

IndInf02_Ampelsteuerung_Weinb_5BHIT
1.0

Generated by Doxygen 1.8.10

Sun Nov 29 2015 17:33:05

Contents

1	Module Index	1
1.1	Modules	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Module Documentation	7
4.1	CMSIS	7
4.1.1	Detailed Description	7
4.2	Stm32f3xx_system	8
4.2.1	Detailed Description	8
4.3	STM32F3xx_System_Private_Includes	9
4.4	STM32F3xx_System_Private_TypesDefinitions	10
4.5	STM32F3xx_System_Private_Defines	11
4.5.1	Detailed Description	11
4.5.2	Macro Definition Documentation	11
4.5.2.1	HSE_VALUE	11
4.5.2.2	HSI_VALUE	11
4.5.2.3	VECT_TAB_OFFSET	11
4.6	STM32F3xx_System_Private_Macros	12
4.7	STM32F3xx_System_Private_Variables	13
4.7.1	Detailed Description	13
4.8	STM32F3xx_System_Private_FunctionPrototypes	14
4.9	STM32F3xx_System_Private_Functions	15
4.9.1	Detailed Description	15
4.9.2	Function Documentation	15
4.9.2.1	SystemCoreClockUpdate(void)	15
4.9.2.2	SystemInit(void)	16
5	Data Structure Documentation	17

5.1	traffic_light_data Struct Reference	17
6	File Documentation	19
6.1	src/leds.c File Reference	19
6.1.1	Detailed Description	20
6.1.2	Function Documentation	20
6.1.2.1	led_green()	20
6.1.2.2	led_green_blink()	20
6.1.2.3	led_init()	20
6.1.2.4	led_red()	21
6.1.2.5	led_red_yellow()	21
6.1.2.6	led_reset()	21
6.1.2.7	led_yellow()	21
6.1.2.8	led_yellow_blink()	21
6.2	src/leds.h File Reference	22
6.2.1	Detailed Description	22
6.2.2	Function Documentation	23
6.2.2.1	led_green()	23
6.2.2.2	led_green_blink()	23
6.2.2.3	led_init()	23
6.2.2.4	led_red()	23
6.2.2.5	led_red_yellow()	24
6.2.2.6	led_reset()	24
6.2.2.7	led_yellow()	24
6.2.2.8	led_yellow_blink()	24
6.3	src/main.c File Reference	25
6.3.1	Detailed Description	25
6.4	src/stm32f3xx_it.c File Reference	26
6.4.1	Detailed Description	26
6.4.2	Function Documentation	26
6.4.2.1	EXTI0_IRQHandler(void)	26
6.4.2.2	SysTick_Handler(void)	27
6.5	src/system_stm32f3xx.c File Reference	27
6.5.1	Detailed Description	28
6.6	src/traffic_light.h File Reference	29
6.6.1	Detailed Description	30
6.6.2	Function Documentation	30
6.6.2.1	traffic_light_control(traffic_light_data *traffic_light)	30
6.7	src/traffic_light_control.c File Reference	31
6.7.1	Detailed Description	31

6.7.2	Function Documentation	32
6.7.2.1	traffic_light_control(traffic_light_data *p_traffic_light)	32
Index		33

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

CMSIS	7
Stm32f3xx_system	8
STM32F3xx_System_Private_Includes	9
STM32F3xx_System_Private_TypesDefinitions	10
STM32F3xx_System_Private_Defines	11
STM32F3xx_System_Private_Macros	12
STM32F3xx_System_Private_Variables	13
STM32F3xx_System_Private_FunctionPrototypes	14
STM32F3xx_System_Private_Functions	15

