdDrv : \texttt{stm32f429i\_discovery\_lcd.c}
- LED3 : \texttt{stm32f429i\_discovery.h}
- LED3_GPIO_CLK_DISABLE : \texttt{stm32f429i\_discovery.h}
- LED3_GPIO_CLK_ENABLE : \texttt{stm32f429i\_discovery.h}
- LED3_GPIO_PORT : \texttt{stm32f429i\_discovery.h}
- LED3_PIN : \texttt{stm32f429i\_discovery.h}
- LED4 : \texttt{stm32f429i\_discovery.h}
- LED4_GPIO_CLK_DISABLE : \texttt{stm32f429i\_discovery.h}
- LED4_GPIO_CLK_ENABLE : \texttt{stm32f429i\_discovery.h}
- LED4_GPIO_PORT : \texttt{stm32f429i\_discovery.h}
- LED4_PIN : \texttt{stm32f429i\_discovery.h}
- Led_TypeDef : \texttt{stm32f429i\_discovery.h}
- LEDn : \texttt{stm32f429i\_discovery.h}
- LEDx_GPIO_CLK_DISABLE : \texttt{stm32f429i\_discovery.h}
- LEDx_GPIO_CLK_ENABLE : \texttt{stm32f429i\_discovery.h}
- LEFT_MODE : \texttt{stm32f429i\_discovery\_lcd.h}
- LtdcHandler : \texttt{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](#)
- 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](#)
- POLY_X : [stm32f429i_discovery_lcd.c](#)
- POLY_Y : [stm32f429i_discovery_lcd.c](#)
- pPoint : [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 : [link](stm32f429i_discovery_sdram.h)
- SDRAM_CAS_LATENCY : [link](stm32f429i_discovery_sdram.h)
- SDRAM_DEVICE_ADDR : [link](stm32f429i_discovery_sdram.h)
- SDRAM_DEVICE_SIZE : [link](stm32f429i_discovery_sdram.h)
- SDRAM_DMAx_CHANNEL : [link](stm32f429i_discovery_sdram.h)
- SDRAM_DMAx_IRQHandler : [link](stm32f429i_discovery_sdram.h)
- SDRAM_DMAx_IRQn : [link](stm32f429i_discovery_sdram.h)
- SDRAM_DMAx_STREAM : [link](stm32f429i_discovery_sdram.h)
- SDRAM_ERROR : [link](stm32f429i_discovery_sdram.h)
- SDRAM_MEMORY_WIDTH : [link](stm32f429i_discovery_sdram.h)
- SDRAM_MODEREG_BURST_LENGTH_1 : [link](stm32f429i_discovery_sdram.h)
- SDRAM_MODEREG_BURST_LENGTH_2 : [link](stm32f429i_discovery_sdram.h)
- SDRAM_MODEREG_BURST_LENGTH_4 : [link](stm32f429i_discovery_sdram.h)
- SDRAM_MODEREG_BURST_LENGTH_8 : [link](stm32f429i_discovery_sdram.h)
- SDRAM_MODEREG_BURST_TYPE_INTERLEAVED :
stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_BURST_TYPE_SEQUENCEAL :
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_CAS_LATENCY_2 :
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_CAS_LATENCY_3 :
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_OPERATING_MODE_STANDARD :
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_WRITEBURST_MODE_PROGRAMMED :
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_WRITEBURST_MODE_SINGLE :
  stm32f429i_discovery_sdram.h
- SDRAM_OK :
  stm32f429i_discovery_sdram.h
- SDRAM_READBURST :
  stm32f429i_discovery_sdram.h
- SDRAM_TIMEOUT :
  stm32f429i_discovery_sdram.h
- SdramHandle :
  stm32f429i_discovery_sdram.c
- SpiHandle :
  stm32f429i_discovery.c
- SPIx_Error() :
  stm32f429i_discovery.c
- SPIx_Init() :
  stm32f429i_discovery.c
- SPIx_MspInit() :
  stm32f429i_discovery.c
- SPIx_Read() :
  stm32f429i_discovery.c
- SPIx_TIMEOUT_MAX :
  stm32f429i_discovery.h
- SPIx_Write() :
  stm32f429i_discovery.c
- SPIx_WriteRead() :
  stm32f429i_discovery.c
- SpixTimeout :
  stm32f429i_discovery.c
- STMPE811_INT_CLK_DISABLE :
  stm32f429i_discovery.h
- STMPE811_INT_CLK_ENABLE :
  stm32f429i_discovery.h
- STMPE811_INT_EXTI :
  stm32f429i_discovery.h
- STMPE811_INT_EXTIHandler :
  stm32f429i_discovery.h
- STMPE811_INT_GPIO_PORT :
  stm32f429i_discovery.h
- STMPE811_INT_PIN :
  stm32f429i_discovery.h
Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- **t** -

- 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

## Directories

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

## File List

<table>
<thead>
<tr>
<th>File List</th>
<th>Defines</th>
<th>Functions</th>
<th>Variables</th>
<th>Typedefs</th>
<th>Enumerations</th>
<th>Enumerator</th>
</tr>
</thead>
</table>

### Defines

- b

### Functions

- 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
• `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_MspInit() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_ReadPixel() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_Relaod() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_ResetColorKeying() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_ResetColorKeying_NoReload() : \n  stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SelectLayer() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetBackColor() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetColorKeying() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetColorKeying_NoReload() : \n  stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetFont() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerAddress() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerAddress_NoReload() : \n  stm32f429i_discovery_lcd.h, stm32f429i_discovery_lcd.c
- BSP_LCD_SetLayerVisible() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerVisible_NoReload() : \n  stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerWindow() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetLayerWindow_NoReload() : \n  stm32f429i_discovery_lcd.c, stm32f429i_discovery_lcd.h
- BSP_LCD_SetTextColor() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetTransparency() : stm32f429i_discovery_lcd.c, \n  stm32f429i_discovery_lcd.h
- BSP_LCD_SetTransparency_NoReload() :
stm32f429i_discovery_lcd.c , stm32f429i_discovery_lcd.h

• BSP_LED_Init() : stm32f429i_discovery.h ,
  stm32f429i_discovery.c
• BSP_LED_Off() : stm32f429i_discovery.h ,
  stm32f429i_discovery.c
• BSP_LED_On() : stm32f429i_discovery.h ,
  stm32f429i_discovery.c
• BSP_LED_Toggle() : stm32f429i_discovery.h ,
  stm32f429i_discovery.c
• BSP_PB_GetState() : stm32f429i_discovery.h ,
  stm32f429i_discovery.c
• BSP_PB_Init() : stm32f429i_discovery.h ,
  stm32f429i_discovery.c
• 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.c ,
  stm32f429i_discovery_sdram.h
• BSP_SDRAM_MspDeInit() : stm32f429i_discovery_sdram.c ,
  stm32f429i_discovery_sdram.h
• BSP_SDRAM_MspInit() : 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.h ,
  stm32f429i_discovery_sdram.c
• 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.h ,
  stm32f429i_discovery_ts.c
- BSP_TS_ITConfig() : stm32f429i_discovery_ts.c ,
  stm32f429i_discovery_ts.h
- BSP_TS_ITGetStatus() : stm32f429i_discovery_ts.c ,
  stm32f429i_discovery_ts.h
STM32F429I-Discovery BSP User Manual

- c -

- ConvertLineToARGB8888() : `stm32f429i_discovery_lcd.c`

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

- 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`

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org) 1.7.6.1
- g -

- GYRO_IO_Init() : stm32f429i_discovery.c
- GYRO_IO_Read() : stm32f429i_discovery.c
- GYRO_IO_Write() : stm32f429i_discovery.c
STM32F429I-Discovery BSP User Manual

- 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

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

- L -

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

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
### Files

- 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`
- **SpiTimeout**: `stm32f429i_discovery.c`
- **Timing**: `stm32f429i_discovery_sdram.c`
- **TsDrv**: `stm32f429i_discovery_ts.c`
- **TsXBoundary**: `stm32f429i_discovery_ts.c`
• TsYBoundary: `stm32f429i_discovery_ts.c`

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery
BSP User Manual by doxygen 1.7.6.1
STM32F429I-Discovery BSP User Manual

- pPoint : *stm32f429i_discovery_lcd.h*

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by *doxygen* 1.7.6.1
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`

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery 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

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery
STM32F429I-Discovery BSP User Manual

- __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

- ABS : stm32f429i_discovery_lcd.c

- BUFFER_OFFSET : stm32f429i_discovery_lcd.h
- BUTTONn : stm32f429i_discovery.h
- d -

- EEPROM_FAIL : stm32f429i_discovery_eeprom.h

- e -

- EEPROM_FAIL : stm32f429i_discovery_eeprom.h
- g -

- 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

- i -

- I2Cx_TIMEOUT_MAX : stm32f429i_discovery.h
- IO_I2C_ADDRESS : stm32f429i_discovery.h
- IO_PIN_0 : stm32f429i_discovery_io.h
- IO_PIN_1 : stm32f429i_discovery_io.h
- IO_PIN_2 : stm32f429i_discovery_io.h
- IO_PIN_3 : stm32f429i_discovery_io.h
- IO_PIN_4 : 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
- k -

- KEY_BUTTON_EXTIIRQn : 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

- 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_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_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_RELOAD_IMMEDIATE: `stm32f429i_discovery_lcd.h`
- LCD_RELOAD_VERTICAL_BLANKING: `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`
- 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`
- LED4_GPIO_CLK_ENABLE : `stm32f429i_discovery.h`
- LED4_GPIO_PORT : `stm32f429i_discovery.h`
- LED4_PIN : `stm32f429i_discovery.h`
- LEDn : `stm32f429i_discovery.h`
- LEDx_GPIO_CLK_DISABLE : `stm32f429i_discovery.h`
- LEDx_GPIO_CLK_ENABLE : `stm32f429i_discovery.h`

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

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

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

- 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_ERROR : `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 : 
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_CAS_LATENCY_2 : 
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_CAS_LATENCY_3 : 
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_OPERATING_MODE_STANDARD : 
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_WRITEBURST_MODE_PROGRAMMED : 
  stm32f429i_discovery_sdram.h
- SDRAM_MODEREG_WRITEBURST_MODE_SINGLE : 
  stm32f429i_discovery_sdram.h
- SDRAM_OK : stm32f429i_discovery_sdram.h
- SDRAM_READBURST : stm32f429i_discovery_sdram.h
- SDRAM_TIMEOUT : stm32f429i_discovery_sdram.h
- SPIx_TIMEOUT_MAX : stm32f429i_discovery.h
- STMPE811_INT_CLK_DISABLE : stm32f429i_discovery.h
- STMPE811_INT_CLK_ENABLE : stm32f429i_discovery.h
- STMPE811_INT_EXTI : stm32f429i_discovery.h
- STMPE811_INT_EXTIHandler : stm32f429i_discovery.h
- STMPE811_INT_GPIO_PORT : stm32f429i_discovery.h
- STMPE811_INT_PIN : stm32f429i_discovery.h

- t -

- TS_I2C_ADDRESS : stm32f429i_discovery.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`
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...

#include "stm32f429i_discovery.h"

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

<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION_MAIN</td>
<td>(0x02)</td>
</tr>
<tr>
<td>STM32F429I DISCO BSP Driver version number</td>
<td>V2.1.5</td>
</tr>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION_SUB1</td>
<td>(0x01)</td>
</tr>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION_SUB2</td>
<td>(0x05)</td>
</tr>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION_RC</td>
<td>(0x00)</td>
</tr>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION</td>
<td></td>
</tr>
</tbody>
</table>
### Functions

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static void</td>
<td><code>I2Cx_Init</code> (void)</td>
<td>I2Cx Bus initialization.</td>
</tr>
<tr>
<td>static void</td>
<td><code>I2Cx_ITConfig</code> (void)</td>
<td>Configures Interruption pin for I2C communication.</td>
</tr>
<tr>
<td>static void</td>
<td><code>I2Cx_WriteData</code> (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><code>I2Cx_WriteBuffer</code> (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><code>I2Cx_ReadData</code> (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><code>I2Cx_ReadBuffer</code> (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><code>I2Cx_Error</code> (void)</td>
<td>I2Cx error treatment function.</td>
</tr>
<tr>
<td>static void</td>
<td><code>I2Cx_MspInit</code> (I2C_HandleTypeDef *hi2c)</td>
<td>I2Cx MSP Initialization.</td>
</tr>
<tr>
<td>static void</td>
<td><code>SPIx_Init</code> (void)</td>
<td>SPIx Bus initialization.</td>
</tr>
<tr>
<td>static void</td>
<td><code>SPIx_Write</code> (uint16_t Value)</td>
<td>Writes a byte to device.</td>
</tr>
<tr>
<td>static uint32_t</td>
<td><code>SPIx_Read</code> (uint8_t ReadSize)</td>
<td>Reads 4 bytes from device.</td>
</tr>
<tr>
<td>static uint8_t</td>
<td><code>SPIx_WriteRead</code> (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><code>SPIx_Error</code> (void)</td>
<td>SPIx error treatment function.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>static void SPIx_MspInit (SPI_HandleTypeDef *hspi)</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>Variable Declaration</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>
</tbody>
</table>
| GPIO_TypeDef                        | BUTTON_PORT [BUTTONn] = 
| const uint16_t                       | BUTTON_PIN [BUTTONn] = 
| const uint8_t                       | BUTTON_IRQn [BUTTONn] = |
| uint32_t                             | I2cxTimeout = I2Cx_TIMEOUT_MAX |
| uint32_t                             | SpixTimeout = SPIx_TIMEOUT_MAX |
| I2C_HandleTypeDef                   | I2cHandle |
| static SPI_HandleTypeDef            | SpiHandle |
| static uint8_t                      | Is_LCD_IO_Initialized = 0 |
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:
V2.1.5

Date:
27-January-2017

Attention:
© COPYRIGHT(c) 2017 STMicroelectronics

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

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

Definition in file stm32f429i_discovery.c.
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

```c
#define LEDn 2
#define LED3_PIN GPIO_PIN_13
#define LED3_GPIO_PORT GPIOG
#define LED3_GPIO_CLK_ENABLE() __HAL_RCC_GPIOG_CLK_ENABLE()
#define LED3_GPIO_CLK_DISABLE() __HAL_RCC_GPIOG_CLK_DISABLE()
#define LED4_PIN GPIO_PIN_14
#define LED4_GPIO_PORT GPIOG
#define LED4_GPIO_CLK_ENABLE() __HAL_RCC_GPIOG_CLK_ENABLE()
#define LED4_GPIO_CLK_DISABLE() __HAL_RCC_GPIOG_CLK_DISABLE()
#define LEDx_GPIO_CLK_ENABLE(__INDEX__) __HAL_RCC_GPIOG_CLK_ENABLE()
#define LEDx_GPIO_CLK_DISABLE(__INDEX__) __HAL_RCC_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() __HAL_RCC_GPIOA_CLK_ENABLE()
#define KEY_BUTTON_GPIO_CLK_DISABLE() __HAL_RCC_GPIOA_CLK_DISABLE()
#define KEY_BUTTON_EXTI_IRQn EXTI0_IRQn
#define BUTTONx_GPIO_CLK_ENABLE(__INDEX__) __HAL_RCC_GPIOG_CLK_ENABLE()
#define BUTTONx_GPIO_CLK_DISABLE(__INDEX__) __HAL_RCC_GPIOG_CLK_DISABLE()
#define IO_I2C_ADDRESS 0x82
#define TS_I2C_ADDRESS 0x82
#define DISCOVERY_I2Cx I2C3
#define DISCOVERY_I2Cx_CLOCK_ENABLE() __HAL_RCC_I2C3_CLK_ENABLE()
#define DISCOVERY_I2Cx_FORCE_RESET() __HAL_RCC_I2C3_FORCE_RESET()
#define DISCOVERY_I2Cx_RELEASE_RESET() __HAL_RCC_I2C3_RELEASE_RESET()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_ENABLE()
#define DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()
#define DISCOVERY_I2Cx_SCL_GPIO_PORT GPIOB
#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 for I2C waiting loops */
#define DISCOVERY_SPIx_SPI5
#define DISCOVERY_SPIx_CLK_ENABLE() __HAL_RCC_SPI5_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_PORT  GPIOF /* GPIOF */
#define DISCOVERY_SPIx_AF        GPIO_AF5_SPI5
#define DISCOVERY_SPIx_GPIO_CLK_ENABLE() __HAL_RCC_GPIOF_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_CLK_DISABLE() __HAL_RCC_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() __HAL_RCC_GPIOA_CLK_ENABLE()
#define STMPE811_INT_CLK_DISABLE() __HAL_RCC_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, GPIO_PIN_RESET)
#define LCD_CS_HIGH() HAL_GPIO_WritePin(LCD_NCS_GPIO_PORT, GPIO_PIN_SET)
#define LCD_WRX_LOW()  HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, GPIO_PIN_RESET)
#define LCD_WRX_HIGH() HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, GPIO_PIN_SET)
#define LCD_RDX_LOW()  HAL_GPIO_WritePin(LCD_RDX_GPIO_PORT, GPIO_PIN_RESET)
#define LCD_RDX_HIGH() HAL_GPIO_WritePin(LCD_RDX_GPIO_PORT, GPIO_PIN_SET)
```
```c
#define 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() __HAL_RCC_GPIOC_CLK_ENABLE()
#define LCD_NCS_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()

#define LCD_WRX_PIN GPIO_PIN_13
LCD Command/data pin.

#define LCD_WRX_GPIO_PORT GPIOD
#define LCD_WRX_GPIO_CLK_ENABLE() __HAL_RCC_GPIOD_CLK_ENABLE()
#define LCD_WRX_GPIO_CLK_DISABLE() __HAL_RCC_GPIOD_CLK_DISABLE()

#define LCD_RDX_PIN GPIO_PIN_12
#define LCD_RDX_GPIO_PORT GPIOD
#define LCD_RDX_GPIO_CLK_ENABLE() __HAL_RCC_GPIOD_CLK_ENABLE()
#define LCD_RDX_GPIO_CLK_DISABLE() __HAL_RCC_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, GPIO_PIN_RESET)
#define GYRO_CS_HIGH() HAL_GPIO_WritePin(GYRO_CS_GPIO_PORT, 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() __HAL_RCC_GPIOC_CLK_ENABLE()
#define GYRO_CS_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()

#define GYRO_INT_GPIO_PORT GPIOA /* GPIOA */
#define GYRO_INT_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
#define GYRO_INT_GPIO_CLK_DISABLE() __HAL_RCC_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
```
### Enumerations

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

<table>
<thead>
<tr>
<th>Data 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:
V2.1.5

Date:
27-January-2017

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.
stm32f429i_discovery_eeprom.h File Reference

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

#include "stm32f429i_discovery.h"

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

<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>
### Functions

<table>
<thead>
<tr>
<th></th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint32_t</code></td>
<td><strong>BSP_EEPROM_Init</strong> (void)</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><strong>BSP_EEPROM_ReadBuffer</strong> (uint8_t *pBuffer, uint16_t ReadAddr, uint16_t *NumByteToRead)</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><strong>BSP_EEPROM_WritePage</strong> (uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite)</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><strong>BSP_EEPROM_WriteBuffer</strong> (uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite)</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><strong>BSP_EEPROM_WaitEepromStandbyState</strong> (void)</td>
<td></td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_EEPROM_TIMEOUT_UserCallback</strong> (void)</td>
<td></td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>EEPROM_IO_Init</strong> (void)</td>
<td></td>
</tr>
<tr>
<td><code>HAL_StatusTypeDef</code></td>
<td><strong>EEPROM_IO_WriteData</strong> (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize)</td>
<td></td>
</tr>
<tr>
<td><code>HAL_StatusTypeDef</code></td>
<td><strong>EEPROM_IO_ReadData</strong> (uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize)</td>
<td></td>
</tr>
<tr>
<td><code>HAL_StatusTypeDef</code></td>
<td><strong>EEPROM_IO_IsDeviceReady</strong> (uint16_t DevAddress, uint32_t Trials)</td>
<td></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:
V2.1.5

Date:
27-January-2017

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.
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>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t BSP_GYRO_Init (void)</code></td>
<td>Set Gyroscope Initialization.</td>
</tr>
<tr>
<td><code>uint8_t BSP_GYRO_ReadID (void)</code></td>
<td>Read ID of Gyroscope component.</td>
</tr>
<tr>
<td><code>void BSP_GYRO_Reset (void)</code></td>
<td>Reboot memory content of Gyroscope.</td>
</tr>
<tr>
<td><code>void BSP_GYRO_ITConfig (GYRO_InterruptConfigTypeDef *pIntConfig)</code></td>
<td>Configures INT1 interrupt.</td>
</tr>
<tr>
<td><code>void BSP_GYRO_EnableIT (uint8_t IntPin)</code></td>
<td>Enables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td><code>void BSP_GYRO_DisableIT (uint8_t IntPin)</code></td>
<td>Disables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td><code>void 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:
V2.1.5

Date:
27-January-2017

Attention:
© COPYRIGHT(c) 2017 STMicroelectronics

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

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

Definition in file 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>Data Type</th>
<th>Function Name</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>void</code></td>
<td><code>BSP_GYRO_Reset (void)</code></td>
<td>Reboot memory content of Gyroscope.</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_ITConfig (GYRO_InterruptConfigTypeDef *pIntConfigStruct)</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>
Detailed Description

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

Author:  
MCD Application Team

Version:  
V2.1.5

Date:  
27-January-2017

| Attention: |
© COPYRIGHT(c) 2017 STMicroelectronics

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

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

Definition in file stm32f429i_discovery_gyroscope.h.
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>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t BSP_IO_Init (void)</code></td>
<td>Initializes and configures the IO functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
<tr>
<td><code>uint8_t BSP_IO_ITGetStatus (uint16_t IoPin)</code></td>
<td>Gets the selected pins IT status.</td>
</tr>
<tr>
<td><code>void BSP_IO_ITClear (void)</code></td>
<td>Clears all the IO IT pending bits.</td>
</tr>
<tr>
<td><code>void BSP_IO_ConfigPin (uint16_t IoPin, IO_ModeTypedef IoMode)</code></td>
<td>Configures the IO pin(s) according to IO mode structure value.</td>
</tr>
<tr>
<td><code>void BSP_IO_WritePin (uint16_t IoPin, uint8_t PinState)</code></td>
<td>Sets the selected pins state.</td>
</tr>
<tr>
<td><code>uint16_t BSP_IO_ReadPin (uint16_t IoPin)</code></td>
<td>Gets the selected pins current state.</td>
</tr>
<tr>
<td><code>void BSP_IO_TogglePin (uint16_t IoPin)</code></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:
V2.1.5

Date:
27-January-2017

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

This file contains all the functions prototypes for the stk32f429i_discovery_io.c driver.

Author:  
MCD Application Team

Version:  
V2.1.5

Date:  
27-January-2017

| Attention: |
© COPYRIGHT(c) 2017 STMicroelectronics

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

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

Definition in file 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

```c
#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>static void DrawChar (uint16_t Xpos, uint16_t Ypos, const uint8_t *c)</code></td>
<td>Draws a character on LCD.</td>
</tr>
<tr>
<td><code>static void 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>static void 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>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 FB_Address)</code></td>
<td>Initializes the LCD layers.</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_SetLayerVisible_NoReload (uint32_t LayerIndex, FunctionalState State)</code></td>
<td>Sets an LCD Layer visible without reloading.</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_SetTransparency_NoReload (uint32_t LayerIndex, uint8_t Transparency)</code></td>
<td></td>
</tr>
</tbody>
</table>

---

For more information, please refer to the documentation available at [GitHub](https://github.com/YourRepo).
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void LayerIndex, uint8_t Transparency)</code></td>
<td>Configures the transparency without reloading.</td>
<td></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>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerAddress_NoReload (uint32_t LayerIndex, uint32_t Address)</code></td>
<td>Sets an LCD layer frame buffer address without reloading.</td>
<td></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>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerWindow_NoReload (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</code></td>
<td>Sets display window without reloading.</td>
<td></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>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetColorKeying_NoReload (uint32_t LayerIndex, uint32_t RGBValue)</code></td>
<td>Configures and sets the color keying without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_ResetColorKeying (uint32_t LayerIndex)</code></td>
<td>Disables the color Keying.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_ResetColorKeying_NoReload (uint32_t LayerIndex)</code></td>
<td>Disables the color keying without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_Relaod (uint32_t ReloadType)</code></td>
<td>Disables the color keying without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_GetTextColor (void)</code></td>
<td>Gets the LCD Text color.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_GetBackColor (void)</code></td>
<td>Gets the LCD Background color.</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_SetTextColor(uint32_t Color)</td>
<td>Sets the Text color.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_SetBackColor(uint32_t Color)</td>
<td>Sets the Background color.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_SetFont(sFONT *pFonts)</td>
<td>Sets the Text Font.</td>
<td></td>
</tr>
<tr>
<td>sFONT * BSP_LCD_GetFont(void)</td>
<td>Gets the Text Font.</td>
<td></td>
</tr>
<tr>
<td>uint32_t BSP_LCD_ReadPixel(uint16_t Xpos, uint16_t Ypos)</td>
<td>Reads Pixel.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_Clear(uint32_t Color)</td>
<td>Clears the hole LCD.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_ClearStringLine(uint32_t Line)</td>
<td>Clears the selected line.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DisplayChar(uint16_t Xpos, uint16_t Ypos, uint8_t Ascii)</td>
<td>Displays one character.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DisplayStringAt(uint16_t X, uint16_t Y, uint8_t *pText, Text_AlignModeTypdef mode)</td>
<td>Displays a maximum of 60 char on the LCD.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DisplayStringAtLine(uint16_t Line, uint8_t *ptr)</td>
<td>Displays a maximum of 20 char on the LCD.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DrawHLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length)</td>
<td>Displays an horizontal line.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DrawVLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length)</td>
<td>Displays a vertical line.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DrawLine(uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2)</td>
<td>Displays an uni-line (between two points).</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DrawRect(uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DrawCircle</td>
<td>Displays a circle.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DrawPolygon</td>
<td>Displays an poly-line (between many points).</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DrawEllipse</td>
<td>Displays an Ellipse.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DrawBitmap</td>
<td>Displays a bitmap picture loaded in the internal Flash (32 bpp).</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_FillRect</td>
<td>Displays a full rectangle.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_FillCircle</td>
<td>Displays a full circle.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_FillTriangle</td>
<td>Fill triangle.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_FillPolygon</td>
<td>Displays a full poly-line (between many points).</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_FillEllipse</td>
<td>Draw a full ellipse.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DisplayOn</td>
<td>Enables the Display.</td>
<td></td>
</tr>
<tr>
<td>void BSP_LCD_DisplayOff</td>
<td>Disables the Display.</td>
<td></td>
</tr>
<tr>
<td>__weak void BSP_LCD_MspInit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------</td>
<td></td>
</tr>
<tr>
<td>void <strong>BSP_LCD_DrawPixel</strong> (uint16_t Xpos, uint16_t Ypos, uint32_t RGB_Code)</td>
<td>Writes Pixel.</td>
<td></td>
</tr>
</tbody>
</table>
### Variables

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTDC_HandleTypeDef</td>
<td>LtdcHandler</td>
</tr>
<tr>
<td>static DMA2D_HandleTypeDef</td>
<td>Dma2dHandler</td>
</tr>
<tr>
<td>static RCCPeriphCLKInitTypeDef</td>
<td>PeriphClkInitStruct</td>
</tr>
<tr>
<td>static uint32_t</td>
<td>ActiveLayer = 0</td>
</tr>
<tr>
<td>static LCD_DrawPropTypeDef</td>
<td>DrawProp</td>
</tr>
<tr>
<td>LCD_DrvTypeDef *</td>
<td>LcdDrv</td>
</tr>
</tbody>
</table>

Note: The table lists variables and their corresponding types and values.
Detailed Description

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

Author:
MCD Application Team

Version:
V2.1.5

Date:
27-January-2017

| 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_lcd.c.
STM32F429I-Discovery BSP User Manual

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