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

traffic_light_data	17
------------------------------------	-------	----

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

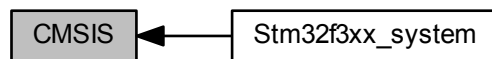
src/ leds.c	
Functions to control LEDs	19
src/ leds.h	
Definition of LED functions	22
src/ main.c	
Default main function	25
src/ stm32f3xx_it.c	
Default Interrupt Service Routines	26
src/ system_stm32f3xx.c	
CMSIS Cortex-M4 Device Peripheral Access Layer System Source File	27
src/ traffic_light.h	
Definition of the traffic lights (including states and events)	29
src/ traffic_light_control.c	
Event centric state machine to control the traffic light	31

Chapter 4

Module Documentation

4.1 CMSIS

Collaboration diagram for CMSIS:



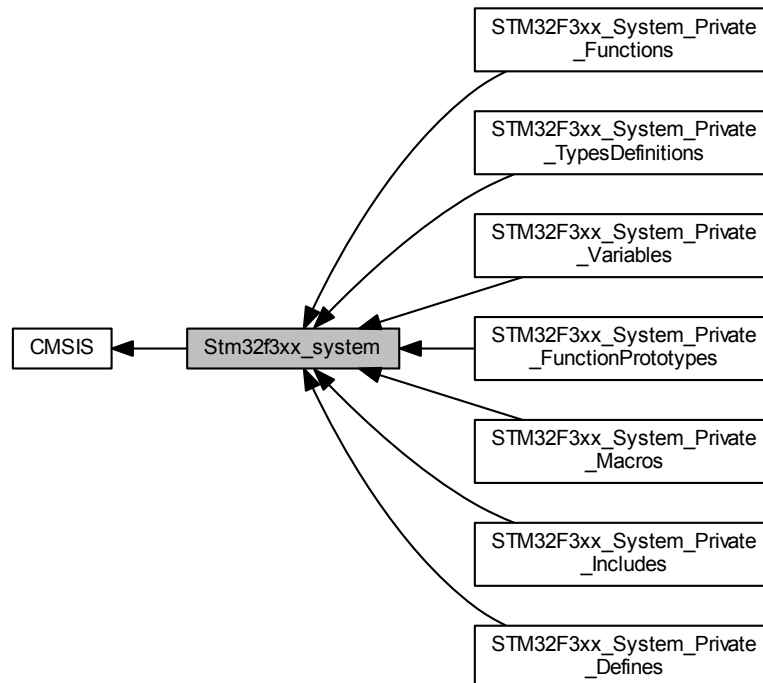
Modules

- [Stm32f3xx_system](#)

4.1.1 Detailed Description

4.2 Stm32f3xx_system

Collaboration diagram for Stm32f3xx_system:



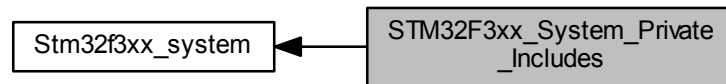
Modules

- [STM32F3xx_System_Private_Includes](#)
- [STM32F3xx_System_Private_TypesDefinitions](#)
- [STM32F3xx_System_Private_Defines](#)
- [STM32F3xx_System_Private_Macros](#)
- [STM32F3xx_System_Private_Variables](#)
- [STM32F3xx_System_Private_FunctionPrototypes](#)
- [STM32F3xx_System_Private_Functions](#)

4.2.1 Detailed Description

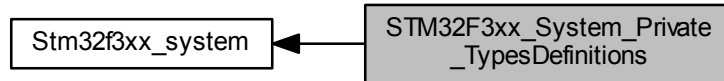
4.3 STM32F3xx_System_Private_Includes

Collaboration diagram for STM32F3xx_System_Private_Includes:



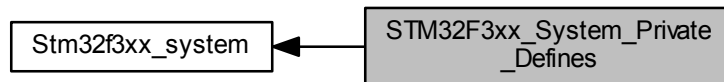
4.4 STM32F3xx_System_Private_TypesDefinitions

Collaboration diagram for STM32F3xx_System_Private_TypesDefinitions:



4.5 STM32F3xx_System_Private_Defines

Collaboration diagram for STM32F3xx_System_Private_Defines:



Macros

- `#define HSE_VALUE ((uint32_t)8000000)`
- `#define HSI_VALUE ((uint32_t)8000000)`
- `#define VECT_TAB_OFFSET 0x0`

4.5.1 Detailed Description

4.5.2 Macro Definition Documentation

4.5.2.1 `#define HSE_VALUE ((uint32_t)8000000)`

Default value of the External oscillator in Hz. This value can be provided and adapted by the user application.