```
struct LCD_DrawPropTypeDef
struct Point
```
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_TRANSPARENT 0xFF000000
#define LCD_DEFAULT_FONT Font24
   LCD default font.
#define LCD_RELOAD_IMMEDIATE ((uint32_t)LTDC_SRCSR_IMR)
   LCD Reload Types.
#define LCD_RELOAD_VERTICAL_BLANKING ((uint32_t)LTDC_SRCSR_VBR)
#define LCD_BACKGROUND_LAYER 0x0000
   LCD Layer.
#define LCD_FOREGROUND_LAYER 0x0001
#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
typedef struct Point * pPoint
### Enumerations

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

Line mode structures definition. More...
## 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 FrameBuffer)</td>
<td>Initializes the LCD layers.</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_SetTransparency_NoReload</strong> (uint32_t LayerIndex, uint8_t Transparency)</td>
<td>Configures the transparency without reloading.</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_SetLayerAddress_NoReload</strong> (uint32_t LayerIndex, uint32_t Address)</td>
<td>Sets an LCD layer frame buffer address without reloading.</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_SetColorKeying_NoReload</strong> (uint32_t LayerIndex, uint32_t RGBValue)</td>
<td>Configures and sets the color keying without reloading.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_ResetColorKeying</strong> (uint32_t LayerIndex)</td>
<td>Disables the color Keying.</td>
</tr>
</tbody>
</table>
| void      | **BSP_LCD_ResetColorKeying_NoReload** (uint32_t LayerIndex) | }
Disables the color keying without reloading.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>BSP_LCD_SetLayerWindow</code> (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><code>BSP_LCD_SetLayerWindow_NoReload</code> (uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</td>
<td>Sets display window without reloading.</td>
</tr>
<tr>
<td><code>BSP_LCD_SelectLayer</code> (uint32_t LayerIndex)</td>
<td>Selects the LCD Layer.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetLayerVisible</code> (uint32_t LayerIndex, FunctionalState state)</td>
<td>Sets a LCD Layer visible.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetLayerVisible_NoReload</code> (uint32_t LayerIndex, FunctionalState State)</td>
<td>Sets an LCD Layer visible without reloading.</td>
</tr>
<tr>
<td><code>BSP_LCD_Relaod</code> (uint32_t ReloadType)</td>
<td>Disables the color keying without reloading.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetTextColor</code> (uint32_t Color)</td>
<td>Sets the Text color.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetBackColor</code> (uint32_t Color)</td>
<td>Sets the Background color.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetTextColor</code> (void)</td>
<td>Gets the LCD Text color.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetBackColor</code> (void)</td>
<td>Gets the LCD Background color.</td>
</tr>
<tr>
<td><code>BSP_LCD_SetFont</code> (sFONT *pFonts)</td>
<td>Sets the Text Font.</td>
</tr>
<tr>
<td><code>BSP_LCD_GetFont</code> (void)</td>
<td>Gets the Text Font.</td>
</tr>
<tr>
<td><code>BSP_LCD_ReadPixel</code> (uint16_t Xpos, uint16_t Ypos)</td>
<td>Reads Pixel.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawPixel</code> (uint16_t Xpos, uint16_t Ypos)</td>
<td></td>
</tr>
</tbody>
</table>
void uint32_t pixel)
  Writes Pixel.

void BSP_LCD_Clear (uint32_t Color)
  Clears the hole LCD.

void BSP_LCD_ClearStringLine (uint32_t Line)
  Clears the selected line.

void BSP_LCD_DisplayStringAtLine (uint16_t Line, uint8_t *ptr)
  Displays a maximum of 20 char on the LCD.

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.

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

void BSP_LCD_DrawHLine (uint16_t Xpos, uint16_t Ypos, uint16_t Length)
  Displays an horizontal line.

void BSP_LCD_DrawVLine (uint16_t Xpos, uint16_t Ypos, uint16_t Length)
  Displays a vertical line.

void BSP_LCD_DrawLine (uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2)
  Displays an uni-line (between two points).

void BSP_LCD_DrawRect (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
  Displays a rectangle.

void BSP_LCD_DrawCircle (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)
  Displays a circle.

void BSP_LCD_DrawPolygon (pPoint Points, uint16_t PointCount)
  Displays an poly-line (between many points).
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>void BSP_LCD_DrawEllipse (int Xpos, int Ypos, int XRADIUS, int YRADIUS)</strong></td>
<td>Displays an ellipse.</td>
</tr>
<tr>
<td>*<em>void BSP_LCD_DrawBitmap (uint32_t X, uint32_t Y, uint8_t <em>pBmp)</em></em></td>
<td>Displays a bitmap picture loaded in the internal Flash (32 bpp).</td>
</tr>
<tr>
<td><strong>void BSP_LCD_FillRect (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</strong></td>
<td>Displays a full rectangle.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_FillCircle (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</strong></td>
<td>Displays a full circle.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_FillTriangle (uint16_t X1, uint16_t X2, uint16_t X3, uint16_t Y1, uint16_t Y2, uint16_t Y3)</strong></td>
<td>Fill triangle.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_FillPolygon (pPoint Points, uint16_t PointCount)</strong></td>
<td>Displays a full poly-line (between many points).</td>
</tr>
<tr>
<td><strong>void BSP_LCD_DrawEllipse (int Xpos, int Ypos, int XRADIUS, int YRADIUS)</strong></td>
<td>Draw a full ellipse.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_DisplayOff (void)</strong></td>
<td>Disables the Display.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_DisplayOn (void)</strong></td>
<td>Enables the Display.</td>
</tr>
<tr>
<td><strong>void BSP_LCD_MspInit (void)</strong></td>
<td>Initializes the LTDC MSP.</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:
   V2.1.5

Date:
   27-January-2017

| 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_lcd.h.
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>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><code>BSP_SDRAM_Init (void)</code></td>
<td>Initializes the SDRAM device.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_SDRAM_Initialization_sequence (uint32_t RefreshCount)</code></td>
<td>Programs the SDRAM device.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><code>BSP_SDRAM_ReadData (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</code></td>
<td>Reads an amount of data from the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><code>BSP_SDRAM_ReadData_DMA (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</code></td>
<td>Reads an amount of data from the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><code>BSP_SDRAM_WriteData (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</code></td>
<td>Writes an amount of data to the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><code>BSP_SDRAM_WriteData_DMA (uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)</code></td>
<td>Writes an amount of data to the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><code>BSP_SDRAM_Sendcmd (FMC_SDRAM_CommandTypeDef *SdramCmd)</code></td>
<td>Sends command to the SDRAM bank.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_SDRAM_DMA_IRQHandler (void)</code></td>
<td>Handles SDRAM DMA transfer interrupt request.</td>
</tr>
<tr>
<td>__weak void</td>
<td><code>BSP_SDRAM_MspInit (SDRAM_HandleTypeDef *hsdram, void *Params)</code></td>
<td>Initializes SDRAM MSP.</td>
</tr>
<tr>
<td>__weak void</td>
<td><code>BSP_SDRAM_MspDelInit (SDRAM_HandleTypeDef *hsdram, void *Params)</code></td>
<td>DeInitializes SDRAM MSP.</td>
</tr>
</tbody>
</table>
### 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>
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:
V2.1.5

Date:
27-January-2017

| 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.
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.
**Defines**

```c
#define SDRAM_OK  ((uint8_t)0x00)
  SDRAM status structure definition.
#define SDRAM_ERROR ((uint8_t)0x01)
#define SDRAM_DEVICE_ADDR  ((uint32_t)0xD0000000)
  FMC SDRAM Bank address.
#define SDRAM_DEVICE_SIZE  ((uint32_t)0x800000)  /* SDRAM device size in Bytes */
#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 counter */
#define SDRAM_TIMEOUT  ((uint32_t)0xFFFF)
#define __DMAx_CLK_ENABLE __HAL_RCC_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_TYPESEQUENTIAL  ((uint16_t)0x0000)
#define SDRAM_MODEREG_BURST_TYPEINTERLEAVED  ((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)
### 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_SDRAM_Init</code> (void)</td>
<td>Initializes the SDRAM device.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_SDRAM_Initialization_sequence</code> (uint32_t <code>RefreshCount</code>)</td>
<td>Programs the SDRAM device.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_SDRAM_ReadData</code> (uint32_t <code>uwStartAddress</code>, uint32_t *<code>pData</code>, uint32_t <code>uwDataSize</code>)</td>
<td>Reads an amount of data from the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_SDRAM_ReadData_DMA</code> (uint32_t <code>uwStartAddress</code>, uint32_t *<code>pData</code>, uint32_t <code>uwDataSize</code>)</td>
<td>Reads an amount of data from the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_SDRAM_WriteData</code> (uint32_t <code>uwStartAddress</code>, uint32_t *<code>pData</code>, uint32_t <code>uwDataSize</code>)</td>
<td>Writes an amount of data to the SDRAM memory in polling mode.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_SDRAM_WriteData_DMA</code> (uint32_t <code>uwStartAddress</code>, uint32_t *<code>pData</code>, uint32_t <code>uwDataSize</code>)</td>
<td>Writes an amount of data to the SDRAM memory in DMA mode.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_SDRAM_Sendcmd</code> (FMC_SDRAM_CommandTypeDef *<code>SdramCmd</code>)</td>
<td>Sends command to the SDRAM bank.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_SDRAM_DMA_IRQHandler</code> (void)</td>
<td>Handles SDRAM DMA transfer interrupt request.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_SDRAM_MspInit</code> (SDRAM_HandleTypeDef *<code>hsdram</code>, void *<code>Params</code>)</td>
<td>Initializes SDRAM MSP.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_SDRAM_MspDeInit</code> (SDRAM_HandleTypeDef *<code>hsdram</code>, void *<code>Params</code>)</td>
<td>DeInitializes SDRAM MSP.</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:
    V2.1.5

Date:
    27-January-2017

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`. 
stm32f429i_discovery_ts.c File Reference

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>Data Type</th>
<th>Function Name</th>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint8_t</code></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><code>uint8_t</code></td>
<td><strong>BSP_TS_ITConfig</strong> (void)</td>
<td></td>
<td>Configures and enables the touch screen interrupts.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><strong>BSP_TS_ITGetStatus</strong> (void)</td>
<td></td>
<td>Gets the TS IT status.</td>
</tr>
<tr>
<td><code>void</code></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><code>void</code></td>
<td><strong>BSP_TS_ITClear</strong> (void)</td>
<td></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:
V2.1.5

Date:
27-January-2017

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

define TS_StatusTypeDef { TS_OK = 0x00, TS_ERROR = 0x01, TS_TIMEOUT = 0x02 }
### 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_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>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:
V2.1.5

Date:
27-January-2017

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 TypesDefinitions
- STM32F429I DISCOVERY LCD Private Defines
- STM32F429I DISCOVERY LCD Private Macros
- STM32F429I DISCOVERY LCD Private Variables
- STM32F429I DISCOVERY LCD Private FunctionPrototypes
- 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
Here are the data structures with brief descriptions:

<table>
<thead>
<tr>
<th>Data Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_DrawPropTypeDef</td>
</tr>
<tr>
<td>Point</td>
</tr>
<tr>
<td>TS_StateTypeDef</td>
</tr>
</tbody>
</table>
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</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.h</td>
<td>This file contains definitions for STM32F429I-Discovery Kit LEDs, push-buttons hardware resources</td>
</tr>
<tr>
<td>stm32f429i_discovery_eeprom.c</td>
<td>This file provides a set of functions needed to manage an I2C 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</td>
<td>This file contains all the function prototypes for the firmware driver for the STM32F429I-Discovery Kit's EEPROM memory.</td>
</tr>
<tr>
<td>stm32f429i_discovery_gyroscope.c</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>
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>

Directories

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

- **Drivers**
  - **BSP**
    - STM32F429I-Discovery

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org) 1.7.6.1
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>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private TypesDefinitions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Defines</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Variables</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private FunctionPrototypes</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Functions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Exported Constants</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Exported Macros</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Exported Functions</td>
</tr>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Exported Types</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.

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery 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 GYROSCOPE**

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

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1
# STM32F429I-Discovery BSP User Manual

<table>
<thead>
<tr>
<th>Directories</th>
<th>Modules</th>
</tr>
</thead>
</table>

**STM32F429I DISCOVERY IO**

STM32F429I DISCOVERY
# Modules

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

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org) 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LCD

STM32F429I DISCOVERY

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

<table>
<thead>
<tr>
<th>Module</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)
## Data Structure Index

<table>
<thead>
<tr>
<th>Data Structure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_DrawPropTypeDef</td>
<td></td>
</tr>
<tr>
<td>Point</td>
<td></td>
</tr>
<tr>
<td>TS_StateTypeDef</td>
<td></td>
</tr>
</tbody>
</table>

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery 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 LCD Exported Types**

**STM32F429I DISCOVERY LCD**
## 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>
typedef struct Point * pPoint
## Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LCD_StatusTypeDef</strong></td>
<td>{ LCD_OK = 0, LCD_ERROR = 1, LCD_TIMEOUT = 2 }</td>
</tr>
<tr>
<td><strong>Text_AlignModeTypeDef</strong></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

<table>
<thead>
<tr>
<th>Enumerator:</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_OK</td>
</tr>
<tr>
<td>LCD_ERROR</td>
</tr>
<tr>
<td>LCD_TIMEOUT</td>
</tr>
</tbody>
</table>

Definition at line 70 of file stm32f429i_discovery_lcd.h.

enum Text.AlignModeTypdef

Line mode structures definition.

<table>
<thead>
<tr>
<th>Enumerator:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CENTER_MODE</td>
</tr>
<tr>
<td>RIGHT_MODE</td>
</tr>
<tr>
<td>LEFT_MODE</td>
</tr>
</tbody>
</table>

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

Go to the documentation of this file.

```c
/** 
 ******************************************
 ******************************************
 *
 * @file stm32f429i_discovery_lcd.h 
 * @author MCD Application Team 
 * @version V2.1.5 
 * @date 27-January-2017 
 * @brief This file contains all the function prototypes for the 
 * stm32f429i_discovery_lcd.c driver. 
 * 
 ******************************************
 ******************************************
 *
 * @attention 
 * 
 * <h2><center>&copy; COPYRIGHT(c) 2017 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.
/* Define to prevent recursive inclusion ----------------------------------*/

#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_AlignModeTypeDef;
/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Exported_Constructors
 * STM32F429I DISCOVERY LCD Exported Constants
 */

#define LCD_LayerCfgTypeDef LTDC_LayerCfgTypeDef

#define MAX_LAYER_NUMBER 2
#define LCD_FRAME_BUFFER ((uint32_t)0x10000000)
#define BUFFER_OFFSET ((uint32_t)0x50000)

#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
`#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 Reload Types
 */
#define LCD_RELOAD_IMMEDIATE    (uint32_t)LTDC_SRCR_IMR)
#define LCD_RELOAD_VERTICAL_BLANKING (uint32_t)LTDC_SRCR_VBR)

`/**
 * @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

/**
 */
#define BSP_LCD_Init(void)
#endif

/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Exported_Functions
 * STM32F429I DISCOVERY LCD Exported Functions
 */
#define BSP_LCD_Init(void)
#endif

uint8_t BSP_LCD_Init(void);
uint32_t BSP_LCD_GetXSize(void);
uint32_t BSP_LCD_GetYSize(void);
/* functions using the LTDC controller */

void BSP_LCD_LayerDefaultInit(uint16_t LayerIndex, uint32_t FrameBuffer);
void BSP_LCD_SetTransparency(uint32_t LayerIndex, uint8_t Transparency);
void BSP_LCD_SetTransparency_NoReload(uint32_t LayerIndex, uint8_t Transparency);
void BSP_LCD_SetLayerAddress(uint32_t LayerIndex, uint32_t Address);
void BSP_LCD_SetLayerAddress_NoReload(uint32_t LayerIndex, uint32_t Address);
void BSP_LCD_SetColorKeying(uint32_t LayerIndex, uint32_t RGBValue);
void BSP_LCD_SetColorKeying_NoReload(uint32_t LayerIndex, uint32_t RGBValue);
void BSP_LCD_ResetColorKeying(uint32_t LayerIndex);
void BSP_LCD_ResetColorKeying_NoReload(uint32_t LayerIndex);
void BSP_LCD_SetLayerWindow(uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height);
void BSP_LCD_SetLayerWindow_NoReload(uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height);
void BSP_LCD_SelectLayer(uint32_t LayerIndex);
void BSP_LCD_SetLayerVisible(uint32_t LayerIndex, FunctionalState state);
void BSP_LCD_SetLayerVisible_NoReload(uint32_t LayerIndex, FunctionalState State);
void BSP_LCD_Relaod(uint32_t ReloadType);
void BSP_LCD_SetTextColor(uint32_t Color);
void BSP_LCD_SetBackColor(uint32_t Color);
uint32_t BSP_LCD_GetTextColor(void);
uint32_t BSP_LCD_GetBackColor(void);

void BSP_LCD_SetFont(sFONT *pFonts);
sFONT *BSP_LCD_GetFont(void);

void BSP_LCD_SetFont(sFONT *pFonts);

uint32_t BSP_LCD_ReadPixel(uint16_t Xpos, uint16_t Ypos);

void BSP_LCD_DrawPixel(uint16_t Xpos, uint16_t Ypos, uint32_t pixel);
void BSP_LCD_Clear(uint32_t Color);
void BSP_LCD_ClearStringLine(uint32_t Line);

void BSP_LCD_DisplayStringAtLine(uint16_t Xpos, uint8_t *ptr);

void BSP_LCD_DisplayStringAt(uint16_t X, uint16_t Y, uint8_t *pText, Text_AlignModeTypeDef mode);
void BSP_LCD_DisplayChar(uint16_t Xpos, uint16_t Ypos, uint8_t Ascii);

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, int16_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);
```c
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, int16_t Ypos, 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);

/* This function can be modified in case the current settings need to be changed for specific application needs */
void BSP_LCD_MspInit(void);

/**
 * @}
 */
/**
 * @}
 */
/**
 * @}
 */
/**
 * @}
 */
```
```c
#ifdef __cplusplus
#endif
#endif /* __STM32F429I_DISCOVERY_LCD_H */
#endif
/*	__STM32F429I_DISCOVERY_LCD_H	*/
/****************************	(C)	COPYRIGHT	STMicroelectronics	*****END	OF	FILE****/
```

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery
BSP User Manual by doxygen 1.7.6.1
stm32f429i_discovery_lcd.c

Go to the documentation of this file.

```c
/**
 * @file    stm32f429i_discovery_lcd.c
 * @author  MCD Application Team
 * @version V2.1.5
 * @date    27-January-2017
 * @brief   This file includes the LCD driver for ILI9341 Liquid Crystal Display Modules of STM32F429I-Discovery kit (MB1075).
 * @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.
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 :

   o Initialize the LCD using the LCD_Init() function.
   o Apply the Layer configuration using LCD_LayerDefaultInit() function
   o Select the LCD layer to be used using LCD_SelectLayer() function.
   o Enable the LCD display using LCD_DispPlayOn() function.

   + Options

   o Configure and enable the color keying functionality using LCD_SetColorKeying() function.
Modify in the fly the transparency and/or the frame buffer address using the following functions:
- LCD_SetTransparency()
- LCD_SetLayerAddress()

+ Display on LCD
  - Clear the hole LCD using LCD_Clear() function or only one specified string
  - 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
  - Draw and fill a basic shapes (dot, line, rectangle, circle, ellipse, .. bitmap)

--------------------------------------------

/* Includes ----------------------------------*/
#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	STM32F429I_DISCOVERY_LCD
@brief This file includes the LCD driver for (ILI9341)
*/

/**
@defgroup STM32F429I_DISCOVERY_LCD_Private_TypesDefinitions	STM32F429I_DISCOVERY_LCD_Private_TypesDefinitions
*/

/**
@defgroup STM32F429I_DISCOVERY_LCD_Private_Defines	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_Macros STM32F429I DISCOVERY LCD Private Macros
 * @{
 */
#define ABS(X) ((X) > 0 ? (X) : -(X))
/**
 * @}*/

/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Private_Variables STM32F429I DISCOVERY LCD Private Variables
 * @{
 */

LTDC_HandleTypeDef LdttcHandler;
static DMA2D_HandleTypeDef Dma2dHandler;
static RCC_PeriphCLKInitTypeDef PeriphClkInitStruct;

/* Default LCD configuration with LCD Layer 1 */
static uint32_t ActiveLayer = 0;
static LCD_DrawPropTypeDef DrawProp[MAX_LAYER_NUMBER];

LCD_DrvTypeDef *LcdDrv;
/**
 * @}*/

/**
 * @defgroup STM32F429I_DISCOVERY_LCD_Private_FunctionPrototypes STM32F429I DISCOVERY LCD Private FunctionPrototypes
 * @{
 */

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 th
en LCD_READ_ID4 is not accessible. */
    /* In this case, ReadID function is bypassed. */
    /*if(ili9341_drv.ReadID() == ILI9341_ID)*/
    /* LTDC Configuration ------------------
----------------------------------*/
    LdtcHandler.Instance = LTDC;
    /* Timing configuration (Typical configurat
ion from ILI9341 datasheet)
    HSYNC=10 (9+1)
    HBP=20 (29-10+1)
    ActiveW=240 (269-20-10+1)*/
HFP=10 (279-240-20-10+1)
VSYNC=2 (1+1)
VBP=2 (3-2+1)
ActiveH=320 (323-2-2+1)
VFP=4 (327-320-2-2+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 */
00195    LtdcHandler.Init.Backcolor.Red = 0;
00196    LtdcHandler.Init.Backcolor.Blue = 0;
00197    LtdcHandler.Init.Backcolor.Green = 0;
00198
00199    /* LCD clock configuration */
00200    /* PLLSAI_VCO Input = HSE_VALUE/PLL_M = 1 Mhz */
00201    /* PLLSAI_VCO Output = PLLSAI_VCO Input * PLLSAIN = 192 Mhz */
00202    /* PLLLCDCLK = PLLSAI_VCO Output/PLLSAIR = 192/4 = 48 Mhz */
00203    /* LTDC clock frequency = PLLLCDCLK / LTDC_PLLSAI_DIVR_8 = 48/4 = 6Mhz */
00204   PeriphClkInitStruct.PeriphClockSelection = RCC_PERIPHCLK_LTDC;
00205   PeriphClkInitStruct.PLLSAI.PLLSAIN = 192;
00206   PeriphClkInitStruct.PLLSAI.PLLSAIR = 4;
00207   PeriphClkInitStruct.PLLSAIDivR = RCC_PLLSAIDIVR_8;
00208    HAL_RCCEx_PeriphCLKConfig(&PeriphClkInitStruct);
00209
00210    /* Polarity */
00211    LtdcHandler.Init.HSPolarity = LTDC_HSPOLARITY_AL;
00212    LtdcHandler.Init.VSPolarity = LTDC_VSPOLARITY_AL;
00213    LtdcHandler.Init.DEPolarity = LTDC_DEPOLARITY_AL;
00214    LtdcHandler.Init.PCPolarity = LTDC_PCPOLARITY_IPC;
00215
00216    BSP_LCD_MspInit();
00217    HAL_LTDC_Init(&LtcHandler);
/* 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.FBStartAddress = 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(&LttcHandler, &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 Sets an LCD Layer visible without reloading.
 * @param LayerIndex: Visible Layer
 * @param State: New state of the specified layer
 * This parameter can be one of the following values:
 * @arg ENABLE
 * @arg DISABLE
 * @retval None
 */

void BSP_LCD_SetLayerVisible_NoReload(uint32_t LayerIndex, FunctionalState State)
{
    if (State == ENABLE)
    {
        __HAL_LTDC_LAYER_ENABLE(&LtdcHandler, LayerIndex);
    }
    else
    {
        __HAL_LTDC_LAYER_DISABLE(&LtdcHandler, LayerIndex);
    }
    /* Do not Sets the Reload */

/**
 * @brief Configures the Transparency.
 * @param LayerIndex: the Layer foreground or background.
 * @param Transparency: the Transparency, This parameter must range from 0x00 to 0xFF.
 */
void BSP_LCD_SetTransparency(uint32_t LayerIndex, uint8_t Transparency) {
    HAL_LTDC_SetAlpha(&LtdcHandler, Transparency, LayerIndex);
}

/**
 * @brief Configures the transparency without reloading.
 * @param LayerIndex: Layer foreground or background.
 * @param Transparency: Transparency
 * This parameter must be a number between Min_Data = 0x00 and Max_Data = 0xFF
 * @retval None
 */
void BSP_LCD_SetTransparency_NoReload(uint32_t LayerIndex, uint8_t Transparency) {
    HAL_LTDC_SetAlpha_NoReload(&LtdcHandler, Transparency, LayerIndex);
}

/**
 * @brief Sets a LCD layer frame buffer address.
 * @param LayerIndex: specifies the Layer foreground or background
 * @param Address: new LCD frame buffer value
 * @retval None
 */
void BSP_LCD_SetLayerAddress(uint32_t LayerIndex, uint32_t Address) {
    HAL_LTDC_SetAddress(&LtdcHandler, Address,

/**
 * @brief Sets an LCD layer frame buffer address without reloading.
 * @param LayerIndex: Layer foreground or background
 * @param Address: New LCD frame buffer value
 * @retval None
 */
void BSP_LCD_SetLayerAddress_NoReload(uint32_t LayerIndex, uint32_t Address)
{
    HAL_LTDC_SetAddress_NoReload(&LtdcHandler, Address, LayerIndex);
}

/**
 * @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, X
void BSP_LCD_SetLayerWindow_NoReload(uint16_t LayerIndex,
uint16_t Xpos, uint16_t Ypos, uint16_t Width,
uint16_t Height)
{
/* Reconfigure the layer size */
HAL_LTDC_SetWindowSize_NoReload(&LtdcHandle,
Width, Height, LayerIndex);
/* Reconfigure the layer position */
HAL_LTDC_SetWindowPosition_NoReload(&LtdcHandle,
Xpos, Ypos, LayerIndex);

/* @brief Configures and sets the color Keying.
@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 f
/**
 * @brief Configures and sets the color keying without reloading.
 * @param LayerIndex: Layer foreground or background
 * @param RGBValue: Color reference
 * @retval None
 */

void BSP_LCD_SetColorKeying_NoReload(uint32_t LayerIndex, uint32_t RGBValue) {
    /* Configure and Enable the color Keying for LCD Layer */
    HAL_LTDC_ConfigColorKeying_NoReload(&LtdcHandler, RGBValue, LayerIndex);
    HAL_LTDC_EnableColorKeying_NoReload(&LtdcHandler, LayerIndex);
}

/**
 * @brief Disables the color Keying.
 * @param LayerIndex: the Layer foreground or background
 */

void BSP_LCD_ResetColorKeying(uint32_t LayerIndex) {
    /* Disable the color Keying for LCD Layer */
    HAL_LTDC_DisableColorKeying(&LtdcHandler,
    }
LayerIndex;  
00450 }
00451 
00452 /**<
00453 * @brief Disables the color keying without reloading.
00454 * @param LayerIndex: Layer foreground or background
00455 * @retval None
00456 */
00457 void BSP_LCD_ResetColorKeying_NoReload(uint32_t LayerIndex)
00458 {
00459    /* Disable the color Keying for LCD Layer */
00460    HAL_LTDC_DisableColorKeying_NoReload(&LtdcHandler, LayerIndex);
00461 }
00462 
00463 /**<
00464 * @brief Disables the color keying without reloading.
00465 * @param ReloadType: can be one of the following values
00466 *     - LCD_RELOAD_IMMEDIATE
00467 *     - LCD_RELOAD_VERTICAL_BLANKING
00468 * @retval None
00469 */
00470 void BSP_LCD_ReLoad(uint32_t ReloadType)
00471 {
00472    HAL_LTDC_ReLoad (&LtdcHandler, ReloadType);
00473 }
00474 
00475 /**<
00476 * @brief Gets the LCD Text color.
00477 * @retval Text color

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;
}

void BSP_LCD_SetBackColor(uint32_t Color) {
    DrawProp[ActiveLayer].BackColor = Color;
}

/*
 * @brief Gets the LCD Background color.
 * @retval Background color
 */
uint32_t BSP_LCD_GetBackColor(void) {
    return DrawProp[ActiveLayer].BackColor;
}

/*
 * @brief Sets the Text color.
 * @param Color: the Text color code ARGB(8-8-8-8)
 */
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.
 */
```c
/**
 * @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 + (4*(Ypos*BSP_LCD_GetXSize()+Xpos)));}
    else if(LtdcHandler.LayerCfg[ActiveLayer].PixelFormat == LTDC_PIXEL_FORMAT_BGR8888)
    {
        /* Read data value from SDRAM memory */
        ret = *(__IO.uint32_t*)(LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (4*(Ypos*BSP_LCD_GetXSize()+Xpos)));}
    else
    {
        /* Read data value from SDRAM memory */
        ret = *(__IO.uint32_t*)(LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (4*(Ypos*BSP_LCD_GetXSize()+Xpos)));}
    }
```
PixelFormat == LTDC_PIXEL_FORMAT_RGB888) {
    /* Read data value from SDRAM memory */
    ret = (*(__IO uint32_t*) (LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (4*(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 */

```c
FillBuffer(ActiveLayer, (uint32_t *)(LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress), BSP_LCD_GetXSize(), BSP_LCD_GetYSize(), 0, Color);
```

00573 }

00574 00575 /**
00576 * @brief Clears the selected line.
00577 * @param Line: the line to be cleared
00578 */

```c
void BSP_LCD_ClearStringLine(uint32_t Line)
```

00579 {

00580 uint32_t colorbackup = DrawProp[ActiveLayer].TextColor;

00581 DrawProp[ActiveLayer].TextColor = DrawProp[ActiveLayer].BackColor;

00582 /* Draw rectangle with background color */

00583 BSP_LCD_FillRect(0, (Line * DrawProp[ActiveLayer].pFont->Height), BSP_LCD_GetXSize(), DrawProp[ActiveLayer].pFont->Height);

00584 DrawProp[ActiveLayer].TextColor = colorbackup;

00585 BSP_LCD_SetTextColor(DrawProp[ActiveLayer].TextColor);

00586 }

00587 00588 /**
00589 * @brief Displays one character.
00590 * @param Xpos: start column address
00591 * @param Ypos: the Line where to display the character shape
00592 * @param Ascii: character ascii code, must be between 0x20 and 0x7E
00593 */

```c
void BSP_LCD_DisplayChar(uint16_t Xpos, uint
```
16_t Ypos, uint8_t Ascii)
00598 { 
00599   DrawChar(Xpos, Ypos, &DrawProp[ActiveLayer ].pFont->table[(Ascii-' ') * 
00600       DrawProp[ActiveLayer].pFont->Height * ((DrawProp[ActiveLayer].pFont->Width + 7) / 8)));
00601 }
00602
00603 /**<
00604   * @brief  Displays a maximum of 60 char on the LCD.
00605   * @param  X: pointer to x position (in pixel)
00606   * @param  Y: pointer to y position (in pixel)
00607   * @param  pText: pointer to string to display on LCD
00608   * @param  mode: The display mode
00609   *   This parameter can be one of the following values:
00610   *   @arg CENTER_MODE
00611   *   @arg RIGHT_MODE
00612   *   @arg LEFT_MODE
00613 */
00614 void BSP_LCD_DisplayStringAt(uint16_t X, uint16_t Y, uint8_t *pText, Text_AlignModeTypeDef mode 
00615 { 
00616   uint16_t refcolumn = 1, i = 0;
00617   uint32_t size = 0, xsize = 0;
00618   uint8_t  *ptr = pText;
00619
00620   /* Get the text size */
00621   while (*ptr++) size ++ ;
00622
00623   /* 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)
{
BSP_LCD_DisplayStringAt(0, LINE(Line), ptr, LEFT_MODE);
}

/**
 * @brief Displays an horizontal line.
 * @param Xpos: the X position
 * @param Ypos: the Y position
 * @param Length: line length
 */

void BSP_LCD_DrawHLine(uint16_t Xpos, uint16_t Ypos, uint16_t Length)
{
uint32_t xaddress = 0;

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

**@brief** Displays a vertical line.
**@param** Xpos: the X position
**@param** Ypos: the Y position
**@param** Length: line length

**@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 (X2 >= X1)    /* The x-values are increasing */
    {
        xinc1 = 1;
        xinc2 = 1;
    }
    else    /* 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;
    }
    while (numpixels < num + 1)
    {
        if (deltay > deltax)
        {
            deltay--; /* The y-value needs to be decremented */
            y -= yinc2;
        }
        else
        {
            deltax--; /* The x-value needs to be decremented */
            x -= xinc2;
        }
        curpixel += 1;
        if (curpixel < num + 1)
        {
            curpixel = (deltax | (deltay << 8)) ? deltay : deltax;
        }
        BSP_LCD_DrawPixel(x, y);
    }
}
00737   {  
00738       yinc1 = 1;  
00739       yinc2 = 1;  
00740   }
00741   else /* The y-values are decreasing */
00742   {
00743       yinc1 = -1;  
00744       yinc2 = -1;  
00745   }
00746
00747   if (deltax >= deltay) /* There is at least one x-value for every y-value */
00748     {
00749       xinc1 = 0; /* Don't change the x when numerator >= denominator */
00750       yinc2 = 0; /* Don't change the y for every iteration */
00751       den = deltax;
00752       num = deltax / 2;
00753       numadd = deltay;
00754       numpixels = deltax; /* There are more x-values than y-values */
00755   }
00756   else /* There is at least one y-value for every x-value */
00757     {
00758       xinc2 = 0; /* Don't change the x for every iteration */
00759       yinc1 = 0; /* Don't change the y when numerator >= denominator */
00760       den = deltay;
00761       num = deltay / 2;
00762       numadd = deltax;
00763       numpixels = deltay; /* There are more y-values than x-values */
00764   }
for (curpixel = 0; curpixel <= numpixels; curpixel++)
    
    BSP_LCD_DrawPixel(x, y, DrawProp[ActiveLayer].TextColor);  /* Draw the current pixel */

    num += numadd;
    /* Increase the numerator by the top of the fraction */

    if (num >= den)
        /* Check if numerator >= denominator */
        
        num -= den;
        /* Calculate the new numerator value */

        x += xinc1;
        /* Change the x as appropriate */

        y += yinc1;
        /* Change the y as appropriate */

    }

    x += xinc2;
    /* Change the x as appropriate */

    y += yinc2;
    /* Change the y as appropriate */

/**
 * @brief Displays a rectangle.
 * @param Xpos: the X position
 * @param Ypos: the Y position
 * @param Height: display rectangle height
 * @param Width: display rectangle width
 */

void BSP_LCD_DrawRect(uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
    
    /* Draw horizontal lines */
BSP_LCD_DrawHLine(Xpos, Ypos, Width);
BSP_LCD_DrawHLine(Xpos, (Ypos + Height), Width);

/* Draw vertical lines */
BSP_LCD_DrawVLine(Xpos, Ypos, Height);
BSP_LCD_DrawVLine((Xpos + Width), Ypos, Height);
}

/**
 * @brief Displays a circle.
 * @param Xpos: the X position
 * @param Ypos: the Y position
 * @param Radius: the circle radius
 */
void BSP_LCD_DrawCircle(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;

    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 + curx), (Ypos + cury), DrawProp[ActiveLayer].TextColor);
        BSP_LCD_DrawPixel((Xpos - curx), (Ypos + cury), DrawProp[ActiveLayer].TextColor);
        curx++;
    }
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);

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;
    }
void 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
 */
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);

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

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

/* Read bit/pixel */
bitpixel = *(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)
{
    inputcolormode = CM_ARGB8888;
}
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
 * @param Width: rectangle width
 */

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 *)xadresse, Width, Height, (BSP_LCD_GetXSize() - Width), DrawProp[ActiveLayer].TextColor);

/**
 * @brief Displays a full circle.
 * @param Xpos: the X position
 * @param Ypos: the Y position
 * @param Radius: the circle radius
 */
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;
}

else /* 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 */
01062  {
01063      yinc1 = -1;
01064      yinc2 = -1;
01065  }
01066
01067  if (deltax >= deltay)    /* There is
01068      at least one x-value for every y-value */
01069  {
01070      xinc1 = 0;       /* Don't cha
01071          nge the x when numerator >= denominator */
01072      yinc2 = 0;       /* Don't cha
01073          nge the y for every iteration */
01074      den = deltax;
01075      num = deltax / 2;
01076      numadd = deltay;
01077      numpixels = deltax;    /* There are
01078          more x-values than y-values */
01079  }
01080  else    /* There is
01081      at least one y-value for every x-value */
01082  {
01083      xinc2 = 0;       /* Don't cha
01084          nge the x for every iteration */
01085      yinc1 = 0;       /* Don't cha
01086          nge the y when numerator >= denominator */
01087      den = deltay;
01088      num = deltay / 2;
01089      numadd = deltax;
01090      numpixels = deltay;    /* There are
01091          more y-values than x-values */
01092  }
01093
01094  for (curpixel = 0; curpixel <= numpixels;
01095      curpixel++)
01096  {
01097      BSP_LCD_DrawLine(x, y, X3, Y3);
01090  num += numadd;            /* Increase the numerator by the top of the fraction */
01091  if (num >= den)            /* Check if numerator >= denominator */
01092  {
01093      num -= den;          /* Calculate the new numerator value */
01094      x += xinc1;         /* Change the x as appropriate */
01095      y += yinc1;         /* Change the y as appropriate */
01096  }
01097      x += xinc2;         /* Change the x as appropriate */
01098      y += yinc2;         /* Change the y as appropriate */
01099  }
01100 }
01101
01102 /**
01103  * @brief Displays a full poly-line (between many points).
01104  * @param Points: pointer to the points array
01105  * @param PointCount: Number of points
01106  */
01107 void BSP_LCD_FillPolygon(pPoint Points, uint16_t PointCount)
01108 {
01109  int16_t x = 0, y = 0, x2 = 0, y2 = 0, xcenter = 0, ycenter = 0, xfirst = 0, yfirst = 0, pixelx = 0, pixely = 0, counter = 0;
01110  uint16_t imageleft = 0, imageright = 0, imagetop = 0, imagebottom = 0;
01111  imageleft = imageright = Points->X;
imagetop = imagebottom = Points->Y;

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, yfirst, y2, ycenter);
BSP_LCD_FillTriangle(xfirst, xcenter, x2, yfirst, 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.
*/
__weak void BSP_LCD_MspInit(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;

    /* Enable the LTDC and DMA2D Clock */
    __HAL_RCC_LTDC_CLK_ENABLE();
    __HAL_RCC_DMA2D_CLK_ENABLE();

    /* Enable GPIOs clock */
    __HAL_RCC_GPIOA_CLK_ENABLE();
    __HAL_RCC_GPIOB_CLK_ENABLE();
    __HAL_RCC_GPIOC_CLK_ENABLE();
    __HAL_RCC_GPIOD_CLK_ENABLE();
    __HAL_RCC_GPIOE_CLK_ENABLE();
    __HAL_RCC_GPIOF_CLK_ENABLE();
    __HAL_RCC_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 |

GPIOA configuration:

```c
GPIO_InitStructure.Pin = GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_6 |
GPIO_PIN_11 | GPIO_PIN_12;
GPIO_InitStructure.Mode = GPIO_MODE_AF_PP;
GPIO_InitStructure.Pull = GPIO_NOPULL;
GPIO_InitStructure.Speed = GPIO_SPEED_FAST;
GPIO_InitStructure.Alternate = GPIO_AF14_LT
```
DC;
01269    HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
01270
01271  /* GPIOB configuration */
01272    GPIO_InitStructure.Pin = GPIO_PIN_8 | \
01273              GPIO_PIN_9 | GPIO_PIN_10 | GPIO_PIN_11;
01274    HAL_GPIO_Init(GPIOB, &GPIO_InitStructure);
01275
01276  /* GPIOC configuration */
01277    GPIO_InitStructure.Pin = GPIO_PIN_6 | GPIO_PIN_7 | GPIO_PIN_10;
01278    HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);
01279
01280  /* GPIOD configuration */
01281    GPIO_InitStructure.Pin = GPIO_PIN_3 | GPIO_PIN_6;
01282    HAL_GPIO_Init(GPIOD, &GPIO_InitStructure);
01283
01284  /* GPIOF configuration */
01285    GPIO_InitStructure.Pin = GPIO_PIN_10;
01286    HAL_GPIO_Init(GPIOF, &GPIO_InitStructure);
01287
01288  /* GPIOG configuration */
01289    GPIO_InitStructure.Pin = GPIO_PIN_6 | GPIO_PIN_7 | \n01290              GPIO_PIN_11;
01291    HAL_GPIO_Init(GPIOG, &GPIO_InitStructure);
01292
01293  /* GPIOB configuration */
01294    GPIO_InitStructure.Pin = GPIO_PIN_0 | GPIO_PIN_1;
01295    GPIO_InitStructure.Alternate= GPIO_AF9_LTD;
01296    HAL_GPIO_Init(GPIOB, &GPIO_InitStructure);
01297
/* GPIOG configuration */
GPIO_InitStructure.Pin = GPIO_PIN_10 | GPIO_PIN_12;
HAL_GPIO_Init(GPIOG, &GPIO_InitStructure);

/*******************************************
* Static Functions
_______________________________________

/**
* @brief Writes Pixel.
* @param Xpos: the X position
* @param Ypos: the Y position
* @param RGB_Code: the pixel color in ARG B mode (8-8-8-8)
*/
void BSP_LCD_DrawPixel(uint16_t Xpos, uint16_t Ypos, uint32_t RGB_Code) {
    /* Write data value to all SDRAM memory */
    *((__IO uint32_t*)) (LtdcHandler.LayerCfg[ActiveLayer].FBStartAdress + (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
 * @param ySize: buffer height
 * @param OffLine: offset
 * @param ColorIndex: color Index
 */

static void FillBuffer(uint32_t LayerIndex, void * pDst, uint32_t xSize, uint32_t ySize, uint32_t OffLine, uint32_t ColorIndex)
{
    /* Register to memory mode with ARGB8888 as color Mode */
    Dma2dHandler.Init.Mode = DMA2D_R2M;
    Dma2dHandler.Init.ColorMode = DMA2D_ARGB8888;
Dma2dHandler.Init.OutputOffset = OffLine;

Dma2dHandler.Instance = DMA2D;

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

/**
 * @}
<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 SDRAM</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STM32F429I DISCOVERY</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Modules

<table>
<thead>
<tr>
<th>Category</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>

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery 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>Modules</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I**

**DISCOVERY TS**

STM32F429I DISCOVERY
## Modules

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

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery 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>Data Structures</td>
<td></td>
</tr>
</tbody>
</table>

## STM32F429I DISCOVERY TS Exported Types

STM32F429I DISCOVERY TS

---

**Tags:**
- STM32F429I
- DISCOVERY TS
- Exported Types
## Data Structures

```c
struct TS_StateTypeDef
```

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery BSP User Manual by [doxygen](https://www.doxygen.org/) 1.7.6.1
stm32f429i_discovery_ts.h

Go to the documentation of this file.

00001 **
00002 ******************************************
00003 * @file stm32f429i_discovery_ts.h
00004 * @author MCD Application Team
00005 * @version V2.1.5
00006 * @date 27-January-2017
00007 * @brief This file contains all the function prototypes for the
00008 * stm32f429i_discovery_ts.c driver.
00009 ******************************************
00010 * @attention
00011 *
00012 * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00013 *
00014 * Redistribution and use in source and binary forms, with or without modification,
00015 * are permitted provided that the following conditions are met:
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_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
 * @{
 */

/** @defgroup STM32F429I_DISCOVERY_TS_Export ed_Types STM32F429I DISCOVERY TS Exported Types
 * @{
 */

00035  *
00036  ******************************************************
00037  */
00038
00039  /* Define to prevent recursive inclusion -------------------------------*/
00040  ifndef __STM32F429I_DISCOVERY_TS_H
00041  define __STM32F429I_DISCOVERY_TS_H
00042
00043  ifdef __cplusplus
00044    extern "C" {
00045    endif
00046
00047  /* Includes ---------------------------------- ----------------------------------*/
00048  include "stm32f429i_discovery.h"
00049  /* Include TouchScreen component driver */
00050  include "../Components/stmpe811/stmpe811.h"
00051
00052  /** @addtogroup BSP
00053    * @{
00054    */
00055
00056  /** @addtogroup STM32F429I_DISCOVERY
00057    * @{
00058    */
00059
00060  /** @addtogroup STM32F429I_DISCOVERY_TS
00061    * @{
00062    */
00063
00064  /** @defgroup STM32F429I_DISCOVERY_TS_Export ed_Types STM32F429I DISCOVERY TS Exported Types
00065    * @{
00066    */
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
 * }
/**
 * @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.

```c
/**
 * @file stm32f429i_discovery_ts.c
 * @author MCD Application Team
 * @version V2.1.5
 * @date 27-January-2017
 * @brief This file provides a set of functions needed to manage Touch screen available with STMPE811 IO Expander device mounted on STM32F429I-Discovery Kit.

 * Redistributions and use in source and binary forms, with or without modification,
 * are permitted provided that the following
```
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, 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_ts.h"
#include "stm32f429i_discovery_io.h"

/** @addtogroup BSP */
* @{
*/

/** @addtogroup STM32F429I_DISCOVERY */
* @{
*/

/** @addtogroup STM32F429I_DISCOVERY_TS */
* @{
*/

/** @defgroup STM32F429I_DISCOVERY_TS_Private_Types_Definitions */
* @{
*/

/** @defgroup 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
 */
/*
@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;

    /* Initialize x and y positions boundaries */
    TsXBoundary = XSize;
    TsYBoundary = YSize;

    /* 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 the TS driver structure */
        }
/* 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
 */
current state structure

```c
00156   */
00157 void BSP_TS_GetState(TS_StateTypeDef* TsState) {
00159     static uint32_t _x = 0, _y = 0;
00160     uint16_t xDiff, yDiff, x, y, xr, yr;
00161     TsState->TouchDetected = TsDrv->DetectTouch(TS_I2C_ADDRESS);
00163     if(TsState->TouchDetected) {
00166         TsDrv->GetXY(TS_I2C_ADDRESS, &x, &y);
00168         /* Y value first correction */
00169         y -= 360;
00170         /* Y value second correction */
00172         yr = y / 11;
00174         /* Return y position value */
00175         if(yr <= 0) {
00177             yr = 0;
00178         }
00179         else if (yr > TsYBoundary) {
00181             yr = TsYBoundary - 1;
00182         }
00183         else
00184             {};
00185         y = yr;
00186     /* X value first correction */
00188     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****/
<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 EEPROM**

STM32F429I DISCOVERY
### Modules

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

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery

BSP User Manual by [doxygen](https://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>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drivers</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Drivers 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><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drivers</td>
<td>BSP</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><strong>Directories</strong></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><code>stm32f429i_discovery.c</code></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><code>stm32f429i_discovery.h</code></td>
<td>This file contains definitions for STM32F429I-Discovery Kit LEDs, push-buttons hardware resources.</td>
</tr>
<tr>
<td><code>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 toolchain compiler preprocessor.</td>
</tr>
<tr>
<td><code>stm32f429i_discovery_eeprom.h</code></td>
<td>This file contains all the functions prototypes for the firmware driver.</td>
</tr>
<tr>
<td><code>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><code>stm32f429i_discovery_gyroscope.h</code></td>
<td></td>
</tr>
</tbody>
</table>


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

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>stm32f429i_discovery_io.c</strong> [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><strong>stm32f429i_discovery_io.h</strong> [code]</td>
<td>This file contains all the functions prototypes for the <strong>stm32f429i_discovery_io.c</strong> driver.</td>
</tr>
<tr>
<td><strong>stm32f429i_discovery_lcd.c</strong> [code]</td>
<td>This file includes the LCD driver for ILI9341 Liquid Crystal Display Modules of STM32F429I-Discovery kit (MB1075).</td>
</tr>
<tr>
<td><strong>stm32f429i_discovery_lcd.h</strong> [code]</td>
<td>This file contains all the functions prototypes for the <strong>stm32f429i_discovery_lcd.c</strong> driver.</td>
</tr>
<tr>
<td><strong>stm32f429i_discovery_sdram.c</strong> [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><strong>stm32f429i_discovery_sdram.h</strong> [code]</td>
<td></td>
</tr>
</tbody>
</table>
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.

```plaintext
00001 /**
00002 **************************************************
00003 * @file    stm32f429i_discovery_eeprom.h
00004 * @author  MCD Application Team
00005 * @version V2.1.5
00006 * @date    27-January-2017
00007 * @brief   This file contains all the functions prototypes for
00008 *           the stm32f429i_discovery_eeprom.c firmware driver.
00009 **************************************************
00010 * @attention
00011 *
00012 *  <h2><center>&copy; COPYRIGHT(c) 2017 STM microelectronics</center></h2>
00013 *
00014 * Redistribution and use in source and binary forms, with or without modification,
00015 * are permitted provided that the following conditions are met:
```
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

#ifdef __cplusplus
extern "C" {
#endif

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

/** @addtogroup BSP
 * @{
 */

/** @addtogroup STM32F429I_DISCOVERY
 * @{
 */

/** @addtogroup STM32F429I_DISCOVERY_EEPROM
STM32F429I DISCOVERY EEPROM
*/

/** @defgroup STM32F429I_DISCOVERY_EEPROM_Ex
ported_Types STM32F429I DISCOVERY EEPROM Exported Types
*/

/** @defgroup STM32F429I_DISCOVERY_EEPROM_Ex
ported_Types STM32F429I DISCOVERY EEPROM Exported Types
*/
/**
 * @defgroup STM32F429I_DISCOVERY_EEPROM_Exported_Constants
 * @brief 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*/
#define EEPROM_READ_TIMEOUT ((uint32_t)(1000))
#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. 
BSP_EEPROM_TIMEOUT_UserCallback() function is called whenever a timeout condition occurs during communication (waiting on an event that doesn't occur, bus errors, busy devices ...). */
void BSP_EEPROM_TIMEOUT_UserCallback(void);
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 /* __cplusplus */

#if defined(__cplusplus)
}
#endif /* __STM32F429I_DISCOVERY_EEPROM_H */

/**************************** (C) COPYRIGHT STMi
stm32f429i_discovery_eeprom.c

Go to the documentation of this file.

00001 /**
00002 ******************************************
00003 **                                    
00004 * @file stm32f429i_discovery_eeprom.c
00005 * @author MCD Application Team
00006 * @version V2.1.5
00007 * @date 27-January-2017
00008 * @brief This file provides a set of functions needed to manage an I2C M24LR64
00009 * EEPROM memory.
00010 * To be able to use this driver, the switch EE_M24LR64 must be defined
00011 * in your toolchain compiler preprocessor
00012 *================================================
00013 * Notes:
00014 * - This driver is intended for STM32F4xx families devices only.
00015 * - The I2C EEPROM memory (M24LR64) is available on separate daughter
board AN7-M24LR-A, which is not provided with the STM32F429I DISCOVERY board.

To use this driver you have to connect the AN7-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 a 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>STM32F4xx I2C Pins</th>
</tr>
</thead>
<tbody>
<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>SDA</td>
</tr>
<tr>
<td>SCL</td>
<td>SCL</td>
</tr>
<tr>
<td>E1(GND)</td>
<td>7 (0V)</td>
</tr>
</tbody>
</table>
* 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
 DISCLAIMS. 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_eeprom.h"

#ifdef EE_M24LR64

/** @addtogroup BSP
 */

/** @addtogroup STM32F429I_DISCOVERY
 */

/** @defgroup STM32F429I_DISCOVERY_EEPROM STM32F429I_DISCOVERY_EEPROM
 */

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

```c
/**
 * @defgroup STM32F429I_DISCOVERY_EEPROM_Private_Types STM32F429I DISCOVERY EEPROM Private Types
 */

/**
 * @defgroup STM32F429I_DISCOVERY_EEPROM_Private_Defines STM32F429I DISCOVERY EEPROM Private Defines
 */

/**
 * @defgroup STM32F429I_DISCOVERY_EEPROM_Private_Macros STM32F429I DISCOVERY EEPROM Private Macros
 */

/**
 * @defgroup STM32F429I_DISCOVERY_EEPROM_Private_Variables STM32F429I DISCOVERY EEPROM Private Variables
 */

__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
 * the EEPROM.
 * @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)
BSP_EEPROM_TIMEOUT_UserCallback();
return EEPROM_TIMEOUT;
}
}

/* If all operations OK, return EEPROM_OK (0) */
return EEPROM_OK;
}

/**
 * @brief Writes more than one byte to the EEPROM with a single WRITE cycle.
 * @note The number of bytes (combined to write start address) must not cross the EEPROM page boundary. This function can only write into the boundaries of an EEPROM page.
 * This function doesn't check on boundaries condition (in this driver the function BSP_EEPROM_WriteBuffer() which calls BSP_EEPROM_WritePage() is responsible of checking on Page boundaries).
 * @param pBuffer : pointer to the buffer containing the data to be written to the EEPROM.
 * @param WriteAddr : EEPROM's internal address to write to.
 * @param NumByteToWrite : pointer to the variable holding number of bytes to be written into the EEPROM.
The variable pointed by NumByteToWrite is reset to 0 when all the data are written to the EEPROM. Application should monitor this variable in order know when the transfer is complete.