4.5.2.2 `#define HSI_VALUE ((uint32_t)8000000)`

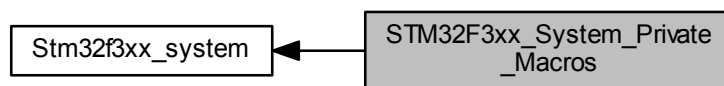
Default value of the Internal oscillator in Hz. This value can be provided and adapted by the user application.

4.5.2.3 `#define VECT_TAB_OFFSET 0x0`

< Uncomment the following line if you need to relocate your vector Table in Internal SRAM. Vector Table base offset field. This value must be a multiple of 0x200.

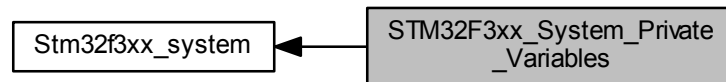
4.6 STM32F3xx_System_Private_Macros

Collaboration diagram for STM32F3xx_System_Private_Macros:



4.7 STM32F3xx_System_Private_Variables

Collaboration diagram for STM32F3xx_System_Private_Variables:



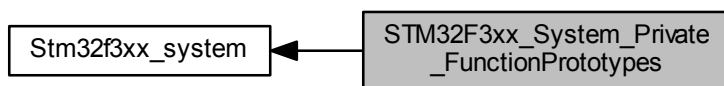
Variables

- uint32_t **SystemCoreClock** = 8000000
- __IO const uint8_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}

4.7.1 Detailed Description

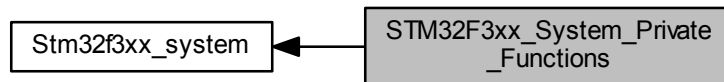
4.8 STM32F3xx_System_Private_FunctionPrototypes

Collaboration diagram for STM32F3xx_System_Private_FunctionPrototypes:



4.9 STM32F3xx_System_Private_Functions

Collaboration diagram for STM32F3xx_System_Private_Functions:



Functions

- void [SystemInit](#) (void)
Setup the microcontroller system Initialize the FPU setting, vector table location and the PLL configuration is reset.
- void [SystemCoreClockUpdate](#) (void)
Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

4.9.1 Detailed Description

4.9.2 Function Documentation

4.9.2.1 void SystemCoreClockUpdate (void)

Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

Note

Each time the core clock (HCLK) changes, this function must be called to update SystemCoreClock variable value. Otherwise, any configuration based on this variable will be incorrect.

- The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:

- If SYSCLK source is HSI, SystemCoreClock will contain the [HSI_VALUE\(*\)](#)
- If SYSCLK source is HSE, SystemCoreClock will contain the [HSE_VALUE\(**\)](#)
- If SYSCLK source is PLL, SystemCoreClock will contain the [HSE_VALUE\(**\)](#) or [HSI_VALUE\(*\)](#) multiplied/divided by the PLL factors.

(*) HSI_VALUE is a constant defined in stm32f3xx_hal.h file (default value 8 MHz) but the real value may vary depending on the variations in voltage and temperature.

(**) HSE_VALUE is a constant defined in stm32f3xx_hal.h file (default value 8 MHz), user has to ensure that HSE_VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.

- The result of this function could be not correct when using fractional value for HSE crystal.

Parameters

<i>None</i>	
-------------	--

Return values

<i>None</i>	
-------------	--

4.9.2.2 void SystemInit (void)

Setup the microcontroller system Initialize the FPU setting, vector table location and the PLL configuration is reset.