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

@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;
    }
    return status;
}
```
    } /* Wait transfer through DMA to be complete */
    EEPROMTimeout = HAL_GetTick() + EEPROM_WRITE_TIMEOUT;
    while (EEPROMDataWrite > 0) {
        if(HAL_GetTick() > EEPROMTimeout) {
            BSP_EEPROM_TIMEOUT_UserCallback();
            return EEPROM_TIMEOUT;
        }
    }
    if (BSP_EEPROM_WaitEepromStandbyState() != EEPROM_OK) {
        return EEPROM_FAIL;
    }
    /* If all operations OK, return EEPROM_OK (0) */
    return status;
}
/**
 * @brief  Writes buffer of data to the I2C EEPROM.
 * @param  pBuffer : pointer to the buffer containing the data to be written
 * @param  WriteAddr : EEPROM's internal address to write to.
 * @param  NumByteToWrite : number of bytes to write to the EEPROM.
 * @retval EEPROM_OK (0) if operation is co
correctly performed, else return value
00291 * different from EEPROM_OK (0) or
the timeout user callback.
00292 */
00293 uint32_t BSP_EEPROM_WriteBuffer(uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite)
00294 {
00295   uint16_t numofpage = 0, numofsingle = 0, count = 0;
00296   uint16_t addr = 0;
00297   uint8_t dataindex = 0;
00298   uint32_t status = EEPROM_OK;
00299   addr = WriteAddr % EEPROM_PAGESIZE;
00300   count = EEPROM_PAGESIZE - addr;
00301   numofpage = NumByteToWrite / EEPROM_PAGESIZE;
00302   numofsingle = NumByteToWrite % EEPROM_PAGESIZE;
00303   /* If WriteAddr is EEPROM_PAGESIZE aligned */
00304   if(addr == 0)
00305     {
00306       /* If NumByteToWrite < EEPROM_PAGESIZE */
00307          if(numofpage == 0)
00308             {
00309                /* Store the number of data to be written */
00310                 dataindex = numofsingle;
00311                /* Start writing data */
00312                 status = BSP_EEPROM_WritePage(pBuffer,
00313                      WriteAddr, (uint8_t*)(&dataindex));
00314                if (status != EEPROM_OK)
00315                  {
00316                     return status;
00317                 }
00318     }  
00319 }  
00320 /* If NumByteToWrite > EEPROM_PAGESIZE */
00321 else  
00322 {
00323     while(numofpage--)
00324     {
00325         /* Store the number of data to be written */
00326         dataindex = EEPROM_PAGESIZE;
00327         status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
00328         if (status != EEPROM_OK)
00329             {
00330                 return status;
00331             }
00332         WriteAddr += EEPROM_PAGESIZE;
00333         pBuffer += EEPROM_PAGESIZE;
00334     }
00335 }
00336
00337    if(numofsingle!=0)
00338    {
00339        /* Store the number of data to be written */
00340        dataindex = numofsingle;
00341        status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
00342        if (status != EEPROM_OK)
00343            {
00344                return status;
00345            }
00346    }
00347 }
00348 /* If WriteAddr is not EEPROM_PAGESIZE ali
signed */
    00350  else
    00351  {
    00352    /* If NumByteToWrite < EEPROM_PAGESIZE */

    00353    if(numofpage== 0)
    00354    {
    00355        /* If the number of data to be written is more than the remaining space in the current page: */
    00356        if (NumByteToWrite > count)
    00357        {
    00358            /* Store the number of data to be written */
    00359            dataindex = count;
    00360            /* Write the data contained in same page */
    00361            status = BSP_EEPROM_WritePage(pBuffer, WriteAddr, (uint8_t*)(&dataindex));
    00362            if (status != EEPROM_OK)
    00363            {
    00364                return status;
    00365            }
    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    }
else {
/* 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 addresses.
 *
 * @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)
{
}

#endif /* EE_M24LR64 */
/**
 * @}
 */

/**
 * @}
 */

/************************	(C)	COPYRIGHT	STMi
croelectronics *****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 Exported Functions

**STM32F429I DISCOVERY EEPROM**
## Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint32_t</td>
<td>BSP_EEPROM_Init</td>
<td>(void)</td>
</tr>
<tr>
<td>uint32_t</td>
<td>BSP_EEPROM_ReadBuffer</td>
<td>(uint8_t *pBuffer, uint16_t ReadAddr, uint16_t *NumByteToRead)</td>
</tr>
<tr>
<td>uint32_t</td>
<td>BSP_EEPROM_WritePage</td>
<td>(uint8_t *pBuffer, uint16_t WriteAddr, uint8_t *NumByteToWrite)</td>
</tr>
<tr>
<td>uint32_t</td>
<td>BSP_EEPROM_WriteBuffer</td>
<td>(uint8_t *pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite)</td>
</tr>
<tr>
<td>void</td>
<td>BSP_EEPROM_WaitEepromStandbyState</td>
<td>(void)</td>
</tr>
<tr>
<td>void</td>
<td>BSP_EEPROM_TIMEOUT_UserCallback</td>
<td>(void)</td>
</tr>
<tr>
<td>void</td>
<td>EEPROM_IO_Init</td>
<td>(void)</td>
</tr>
<tr>
<td>HAL_StatusTypeDef</td>
<td>EEPROM_IO_WriteData</td>
<td>(uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize)</td>
</tr>
<tr>
<td>HAL_StatusTypeDef</td>
<td>EEPROM_IO_ReadData</td>
<td>(uint16_t DevAddress, uint16_t MemAddress, uint8_t *pBuffer, uint32_t BufferSize)</td>
</tr>
<tr>
<td>HAL_StatusTypeDef</td>
<td>EEPROM_IO_IsDeviceReady</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)
<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 SDRAM Exported Constants</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY SDRAM</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```c
#define SDRAM_OK            ((uint8_t)0x00)  // SDRAM status structure definition.
#define SDRAM_ERROR         ((uint8_t)0x01)
#define SDRAM_DEVICE_ADDR   ((uint32_t)0xD0000000)  // FMC SDRAM Bank address.
#define SDRAM_DEVICE_SIZE   ((uint32_t)0x800000)  // FMC SDRAM device size in Bytes
#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
#define SDRAM_READBURST     FMC_SDRAM_RBURST_DISABLE  // FMC SDRAM Memory Read Burst feature.
#define REFRESH_COUNT       ((uint32_t)1386)  // SDRAM refresh counter
#define SDRAM_TIMEOUT       ((uint32_t)0xFFFF)  // FMC SDRAM Bank Remap.
#define __DMAx_CLK_ENABLE   __HAL_RCC_DMA2_CLK_ENABLE  // FMC SDRAM DMA Channel
#define SDRAM_DMAx_CHANNEL  DMA_CHANNEL_0  // DMA CHANNEL 0
#define SDRAM_DMAx_STREAM   DMA2_Stream0  // DMA Channel 0
#define SDRAM_DMAx_IRQn     DMA2_Stream0_IRQn  // DMA Stream 0 IRQ
#define SDRAM_DMAx_IRQHandler DMA2_Stream0_IRQHandler  // DMA Stream 0_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)
```
```c
#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

#define __DMAx_CLK_ENABLE  __HAL_RCC_DMA2_CLK_ENABLE

Definition at line 120 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_MspInit().

#define REFRESH_COUNT  ((uint32_t)1386) /* SDRAM refresh count*/

FMC SDRAM Bank Remap.
Definition at line 116 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Init().

#define SDCLOCK_PERIOD  FMC_SDRAM_CLOCK_PERIOD_2 /* FMC SDRAM Memory clock period.*/

Definition at line 100 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Init().

#define SDRAM_CAS_LATENCY  FMC_SDRAM_CAS_LATENCY_3 /* FMC SDRAM CAS Latency.*/

Definition at line 95 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_Init().

#define SDRAM_DEVICE_ADDR  ((uint32_t)0xD0000000)

#define SDRAM_DEVICE_ADDR  ((uint32_t)0xD0000000)
FMC SDRAM Bank address.
Definition at line 82 of file stm32f429i_discovery_sdram.h.

#define SDRAM_DEVICE_SIZE ((uint32_t)0x800000) /* SDRAM device size in Bytes*/
Definition at line 83 of file stm32f429i_discovery_sdram.h.

#define SDRAM_DMAx_CHANNEL DMA_CHANNEL_0
Definition at line 121 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_MspInit().

#define SDRAM_DMAx_IRQHandler DMA2_Stream0_IRQHandler
Definition at line 124 of file stm32f429i_discovery_sdram.h.

#define SDRAM_DMAx_IRQn DMA2_Stream0_IRQn
Definition at line 123 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_MspDeInit(), and
BSP_SDRAM_MspInit().

#define SDRAM_DMAx_STREAM DMA2_Stream0
Definition at line 122 of file stm32f429i_discovery_sdram.h.
Referenced by BSP_SDRAM_MspDeInit(), and
BSP_SDRAM_MspInit().
```
#define SDRAM_ERROR ((uint8_t)0x01)

Definition at line 77 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Init(), BSP_SDRAM_ReadData(),
BSP_SDRAM_ReadData_DMA(), BSP_SDRAM_Sendcmd(),
BSP_SDRAM_WriteData(), and BSP_SDRAM_WriteData_DMA().
```

```
#define SDRAM_MEMORY_WIDTH FMC_SDRAM_MEM_BUS_WIDTH_16

FMC SDRAM Memory Width.

Definition at line 89 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 129 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Initialization_sequence().
```

```
#define SDRAM_MODEREG_BURST_LENGTH_2 ((uint16_t)0x0001)

Definition at line 130 of file stm32f429i_discovery_sdram.h.
```

```
#define SDRAM_MODEREG_BURST_LENGTH_4 ((uint16_t)0x0002)

Definition at line 131 of file stm32f429i_discovery_sdram.h.
```
#define SDRAM_MODEREG_BURST_LENGTH_8  ((uint16_t)0x0004)

Definition at line 132 of file stm32f429i_discovery_sdram.h.

#define SDRAM_MODEREG_BURST_TYPE_INTERLEAVED  ((uint16_t)0x0008)

Definition at line 134 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_BURST_TYPE_SEQUENTIAL  ((uint16_t)0x0000)

Definition at line 133 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_CAS_LATENCY_2  ((uint16_t)0x0020)

Definition at line 135 of file stm32f429i_discovery_sdram.h.

Definition at line 135 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_CAS_LATENCY_3  ((uint16_t)0x0030)

Definition at line 136 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_OPERATING_MODE_STANDARD  ((uint16_t)0x0000)

Definition at line 137 of file stm32f429i_discovery_sdram.h.

Referenced by BSP_SDRAM_Initialization_sequence().

#define SDRAM_MODEREG_WRITEBURST_MODE_PROGRAMMED
Definition at line 138 of file `stm32f429i_discovery_sdram.h`.

```c
#define SDRAM_MODEREG_WRITEBURST_MODE_SINGLE ((uint16_t)0x0200)
```

Definition at line 139 of file `stm32f429i_discovery_sdram.h`.

Referenced by `BSP_SDRAM_Initialization_sequence()`.

```c
#define SDRAM_OK ((uint8_t)0x00)
```

SDRAM status structure definition.

Definition at line 76 of file `stm32f429i_discovery_sdram.h`.

Referenced by `BSP_SDRAM_Init()`, `BSP_SDRAM_ReadData()`, `BSP_SDRAM_ReadData_DMA()`, `BSP_SDRAM_Sendcmd()`, `BSP_SDRAM_WriteData()`, and `BSP_SDRAM_WriteData_DMA()`.

```c
#define SDRAM_READBURST FMC_SDRAM_RBURST_DISABLE /*
Default configuration used with LCD*/
```

FMC SDRAM Memory Read Burst feature.

Definition at line 106 of file `stm32f429i_discovery_sdram.h`.

Referenced by `BSP_SDRAM_Init()`.

```c
#define SDRAM_TIMEOUT ((uint32_t)0xFFFF)
```

Definition at line 117 of file `stm32f429i_discovery_sdram.h`.

Referenced by `BSP_SDRAM_Initialization_sequence()`, and `BSP_SDRAM_Sendcmd()`.
<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 Defines**

STM32F429I DISCOVERY LOW LEVEL
# Defines

<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION_MAIN</td>
<td>0x02</td>
</tr>
<tr>
<td>STM32F429I DISCO BSP Driver version number</td>
<td>V2.1.5</td>
</tr>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION_SUB1</td>
<td>0x01</td>
</tr>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION_SUB2</td>
<td>0x05</td>
</tr>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION_RC</td>
<td>0x00</td>
</tr>
<tr>
<td>__STM32F429I_DISCO_BSP_VERSION</td>
<td></td>
</tr>
</tbody>
</table>
Define Documentation

#define __STM32F429I_DISCO_BSP_VERSION

Value:

\[
( ( __STM32F429I_DISCO_BSP_VERSION_MAIN \ll 24 ) \mid ( __STM32F429I_DISCO_BSP_VERSION_SUB1 \ll 16 ) \mid ( __STM32F429I_DISCO_BSP_VERSION_SUB2 \ll 8 ) \mid ( __STM32F429I_DISCO_BSP_VERSION_RC ) )
\]

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 V2.1.5.

[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 (0x05)

[15:8] sub2 version
Definition at line 72 of file stm32f429i_discovery.c.

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery 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>
<th>Defines</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>STM32F429I</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DISCOVERY LCD Private Macros</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>STM32F429I DISCOVERY LCD</td>
<td></td>
</tr>
</tbody>
</table>
Defines

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

\#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

STM32F429I DISCOVERY LCD
## Variables

<table>
<thead>
<tr>
<th>Type Definition</th>
<th>Variable Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTDC_HandleTypeDef</td>
<td>LcmdHandler</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>LcmdDrv</td>
</tr>
</tbody>
</table>
Variable Documentation

**uint32_t** ActiveLayer = 0  [static]

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

Referenced by **BSP_LCD_Clear()**, **BSP_LCD_ClearStringLine()**, **BSP_LCD_DisplayChar()**, **BSP_LCD_DisplayStringAt()**, **BSP_LCD_DrawBitmap()**, **BSP_LCD_DrawCircle()**, **BSP_LCD_DrawEllipse()**, **BSP_LCD_DrawHLine()**, **BSP_LCD_DrawLine()**, **BSP_LCD_DrawPixel()**, **BSP_LCD_DrawVLine()**, **BSP_LCD_FillCircle()**, **BSP_LCD_FillRect()**, **BSP_LCD_GetBackColor()**, **BSP_LCD_GetFont()**, **BSP_LCD_GetTextColor()**, **BSP_LCD_ReadPixel()**, **BSP_LCD_SelectLayer()**, **BSP_LCD_SetBackColor()**, **BSP_LCD_SetFont()**, **BSP_LCD_SetTextColor()**, and **DrawChar()**.

**DMA2D_HandleTypeDef** Dma2dHandler  [static]

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

Referenced by **ConvertLineToARGB8888()**, and **FillBuffer()**.

**LCD_DrawPropTypeDef** DrawProp[MAX_LAYER_NUMBER]  [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_LCD_GetXSize(), BSP_LCD_GetYSize(), and 
BSP_LCD_Init.

LTDC_HandleTypeDef LtdcHandler

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_Relaod(), BSP_LCD_ResetColorKeying(), 
BSP_LCD_ResetColorKeying_NoReload(), 
BSP_LCD_SetColorKeying(), 
BSP_LCD_SetColorKeying_NoReload(), 
BSP_LCD_SetLayerAddress(), 
BSP_LCD_SetLayerAddress_NoReload(), 
BSP_LCD_SetLayerVisible(), 
BSP_LCD_SetLayerVisible_NoReload(), 
BSP_LCD_SetLayerWindow(), 
BSP_LCD_SetLayerWindow_NoReload(), 
BSP_LCD_SetTransparency(), and 
BSP_LCD_SetTransparency_NoReload().

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
<table>
<thead>
<tr>
<th>Function Type</th>
<th>Function Name</th>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint32_t</td>
<td>BSP_GetVersion (void)</td>
<td></td>
<td>This method returns the STM32F429I DISCO BSP Driver revision.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_LED_Init (Led_TypeDef Led)</td>
<td>Led_TypeDef Led</td>
<td>Configures LED GPIO.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_LED_On (Led_TypeDef Led)</td>
<td>Led_TypeDef Led</td>
<td>Turns selected LED On.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_LED_Off (Led_TypeDef Led)</td>
<td>Led_TypeDef Led</td>
<td>Turns selected LED Off.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_LED_Toggle (Led_TypeDef Led)</td>
<td>Led_TypeDef Led</td>
<td>Toggles the selected LED.</td>
</tr>
<tr>
<td>void</td>
<td>BSP_PB_Init (Button_TypeDef Button, ButtonMode_TypeDef ButtonMode)</td>
<td>Button_TypeDef Button, ButtonMode_TypeDef ButtonMode</td>
<td>Configures Button GPIO and EXTI Line.</td>
</tr>
<tr>
<td>uint32_t</td>
<td>BSP_PB_GetState (Button_TypeDef Button)</td>
<td>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
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_IRQHandler`, `BUTTON_MODE_GPIO`, and `BUTTON_MODE_EXTI`,
BUTTON_MODE_GPIO, BUTTON_PIN, BUTTON_PORT, and BUTTONx_GPIO_CLK_ENABLE.
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LOW LEVEL Private Functions

STM32F429I DISCOVERY LOW LEVEL
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>uint32_t BSP_GetVersion(void)</code></td>
<td>This method returns the STM32F429I DISCO BSP Driver revision.</td>
</tr>
<tr>
<td><code>void BSP_LED_Init(Led_TypeDef Led)</code></td>
<td>Configures LED GPIO.</td>
</tr>
<tr>
<td><code>void BSP_LED_On(Led_TypeDef Led)</code></td>
<td>Turns selected LED On.</td>
</tr>
<tr>
<td><code>void BSP_LED_Off(Led_TypeDef Led)</code></td>
<td>Turns selected LED Off.</td>
</tr>
<tr>
<td><code>void BSP_LED_Toggle(Led_TypeDef Led)</code></td>
<td>Toggles the selected LED.</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>
</tr>
<tr>
<td><code>uint32_t BSP_PB_GetState(Button_TypeDef Button)</code></td>
<td>Returns the selected Button state.</td>
</tr>
<tr>
<td><code>static void I2Cx_MspInit(I2C_HandleTypeDef *hi2c)</code></td>
<td>I2Cx MSP Initialization.</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>
</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>
</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>
</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>
</tr>
<tr>
<td><code>static uint32_t SPIx_Read(uint8_t ReadSize)</code></td>
<td>Reads 4 bytes from device.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>SPIx_Write</strong> (uint16_t Value)</td>
<td>Writes a byte to device.</td>
</tr>
<tr>
<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><strong>SPIx_MspInit</strong> (SPI_HandleTypeDef *hspi)</td>
<td>SPI MSP Init.</td>
</tr>
<tr>
<td><strong>LCD_IO_ReadData</strong> (uint16_t RegValue, uint8_t ReadSize)</td>
<td>Reads register value.</td>
</tr>
<tr>
<td><strong>LCD_Delay</strong> (uint32_t Delay)</td>
<td>Wait for loop in ms.</td>
</tr>
<tr>
<td><strong>IOE_Write</strong> (uint8_t Addr, uint8_t Reg, uint8_t Value)</td>
<td>IOE Writes single data operation.</td>
</tr>
<tr>
<td><strong>IOE_Read</strong> (uint8_t Addr, uint8_t Reg)</td>
<td>IOE Reads single data.</td>
</tr>
<tr>
<td><strong>IOE_WriteMultiple</strong> (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)</td>
<td>IOE Writes multiple data.</td>
</tr>
<tr>
<td><strong>IOE_ReadMultiple</strong> (uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)</td>
<td>IOE Reads multiple data.</td>
</tr>
<tr>
<td><strong>IOE_Delay</strong> (uint32_t Delay)</td>
<td>IOE Delay.</td>
</tr>
<tr>
<td><strong>GYRO_IO_Write</strong> (uint8_t *pBuffer, uint8_t WriteAddr, uint16_t NumByteToWrite)</td>
<td>Writes one byte to the Gyroscope.</td>
</tr>
<tr>
<td><strong>GYRO_IO_Read</strong> (uint8_t *pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead)</td>
<td>Reads a block of data from the Gyroscope.</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
one of following parameters:
  
  - LED3
  - LED4

Definition at line 229 of file stm32f429i_discovery.c.

References GPIO_PIN, and GPIO_PORT.

```
void BSP_LED_On ( Led_TypeDef Led )

Turns selected LED On.

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

Definition at line 217 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_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 )
```

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
                            ) [static]
```

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()**.

### `void IOE_Delay ( uint32_t Delay )`

IOE Delay.

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

Definition at line 951 of file *stm32f429i_discovery.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()**.

### `uint16_t IOE_ReadMultiple ( uint8_t Addr,
uint8_t Reg,
uint8_t * pBuffer,`
# IOE_ReadBuffer

**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_Write

** 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().

---

```c
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().

---

```c
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

STM32F429I
DISCOVERY GYROSCOPE Private Functions

STM32F429I DISCOVERY GYROSCOPE
## 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</code> (void)</td>
<td>Set Gyroscope Initialization.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_GYRO_ReadID</code> (void)</td>
<td>Read ID of Gyroscope component.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_Reset</code> (void)</td>
<td>Reboot memory content of Gyroscope.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_ITConfig</code> (GYRO_InterruptConfigTypeDef *pIntConfig)</td>
<td>Configures INT1 interrupt.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_EnableIT</code> (uint8_t IntPin)</td>
<td>Enables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_DisableIT</code> (uint8_t IntPin)</td>
<td>Disables INT1 or INT2 interrupt.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_GYRO_GetXYZ</code> (float *pfData)</td>
<td>Gets XYZ angular acceleration.</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_InterruptConfigTypeDef 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**
void BSP_GYRO_Reset ( void )

Reboot memory content of Gyroscope.

Definition at line 154 of file stm32f429i_discovery_gyroscope.c.
References GyroscopeDrv.

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

Set Gyroscope Initialization.

Return values:
GYRO_OK if no problem during initialization

void BSP_GYRO_ITConfig ( GYRO_InterruptConfigTypeDef * pIntC)

Configures INT1 interrupt.

Parameters:
 pIntConfig: pointer to a L3GD20_InterruptConfigTypeDef structure that contains the configuration setting for the L3GD20 Interrupt.

uint8_t BSP_GYRO_ReadID ( void )

Read ID of Gyroscope component.

Return values:
ID
<table>
<thead>
<tr>
<th>Definition at line 154 of file <code>stm32f429i_discovery_gyroscope.c</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>References <strong>GyroscopeDrv</strong>.</td>
</tr>
</tbody>
</table>

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

Reboot memory content of Gyroscope.

<table>
<thead>
<tr>
<th>Definition at line 168 of file <code>stm32f429i_discovery_gyroscope.c</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>References <strong>GyroscopeDrv</strong>.</td>
</tr>
</tbody>
</table>

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery
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>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY IO Private Functions**

STM32F429I DISCOVERY IO
## Functions

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

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>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>
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 `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.**


```c
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`.

```c
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`.

```c
void BSP_IO_WritePin ( uint16_t IoPin, uint8_t PinState )
```

Sets the selected pins state.