Parameters

<i>None</i>	
-------------	--

Return values

<i>None</i>	
-------------	--

Chapter 5

Data Structure Documentation

5.1 traffic_light_data Struct Reference

Data Fields

- traffic_state **state**
- traffic_event **event**

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

- src/[traffic_light.h](#)

Chapter 6

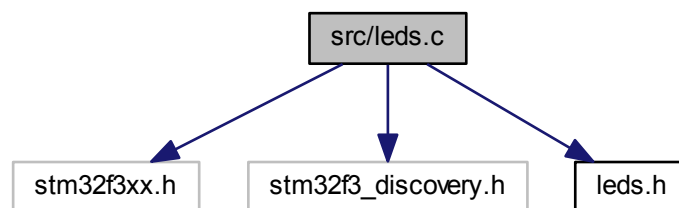
File Documentation

6.1 src/leds.c File Reference

Functions to control LEDs.

```
#include "stm32f3xx.h"
#include "stm32f3_discovery.h"
#include "leds.h"
```

Include dependency graph for leds.c:



Functions

- void `led_init()`
This function initializes the LEDs of the board.
- void `led_reset()`
This function switches all LEDs (red, orange, green) off.
- void `led_red()`
This function switches on the red LED for 4s.
- void `led_red_yellow()`
This function switches on the red & yellow LED for 2s.
- void `led_green()`
This function switches on the green LED for 2s.
- void `led_green_blink()`
This function switches on/off the green LED 4 times for 0,5s.
- void `led_yellow()`

This function switches on the orange LED for 2s.

- void `led_yellow_blink` ()

This function switches on/off the yellow led twice for 0,5s.

6.1.1 Detailed Description

Functions to control LEDs.

Author

Mathias Ritter

Version

V1.0

Date

13-November-2015

6.1.2 Function Documentation

6.1.2.1 void `led_green` ()

This function switches on the green LED for 2s.

Parameters

<i>none</i>	
-------------	--

Return values

--	--

6.1.2.2 void `led_green_blink` ()

This function switches on/off the green LED 4 times for 0,5s.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.1.2.3 void `led_init` ()

This function initializes the LEDs of the board.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.1.2.4 void led_red ()

This function switches on the red LED for 4s.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.1.2.5 void led_red_yellow ()

This function switches on the red & yellow LED for 2s.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.1.2.6 void led_reset ()

This function switches all LEDs (red, orange, green) off.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.1.2.7 void led_yellow ()

This function switches on the orange LED for 2s.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.1.2.8 void led_yellow_blink ()

This function switches on/off the yellow led twice for 0,5s.

Parameters

<i>none</i>	
-------------	--

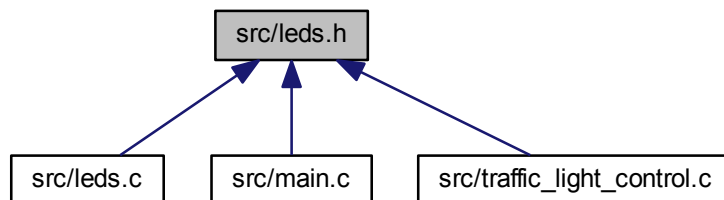
Return values

<i>none</i>	
-------------	--

6.2 src/leds.h File Reference

Definition of LED functions.

This graph shows which files directly or indirectly include this file:



Functions

- void [led_init](#) ()
This function initializes the LEDs of the board.
- void [led_reset](#) ()
This function switches all LEDs (red, orange, green) off.
- void [led_red](#) ()
This function switches on the red LED for 4s.
- void [led_red_yellow](#) ()
This function switches on the red & yellow LED for 2s.
- void [led_green](#) ()
This function switches on the green LED for 2s.
- void [led_green_blink](#) ()
This function switches on/off the green LED 4 times for 0,5s.
- void [led_yellow](#) ()
This function switches on the orange LED for 2s.
- void [led_yellow_blink](#) ()
This function switches on/off the yellow led twice for 0,5s.