**Parameters:**

- **IoPin:**
- **PinState:**
**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`.
<table>
<thead>
<tr>
<th>Directories</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY LCD Private Functions</td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY LCD
## Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><code>BSP_LCD_Init (void)</code></td>
<td>Initializes the LCD.</td>
</tr>
<tr>
<td>uint32_t</td>
<td><code>BSP_LCD_GetXSize (void)</code></td>
<td>Gets the LCD X size.</td>
</tr>
<tr>
<td>uint32_t</td>
<td><code>BSP_LCD_GetYSize (void)</code></td>
<td>Gets the LCD Y size.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_LayerDefaultInit (uint16_t LayerIndex, uint32_t FB_Address)</code></td>
<td>Initializes the LCD layers.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SelectLayer (uint32_t LayerIndex)</code></td>
<td>Selects the LCD Layer.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SetLayerVisible (uint32_t LayerIndex, FunctionalState state)</code></td>
<td>Sets a LCD Layer visible.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SetLayerVisible_NoReload (uint32_t LayerIndex, FunctionalState State)</code></td>
<td>Sets an LCD Layer visible without reloading.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SetTransparency (uint32_t LayerIndex, uint8_t Transparency)</code></td>
<td>Configures the Transparency.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SetTransparency_NoReload (uint32_t LayerIndex, uint8_t Transparency)</code></td>
<td>Configures the transparency without reloading.</td>
</tr>
<tr>
<td>void</td>
<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><code>BSP_LCD_SetLayerAddress_NoReload (uint32_t LayerIndex, uint32_t Address)</code></td>
<td>Sets an LCD layer frame buffer address without reloading.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_LCD_SetLayerWindow (uint16_t LayerIndex, uint32_t WindowAddress)</code></td>
<td>Sets a LCD layer window address.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerWindow_NoReload(uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</code></td>
<td>Sets display window without reloading.</td>
<td></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>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetColorKeying_NoReload(uint32_t LayerIndex, uint32_t RGBValue)</code></td>
<td>Configures and sets the color keying without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_ResetColorKeying(uint32_t LayerIndex)</code></td>
<td>Disables the color Keying.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_ResetColorKeying_NoReload(uint32_t LayerIndex)</code></td>
<td>Disables the color keying without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_Relaod(uint32_t ReloadType)</code></td>
<td>Disables the color keying without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_GetTextColor(void)</code></td>
<td>Gets the LCD Text color.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_GetBackColor(void)</code></td>
<td>Gets the LCD Background color.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetTextColor(uint32_t Color)</code></td>
<td>Sets the Text color.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetBackColor(uint32_t Color)</code></td>
<td>Sets the Background color.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetFont(sFONT *pFonts)</code></td>
<td>Sets the Text Font.</td>
<td></td>
</tr>
<tr>
<td><code>sFONT * BSP_LCD_GetFont(void)</code></td>
<td>Gets the Text Font.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_ReadPixel(uint16_t Xpos, uint16_t Ypos)</code></td>
<td>Reads Pixel.</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><code>BSP_LCD_Clear</code></td>
<td>(uint32_t Color)</td>
<td>Clears the hole LCD.</td>
</tr>
<tr>
<td><code>BSP_LCD_ClearStringLine</code></td>
<td>(uint32_t Line)</td>
<td>Clears the selected line.</td>
</tr>
<tr>
<td><code>BSP_LCD_DisplayChar</code></td>
<td>(uint16_t Xpos, uint16_t Ypos, uint8_t Ascii)</td>
<td>Displays one character.</td>
</tr>
<tr>
<td><code>BSP_LCD_DisplayStringAt</code></td>
<td>(uint16_t X, uint16_t Y, uint8_t *pText, Text_AlignModeTypeDef mode)</td>
<td>Displays a maximum of 60 char on the LCD.</td>
</tr>
<tr>
<td><code>BSP_LCD_DisplayStringAtLine</code></td>
<td>(uint16_t Line, uint8_t *ptr)</td>
<td>Displays a maximum of 20 char on the LCD.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawHLine</code></td>
<td>(uint16_t Xpos, uint16_t Ypos, uint16_t Length)</td>
<td>Displays an horizontal line.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawVLine</code></td>
<td>(uint16_t Xpos, uint16_t Ypos, uint16_t Length)</td>
<td>Displays a vertical line.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawLine</code></td>
<td>(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></td>
<td>(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></td>
<td>(uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</td>
<td>Displays a circle.</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawPolygon</code></td>
<td>(pPoint Points, uint16_t PointCount)</td>
<td>Displays an poly-line (between many points).</td>
</tr>
<tr>
<td><code>BSP_LCD_DrawEllipse</code></td>
<td>(int Xpos, int Ypos, int XRadius, int YRadius)</td>
<td>Displays an Ellipse.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DrawBitmap (uint32_t X, uint32_t Y, uint8_t *pBmp)</code></td>
<td>Displays a bitmap picture loaded in the internal Flash (32 bpp).</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>Displays a full rectangle.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_FillCircle (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)</code></td>
<td>Displays a full circle.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_FillTriangle (uint16_t X1, uint16_t X2, uint16_t X3, uint16_t Y1, uint16_t Y2, uint16_t Y3)</code></td>
<td>Fill triangle.</td>
<td></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>
<td></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>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DisplayOn (void)</code></td>
<td>Enables the Display.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_DisplayOff (void)</code></td>
<td>Disables the Display.</td>
<td></td>
</tr>
<tr>
<td><code>__weak void BSP_LCD_MspInit (void)</code></td>
<td>Initializes the LTDC MSP.</td>
<td></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>
<td></td>
</tr>
<tr>
<td><code>static void DrawChar (uint16_t Xpos, uint16_t Ypos, const uint8_t *c)</code></td>
<td>Draws a character on LCD.</td>
<td></td>
</tr>
<tr>
<td><code>static void FillBuffer (uint32_t LayerIndex, void *pDst, uint32_t xSize, uint32_t ySize, uint32_t OffLine, uint32_t ColorIndex)</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Fills buffer.

| static void ConvertLineToARGB8888 (void *pSrc, void *pDst, uint32_t xSize, uint32_t ColorMode) |
| Converting Line to ARGB8888 pixel format. |

- Converts Line to ARGB8888 pixel format.
Function Documentation

void **BSP_LCD_Clear** ( uint32_t  **Color** )

Clears the hole LCD.

**Parameters:**

**Color:** the color of the background

Definition at line 569 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 579 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 597 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 1214 of file `stm32f429i_discovery_lcd.c`.

References `LcdDrv`.

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

Enables the Display.

Definition at line 1203 of file `stm32f429i_discovery_lcd.c`.

References `LcdDrv`.

```c
void BSP_LCD_DisplayStringAt ( uint16_t X, uint16_t Y, uint8_t * pText,
```
Text_AlignMode_TypeDef  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 614 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 668 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 903 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 805 of file *stm32f429i_discovery_lcd.c*.

References **ActiveLayer**, and **BSP_LCD_DrawPixel()**.
void **BSP_LCD_Dra**w**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 871 of file stm32f429i_discovery_lcd.c.

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

void **BSP_LCD_Dra**w**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 679 of file stm32f429i_discovery_lcd.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 714 of file stm32f429i_discovery_lcd.c.

References ABS, ActiveLayer, and BSP_LCD_DrawPixel().

Referenced by BSP_LCD_DrawPolygon(), and BSP_LCD_FillTriangle().

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 1313 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 844 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:
  \textbf{Xpos,}: \text{the X position}
  \textbf{Ypos,}: \text{the Y position}
  \textbf{Height,}: \text{display rectangle height}
  \textbf{Width,}: \text{display rectangle width}

Definition at line 788 of file \texttt{stm32f429i\_discovery\_lcd.c}.

References \texttt{BSP\_LCD\_DrawHLine()}, and \texttt{BSP\_LCD\_DrawVLine()}.  

\begin{verbatim}
void BSP_LCD_DrawVLine ( uint16_t Xpos,
                        uint16_t Ypos,
                        uint16_t Length )
\end{verbatim}

Displays a vertical line.

Parameters:
  \textbf{Xpos,}: \text{the X position}
  \textbf{Ypos,}: \text{the Y position}
  \textbf{Length,}: \text{line length}

Definition at line 696 of file \texttt{stm32f429i\_discovery\_lcd.c}.

References \texttt{ActiveLayer, BSP\_LCD\_GetXSize()}, \texttt{FillBuffer()}, and \texttt{LtdcHandler}.

Referenced by \texttt{BSP\_LCD\_DrawRect()}.  

\begin{verbatim}
void BSP_LCD_FillCircle ( uint16_t Xpos,
                         uint16_t Ypos,
                         uint16_t Radius )
\end{verbatim}
Displays a full circle.

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

Definition at line 983 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 1175 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 1107 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 963 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 1034 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 488 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 524 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 479 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 238 of file stm32f429i_discovery_lcd.c.

references lcdDrv.

referred by bsp_lcd_clear(), bsp_lcd_clearstringline(),
bsp_lcd_displaystringat(), bsp_lcd_drawbitmap(),
bsp_lcd_drawhline(), bsp_lcd_drawpixel(),
bsp_lcd_drawvline(), bsp_lcd_fillrect(),
bsp_lcd_layerdefaultinit(), and bsp_lcd_readpixel.
**uint32_t BSP_LCD_GetYSize ( void )**

Gets the LCD Y size.

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

Definition at line 247 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 155 of file `stm32f429i_discovery_lcd.c`.

References `BSP_LCD_MspInit()`, `BSP_LCD_SetFont()`, `BSP_SDRAM_Init()`, `LCD_DEFAULT_FONT`, `LCD_OK`, LcdDrv, LtdcHandler, 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 257 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.

__weak void BSP_LCD_MspInit ( void )

Initializes the LTDC MSP.

Definition at line 1229 of file stm32f429i_discovery_lcd.c.

Referenced by BSP_LCD_Init().

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 535 of file stm32f429i_discovery_lcd.c.
void BSP_LCD_Relaod ( uint32_t ReloadType )

Disables the color keying without reloading.

**Parameters:**

- **ReloadType:** can be one of the following values
  - LCD_RELOAD_IMMEDIATE
  - LCD_RELOAD_VERTICAL_BLANKING

**Return values:**

- **None**

Definition at line 470 of file stm32f429i_discovery_lcd.c.

References **LtdcHandler**.

---

void BSP_LCD_ResetColorKeying ( uint32_t LayerIndex )

Disables the color Keying.

**Parameters:**

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

Definition at line 446 of file stm32f429i_discovery_lcd.c.

References **LtdcHandler**.

---

void BSP_LCD_ResetColorKeying_NoReload ( uint32_t LayerIndex )

Disables the color keying without reloading.

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

Return values:
None

Definition at line 457 of file stm32f429i_discovery_lcd.c.
References LtdcHandler.

void BSP_LCD_SelectLayer ( uint32_t LayerIndex )

Selects the LCD Layer.

Parameters:
LayerIndex,: the Layer foreground or background.

Definition at line 292 of file stm32f429i_discovery_lcd.c.
References ActiveLayer.

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 506 of file stm32f429i_discovery_lcd.c.
References ActiveLayer, and LCD_DrawPropTypeDef::BackColor.

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 422 of file *stm32f429i_discovery_lcd.c*.

References **LtdcHandler**.

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

Configures and sets the color keying without reloading.

**Parameters:**
- **LayerIndex:** Layer foreground or background
- **RGBValue:** Color reference

**Return values:**
- **None**

Definition at line 435 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 515 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 366 of file *stm32f429i_discovery_lcd.c*.

References *LtdcHandler*.

```c
void BSP_LCD_SetLayerAddress_NoReload ( uint32_t LayerIndex, uint32_t Address )
```

Sets an LCD layer frame buffer address without reloading.

**Parameters:**
- `LayerIndex`: Layer foreground or background
- `Address`: New LCD frame buffer value

**Return values:**
- *None*

Definition at line 377 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 303 of file stm32f429i_discovery_lcd.c.

References **LtdcHandler**.

void BSP_LCD_SetLayerVisible_NoReload ( uint32_t LayerIndex, FunctionalState State )

Sets an LCD Layer visible without reloading.

**Parameters:**
- **LayerIndex,:** Visible Layer
- **State,:** New state of the specified layer This parameter can be one of the following values:
  - ENABLE
  - DISABLE

**Return values:**
- **None**

Definition at line 325 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 390 of file **stm32f429i_discovery_lcd.c**.

References **LtdcHandler**.

void **BSP_LCD_SetLayerWindow_NoReload**( uint16_t  LayerIndex,
uint16_t   Xpos,
uint16_t   Ypos,
uint16_t   Width,
uint16_t   Height
)

Sets display window without reloading.

**Parameters:**

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

Return values:
None

Definition at line 408 of file stm32f429i_discovery_lcd.c.
References LtdcHandler.

```c
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 497 of file stm32f429i_discovery_lcd.c.
References ActiveLayer, and LCD_DrawPropTypeDef::TextColor.

Referenced by BSP_LCD_ClearStringLine(), BSP_LCD_FillCircle(),
and BSP_LCD_FillRect().

```c
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 344 of file \texttt{stm32f429i_discovery_lcd.c}.

References \texttt{LtdcHandler}.

\begin{verbatim}
void BSP_LCD_SetTransparency_NoReload (uint32_t LayerIndex, uint8_t Transparency)
\end{verbatim}

Configures the transparency without reloading.

**Parameters:**
- \texttt{LayerIndex}: Layer foreground or background.
- \texttt{Transparency}: Transparency This parameter must be a number between Min_Data = 0x00 and Max_Data = 0xFF

**Return values:**
- \texttt{None}

Definition at line 356 of file \texttt{stm32f429i_discovery_lcd.c}.

References \texttt{LtdcHandler}.

\begin{verbatim}
static void ConvertLineToARGB8888 ( void * pSrc, void * pDst, uint32_t xSize, uint32_t ColorMode )
\end{verbatim}

Converts Line to ARGB8888 pixel format.

**Parameters:**
- \texttt{pSrc}: pointer to source buffer
pDst,: output color
xSize,: buffer width
ColorMode,: input color mode

Definition at line 1413 of file stm32f429i_discovery_lcd.c.

References Dma2dHandler.

Referenced by BSP_LCD_DrawBitmap().

static void DrawChar ( uint16_t Xpos,
    uint16_t Ypos,
    const uint8_t * c
) [static]

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 1325 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, BSP_LCD_DrawPixel(), and LCD_DrawPropTypeDef::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
) [static]

Fills buffer.

**Parameters:**

- **LayerIndex:** layer index
- **pDst:** output color
- **xSize:** buffer width
- **ySize:** buffer height
- **OffLine:** offset
- **ColorIndex:** color Index

Definition at line 1382 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
### 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><strong>BSP_LCD_Init (void)</strong></td>
<td>Initializes the LCD.</td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><strong>BSP_LCD_GetXSize (void)</strong></td>
<td>Gets the LCD X size.</td>
</tr>
<tr>
<td><code>uint32_t</code></td>
<td><strong>BSP_LCD_GetYSize (void)</strong></td>
<td>Gets the LCD Y size.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_LayerDefaultInit (void)</strong></td>
<td>(uint16_t LayerIndex, uint32_t FrameBuffer)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_SetTransparency (void)</strong></td>
<td>(uint32_t LayerIndex, uint8_t Transparency)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_SetTransparency_NoReload (void)</strong></td>
<td>(uint32_t LayerIndex, uint8_t Transparency)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_SetLayerAddress (void)</strong></td>
<td>(uint32_t LayerIndex, uint32_t Address)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_SetLayerAddress_NoReload (void)</strong></td>
<td>(uint32_t LayerIndex, uint32_t Address)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_SetColorKeying (void)</strong></td>
<td>(uint32_t LayerIndex, uint32_t RGBValue)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_SetColorKeying_NoReload (void)</strong></td>
<td>(uint32_t LayerIndex, uint32_t RGBValue)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_ResetColorKeying (void)</strong></td>
<td>(uint32_t LayerIndex)</td>
</tr>
<tr>
<td><code>void</code></td>
<td><strong>BSP_LCD_ResetColorKeying_NoReload (void)</strong></td>
<td>(uint32_t LayerIndex)</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></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>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerWindow_NoReload(uint16_t LayerIndex, uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)</code></td>
<td>Sets display window without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SelectLayer(uint32_t LayerIndex)</code></td>
<td>Selects the LCD Layer.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerVisible(uint32_t LayerIndex, FunctionalState state)</code></td>
<td>Sets a LCD Layer visible.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetLayerVisible_NoReload(uint32_t LayerIndex, FunctionalState State)</code></td>
<td>Sets an LCD Layer visible without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_Relaod(uint32_t ReloadType)</code></td>
<td>Disables the color keying without reloading.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetTextColor(uint32_t Color)</code></td>
<td>Sets the Text color.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetBackColor(uint32_t Color)</code></td>
<td>Sets the Background color.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_GetTextColor(void)</code></td>
<td>Gets the LCD Text color.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_GetBackColor(void)</code></td>
<td>Gets the LCD Background color.</td>
<td></td>
</tr>
<tr>
<td><code>void BSP_LCD_SetFont(sFONT *pFonts)</code></td>
<td>Sets the Text Font.</td>
<td></td>
</tr>
<tr>
<td><code>sFONT * BSP_LCD_GetFont(void)</code></td>
<td>Gets the Text Font.</td>
<td></td>
</tr>
<tr>
<td><code>uint32_t BSP_LCD_ReadPixel(uint16_t Xpos, uint16_t Ypos)</code></td>
<td>Reads Pixel.</td>
<td></td>
</tr>
<tr>
<td><code>BSP_LCD_DrawPixel(uint16_t Xpos, uint16_t Ypos, uint32_t Color)</code></td>
<td>Draws a Pixel.</td>
<td></td>
</tr>
</tbody>
</table>
void uint32_t pixel)
  Writes Pixel.

void BSP_LCD_Clear (uint32_t Color)
  Clears the hole LCD.

void BSP_LCD_ClearStringLine (uint32_t Line)
  Clears the selected line.

void BSP_LCD_DisplayStringAtLine (uint16_t Line, uint8_t *ptr)
  Displays a maximum of 20 char on the LCD.

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.

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

void BSP_LCD_DrawHLine (uint16_t Xpos, uint16_t Ypos, uint16_t Length)
  Displays an horizontal line.

void BSP_LCD_DrawVLine (uint16_t Xpos, uint16_t Ypos, uint16_t Length)
  Displays a vertical line.

void BSP_LCD_DrawLine (uint16_t X1, uint16_t Y1, uint16_t X2, uint16_t Y2)
  Displays an uni-line (between two points).

void BSP_LCD_DrawRect (uint16_t Xpos, uint16_t Ypos, uint16_t Width, uint16_t Height)
  Displays a rectangle.

void BSP_LCD_DrawCircle (uint16_t Xpos, uint16_t Ypos, uint16_t Radius)
  Displays a circle.

void BSP_LCD_DrawPolygon (pPoint Points, uint16_t PointCount)
  Displays an poly-line (between many points).
<table>
<thead>
<tr>
<th>void</th>
<th><strong>BSP_LCD_DrawEllipse</strong> (int Xpos, int Ypos, int XR radius, int YRadius)</th>
</tr>
</thead>
<tbody>
<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_DrawEllipse</strong> (int Xpos, int Ypos, int XR radius, 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>Disables the Display.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_DisplayOn</strong> (void)</td>
</tr>
<tr>
<td></td>
<td>Enables the Display.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_LCD_MspInit</strong> (void)</td>
</tr>
<tr>
<td></td>
<td>Initializes the LTDC MSP.</td>
</tr>
</tbody>
</table>
Function Documentation

```c
void BSP_LCD_Clear ( uint32_t Color )
```

Clears the hole LCD.

**Parameters:**

- **Color:** the color of the background

Definition at line 569 of file `stm32f429i_discovery_lcd.c`.

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

```c
void BSP_LCD_ClearStringLine ( uint32_t Line )
```

Clears the selected line.

**Parameters:**

- **Line:** the line to be cleared

Definition at line 579 of file `stm32f429i_discovery_lcd.c`.

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

```c
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 597 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 1214 of file `stm32f429i_discovery_lcd.c`.

References `LcdDrv`.

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

Enables the Display.

Definition at line 1203 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)  
- \( \text{pText} \): pointer to string to display on LCD  
- \( \text{mode} \): The display mode This parameter can be one of the following values:  
  - CENTER_MODE  
  - RIGHT_MODE  
  - LEFT_MODE

Definition at line 614 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:**

- \( \text{Line} \): the Line where to display the character shape  
- \( \text{ptr} \): pointer to string to display on LCD

Definition at line 668 of file `stm32f429i_discovery_lcd.c`.  

```c
Text_AlignModeTypdef mode
)
```
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 903 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 805 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 871 of file `stm32f429i_discovery_lcd.c`.

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

```c
void BSP_LCD_DrawHLine ( uint16_t Xpos,\n                         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 679 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 714 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 1313 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 844 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 788 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 696 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 983 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 XRadian,
                           int YRadius )
```

Draw a full ellipse.

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

Definition at line 1175 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 1107 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 963 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 1034 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 488 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 524 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 479 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 238 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 247 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 155 of file stm32f429i_discovery_lcd.c.

References BSP_LCD_MspInit(), BSP_LCD_SetFont(), BSP_SDRAM_Init(), LCD_DEFAULT_FONT, LCD_OK, LcdDrv, LtdcHandler, 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 257 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
void BSP_LCD_MspInit ( void )
```

Initializes the LTDC MSP.

Definition at line 1229 of file stm32f429i_discovery_lcd.c.

Referenced by BSP_LCD_Init().

```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 535 of file stm32f429i_discovery_lcd.c.
void BSP_LCD_Relaod ( uint32_t ReloadType )

Disables the color keying without reloading.

Parameters:

   ReloadType,: can be one of the following values
   • LCD_RELOAD_IMMEDIATE
   • LCD_RELOAD_VERTICAL_BLANKING

Return values:

   None

Definition at line 470 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.

void BSP_LCD_ResetColorKeying ( uint32_t LayerIndex )

Disables the color Keying.

Parameters:

   LayerIndex,: the Layer foreground or background

Definition at line 446 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.

void BSP_LCD_ResetColorKeying_NoReload ( uint32_t LayerIndex )

Disables the color keying without reloading.

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

**Return values:**

None

Definition at line 457 of file `stm32f429i_discovery_lcd.c`.

References **LtdcHandler**.

**void** **BSP_LCD_SelectLayer** ( **uint32_t** LayerIndex )

Selects the LCD Layer.

**Parameters:**

LayerIndex,: the Layer foreground or background.

Definition at line 292 of file `stm32f429i_discovery_lcd.c`.

References **ActiveLayer**.

**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 506 of file `stm32f429i_discovery_lcd.c`.

References **ActiveLayer**, and **LCD_DrawPropTypeDef::BackColor**.

**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 422 of file `stm32f429i_discovery_lcd.c`.

References **LtdcHandler**.

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

Configures and sets the color keying without reloading.

**Parameters:**
- **LayerIndex:** Layer foreground or background
- **RGBValue:** Color reference

**Return values:**
- **None**

Definition at line 435 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 515 of file `stm32f429i_discovery_lcd.c`.  

References **LtdcHandler**.
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 366 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.

void BSP_LCD_SetLayerAddress_NoReload ( uint32_t LayerIndex, uint32_t Address )

Sets an LCD layer frame buffer address without reloading.

Parameters:
    LayerIndex,: Layer foreground or background
    Address,: New LCD frame buffer value

Return values:
    None

Definition at line 377 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 303 of file stm32f429i_discovery_lcd.c.

References **LtdcHandler**.

---

void BSP_LCD_SetLayerVisible_NoReload ( uint32_t LayerIndex, FunctionalState State )

Sets an LCD Layer visible without reloading.

**Parameters:**
- **LayerIndex,:** Visible Layer
- **State,:** New state of the specified layer This parameter can be one of the following values:
  - ENABLE
  - DISABLE

**Return values:**
- None

Definition at line 325 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 390 of file **stm32f429i_discovery_lcd.c**.

References **LtdcHandler**.

void **BSP_LCD_SetLayerWindow_NoReload** ( uint16_t **LayerIndex**,  
    uint16_t **Xpos**,  
    uint16_t **Ypos**,  
    uint16_t **Width**,  
    uint16_t **Height**  
  )

Sets display window without reloading.

**Parameters:**

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

Return values:
None

Definition at line 408 of file stm32f429i_discovery_lcd.c.

References LtdcHandler.

```c
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 497 of file stm32f429i_discovery_lcd.c.

References ActiveLayer, and LCD_DrawPropTypeDef::TextColor.

Referenced by BSP_LCD_ClearStringLine(), BSP_LCD_FillCircle(), and BSP_LCD_FillRect().

```c
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.
void BSP_LCD_SetTransparency_NoReload (uint32_t LayerIndex, uint8_t Transparency)

Configures the transparency without reloading.

**Parameters:**
- **LayerIndex:** Layer foreground or background.
- **Transparency:** Transparency This parameter must be a number between Min_Data = 0x00 and Max_Data = 0xFF

**Return values:**
- None

Definition at line 356 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>Data Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><code>BSP_SDRAM_Init</code> (void)</td>
<td>Initializes the SDRAM device.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_SDRAM_Initialization_sequence</code> (uint32_t RefreshCount)</td>
<td>Programs the SDRAM device.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><code>BSP_SDRAM_ReadData</code> (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>uint8_t</td>
<td><code>BSP_SDRAM_ReadData_DMA</code> (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>uint8_t</td>
<td><code>BSP_SDRAM_WriteData</code> (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>uint8_t</td>
<td><code>BSP_SDRAM_WriteData_DMA</code> (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>uint8_t</td>
<td><code>BSP_SDRAM_Sendcmd</code> (FMC_SDRAM_CommandTypeDef *SdramCmd)</td>
<td>Sends command to the SDRAM bank.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_SDRAM_DMA_IRQHandler</code> (void)</td>
<td>Handles SDRAM DMA transfer interrupt request.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_SDRAM_MspInit</code> (SDRAM_HandleTypeDef *hsdram, void *Params)</td>
<td>Initializes SDRAM MSP.</td>
</tr>
<tr>
<td>void</td>
<td><code>BSP_SDRAM_MspDeInit</code> (SDRAM_HandleTypeDef *hsdram, void *Params)</td>
<td>DeInitializes SDRAM MSP.</td>
</tr>
</tbody>
</table>
Function Documentation

void BSP_SDRAM_DMA_IRQHandler ( void )

Handles SDRAM DMA transfer interrupt request.

Definition at line 311 of file stm32f429i_discovery_sdram.c.

References SdramHandle.

uint8_t BSP_SDRAM_Init ( void )

Initializes the SDRAM device.

Definition at line 99 of file stm32f429i_discovery_sdram.c.

References BSP_SDRAM_Initialization_sequence(),
BSP_SDRAM_MspInit(), REFRESH_COUNT, SDCLOCK_PERIOD,
SDRAM_CAS_LATENCY, SDRAM_ERROR,
SDRAM_MEMORY_WIDTH, SDRAM_OK, 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 160 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_MspDeInit ( SDRAM_HandleTypeDef * hsdram,
                           void * Params)
```

DeInitializes SDRAM MSP.

**Note:**
This function can be surcharged by application code.

**Parameters:**
- **hsdram:** pointer on SDRAM handle
- **Params:** pointer on additional configuration parameters, can be NULL.

Definition at line 443 of file stm32f429i_discovery_sdram.c.

References **SDRAM_DMAx_IRQn**, and **SDRAM_DMAx_STREAM**.

```c
void BSP_SDRAM_MspInit ( SDRAM_HandleTypeDef * hsdram,
                         void * Params)
```

Initializes SDRAM MSP.

**Note:**
This function can be surcharged by application code.

**Parameters:**
**hsdram:** pointer on SDRAM handle

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

Definition at line `322` of file `stm32f429i_discovery_sdram.c`.

References `__DMAx_CLK_ENABLE`, `SDRAM_DMAx_CHANNEL`, `SDRAM_DMAx_IRQn`, and `SDRAM_DMAx_STREAM`.

Referenced by `BSP_SDRAM_Init()`.

```c
uint8_t 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 `221` of file `stm32f429i_discovery_sdram.c`.

References `SDRAM_ERROR`, `SDRAM_OK`, and `SdramHandle`.

```c
uint8_t 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 239 of file *stm32f429i_discovery_sdram.c*.

References **SDRAM_ERROR**, **SDRAM_OK**, and **SdramHandle**.

```c
uint8_t 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 296 of file *stm32f429i_discovery_sdram.c*.

References **SDRAM_ERROR**, **SDRAM_OK**, **SDRAM_TIMEOUT**, and **SdramHandle**.

```c
uint8_t 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 257 of file `stm32f429i_discovery_sdram.c`.