6.2.1 Detailed Description

Definition of LED functions.

Author

Mathias Ritter

Version

V1.0

Date

13-November-2015

6.2.2 Function Documentation

6.2.2.1 void led_green ()

This function switches on the green LED for 2s.

Parameters

<i>none</i>	
-------------	--

Return values

--	--

6.2.2.2 void led_green_blink ()

This function switches on/off the green LED 4 times for 0,5s.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.2.2.3 void led_init ()

This function initializes the LEDs of the board.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.2.2.4 void led_red ()

This function switches on the red LED for 4s.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.2.2.5 void led_red_yellow ()

This function switches on the red & yellow LED for 2s.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.2.2.6 void led_reset ()

This function switches all LEDs (red, orange, green) off.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.2.2.7 void led_yellow ()

This function switches on the orange LED for 2s.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

6.2.2.8 void led_yellow_blink ()

This function switches on/off the yellow led twice for 0,5s.

Parameters

<i>none</i>	
-------------	--

Return values

--

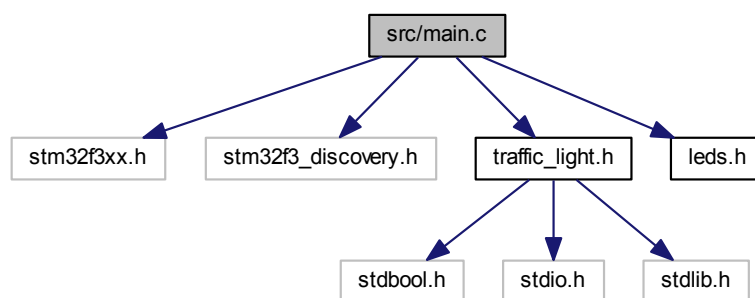
none

6.3 src/main.c File Reference

Default main function.

```
#include "stm32f3xx.h"
#include "stm32f3_discovery.h"
#include "traffic_light.h"
#include "leds.h"
```

Include dependency graph for main.c:



Functions

- `int main` (void)

Variables

- `traffic_light_data` `traffic_light`

6.3.1 Detailed Description

Default main function.

Author

Mathias Ritter

Version

V1.0

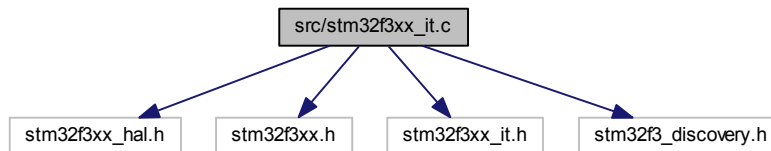
Date

13-November-2015

6.4 src/stm32f3xx_it.c File Reference

Default Interrupt Service Routines.

```
#include "stm32f3xx_hal.h"
#include "stm32f3xx.h"
#include "stm32f3xx_it.h"
#include "stm32f3_discovery.h"
Include dependency graph for stm32f3xx_it.c:
```



Functions

- void [SysTick_Handler](#) (void)
This function handles SysTick Handler.
- void [EXTI0_IRQHandler](#) (void)
This function handles External Interrupt Handler.

6.4.1 Detailed Description

Default Interrupt Service Routines.

Author

Ac6

Version

V1.0

Date

02-Feb-2015

6.4.2 Function Documentation

6.4.2.1 void EXTI0_IRQHandler (void)

This function handles External Interrupt Handler.

Parameters

None	
------	--

Return values

None	
------	--

6.4.2.2 void SysTick_Handler (void)

This function handles SysTick Handler.