References `SDRAM_ERROR`, `SDRAM_OK`, and `SdramHandle`.

```c
uint8_t 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 279 of file `stm32f429i_discovery_sdram.c`.

References `SDRAM_ERROR`, `SDRAM_OK`, and `SdramHandle`.

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery 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>
<th>Functions</th>
</tr>
</thead>
</table>

### STM32F429I DISCOVERY SDRAM Private Functions

STM32F429I DISCOVERY SDRAM
### 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_SDRAM_Init</code> (void)</td>
<td>Initializes the SDRAM device.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_SDRAM_Initialization_sequence</code> (uint32_t RefreshCount)</td>
<td>Programs the SDRAM device.</td>
</tr>
<tr>
<td><code>uint8_t</code></td>
<td><code>BSP_SDRAM_ReadData</code> (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><code>uint8_t</code></td>
<td><code>BSP_SDRAM_ReadData_DMA</code> (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><code>uint8_t</code></td>
<td><code>BSP_SDRAM_WriteData</code> (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><code>uint8_t</code></td>
<td><code>BSP_SDRAM_WriteData_DMA</code> (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><code>uint8_t</code></td>
<td><code>BSP_SDRAM_Sendcmd</code> (FMC_SDRAM_CommandTypeDef *SdramCmd)</td>
<td>Sends command to the SDRAM bank.</td>
</tr>
<tr>
<td><code>void</code></td>
<td><code>BSP_SDRAM_DMA_IRQHandler</code> (void)</td>
<td>Handles SDRAM DMA transfer interrupt request.</td>
</tr>
<tr>
<td>__weak</td>
<td><code>BSP_SDRAM_MspInit</code> (SDRAM_HandleTypeDef *hsdram, void *Params)</td>
<td>Initializes SDRAM MSP.</td>
</tr>
<tr>
<td>__weak</td>
<td><code>BSP_SDRAM_MspDeInit</code> (SDRAM_HandleTypeDef *hsdram, void *Params)</td>
<td>DeInitializes SDRAM MSP.</td>
</tr>
</tbody>
</table>
Function Documentation

```c
void BSP_SDRAM_DMA_IRQHandler ( void )
```

Handles SDRAM DMA transfer interrupt request.

Definition at line 311 of file stm32f429i_discovery_sdram.c.

References SdramHandle.

```c
uint8_t BSP_SDRAM_Init ( void )
```

Initializes the SDRAM device.

Definition at line 99 of file stm32f429i_discovery_sdram.c.

References BSP_SDRAM_Initialization_sequence(), BSP_SDRAM_MspInit(), REFRESH_COUNT, SDCLOCK_PERIOD, SDRAM_CAS_LATENCY, SDRAM_ERROR, SDRAM_MEMORY_WIDTH, SDRAM_OK, 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 160 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
__weak void BSP_SDRAM_MspDeInit ( SDRAM_HandleTypeDef * hsdram, void * Params)
```

DeInitializes SDRAM MSP.

**Note:**
This function can be surcharged by application code.

**Parameters:**
- **hsdram:** pointer on SDRAM handle
- **Params:** pointer on additional configuration parameters, can be NULL.

Definition at line 443 of file stm32f429i_discovery_sdram.c.

References SDRAM_DMAx_IRQHandler, and SDRAM_DMAx_STREAM.

```c
__weak void BSP_SDRAM_MspInit ( SDRAM_HandleTypeDef * hsdram, void * Params)
```

Initializes SDRAM MSP.

**Note:**
This function can be surcharged by application code.

**Parameters:**
**hsdram:** pointer on SDRAM handle

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

Definition at line 322 of file `stm32f429i_discovery_sdram.c`.

References __DMAx_CLK_ENABLE, SDRAM_DMAx_CHANNEL, SDRAM_DMAx_IRQn, and SDRAM_DMAx_STREAM.

Referenced by `BSP_SDRAM_Init()`.

```c
uint8_t 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 221 of file `stm32f429i_discovery_sdram.c`.

References SDRAM_ERROR, SDRAM_OK, and SdramHandle.

```c
uint8_t 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 239 of file `stm32f429i_discovery_sdram.c`.

References **SDRAM_ERROR**, **SDRAM_OK**, and **SdramHandle**.

```c
uint8_t 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 296 of file `stm32f429i_discovery_sdram.c`.

References **SDRAM_ERROR**, **SDRAM_OK**, **SDRAM_TIMEOUT**, and **SdramHandle**.

```c
uint8_t 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 257 of file `stm32f429i_discovery_sdram.c`.

References `SDRAM_ERROR`, `SDRAM_OK`, and `SdramHandle`.

```c
uint8_t 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 279 of file `stm32f429i_discovery_sdram.c`.

References `SDRAM_ERROR`, `SDRAM_OK`, and `SdramHandle`.

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery
BSP User Manual by `doxygen` 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY TS Exported Functions

STM32F429I DISCOVERY TS
## Functions

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Function Name</th>
<th>Parameters/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td><strong>BSP_TS_Init</strong></td>
<td>(uint16_t XSize, uint16_t YSize) 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></td>
<td>(TS_StateTypeDef *TsState) Returns status and positions of the touch screen.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><strong>BSP_TS_ITConfig</strong></td>
<td>(void) Configures and enables the touch screen interrupts.</td>
</tr>
<tr>
<td>uint8_t</td>
<td><strong>BSP_TS_ITGetStatus</strong></td>
<td>(void) Gets the TS IT status.</td>
</tr>
<tr>
<td>void</td>
<td><strong>BSP_TS_ITClear</strong></td>
<td>(void) 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>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### STM32F429I DISCOVERY TS Private Functions

STM32F429I DISCOVERY TS
Functions

<table>
<thead>
<tr>
<th>uint8_t</th>
<th><strong>BSP_TS_Init (uint16_t XSize, uint16_t YSize)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Initializes and configures the touch screen functionalities and configures all necessary hardware resources (GPIOs, clocks..).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>uint8_t</th>
<th><strong>BSP_TS_ITConfig (void)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Configures and enables the touch screen interrupts.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>uint8_t</th>
<th><strong>BSP_TS_ITGetStatus (void)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gets the TS IT status.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th>*<em>BSP_TS_GetState (TS_StateTypeDef <em>TsState)</em></em></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns status and positions of the touch screen.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>void</th>
<th><strong>BSP_TS_ITClear (void)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></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.
<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 Exported Constants**

**STM32F429I DISCOVERY LCD**
Defines

```c
#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  0xFF808080
#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 Statement</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#define LCD_COLOR_TRANSPARENT</code></td>
<td>0xFF000000</td>
</tr>
<tr>
<td><code>#define LCD_DEFAULT_FONT</code></td>
<td>Font24</td>
</tr>
<tr>
<td>LCD default font.</td>
<td></td>
</tr>
<tr>
<td><code>#define LCD_RELOAD_IMMEDIATE</code></td>
<td>((uint32_t)LTDC_SRCR_IMR)</td>
</tr>
<tr>
<td>LCD Reload Types.</td>
<td></td>
</tr>
<tr>
<td><code>#define LCD_RELOAD_VERTICAL_BLANKING</code></td>
<td>((uint32_t)LTDC_SSRCR_VBR)</td>
</tr>
<tr>
<td><code>#define LCD_BACKGROUND_LAYER</code></td>
<td>0x0000</td>
</tr>
<tr>
<td>LCD Layer.</td>
<td></td>
</tr>
<tr>
<td><code>#define LCD_FOREGROUND_LAYER</code></td>
<td>0x0001</td>
</tr>
</tbody>
</table>
Define Documentation

#define BUFFER_OFFSET ((uint32_t)0x50000)

Definition at line 113 of file stm32f429i_discovery_lcd.h.

#define LCD_BACKGROUND_LAYER 0x0000

LCD Layer.
Definition at line 158 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_BLACK 0xFF000000

Definition at line 140 of file stm32f429i_discovery_lcd.h.
Referenced by BSP_LCD_LayerDefaultInit().

#define LCD_COLOR_BLUE 0xFF0000FF

LCD color.
Definition at line 118 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_BROWN 0xFFA52A2A

Definition at line 141 of file stm32f429i_discovery_lcd.h.

#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>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_COLOR_GRAY</td>
<td>0xFF808080</td>
</tr>
<tr>
<td>Definition at line</td>
<td>138</td>
</tr>
<tr>
<td>of file</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>LCD_COLOR_GREEN</td>
<td>0xFF00FF00</td>
</tr>
<tr>
<td>Definition at line</td>
<td>119</td>
</tr>
<tr>
<td>of file</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>LCD_COLOR_LIGHTBLUE</td>
<td>0xFF8080FF</td>
</tr>
<tr>
<td>Definition at line</td>
<td>124</td>
</tr>
<tr>
<td>of file</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>LCD_COLOR_LIGHTCYAN</td>
<td>0xFF80FFFF</td>
</tr>
<tr>
<td>Definition at line</td>
<td>127</td>
</tr>
<tr>
<td>of file</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>LCD_COLOR_LIGHTGRAY</td>
<td>0xFFD3D3D3</td>
</tr>
<tr>
<td>Definition at line</td>
<td>137</td>
</tr>
<tr>
<td>of file</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>LCD_COLOR_LIGHTGREEN</td>
<td>0xFF80FF80</td>
</tr>
<tr>
<td>Definition at line</td>
<td>125</td>
</tr>
<tr>
<td>of file</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>LCD_COLOR_LIGHTMAGENTA</td>
<td>0xFFF80FF0FF</td>
</tr>
<tr>
<td>Definition at line</td>
<td>128</td>
</tr>
<tr>
<td>of file</td>
<td>stm32f429i_discovery_lcd.h</td>
</tr>
<tr>
<td>LCD_COLOR_LIGHTRED</td>
<td>0xFFFF8080</td>
</tr>
</tbody>
</table>
#define LCD_COLOR_LIGHTYELLOW 0xFFFF00FF
Definition at line 126 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_MAGENTA 0xFFFF00FF
Definition at line 129 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_ORANGE 0xFFFFA500
Definition at line 122 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_RED 0xFFFF0000
Definition at line 142 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_TRANSPARENT 0xFF000000
Definition at line 143 of file stm32f429i_discovery_lcd.h.

#define LCD_COLOR_WHITE 0xFFFFFFFF
Definition at line 136 of file stm32f429i_discovery_lcd.h.

Referenced by BSP_LCD_LayerDefaultInit().

#define LCD_COLOR_YELLOW 0xFFFFFF00
#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 159 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 LCD_RELOAD_IMMEDIATE ((uint32_t)LTDC_SRCR_IMR)

LCD Reload Types.

Definition at line 152 of file stm32f429i_discovery_lcd.h.
#define LCD_RELOAD_VERTICAL_BLANKING ((uint32_t)LTDC_SRCR_VBR)

Definition at line 153 of file stm32f429i_discovery_lcd.h.

#define MAX_LAYER_NUMBER 2

LCD status structure definition.

Definition at line 111 of file stm32f429i_discovery_lcd.h.
<table>
<thead>
<tr>
<th>directories</th>
<th>variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY LOW LEVEL Private Variables</td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY LOW LEVEL
### Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</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_IRQHandler [BUTTONn] = {KEY_BUTTON EXTI_IRQHandler}</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

```c
const uint8_t BUTTON_IRQn[BUTTONn] = {KEY_BUTTON_EXTI_IRQn}
```

Definition at line 100 of file stm32f429i_discovery.c.

Referenced by `BSP_PB_Init()`.

```c
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()`.

```c
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()`.

```c
const uint16_t GPIO_PIN[LEDn]
```

Initial value:

```c
{LED3_PIN, 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()`.

```c
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_IOInitialized = 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`. 
Referenced by SPIx_Error(), SPIx_Init(), SPIx_Read(), SPIx_Write(), and SPIx_WriteRead.

```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

<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 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><code>LED3 = 0, LED4 = 1</code></td>
</tr>
<tr>
<td>Button_TypeDef</td>
<td><code>BUTTON_KEY = 0</code></td>
</tr>
<tr>
<td>ButtonMode_TypeDef</td>
<td><code>BUTTON_MODE_GPIO = 0, BUTTON_MODE_EXTI = 1</code></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`.

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery 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><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY LOW LEVEL BUTTON**

STM32F429I DISCOVERY LOW LEVEL Exported Constants
Defines

| #define | BUTTONn | 1 |
| #define | KEY_BUTTON_PIN | GPIO_PIN_0 |
|         |         | Wakeup push-button. |
| #define | KEY_BUTTON_GPIO_PORT | GPIOA |
| #define | KEY_BUTTON_GPIO_CLK_ENABLE() | __HAL_RCC_GPIOA_CLK_ENABLE() |
| #define | KEY_BUTTON_GPIO_CLK_DISABLE() | __HAL_RCC_GPIOA_CLK_DISABLE() |
| #define | KEY_BUTTON_EXTI_IRQn | EXTI0_IRQHandler |
| #define | BUTTONx_GPIO_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:

```c
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:

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

Definition at line 138 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 ( ) __HAL_RCC_GPIOA_CLK_DISABLE()

Definition at line 133 of file stm32f429i_discovery.h.

#define KEY_BUTTON_GPIO_CLK_ENABLE ( ) __HAL_RCC_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 Name</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

**FMC_SDRAM_CommandTypeDef** Command [static]

Definition at line 80 of file *stm32f429i_discovery_sdram.c*.

Referenced by **BSP_SDRAM_Initialization_sequence()**.

**SDRAM_HandleTypeDef** SdramHandle [static]

Definition at line 78 of file *stm32f429i_discovery_sdram.c*.

Referenced by **BSP_SDRAM_DMA_IRQHandler()**, **BSP_SDRAM_Init()**, **BSP_SDRAM_Initialization_sequence()**, **BSP_SDRAM_ReadData()**, **BSP_SDRAM_ReadData_DMA()**, **BSP_SDRAM_Sendcmd()**, **BSP_SDRAM_WriteData()**, and **BSP_SDRAM_WriteData_DMA()**.

**FMC_SDRAM_TimingTypeDef** Timing [static]

Definition at line 79 of file *stm32f429i_discovery_sdram.c*.

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>
<tr>
<td><strong>STM32F429I</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISCOVERY LOW LEVEL BUS</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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() __HAL_RCC_I2C3_CLK_ENABLE()
#define DISCOVERY_I2Cx_FORCE_RESET() __HAL_RCC_I2C3_FORCE_RESET()
#define DISCOVERY_I2Cx_RELEASE_RESET() __HAL_RCC_I2C3_RELEASE_RESET()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_ENABLE()
#define DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()
#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
#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() __HAL_RCC_SPI5_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_PORT GPIOF /* GPIOF */
#define DISCOVERY_SPIx_AF GPIO_AF5_SPI5
#define DISCOVERY_SPIx_GPIO_CLK_ENABLE() __HAL_RCC_GPIOF_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_CLK_DISABLE() __HAL_RCC_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
```

IOE	Control	pin.
#define STMPE811_INT_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()

#define STMPE811_INT_CLK_DISABLE() __HAL_RCC_GPIOA_CLK_DISABLE()

#define STMPE811_INT_EXTI EXTI15_10_IRQHandler

#define STMPE811_INT_EXTIHandler EXTI15_10_IRQHandler

#define LCD_CS_LOW() HAL_GPIO_WritePin(LCD_NCS_GPIO_PORT, GPIO_PIN_RESET)

#define LCD_CS_HIGH() HAL_GPIO_WritePin(LCD_NCS_GPIO_PORT, GPIO_PIN_SET)

#define LCD_WRX_LOW() HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, GPIO_PIN_RESET)

#define LCD_WRX_HIGH() HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, GPIO_PIN_SET)

#define LCD_RDX_LOW() HAL_GPIO_WritePin(LCD_RDX_GPIO_PORT, GPIO_PIN_RESET)

#define LCD_RDX_HIGH() HAL_GPIO_WritePin(LCD_RDX_GPIO_PORT, GPIO_PIN_SET)

#define LCD_NCS_PIN GPIO_PIN_2

#define LCD_NCS_GPIO_PORT GPIOC

#define LCD_NCS_GPIO_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_ENABLE()

#define LCD_NCS_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()
Define Documentation

```c
#define DISCOVERY_I2Cx I2C3
```
Definition at line **159** of file `stm32f429i_discovery.h`.
Referenced by `I2Cx_Init()`, and `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_CLOCK_ENABLE () __HAL_RCC_I2C3_CLK_ENABLE()
```
Definition at line **160** of file `stm32f429i_discovery.h`.
Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_ER_IRQn I2C3_ER_IRQn
```
Definition at line **176** of file `stm32f429i_discovery.h`.
Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_EV_IRQn I2C3_EV_IRQn
```
Definition at line **175** of file `stm32f429i_discovery.h`.
Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_FORCE_RESET () __HAL_RCC_I2C3_FORCE_RESET()
```
Definition at line **161** of file `stm32f429i_discovery.h`.
Referenced by `I2Cx_MspInit()`.
#define DISCOVERY_I2Cx_RELEASE_RESET() __HAL_RCC_I2C3_RELEASE_RESET()

Definition at line 162 of file stm32f429i_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()

Definition at line 164 of file stm32f429i_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_SCL_GPIO_PORT GPIOA

Definition at line 169 of file stm32f429i_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_SCL_PIN GPIO_PIN_8

Definition at line 168 of file stm32f429i_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_SCL_SDA_AF GPIO_AF4_I2C3

Definition at line 170 of file stm32f429i_discovery.h.

Referenced by I2Cx_MspInit().

#define DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()
Definition at line 165 of file stm32f429i_discovery.h.

```c
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE(__HAL_RCC_GPIOC_CLK_ENABLE)
```

Definition at line 163 of file stm32f429i_discovery.h.

Referenced by `I2Cx_MspInit()`.

Definition at line 172 of file stm32f429i_discovery.h.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_I2Cx_SDA_GPIO_PORT GPIOC
```

Definition at line 171 of file stm32f429i_discovery.h.

Referenced by `I2Cx_MspInit()`.

```c
#define DISCOVERY_SPIx_SPI 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 ( ) __HAL_RCC_SPI5_CLK_ENABLE()

Definition at line 190 of file stm32f429i_discovery.h.

Referenced by SPIx_MspInit().

# define DISCOVERY_SPIx_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOF_CLK_DISABLE()

Definition at line 194 of file stm32f429i_discovery.h.

# define DISCOVERY_SPIx_GPIO_CLK_ENABLE ( ) __HAL_RCC_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 maximal timeout for I2C waiting loops*/

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_PORT, LCD_CS_PIN, GPIO_PIN_SET)

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_PORT, LCD_CS_PIN, GPIO_PIN_RESET)

Definition at line 220 of file stm32f429i_discovery.h.
Referenced by LCD_IO_Init(), LCD_IO_ReadData(),
LCD_IO_WriteData(), and LCD_IO_WriteReg().

#define LCD_NCS_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()  
Definition at line 237 of file stm32f429i_discovery.h.

#define LCD_NCS_GPIO_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_ENABLE()  
Definition at line 236 of file stm32f429i_discovery.h.

Referenced by LCD_IO_Init().

#define LCD_NCS_GPIO_PORT GPIOC  
Definition at line 235 of file stm32f429i_discovery.h.

Referenced by LCD_IO_Init().

#define LCD_NCS_PIN GPIO_PIN_2  
LCD Control pin.  
Definition at line 234 of file stm32f429i_discovery.h.

Referenced by LCD_IO_Init().

#define LCD_RDX_HIGH() HAL_GPIO_WritePin(LCD_RDX_GPIO   
Definition at line 229 of file stm32f429i_discovery.h.

#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, GPIO_PIN_SET)
```
Definition at line 228 of file stm32f429i_discovery.h.

Referenced by LCD_IO_ReadData(), and LCD_IO_WriteData().

```c
#define LCD_WRX_LOW() HAL_GPIO_WritePin(LCD_WRX_GPIO_PORT, LCD_WRX_PIN, GPIO_PIN_RESET)
```
Definition at line 225 of file stm32f429i_discovery.h.

Referenced by LCD_IO_ReadData(), and LCD_IO_WriteReg().

```c
#define SPIx_TIMEOUT_MAX ((uint32_t)0x1000)
```
Definition at line 203 of file stm32f429i_discovery.h.

```c
#define STMPE811_INT_CLK_DISABLE() __HAL_RCC_GPIOA_CLK_DISABLE()
```
Definition at line 214 of file stm32f429i_discovery.h.

Referenced by I2Cx_ITConfig().

```c
#define STMPE811_INT_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
```
Definition at line 213 of file stm32f429i_discovery.h.

```c
#define STMPE811_INT_EXTI EXTI15_10_IRQn
```
Definition at line 215 of file stm32f429i_discovery.h.
Referenced by \texttt{I2Cx\_ITConfig()}.  

\begin{verbatim}
#define STMPE811\_INT\_EXTIHandler EXTI15\_10\_IRQHandler
\end{verbatim}

Definition at line 216 of file \texttt{stm32f429i\_discovery.h}.  

Referenced by \texttt{I2Cx\_ITConfig()}.  

\begin{verbatim}
#define STMPE811\_INT\_GPIO\_PORT GPIOA
\end{verbatim}

Definition at line 212 of file \texttt{stm32f429i\_discovery.h}.  

Referenced by \texttt{I2Cx\_ITConfig()}.  

\begin{verbatim}
#define STMPE811\_INT\_PIN GPIO\_PIN\_15
\end{verbatim}

IOE Control pin.  

Definition at line 211 of file \texttt{stm32f429i\_discovery.h}.  

Referenced by \texttt{I2Cx\_ITConfig()}.  

\begin{verbatim}
#define TS\_I2C\_ADDRESS 0x82
\end{verbatim}

Definition at line 149 of file \texttt{stm32f429i\_discovery.h}.  

Referenced by \texttt{BSP\_TS\_GetState()}, \texttt{BSP\_TS\_Init()}, \texttt{BSP\_TS\_ITClear()}, \texttt{BSP\_TS\_ITConfig()}, and \texttt{BSP\_TS\_ITGetStatus()}.  

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery
BSP User Manual by \texttt{doxygen} 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LOW LEVEL Exported Constants

STM32F429I DISCOVERY LOW LEVEL
## Modules

<table>
<thead>
<tr>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>STM32F429I DISCOVERY LOW LEVEL LED</strong></td>
</tr>
<tr>
<td>Define for STM32F429I_DISCO board.</td>
</tr>
<tr>
<td><strong>STM32F429I DISCOVERY LOW LEVEL BUTTON</strong></td>
</tr>
<tr>
<td><strong>STM32F429I DISCOVERY LOW LEVEL BUS</strong></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() __HAL_RCC_GPIOD_CLK_ENABLE()
#define LCD_WRX_GPIO_CLK_DISABLE() __HAL_RCC_GPIOD_CLK_DISABLE()
#define LCD_RDX_PIN GPIO_PIN_12
#define LCD_RDX_GPIO_PORT GPIOD
#define LCD_RDX_GPIO_CLK_ENABLE() __HAL_RCC_GPIOD_CLK_ENABLE()
#define LCD_RDX_GPIO_CLK_DISABLE() __HAL_RCC_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() __HAL_RCC_GPIOC_CLK_ENABLE()
#define GYRO_CS_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()
#define GYRO_INT_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
#define GYRO_INT_GPIO_CLK_DISABLE() __HAL_RCC_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_IRQn
#define GYRO_INT2_PIN GPIO_PIN_2 /* PA.02 */
#define GYRO_INT2_EXTI_IRQn 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 () __HAL_RCC_GPIOC_CLK_DISABLE()
```
Definition at line 272 of file `stm32f429i_discovery.h`.

```c
#define GYRO_CS_GPIO_CLK_ENABLE () __HAL_RCC_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, 0, 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, 0)

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() __HAL_RCC_GPIOA_CLK_DISABLE()
```

Definition at line 275 of file `stm32f429i_discovery.h`.

Referenced by `GYRO_IO_Init()`.

```c
#define GYRO_INT_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
```

Definition at line 274 of file `stm32f429i_discovery.h`.

```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() __HAL_RCC_GPIOD_CLK_DISABLE()
```

Definition at line 252 of file `stm32f429i_discovery.h`.

Referenced by `LCD_IO_Init()`.

```c
#define LCD_RDX_GPIO_CLK_ENABLE() __HAL_RCC_GPIOD_CLK_ENABLE()
```

Definition at line 251 of file `stm32f429i_discovery.h`.

```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 () __HAL_RCC_GPIOD_CLK_DISABLE()
```
Definition at line 247 of file `stm32f429i_discovery.h`. Referenced by `LCD_IO_Init()`.

```c
#define LCD_WRX_GPIO_CLK_ENABLE () __HAL_RCC_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().

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery 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><strong>Directories</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY EEPROM Exported Constants**

STM32F429I DISCOVERY EEPROM
# Defines

<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

<table>
<thead>
<tr>
<th>Definition</th>
<th>Value</th>
<th>File</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>#define EEPROM_FAIL</td>
<td>1</td>
<td>stm32f429i_discovery_eeprom.h</td>
<td>87</td>
</tr>
<tr>
<td>#define EEPROM_MAX_SIZE</td>
<td>0x2000 /* 64Kbit*/</td>
<td>stm32f429i_discovery_eeprom.h</td>
<td>74</td>
</tr>
<tr>
<td>#define EEPROM_MAX_TRIALS</td>
<td>300</td>
<td>stm32f429i_discovery_eeprom.h</td>
<td>84</td>
</tr>
<tr>
<td>#define EEPROM_OK</td>
<td>0</td>
<td>stm32f429i_discovery_eeprom.h</td>
<td>86</td>
</tr>
<tr>
<td>#define EEPROM_PAGESIZE</td>
<td>4</td>
<td>stm32f429i_discovery_eeprom.h</td>
<td>73</td>
</tr>
<tr>
<td>#define EEPROM_READ_TIMEOUT</td>
<td>((uint32_t)(1000))</td>
<td>stm32f429i_discovery_eeprom.h</td>
<td>79</td>
</tr>
<tr>
<td>#define EEPROM_TIMEOUT</td>
<td>2</td>
<td>stm32f429i_discovery_eeprom.h</td>
<td>88</td>
</tr>
</tbody>
</table>
#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>Enumerations</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

```c
enum GYRO_StatusTypeDef
{
    GYRO_OK,
    GYRO_ERROR,
    GYRO_TIMEOUT
};
```

Definition at line 67 of file `stm32f429i_discovery_gyroscope.h`.
<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 FunctionPrototypes**

STM32F429I DISCOVERY LOW LEVEL
### Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static void <code>I2Cx_Init (void)</code></td>
<td>I2Cx Bus initialization.</td>
</tr>
<tr>
<td>static void <code>I2Cx_ITConfig (void)</code></td>
<td>Configures Interruption pin for I2C communication.</td>
</tr>
<tr>
<td>static void <code>I2Cx_Error (void)</code></td>
<td>I2Cx error treatment function.</td>
</tr>
<tr>
<td>static void <code>SPIx_Init (void)</code></td>
<td>SPIx Bus initialization.</td>
</tr>
<tr>
<td>static void <code>SPIx_Error (void)</code></td>
<td>SPIx error treatment function.</td>
</tr>
<tr>
<td>void <code>LCD_IO_Init (void)</code></td>
<td>Configures the LCD_SPI interface.</td>
</tr>
<tr>
<td>void <code>LCD_IO_WriteData (uint16_t RegValue)</code></td>
<td>Writes register value.</td>
</tr>
<tr>
<td>void <code>LCD_IO_WriteReg (uint8_t Reg)</code></td>
<td>Writes register address.</td>
</tr>
<tr>
<td>void <code>IOE_Init (void)</code></td>
<td>IOE Low Level Initialization.</td>
</tr>
<tr>
<td>void <code>IOE_ITConfig (void)</code></td>
<td>IOE Low Level Interrupt configuration.</td>
</tr>
<tr>
<td>void <code>GYRO_IO_Init (void)</code></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().
# 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>Variables</td>
</tr>
</tbody>
</table>

**STM32F429I DISCOVERY GYROSCOPE Private Variables**

**STM32F429I DISCOVERY GYROSCOPE**
Variables

| static GYRO_DrvTypeDef * GyroscopeDrv |
Variable Documentation

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>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 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`. 

---

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by [doxygen](https://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 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.
#define IO_PIN_7 0x80

Definition at line 87 of file stm32f429i_discovery_io.h.

#define IO_PIN_ALL 0xFF

Definition at line 88 of file stm32f429i_discovery_io.h.

Referenced by BSP_IO_Init(), and BSP_IO_ITClear().

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery
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>Directories</td>
<td>Variables</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()**.

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by **doxygen** 1.7.6.1
STM32F429I-Discovery BSP User Manual

STM32F429I DISCOVERY LCD Exported Macros

STM32F429I DISCOVERY LCD
#Defines

<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_ARGB8888</code></td>
<td>LTDC_PIXEL_FORMAT_ARGB8888</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_RGB888</code></td>
<td>LTDC_PIXEL_FORMAT_RGB888</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_RGB565</code></td>
<td>LTDC_PIXEL_FORMAT_RGB565</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_ARGB1555</code></td>
<td>LTDC_PIXEL_FORMAT_ARGB1555</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_ARGB4444</code></td>
<td>LTDC_PIXEL_FORMAT_ARGB4444</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_L8</code></td>
<td>LTDC_PIXEL_FORMAT_L8</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_AL44</code></td>
<td>LTDC_PIXEL_FORMAT_AL44</td>
</tr>
<tr>
<td><code>#define LCD_PIXEL_FORMAT_AL88</code></td>
<td>LTDC_PIXEL_FORMAT_AL88</td>
</tr>
</tbody>
</table>
Define Documentation

```c
#define LCD_PIXEL_FORMAT_AL44 LONGITUDE_FORMAT_AL44
Definition at line 177 of file stm32f429i_discovery_lcd.h.
```

```c
#define LCD_PIXEL_FORMAT_AL88 LTDC_PIXEL_FORMAT_AL88
Definition at line 178 of file stm32f429i_discovery_lcd.h.
```

```c
#define LCD_PIXEL_FORMAT_ARGB1555 LTDC_PIXEL_FORMAT_ARGB1555
Definition at line 174 of file stm32f429i_discovery_lcd.h.
```

```c
#define LCD_PIXEL_FORMAT_ARGB4444 LTDC_PIXEL_FORMAT_ARGB4444
Definition at line 175 of file stm32f429i_discovery_lcd.h.
```

```c
#define LCD_PIXEL_FORMAT_ARGB8888 LTDC_PIXEL_FORMAT_ARGB8888
Definition at line 171 of file stm32f429i_discovery_lcd.h.
```

```c
#define LCD_PIXEL_FORMAT_L8 LTDC_PIXEL_FORMAT_L8
Definition at line 176 of file stm32f429i_discovery_lcd.h.
```

```c
#define LCD_PIXEL_FORMAT_RGB565 LTDC_PIXEL_FORMAT_RGB565
```

LCD Pixel format.
Definition at line 170 of file stm32f429i_discovery_lcd.h.
Definition at line 173 of file `stm32f429i_discovery_lcd.h`.

```
#define LCD_PIXEL_FORMAT_RGB888  LTDC_PIXEL_FORMAT_RGB888
```

Definition at line 172 of file `stm32f429i_discovery_lcd.h`.

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery
BSP User Manual by `doxygen` 1.7.6.1
STM32F429I-Discovery BSP User Manual

Main Page  Modules  Data Structures  Files

Directories

STM32F429I DISCOVERY LOW LEVEL LED

STM32F429I DISCOVERY LOW LEVEL Exported Constants

Define for STM32F429I_DISCO board. More...
#define LEDn 2  
#define LED3_PIN GPIO_PIN_13  
#define LED3_GPIO_PORT GPIOG  
#define LED3_GPIO_CLK_ENABLE() __HAL_RCC_GPIOG_CLK_ENABLE()  
#define LED3_GPIO_CLK_DISABLE() __HAL_RCC_GPIOG_CLK_DISABLE()  
#define LED4_PIN GPIO_PIN_14  
#define LED4_GPIO_PORT GPIOG  
#define LED4_GPIO_CLK_ENABLE() __HAL_RCC_GPIOG_CLK_ENABLE()  
#define LED4_GPIO_CLK_DISABLE() __HAL_RCC_GPIOG_CLK_DISABLE()  
#define LEDx_GPIO_CLK_ENABLE(__INDEX__)  
#define LEDx_GPIO_CLK_DISABLE(__INDEX__)
Detailed Description

Define for STM32F429I_DISCO board.
Define Documentation

```c
#define LED3_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOG_CLK_DISABLE()
Definition at line 105 of file stm32f429i_discovery.h.
```

```c
#define LED3_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOG_CLK_ENABLE()
Definition at line 104 of file stm32f429i_discovery.h.
```

```c
#define LED3_GPIO_PORT GPIOG
Definition at line 103 of file stm32f429i_discovery.h.
```

```c
#define LED3_PIN GPIO_PIN_13
Definition at line 102 of file stm32f429i_discovery.h.
```

```c
#define LED4_GPIO_CLK_DISABLE ( ) __HAL_RCC_GPIOG_CLK_DISABLE()
Definition at line 110 of file stm32f429i_discovery.h.
```

```c
#define LED4_GPIO_CLK_ENABLE ( ) __HAL_RCC_GPIOG_CLK_ENABLE()
Definition at line 109 of file stm32f429i_discovery.h.
```

```c
#define LED4_GPIO_PORT GPIOG
Definition at line 108 of file stm32f429i_discovery.h.
```
#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().
<table>
<thead>
<tr>
<th>Directories</th>
<th>Defines</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM32F429I DISCOVERY LCD Private Defines</td>
<td></td>
</tr>
</tbody>
</table>

STM32F429I DISCOVERY LCD
Defines

#define POLY_X(Z) ((int32_t)((Points + Z)->X))
#define POLY_Y(Z) ((int32_t)((Points + Z)->Y))
Define Documentation

```c
#define POLY_X (Z) ((int32_t)((Points + Z)->X))
```
Definition at line 108 of file `stm32f429i_discovery_lcd.c`.
Referenced by `BSP_LCD_FillPolygon()`.

```c
#define POLY_Y (Z) ((int32_t)((Points + Z)->Y))
```
Definition at line 109 of file `stm32f429i_discovery_lcd.c`.
Referenced by `BSP_LCD_FillPolygon()`.

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery
BSP User Manual by `doxygen` 1.7.6.1
# STM32F429I-Discovery BSP User Manual

## Directories

<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 Constants</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STM32F429I DISCOVERY TS</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Defines

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

```c
enum TS_StatusTypeDef { TS_OK = 0x00, TS_ERROR = 0x01, TS_TIMEOUT = 0x02 }
```
<table>
<thead>
<tr>
<th>Define Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>#define <strong>TS_SWAP_NONE</strong> 0x00</td>
</tr>
<tr>
<td>Definition at line <strong>81</strong> of file <strong>stm32f429i_discovery_ts.h</strong>.</td>
</tr>
</tbody>
</table>

| #define **TS_SWAP_X** 0x01 |
| Definition at line **82** of file **stm32f429i_discovery_ts.h**. |

| #define **TS_SWAP_XY** 0x04 |
| Definition at line **84** of file **stm32f429i_discovery_ts.h**. |

| #define **TS_SWAP_Y** 0x02 |
| Definition at line **83** of file **stm32f429i_discovery_ts.h**. |
Enumeration Type Documentation

```c
enum TS_StatusTypeDef
{
    TS_OK,
    TS_ERROR,
    TS_TIMEOUT
};
```

Definition at line 86 of file `stm32f429i_discovery_ts.h`.

Generated on Fri Feb 17 2017 12:10:39 for STM32F429I-Discovery 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>Directories</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### STM32F429I DISCOVERY TS Private Variables

STM32F429I DISCOVERY TS
## Variables

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable</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   V2.1.5
 * @date      27-January-2017
 * @brief     This file contains definitions for STM32F429I-Discovery Kit LEDs,
 *            push-buttons hardware resources.

 * @attention

 Copyright (c) 2017 STM Microelectronics
 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.

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_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 */
* @{
* /

/** @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
*/
/* @{ */
#define LEDn 2
#define LED3_PIN GPIO_PIN_13
#define LED3_GPIO_PORT GPIOG
#define LED3_GPIO_CLK_ENABLE() __HAL_RCC_GPIOG_CLK_ENABLE()
#define LED3_GPIO_CLK_DISABLE() __HAL_RCC_GPIOG_CLK_DISABLE()
#define LED4_PIN GPIO_PIN_14
#define LED4_GPIO_PORT GPIOG
#define LED4_GPIO_CLK_ENABLE() __HAL_RCC_GPIOG_CLK_ENABLE()
#define LED4_GPIO_CLK_DISABLE() __HAL_RCC_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 BUTTON1 1
 */

/**
 * @brief Wakeup push-button
 */
#define KEY_BUTTON_PIN GPIO_PIN_0
#define KEY BUTTON_GPIO_PORT GPIOA
#define KEY BUTTON_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
#define KEY BUTTON_GPIO_CLK_DISABLE() __HAL_RCC_GPIOA_CLK_DISABLE()
#define KEY BUTTON EXTI_IRQn EXTI0_IRQHandler
#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
 * STM32F429I DISCOVERY LOW LEVEL BUS
 * @{
 */

/*
 * Exported constants
 */

#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 */

/*###############################
 * I2Cx
 * User can use this section to tailor I2Cx instance used and associated resources */

#define DISCOVERY_I2Cx I2C3
#define DISCOVERY_I2Cx_CLOCK_ENABLE() __HAL_RCC_I2C3_CLK_ENABLE()
#define DISCOVERY_I2Cx_FORCE_RESET() __HAL_RCC_I2C3_FORCE_RESET()
#define DISCOVERY_I2Cx_RELEASE_RESET() __HAL_RCC_I2C3_RELEASE_RESET()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_ENABLE()
#define DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
#define DISCOVERY_I2Cx_SDA_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()

#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

#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 100000
#endif /* BSP_I2C_SPEED */
#define I2Cx_TIMEOUT_MAX 0x3000 /*<! The value of the maximal timeout for I2C waiting loops */
#define DISCOVERY_SPIx_SPI5
#define DISCOVERY_SPIx_CLK_ENABLE() __HAL_RCC_SPI5_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_PORT GPIOF
#define DISCOVERY_SPIx_AF GPIO_AF5_SPI5
#define DISCOVERY_SPIx_GPIO_CLK_ENABLE() __HAL_RCC_GPIOF_CLK_ENABLE()
#define DISCOVERY_SPIx_GPIO_CLK_DISABLE() __HAL_RCC_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 */
/* 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 corrupt.
You may modify these timeout values depending on CPU frequency and application conditions (interrupts routines ...). */
#define SPIx_TIMEOUT_MAX ((uint32_t)0x1000)
/* ****************************************** IOE #######
 *******************************************/
/** * @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() __HAL_RCC_GPIOA_CLK_ENABLE()
#define STMPE811_INT_CLK_DISABLE() __HAL_RCC_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)

/* 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)

/* 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() __HAL_RCC_GPIOC_CLK_ENABLE()
#define LCD_NCS_GPIO_CLK_DISABLE() __HAL_RCC_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() __HAL_RCC_GPIOD_CLK_ENABLE()
#define LCD_WRX_GPIO_CLK_DISABLE() __HAL_RCC_GPIOD_CLK_DISABLE()

#define LCD_RDX_PIN GPIO_PIN_12
#define LCD_RDX_GPIO_PORT GPIOD
#define LCD_RDX_GPIO_CLK_ENABLE() __HAL_RCC_GPIOD_CLK_ENABLE()
#define LCD_RDX_GPIO_CLK_DISABLE() __HAL_RCC_GPIOD_CLK_DISABLE()

/*################################GYROSCOPE################################*/

/* Read/Write command */
```c
#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_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_ENABLE()
#define GYRO_CS_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()
#define GYRO_CS_GPIO_PORT GPIOC
#define GYRO_CS_PIN GPIO_PIN_1 /* PC.01 */
#define GYRO_INT_GPIO_CLK_ENABLE() __HAL_RCC_GPIOA_CLK_ENABLE()
#define GYRO_INT_GPIO_CLK_DISABLE() __HAL_RCC_GPIOA_CLK_DISABLE()
#define GYRO_INT_GPIO_PORT GPIOA
#define GYRO_INT1_PIN GPIO_PIN_1 /* PA.01 */
```
#define GYRO_INT1_EXTI_IRQn EXTI1_IRQn

#define GYRO_INT2_PIN GPIO_PIN_2 /* PA.02 */
#define GYRO_INT2_EXTI_IRQn EXTI2_IRQn

/**
   @}
*/

#ifdef EE_M24LR64
/**	@defgroup STM32F429I_DISCOVERY_LOW_LEVEL_I2C_EEPROM STM32F429I DISCOVERY LOW LEVEL I2C EEPROM
*/
 *
#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(__HAL_RCC_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
DMA1_Stream2_IRQHandler

#define EEPROM_I2C_DMA_PREPRIO 0x0F

/**
 * @}
 */

#endif /* EE_M24LR64 */

/**
 * @defgroup STM32F429I_DISCOVERY_LOW_LEVEL_Exported_Macros STM32F429I DISCOVERY LOW LEVEL Exported Macros
 */

/**
 * @defgroup 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 V2.1.5
 * @date    27-January-2017
 * @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) 2017 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.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_TypesDefinitions STM32F429I DISCOVERY LOW LEVEL Private TypesDefinitions
* @}
*/

/**
@defgroup STM32F429I_DISCOVERY_LOW_LEVEL_Private_TypesDefinitions STM32F429I DISCOVERY LOW LEVEL Private TypesDefinitions
* @}
*/
_Private_Defines STM32F429I DISCOVERY LOW LEVEL Private Defines
00064  * @{
00065   */
00066
00067 /**
00068  * @brief STM32F429I DISCO BSP Driver version number V2.1.5
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 (0x05) /*!< [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)\
  |(__STM32F429I_DISCO_BSP_VERSION_SUB1 << 16)\
  |(__STM32F429I_DISCO_BSP_VERSION_SUB2 << 8 )\
  |(__STM32F429I_DISCO_BSP_VERSION_RC))
00075 /**
00076  */
00077
00078 /** @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_EXTIIRQn};

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;

/**
 */

/** @defgroup STM32F429I_DISCOVERY_LOW_LEVEL */
* @{
  * 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 Addr, uint8_t Reg);
ReadSize);
00135 static uint8_t SPIx_WriteRead(uin
t8_t Byte);
00136 static void SPIx_Error(void);
00137 static void SPIx_MspInit(SPI_H
andleTypeDef *hspi);
00138
00139 /* Link function for LCD peripheral */
00140 void LCD_IO_Init(void);
00141 void LCD_IO_WriteData(uint16_t RegValue);
00142 void LCD_IO_WriteReg(uint8_t Reg);
00143 uint32_t LCD_IO_ReadData(uint16_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 A
ddr, uint8_t Reg);
00152 uint16_t IOE_ReadMultiple(u
int8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16
_t Length);
00153 void IOE_WriteMultiple(u
int8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint1
6_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
```c
/*
 * @brief Configures LED GPIO.
 * @param Led: Specifies the Led to be configured.
 * This parameter can be one of following parameters:
 * @arg LED3
 * @arg LED4
 */

void BSP_LED_Init(Led_TypeDef Led) {
    GPIO_InitTypeDef GPIO_InitStruct;

    /* Enable the GPIO_LED Clock */
    LEDx_GPIO_CLK_ENABLE(Led);

    /* Configure the GPIO_LED pin */
    GPIO_InitStruct.Pin = GPIO_PIN[Led];
    GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStruct.Pull = GPIO_PULLUP;
    GPIO_InitStruct.Speed = GPIO_SPEED_FAST;

    HAL_GPIO_Init(GPIO_PORT[Led], &GPIO_InitStruct);
    HAL_GPIO_WritePin(GPIO_PORT[Led], GPIO_PIN[Led], GPIO_PIN_RESET);
```
/**
 * @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)
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;
    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_IRQn[Button]), 0x0F, 0x00);
    HAL_NVIC_EnableIRQ((IRQn_Type)(BUTTON_IRQn[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.
 */
```c
uint32_t BSP_PB_GetState(Button_TypeDef Button)
{
    return HAL_GPIO_ReadPin(BUTTON_PORT[Button], BUTTON_PIN[Button]);
}

uffman_Toplast_ROM00293

 /**************************************************************************/
 /** BUS OPERATIONS */
 /**************************************************************************/

 /**************************************************************************/
 /** I2C Routine */
 /**************************************************************************/

 static void I2Cx_MspInit(I2C_HandleTypeDef *hi2c)
 {
    GPIO_InitTypeDef GPIO_InitStruct;

    ifdef EE_M24LR64
    static DMA_HandleTypeDef hdma_tx;
    static DMA_HandleTypeDef hdma_rx;
    endif /* EE_M24LR64 */

    if (hi2c->Instance == DISCOVERY_I2Cx)
    {
        /* Configure the GPIOs ------------------------
        */
        /* Enable GPIO clock */
```
00323 DISCOVERY_I2Cx_SDA_GPIO_CLK_ENABLE();
00324 DISCOVERY_I2Cx_SCL_GPIO_CLK_ENABLE();
00325 /* Configure I2C Tx as alternate function */
00326 GPIO_InitStruct.Pin = DISCOVERY_I2Cx_SCL_PIN;
00327 GPIO_InitStruct.Mode = GPIO_MODE_AF_OD;
00328 GPIO_InitStruct.Pull = GPIO_NOPULL;
00329 GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
00330 GPIO_InitStruct.Alternate = DISCOVERY_I2Cx_SCL_SDA_AF;
00331 HAL_GPIO_Init(DISCOVERY_I2Cx_SCL_GPIO_PORT, &GPIO_InitStruct);
00332 /* Configure I2C Rx as alternate function */
00333 GPIO_InitStruct.Pin = DISCOVERY_I2Cx_SDA_PIN;
00334 HAL_GPIO_Init(DISCOVERY_I2Cx_SDA_GPIO_PORT, &GPIO_InitStruct);
00335 /* Configure the Discovery I2Cx peripheral --------------------------*/
00336 /* Enable I2C3 clock */
00337 DISCOVERY_I2Cx_CLOCK_ENABLE();
00338 /* Force the I2C Peripheral Clock Reset */
00339 DISCOVERY_I2Cx_FORCE_RESET();
00340 /* Release the I2C Peripheral Clock Reset */
00341 DISCOVERY_I2Cx_RELEASE_RESET();
/* Enable and set Discovery I2Cx Interrupt to the lowest priority */
HAL_NVIC_SetPriority(DISCOVERY_I2Cx_EV_IRQn, 0x0F, 0);
HAL_NVIC_EnableIRQ(DISCOVERY_I2Cx_EV_IRQn);

/* Enable and set Discovery I2Cx Interrupt to the lowest priority */
HAL_NVIC_SetPriority(DISCOVERY_I2Cx_ER_IRQn, 0x0F, 0);
HAL_NVIC_EnableIRQ(DISCOVERY_I2Cx_ER_IRQn);

#ifdef EE_M24LR64
/* I2C DMA TX and RX channels configuration */
/* Enable the DMA clock */
EEPROM_I2C_DMA_CLK_ENABLE();

/* Configure the DMA stream for the EE I2C peripheral TX direction */
/* Configure the DMA Stream */
hdma_tx.Instance = EEPROM_I2C_DMA_STREAM_TX;
/* Set the parameters to be configured */

hdma_tx.Init.Channel = EEPROM_I2C_DMA_CHANNEL;
hdma_tx.Init.Direction = DMA_MEMORY_TO_PERIPH;
hdma_tx.InitPeriphInc = DMA_PINC_DISABLE;
hdma_tx.Init.MemInc = DMA_MINC_ENABLE;
hdma_tx.InitPeriphDataAlignment = DMA_MINC_ENABLE;

```c
```
```c
PDATAALIGN_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
x);
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_PREPRIO, 0);
00387  HAL_NVIC_EnableIRQ((IRQn_Type)(EEPROM_I2 C_DMA_TX_IRQn));
00388
00389   /* Configure the DMA stream for the EE I2C peripheral TX direction */
00390   /* Configure the DMA Stream */
00391  hdma_rx.Instance = EEPR
00392 OM_I2C_DMA_STREAM_RX;
00393  /* 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 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));

#endif /* EE_M24LR64 */

/**
 * @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_DISABLE;
        I2cHandle.Init.OwnAddress2 = 0;
        I2cHandle.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
        I2cHandle.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
        /* 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), 0x0F, 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
 */
00468    */
00469 static void I2Cx_WriteData(uint8_t Addr, uint8_t Reg, uint8_t Value)
00470 {
00471    HAL_StatusTypeDef status = HAL_OK;
00472    status = HAL_I2C_Mem_Write(&I2cHandle, Addr, (uint16_t)Reg, I2C_MEMADD_SIZE_8BIT, &Value, 1, I2cxTimeout);
00473
00474    /* Check the communication status */
00475    if(status != HAL_OK)
00476    {
00477        /* Re-Initialize the BUS */
00478        I2Cx_Error();
00479    }
00480 }
00481
00482
00483    /**
00484    * @brief Writes a value in a register of the device through BUS.
00485    * @param Addr: Device address on BUS Bus.
00486    * @param Reg: The target register address to write
00487    * @param pBuffer: The target register value to be written
00488    * @param Length: buffer size to be written
00489    */
00490 static void I2Cx_WriteBuffer(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length)
00491 {
00492    HAL_StatusTypeDef status = HAL_OK;
00493    status = HAL_I2C_Mem_Write(&I2cHandle, Addr, (uint16_t)Reg, I2C_MEMADD_SIZE_8BIT, pBuffer, Length, I2cxTimeout);
00494
00495    /* Check the communication status */
00496    if(status != HAL_OK)
00497    {
00498        /* Re-Initialize the BUS */
00499        I2Cx_Error();
00500    }
00501    }
length, I2cxTimeout);

    /**
     * @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);

        /* Check the communication status */
        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);
    if(status != HAL_OK)
    {
        /* Re-Initialize the BUS */
        I2Cx_Error();
    }
    return status;
}

@brief Reads multiple data on the BUS in using DMA mode.

@param Addr: I2C Address

@param Reg: Reg Address
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);

    if(status != HAL_OK)
    {
        /* Re-Initialize the BUS */
        I2Cx_Error();
    }

    return status;
}

static HAL_StatusTypeDef I2Cx_IsDeviceReady(
    uint16_t DevAddress, uint32_t Trials)
{
    /* Check the communication status */
    if(DevAddress != HAL_OK)
    {
        /* Re-Initialize the BUS */
        I2Cx_Error();
    }

    // Check the target device is ready for communication.
    // This function is used with Memory devices
    return status;
}
return (HAL_I2C_IsDeviceReady(&I2cHandle, DevAddress, Trials, I2cxTimeout));

#ifdef /* 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 Initialization
 **********************************************************************************/

/**
 * @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 baudrate

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);

}
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;
}

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, SpixTimeout) != 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);
/**
 * @brief Configures the LCD_SPI interface.
 */

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
00816    /* Reset LCD control line(/CS) and Send data */
00817    LCD_CS_LOW();
00818    SPIx_Write(RegValue);
00819
00820    /* Deselect: Chip Select high */
00821    LCD_CS_HIGH();
00822 }
00823
00824    /**
00825    * @brief Writes register address.
00826    */
00827    void LCD_IO_WriteReg(uint8_t Reg)
00828    {
00829    /* Reset WRX to send command */
00830    LCD_WRX_LOW();
00831
00832    /* Reset LCD control line(/CS) and Send command */
00833    LCD_CS_LOW();
00834    SPIx_Write(Reg);
00835
00836    /* Deselect: Chip Select high */
00837    LCD_CS_HIGH();
00838 }
00839
00840    /**
00841    * @brief Reads register value.
00842    * @param RegValue Address of the register to read
00843    * @param ReadSize Number of bytes to read
00844    * @retval Content of the register value
00845    */
00846    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;

void LCD_Delay(uint32_t Delay) {
    HAL_Delay(Delay);
}

/* @brief Wait for loop in ms.
 * @param Delay in ms.
 */

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
 */
@retval The read data
uint8_t IOE_Read(uint8_t Addr, uint8_t Reg) {
    return I2Cx_ReadData(Addr, Reg);
}

/*
@brief IOE Writes multiple data.
@param Addr: I2C Address
@param Reg: Reg Address
@param pBuffer: pointer to data buffer
@param Length: length of the data
*/
void IOE_WriteMultiple(uint8_t Addr, uint8_t Reg, uint8_t *pBuffer, uint16_t Length) {
    I2Cx_WriteBuffer(Addr, Reg, pBuffer, Length);
}

/*
@brief IOE Reads multiple data.
@param Addr: I2C Address
@param Reg: Reg Address
@param pBuffer: pointer to data buffer
@param Length: length of the data
@retval 0 if no problems to read multiple data
*/
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.
 *
 */

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;

}
01003  
01004  /* Set chip select Low at the start of the transmission */
01005    GYRO_CS_LOW();
01006
01007  /* Send the Address of the indexed register */
01008    SPIx_WriteRead(WriteAddr);
01009
01010  /* Send the data that will be written into the device (MSB First) */
01011    while(NumByteToWrite >= 0x01)
01012    {
01013      SPIx_WriteRead(*pBuffer);
01014      NumByteToWrite--;
01015      pBuffer++;
01016    }
01017
01018  /* Set chip select High at the end of the transmission */
01019    GYRO_CS_HIGH();
01020  }
01021
01022  /**
01023    * @brief Reads a block of data from the Gyroscope.
01024    * @param pBuffer: Pointer to the buffer that receives the data read from the Gyroscope.
01025    * @param ReadAddr: Gyroscope's internal address to read from.
01026    * @param NumByteToRead: Number of bytes to read from the Gyroscope.
01027    */
01028  void GYRO_IO_Read(uint8_t* pBuffer, uint8_t ReadAddr, uint16_t NumByteToRead)
01029  {
01030    if(NumByteToRead > 0x01)
01031  {  
01032   ReadAddr |= (uint8_t)(READWRITE_CMD | MULTIPLEBYTE_CMD);  
01033  }  
01034  else  
01035   {  
01036   ReadAddr |= (uint8_t)READWRITE_CMD;  
01037  }  
01038  /* Set chip select Low at the start of the transmission */  
01039  GYRO_CS_LOW();  
01040  
01041  /* Send the Address of the indexed register */  
01042  SPIx_WriteRead(ReadAddr);  
01043  
01044  /* Receive the data that will be read from the device (MSB First) */  
01045  while(NumByteToRead > 0x00)  
01046   {  
01047     /* Send dummy byte (0x00) to generate the SPI clock to Gyroscope (Slave device) */  
01048     *pBuffer = SPIx_WriteRead(DUMMY_BYTE);  
01049     NumByteToRead--;  
01050     pBuffer++;  
01051  }  
01052  
01053  /* Set chip select High at the end of the transmission */  
01054  GYRO_CS_HIGH();  
01055  }  
01056  
01057  
01058  #ifdef EE_M24LR64  
01059  
01060  /************************************************************************** LINK I2C EEPROM **************************************************************************/
/**
 * @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
 * @paramBufferSize: 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));
}
#endif /* EE_M24LR64 */
stm32f429i_discovery_gyroscope.h

Go to the documentation of this file.