Parameters

None	
------	--

Return values

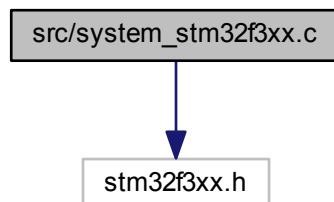
None	
------	--

6.5 src/system_stm32f3xx.c File Reference

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File.

```
#include "stm32f3xx.h"
```

Include dependency graph for system_stm32f3xx.c:



Macros

- #define [HSE_VALUE](#) ((uint32_t)8000000)
- #define [HSI_VALUE](#) ((uint32_t)8000000)
- #define [VECT_TAB_OFFSET](#) 0x0

Functions

- void [SystemInit](#) (void)
Setup the microcontroller system Initialize the FPU setting, vector table location and the PLL configuration is reset.
- void [SystemCoreClockUpdate](#) (void)
Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

Variables

- uint32_t **SystemCoreClock** = 8000000
- __IO const uint8_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}

6.5.1 Detailed Description

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File.

Author

MCD Application Team

Version

V1.2.0

Date

19-June-2015

1. This file provides two functions and one global variable to be called from user application:
 - [SystemInit\(\)](#): This function is called at startup just after reset and before branch to main program. This call is made inside the "startup_stm32f3xx.s" file.
 - SystemCoreClock variable: Contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.
 - [SystemCoreClockUpdate\(\)](#): Updates the variable SystemCoreClock and must be called whenever the core clock is changed during program execution.
2. After each device reset the HSI (8 MHz) is used as system clock source. Then [SystemInit\(\)](#) function is called, in "startup_stm32f3xx.s" file, to configure the system clock before to branch to main program.

3. This file configures the system clock as follows:

Supported STM32F3xx device

System Clock source | HSI

SYSCLK(Hz) | 8000000

HCLK(Hz) | 8000000

AHB Prescaler | 1

APB2 Prescaler | 1

APB1 Prescaler | 1

USB Clock | DISABLE

=====

Attention

© COPYRIGHT(c) 2015 STMicroelectronics

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

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

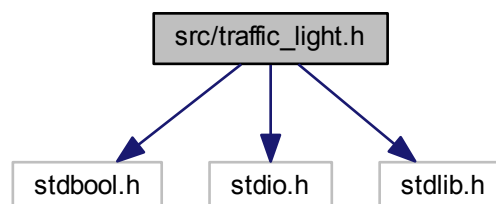
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE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

6.6 src/traffic_light.h File Reference

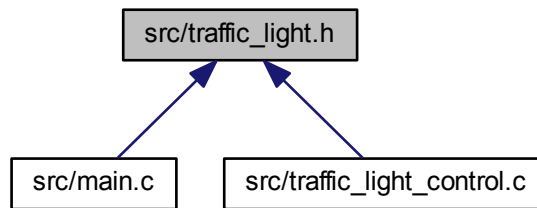
Definition of the traffic lights (including states and events)

```
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
```

Include dependency graph for traffic_light.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [traffic_light_data](#)

Enumerations

- enum **traffic_state** {
 RED, RED_YELLOW, GREEN, GREEN_BLINK,
 YELLOW, YELLOW_BLINK }
- enum **traffic_event** {
 STOP, PREPARE_GO, GO, PREPARE_CAUTION,
 CAUTION, FAULT }

Functions

- void [traffic_light_control](#) ([traffic_light_data](#) *`traffic_light`)
 This function represents a event centric state machine to control the traffic light.

6.6.1 Detailed Description

Definition of the traffic lights (including states and events)

Author

Mathias Ritter

Version

V1.0

Date

13-November-2015

6.6.2 Function Documentation

6.6.2.1 void `traffic_light_control` (`traffic_light_data` * `p_traffic_light`)

This function represents a event centric state machine to control the traffic light.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

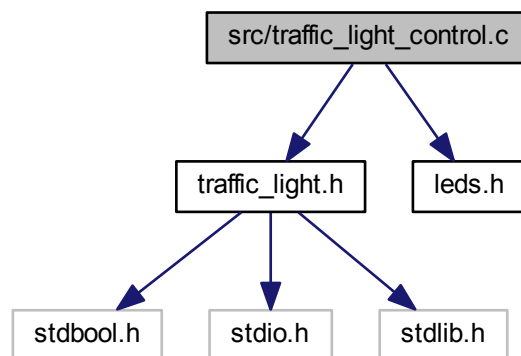
6.7 src/traffic_light_control.c File Reference

event centric state machine to control the traffic light

```
#include "traffic_light.h"
```

```
#include "leds.h"
```

Include dependency graph for traffic_light_control.c:



Functions

- void [traffic_light_control](#) ([traffic_light_data](#) *p_traffic_light)

This function represents a event centric state machine to control the traffic light.

6.7.1 Detailed Description

event centric state machine to control the traffic light

Author

Mathias Ritter

Version

V1.0

Date

13-November-2015

6.7.2 Function Documentation

6.7.2.1 void traffic_light_control (traffic_light_data * p_traffic_light)

This function represents a event centric state machine to control the traffic light.

Parameters

<i>none</i>	
-------------	--

Return values

<i>none</i>	
-------------	--

Index

CMSIS, [7](#)

EXTI0_IRQHandler
stm32f3xx_it.c, [26](#)

HSE_VALUE
STM32F3xx_System_Private_Defines, [11](#)
HSI_VALUE
STM32F3xx_System_Private_Defines, [11](#)

led_green
leds.c, [20](#)
leds.h, [23](#)
led_green_blink
leds.c, [20](#)
leds.h, [23](#)

led_init
leds.c, [20](#)
leds.h, [23](#)

led_red
leds.c, [21](#)
leds.h, [23](#)

led_red_yellow
leds.c, [21](#)
leds.h, [24](#)

led_reset
leds.c, [21](#)
leds.h, [24](#)

led_yellow
leds.c, [21](#)
leds.h, [24](#)

led_yellow_blink
leds.c, [21](#)
leds.h, [24](#)

leds.c
led_green, [20](#)
led_green_blink, [20](#)
led_init, [20](#)
led_red, [21](#)
led_red_yellow, [21](#)
led_reset, [21](#)
led_yellow, [21](#)
led_yellow_blink, [21](#)

leds.h
led_green, [23](#)
led_green_blink, [23](#)
led_init, [23](#)
led_red, [23](#)
led_red_yellow, [24](#)
led_reset, [24](#)

led_yellow, [24](#)
led_yellow_blink, [24](#)

STM32F3xx_System_Private_Defines, [11](#)
HSE_VALUE, [11](#)
HSI_VALUE, [11](#)
VECT_TAB_OFFSET, [11](#)
STM32F3xx_System_Private_FunctionPrototypes, [14](#)
STM32F3xx_System_Private_Functions, [15](#)
SystemCoreClockUpdate, [15](#)
SystemInit, [16](#)

STM32F3xx_System_Private_Includes, [9](#)
STM32F3xx_System_Private_Macros, [12](#)
STM32F3xx_System_Private_TypesDefinitions, [10](#)
STM32F3xx_System_Private_Variables, [13](#)
src/leds.c, [19](#)
src/leds.h, [22](#)
src/main.c, [25](#)
src/stm32f3xx_it.c, [26](#)
src/system_stm32f3xx.c, [27](#)
src/traffic_light.h, [29](#)
src/traffic_light_control.c, [31](#)
stm32f3xx_it.c
EXTI0_IRQHandler, [26](#)
SysTick_Handler, [27](#)

Stm32f3xx_system, [8](#)
SysTick_Handler
stm32f3xx_it.c, [27](#)
SystemCoreClockUpdate
STM32F3xx_System_Private_Functions, [15](#)
SystemInit
STM32F3xx_System_Private_Functions, [16](#)

traffic_light.h
traffic_light_control, [30](#)
traffic_light_control
traffic_light.h, [30](#)
traffic_light_control.c, [32](#)
traffic_light_control.c
traffic_light_control, [32](#)
traffic_light_data, [17](#)

VECT_TAB_OFFSET
STM32F3xx_System_Private_Defines, [11](#)