```c
/**
 * @file stm32f429i_discovery_gyroscope.h
 *
 * @author MCD Application Team
 * @version V2.1.5
 * @date 27-January-2017
 * @brief This file contains definitions for stm32f429i_discovery_gyroscope.c
 * firmware driver.
 */

* @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.
/* Define to prevent recursive inclusion ----------------------------------*/

#ifndef __STM32F429I_DISCOVERY_GYROSCOPE_H
#define __STM32F429I_DISCOVERY_GYROSCOPE_H

#ifdef __cplusplus
extern "C" {

/* 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 */
* @{

*/
#endif
}

/* @} */
#endif /* ifndef __STM32F429I_DISCOVERY_GYROSCOPE_H */
typedef enum {
    GYRO_OK = 0,
    GYRO_ERROR = 1,
    GYRO_TIMEOUT = 2
} GYRO_StatusTypeDef;

/**
 * @}
 */

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Exported_Constants
 * STM32F429I DISCOVERY GYROSCOPE Exported Constants
 * @{
 */

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Exported_Macros
 * STM32F429I DISCOVERY GYROSCOPE Exported Macros
 * @{
 */

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Exported_Functions
 * 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);

/** *
 */

#endif /*__cplusplus*/

/*__STM32F429I_DISCOVERY_GYROSCOPE_H*/

/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
stm32f429i_discovery_gyroscope.c

Go to the documentation of this file.

00001 /**
00002 ************************************************************
00003 * @file     stm32f429i_discovery_gyroscope.c
00004 *
00005 * @author  MCD Application Team
00006 * @version V2.1.5
00007 * @date    27-January-2017
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 *
00013 * <h2><center>&copy; COPYRIGHT(c) 2017 STM microelectronics</center></h2>
00014 *
00015 * Redistribution and use in source and binary forms, with or without modification,
00016 * are permitted provided that the followin
g 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, 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_gyroscope.h"

/**  @addtogroup BSP  */
/**  @addtogroup STM32F429I_DISCOVERY  */
/**  @defgroup STM32F429I_DISCOVERY_GYROSCOPE  */
/**  @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_TypesDefinitions  */
/**  @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_Defines  */
/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_Macros STM32F429I DISCOVERY GYROSCOPE Private Macros
 * @{
 */

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_Variables STM32F429I DISCOVERY GYROSCOPE Private Variables
 * @{

static GYRO_DrvTypeDef *GyroscopeDrv;

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_FunctionPrototypes STM32F429I DISCOVERY GYROSCOPE Private FunctionPrototypes
 * @{

/**
 * @defgroup STM32F429I_DISCOVERY_GYROSCOPE_Private_Functions STM32F429I DISCOVERY GYROSCOPE Private Functions
 * @{

*/
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();
    }
}
@brief Configures INT1 interrupt.
@param pIntConfig: pointer to a L3GD20_InterruptConfig_TypeDef structure that contains the configuration setting for the L3GD20 Interrupt.

*/

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) "

        interruptconfig |= (uint8_t)(pIntConfig->Interrupt_Axes) << 8);

        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);
    }
}
/**
 * @}
 */

/**
 * @}
 */

/**
 * @}
 */

/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE*****/
/**
 ******************************************
 ******************************************
 *
 @file stm32f429i_discovery_io.h
 @author MCD Application Team
 @version V2.1.5
 @date 27-January-2017
 @brief This file contains all the functions prototypes for the
 stm32f429i_discovery_io.c driver.

 ******************************************
 ******************************************
 *
 @attention
 *
 <h2><center>&copy; COPYRIGHT(c) 2017 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.
/* Define to prevent recursive inclusion ----------------------------------*/
#ifndef __STM32F429I_DISCOVERY_IO_H
#define __STM32F429I_DISCOVERY_IO_H

/*	#include "stm32f429i_discovery.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_Macros STM32F429I DISCOVERY IO Exported Macros
 * @{
 */
/**
 * @}
 */
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.

00001 /**
00002 ******************************************
00003 * @file  stm32f429i_discovery_io.c
00004 * @author MCD Application Team
00005 * @version V2.1.5
00006 * @date  27-January-2017
00007 * @brief This file provides a set of functions needed to manage the STMPE811
00008 *        IO Expander device mounted on STM32F429I-Discovery Kit.
00009 ******************************************
00010 * @attention
00011 *
00012 * <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
00013 * Redistribution and use in source and binary forms, with or without modification,
00014 * 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.
/* Includes ------------------------------------*/
#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 I/O functionalities and configures all necessary hardware resources (GP IOs, 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;
    }

    if(ret == IO_OK)
    {
        /*
        * 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;
        }
    }
}
IoDrv->Init(IO_I2C_ADDRESS);
IoDrv->Start(IO_I2C_ADDRESS, IO_PIN_ALL);
}
return ret;
}

/**
 * @brief Gets the selected pins IT status.
 *
 * @param IoPin: The selected pins to check the status.
 * This parameter could be any combination of the IO pins.
 *
 * @retval Status of IO Pin checked.
 */
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 */
```c
00177    IoDrv->WritePin(IO_I2C_ADDRESS, IoPin, Pin State);
00178 }  
00179
00180 /**<
00181 *  @brief  Gets the selected pins current state.
00182 *  @param  IoPin: The selected pins to read.
00183 *  This parameter could be any combination of the IO pins.
00184 *  @retval The current pins state
00185 */
00186 uint16_t BSP_IO_ReadPin(uint16_t IoPin)
00187 {  
00188   return(IoDrv->ReadPin(IO_I2C_ADDRESS, IoPin));
00189 }  
00190
00191 /**<
00192 *  @brief  Toggles the selected pins state.
00193 *  @param  IoPin: The selected pins to toggle.
00194 *  This parameter could be any combination of the IO pins.
00195 */
00196 void BSP_IO_TogglePin(uint16_t IoPin)
00197 {  
00198   /* Toggle the current pin state */  
00199   if(IoDrv->ReadPin(IO_I2C_ADDRESS, IoPin) = = 1 /* Set */)
00200   {  
00201     IoDrv->WritePin(IO_I2C_ADDRESS, IoPin, 0 /* Reset */);
00202   }  
00203 else
00204   {
```
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.

00001  /**
00002  ******************************************
00003  ******************************************
00004  * @file   stm32f429i_discovery_sdram.h
00005  * @author MCD Application Team
00006  * @version V2.1.5
00007  * @date   27-January-2017
00008  * @brief  This file contains all the function prototypes for the
00009  *         stm32f429i_discovery_sdram.c driver.
00010  ******************************************
00011  ******************************************
00012  * @attention
00013  *
00014  * Redistributions and use in source and binary forms, with or without modification,
00015  * 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_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 */
* @{
* /

/** @addtogroup STM32F429I_DISCOVERY_SDRAM_Exported_Types */
* @{
* /

/** @defgroup STM32F429I_DISCOVERY_SDRAM_Exp
orted_Types STM32F429I DISCOVERY SDRAM Exported Ty
pes */
* @{
* /

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/

*/
/**
 * @defgroup STM32F429I_DISCOVERY_SDRAM_Exported_Constants STM32F429I DISCOVERY SDRAM Exported Constants
 *
 * @{ */

/**
 * @brief SDRAM status structure definition
 *
 * #define SDRAM_OK ((uint8_t)0x00)
 * #define SDRAM_ERROR ((uint8_t)0x01)
 */

/**
 * @brief FMC SDRAM Bank address
 *
 * #define SDRAM_DEVICE_ADDR ((uint32_t)0xD0000000)
 * #define SDRAM_DEVICE_SIZE ((uint32_t)0x800000) /* SDRAM device size in Bytes */
 */

/**
 * @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_2 */

#define SDRAM_CAS_LATENCY FMC_SDRAM_CAS_LATENCY_2

#define SDRAM_MEMORY_WIDTH FMC_SDRAM_MEM_BUS_WIDTH_16

/**
 * @brief FMC SDRAM Bank address
 */

#ifdef SDRAM_DEVICE_ADDR
#endif

#define SDRAM_DEVICE_ADDR ((uint32_t)0xD0000000)

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

/**
 * @brief SDRAM status structure definition
 */

#define SDRAM_OK ((uint8_t)0x00)
#define SDRAM_ERROR ((uint8_t)0x01)

}/* */
### FMC SDRAM Memory Clock Period

```c
#define SDCLOCK_PERIOD FMC_SDRAM_CLK

/* Default configuration used with LCD */
#define SDCLOCK_PERIOD FMC_SDRAM_CLK
```

### FMC SDRAM Memory Read Burst Feature

```c
#define SDRAM_READBURST FMC_SDRAM_RB

/* Default configuration used with LCD */
#define SDRAM_READBURST FMC_SDRAM_RB
```

### FMC SDRAM Bank Remap

```c
#define SDRAM_BANK_REMAP

/* Set the refresh rate counter */
#define REFRESH_COUNT (uint32_t)1386 /* SDRAM refresh counter */
#define SDRAM_TIMEOUT (uint32_t)0xFFFF
```

### DMA definitions for SDRAM DMA transfer

```c
#define __DMAx_CLK_ENABLE __HAL_RCC_DMA2_CLK_ENABLE
#define SDRAM_DMAx_CHANNEL DMA_CHANNEL_0
```
```c
#define SDRAM_DMAx_STREAM DMA2_Stream0
#define SDRAM_DMAx_IRQn DMA2_Stream0_IRQHandler
#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)0x0200)
/**
 * @} */
```
/**
 * @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);

uint8_t BSP_SDRAM_ReadData(uint32_t uwStartAddress, uint32_t* pData, uint32_t uwDataSize);

uint8_t BSP_SDRAM_ReadData_DMA(uint32_t uwStartAddress, uint32_t* pData, uint32_t uwDataSize);

uint8_t BSP_SDRAM_WriteData(uint32_t uwStartAddress, uint32_t* pData, uint32_t uwDataSize);

uint8_t BSP_SDRAM_WriteData_DMA(uint32_t uwStartAddress, uint32_t* pData, uint32_t uwDataSize);

void BSP_SDRAM_Sendcmd(FMC_SDRAM_COMMANDTypeDef *SdramCmd);

void BSP_SDRAM_DMA_IRQHandler(void);

/* These function can be modified in case the current settings (e.g. DMA stream) need to be changed for specific application */
```c
void BSP_SDRAM_MspInit(SDRAM_HandleTypeDef *hsdram, void *Params);

void BSP_SDRAM_MspDeInit(SDRAM_HandleTypeDef *hsdram, void *Params);
```
stm32f429i_discovery_sdram.c

Go to the documentation of this file.

```c
/**
 * @file    stm32f429i_discovery_sdram.c
 * @author  MCD Application Team
 * @version V2.1.5
 * @date    27-January-2017
 * @brief   This file provides a set of functions needed to drive the
 *          IS42S16400J SDRAM memory mounted on STM32F429I-Discovery Kit.
 */
```

```c
* @attention
* <h2><center>&copy; COPYRIGHT(c) 2017 STMicroelectronics</center></h2>
```

```c
* 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.
/**
 @addtogroup BSP
 @{
*/

/**
 @addtogroup STM32F429I_DISCOVERY
 @{
*/

/**
 @defgroup STM32F429I_DISCOVERY_SDRAM
 @{
*/

/**
 @defgroup STM32F429I_DISCOVERY_SDRAM_Private_Types_Definitions
 @{
*/

/**
 @defgroup 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
 *
 */

static SDRAM_HandleTypeDef SdramHandle;
static FMC_SDRAM_TimingTypeDef Timing;
static FMC_SDRAM_CommandTypeDef Command;

/**
 * @
 */

/**
 * @defgroup STM32F429I_DISCOVERY_SDRAM_Private_Function_Prototypes STM32F429I DISCOVERY SDRAM Private Function Prototypes
 *
 */

/**
 * @
 */

/**
 * @defgroup STM32F429I_DISCOVERY_SDRAM_Private_Functions STM32F429I DISCOVERY SDRAM Private Functions
 *
 */

* @{
/**
 * @brief Initializes the SDRAM device.
 */

uint8_t BSP_SDRAM_Init(void) {
    static uint8_t sdramstatus = SDRAM_ERROR;

    /* SDRAM device configuration */
    SdramHandle.Instance = FMC_SDRAM_DEVICE;

    /* FMC Configuration --------------------------*/
    /* FMC SDRAM Bank configuration */
    /* Timing configuration for 90 Mhz of SD clock 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;

/* Column addressing: [11:0] */
SdramHandle.Init.RowBitsNumber = FMC_SDRAM_ROW_BITS_NUM_12;
SdramHandle.Init.MemoryDataWidth = SDRAM_MEMORY_WIDTH;
SdramHandle.Init.InternalBankNumber = FMC_SDRAM_INTERN_BANKS_NUM_4;
SdramHandle.Init.CASLatency = SDRAM_CAS_LATENCY;
SdramHandle.Init.WriteProtection = FMC_SDRAM_WRITE_PROTECTION_DISABLE;
SdramHandle.Init.SDClockPeriod = SDCLK_PERIOD;
SdramHandle.Init.ReadBurst = SDRAM_READBURST;
SdramHandle.Init.ReadPipeDelay = FMC_SDRAM_RPIPE_DELAY_1;

/* SDRAM controller initialization */
/* __weak function can be surcharged by the application code */
BSP_SDRAM_MspInit(&SdramHandle, (void *)NULL);
if(HAL_SDRAM_Init(&SdramHandle, &Timing) != HAL_OK)
{
    sdramstatus = SDRAM_ERROR;
}
else
{
    sdramstatus = SDRAM_OK;
}

/* SDRAM initialization sequence */
BSP_SDRAM_Initialization_sequence(REFRESH_
void BSP_SDRAM_Initialization_sequence(uint32_t RefreshCount) {
	__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_SDRA 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_SDRA M_CMD_AUTOREFRESH_MODE;
Command.CommandTarget = FMC_SDRA M_CMD_TARGET_BANK2;
Command.AutoRefreshNumber = 4;
Command.ModeRegisterDefinition = 0;

/* Send the command */
HAL_SDRAM_SendCommand(&SdramHandle, &Command, SDRAM_TIMEOUT);

/* Step 5: Program the external memory mode register */
tmppmrdr = (uint32_t)SDRAM_MODEREG_BURST_LEN
               | SDRAM_MODEREG_BURST_TYP
E_SEQUENTIAL | SDRAM_MODEREG_CAS_LATEN
CY_3 | SDRAM_MODEREG_OPERATING
_MODE_STANDARD |
0200 SDRAM_MODEREG_WRITEBURST_MODE;
0201 Command.CommandMode = FMC_SDRA M_CMD_LOAD_MODE;
00203  Command.CommandTarget = FMC_SDRAMM_CMD_TARGET_BANK2;
00204  Command.AutoRefreshNumber = 1;
00205  Command.ModeRegisterDefinition = tmpmrd;
00206
00207  /* Send the command */
00208  HAL_SDRAM_SendCommand(&SdramHandle, &Command, SDRAM_TIMEOUT);
00209
00210  /* Step 6: Set the refresh rate counter */
00211  /* Set the device refresh rate */
00212  HAL_SDRAM_ProgramRefreshRate(&SdramHandle, RefreshCount);
00213 }
00214
00215  /**
00216  * @brief  Reads an mount of data from the SDRAM memory in polling mode.
00217  * @param  uwStartAddress : Read start address
00218  * @param  pData : Pointer to data to be read
00219  * @param  uwDataSize: Size of read data from the memory
00220  */
00221  uint8_t BSP_SDRAM_ReadData(uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)
00222  {
00223      if(HAL_SDRAM_Read_32b(&SdramHandle, (uint32_t *)uwStartAddress, pData, uwDataSize) != HAL_OK)
00224      {
00225          return SDRAM_ERROR;
00226      }
00227      else
00228      {
00229          return SDRAM_OK;
00230      }
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

uint8_t BSP_SDRAM_ReadData_DMA(uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize)
{
    if(HAL_SDRAM_Read_DMA(&SdramHandle, (uint32_t *)uwStartAddress, pData, uwDataSize) != HAL_OK)
    {
        return SDRAM_ERROR;
    }
    else
    {
        return SDRAM_OK;
    }
}

* @brief Writes an mount of data to the SDRAM memory in polling mode.
* @param uwStartAddress : Write start address
* @param pData : Pointer to data to be written
* @param uwDataSize: Size of written data from the memory
uint8_t BSP_SDRAM_WriteData(uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize) {  
  /* Disable write protection */  
  HAL_SDRAM_WriteProtection_Disable(&SdramHandle);  
  /*Write 32-bit data buffer to SDRAM memory */  
  if(HAL_SDRAM_Write_32b(&SdramHandle, (uint32_t *)uwStartAddress, pData, uwDataSize) != HAL_OK)  
    {  
      return SDRAM_ERROR;  
    }  
  else  
    {  
      return SDRAM_OK;  
    }
}

/**  
 @brief  Writes an mount of data to the SDRAM memory in DMA mode.  
 @param  uwStartAddress : Write start address  
 @param  pData : Pointer to data to be written  
 @param  uwDataSize: Size of written data from the memory  
 */  
uint8_t BSP_SDRAM_WriteData_DMA(uint32_t uwStartAddress, uint32_t *pData, uint32_t uwDataSize) {  
  if(HAL_SDRAM_Write_DMA(&SdramHandle, (uint32_t *)uwStartAddress, pData, uwDataSize) != HAL_OK)  
    {  
      return SDRAM_ERROR;  
    }  
  else  
    {  
      return SDRAM_OK;  
    }
32_t *)uwStartAddress, pData, uwDataSize) != HAL_OK)
    00282    {
    00283        return SDRAM_ERROR;
    00284    }
    00285    else
    00286    {
    00287        return SDRAM_OK;
    00288    }
    00289    }
    00290
    00291 /**
    00292    * @brief Sends command to the SDRAM bank.
    00293    * @param SdramCmd: Pointer to SDRAM command structure
    00294    * @retval HAL status
    00295    */
    00296    uint8_t BSP_SDRAM_Sendcmd(FMC_SDRAM_CommandTypeDef *SdramCmd)
    00297    {
    00298        if(HAL_SDRAM_SendCommand(&SdramHandle, SdramCmd, SDRAM_TIMEOUT) != HAL_OK)
    00299        {
    00300            return SDRAM_ERROR;
    00301        }
    00302        else
    00303        {
    00304            return SDRAM_OK;
    00305        }
    00306    }
    00307
    00308 /**
    00309    * @brief Handles SDRAM DMA transfer interrupt request.
    00310    */
    00311    void BSP_SDRAM_DMA_IRQHandler(void)
    00312    {
HAL_DMA_IRQHandler(SdramHandle.hdma);

/**
 * @brief Initializes SDRAM MSP.
 * @note This function can be surcharged by application code.
 * @param hsdram: pointer on SDRAM handle
 * @param Params: pointer on additional configuration parameters, can be NULL.
 */
__weak void BSP_SDRAM_MspInit(SDRAM_HandleTypeDef *hsdram, void *Params) {
    static DMA_HandleTypeDef dmaHandle;
    GPIO_InitTypeDef GPIO_InitStructure;

    if (hsdram != (SDRAM_HandleTypeDef *)NULL) {
        /* Enable FMC clock */
        __HAL_RCC_FMC_CLK_ENABLE();

        /* Enable chosen DMAx clock */
        __DMAx_CLK_ENABLE();

        /* Enable GPIOs clock */
        __HAL_RCC_GPIOB_CLK_ENABLE();
        __HAL_RCC_GPIOC_CLK_ENABLE();
        __HAL_RCC_GPIOD_CLK_ENABLE();
        __HAL_RCC_GPIOE_CLK_ENABLE();
        __HAL_RCC_GPIOF_CLK_ENABLE();
        __HAL_RCC_GPIOG_CLK_ENABLE();

        /*--- GPIOs Configuration ------------------------*/
    } /*--- GPIOs Configuration ------------------------*/

    /*--- GPIOs Configuration ------------------------*/
} /*--- GPIOs Configuration ------------------------*/
00346 | PD0 <-> FMC_D2 | PE0 <-> FMC_NBL0 |
PF0 <-> FMC_A0 | PG0 <-> FMC_A10 |
00349 | PD1 <-> FMC_D3 | PE1 <-> FMC_NBL1 |
PF1 <-> FMC_A1 | PG1 <-> FMC_A11 |
00350 | PD8 <-> FMC_D13 | PE7 <-> FMC_D4 |
PF0 <-> FMC_A2 | PG8 <-> FMC_SDCLK |
00351 | PD9 <-> FMC_D14 | PE8 <-> FMC_D5 |
PF3 <-> FMC_A3 | PG15 <-> FMC_NCAS |
00352 | PD10 <-> FMC_D15 | PE9 <-> FMC_D6 |
PF4 <-> FMC_A4 |
00353 | PD14 <-> FMC_D0 | PE10 <-> FMC_D7 |
PF5 <-> FMC_A5 |
00354 | PD15 <-> FMC_D1 | PE11 <-> FMC_D8 |
PF11 <-> FMC_NRAS |
00355 | PE12 <-> FMC_D9 |
PF12 <-> FMC_A6 |
00356 | PE13 <-> FMC_D10 |
PF13 <-> FMC_A7 |
00357 | PE14 <-> FMC_D11 |
PF14 <-> FMC_A8 |
00358 | PE15 <-> FMC_D12 |
PF15 <-> FMC_A9 |
00359 | PE16 <-> FMC_D13 |

00360 | PB5 <-> FMC_SDCKE1|
00361 | PB6 <-> FMC_SDNE1 |
00362 | PC0 <-> FMC_SDNWE |
00363 | PE17 <-> FMC_SDCLK |
00364
00355 */
00366
00367 /* 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_PIN_10 | GPIO_PIN_14 |
GPIO_PIN_15;
HAL_GPIO_Init(GPIOD, &GPIO_InitStructure);

/* GPIOE configuration */
GPIO_InitStructure.Pin = GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_7 |
GPIO_PIN_8 | GPIO_PIN_9 | GPIO_PIN_10 |
GPIO_PIN_11 | GPIO_PIN_12 | GPIO_PIN_13 |
GPIO_PIN_14 | GPIO_PIN_15;
HAL_GPIO_Init(GPIOE, &GPIO_InitStructure);
/* GPIOF configuration */
GPIO_InitStruct.Pin = GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5 | GPIO_PIN_11 | GPIO_PIN_12 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15;
HAL_GPIO_Init(GPIOF, &GPIO_InitStruct);

/* GPIOG configuration */
GPIO_InitStruct.Pin = GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_4 | GPIO_PIN_5 | GPIO_PIN_8 | GPIO_PIN_15;
HAL_GPIO_Init(GPIOG, &GPIO_InitStruct);

/* Configure common DMA parameters */
dmaHandle.Init.Channel = SDRAM_DMAx_CHANNEL;
dmaHandle.Init.Direction = DMA_MEMORY_TO_MEMORY;
dmaHandle.Init.PeriphInc = DMA_PERIPH_ENABLE;
dmaHandle.Init.MemInc = DMA_MEM_ENABLE;
dmaHandle.Init.PeriphDataAlignment = DMA_PERIPH_DATAALIGN_WORD;
dmaHandle.Init.MemDataAlignment = DMA_MEMDATAALIGN_WORD;
dmaHandle.Init.Mode = DMA_NORMAL;
dmaHandle.Init.Priority = DMA_PRIORITY_HIGH;
dmaHandle.Init.FIFOMode = DMA_FIFOMODE_DISABLE;
IFOMODE_DISABLE;
00416 dmaHandle.Init.FIFOThreshold = DMA_FIFO_THRESHOLD_FULL;
00417 dmaHandle.Init.MemBurst = DMA_MBURST_SINGLE;
00418 dmaHandle.Init.PeriphBurst = DMA_PBURST_SINGLE;
00419 dmaHandle.Instance = SDRAM_DMAx_STREAM;
00420 /* Associate the DMA handle */
00421 __HAL_LINKDMA(hsdram, hdma, dmaHandle);
00422 /* Deinitialize the stream for new transfer */
00423 HAL_DMA_DeInit(&dmaHandle);
00424 /* Configure the DMA stream */
00425 HAL_DMA_Init(&dmaHandle);
00426 /* NVIC configuration for DMA transfer complete interrupt */
00427 HAL_NVIC_SetPriority(SDRAM_DMAx_IRQn, 0x0F, 0);
00428 HAL_NVIC_EnableIRQ(SDRAM_DMAx_IRQn);
00429 } /* of if(hsdram != (SDRAM_HandleTypeDef *)NULL) */
00430 }
00431 * @brief DeInitializes SDRAM MSP.
00432 * @note This function can be surcharged by application code.
00433 * @param hsdram: pointer on SDRAM handle
00434 * @param Params: pointer on additional configuration parameters, can be NULL.
00435 * /
__weak void BSP_SDRAM_MspDeInit(SDRAM_HandleTypeDef *hsdram, void *Params)
{
    static DMA_HandleTypeDef dma_handle;
    if(hsdram != (SDRAM_HandleTypeDef *)NULL)
    {
        /* Disable NVIC configuration for DMA interrupt */
        HAL_NVIC_DisableIRQ(SDRAM_DMAx_IRQn);
        /* Deinitialize the stream for new transfer */
        dma_handle.Instance = SDRAM_DMAx_STREAM;
        HAL_DMA_DeInit(&dma_handle);
        /* DeInit GPIO pins can be done in the application
        (by surcharging this __weak function) */
        /* GPIO pins clock, FMC clock and DMA clock can be shut down in the application
        by surcharging this __weak function */
    }
    /* of if(hsdram != (SDRAM_HandleTypeDef *)NULL) */
}
/**
  */
# 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>Modules</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**BSP**
<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

BSP
### 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>

Generated on Fri Feb 17 2017 12:10:38 for STM32F429I-Discovery BSP User Manual by doxygen 1.7.6.1