ADAS Graduation Project

Generated on Mon Nov 28 2022 19:04:42 for ADAS Graduation Project by Doxygen 1.9.5

Mon Nov 28 2022 19:04:42

Chapter 1

ADAS Graudation Project

1.1 Description

The graduation project is ADAS and consists of 3 systems:

1. Adaptive Cruise Control
2. Adaptive Light Control
3. Driver Health Care

1.1.1 Adaptive Cruise Control

Adaptive cruise control (ACC) is an enhancement of conventional cruise control. ACC automatically adjusts the speed of your car to match the speed of the car in front of you. If the car ahead slows down, ACC can automatically match it.

1.1.2 Adaptive Light Control

Adaptive light control is a system that changes the brightness of the car's headlights automatically based on the environment's light as well as the cars that are coming in the opposite way. The lighter the environment, the darker the headlights.

1.1.3 Driver Health Care

Driver health care is a system that is responsible for the detection of the driver's sleepiness and give the suitable alters to warn the driver as well as the other cars in real-time using Image Processing and Machine Learning techniques.

1.2 Hardware specifications

This project is developed using:

- Micro controller: `STM32F401CCU6`

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Module Documentation

5.1 Bit Manipulation Math Macros

Macros

- #define SET_BIT(Reg, bitnum) (Reg) |= (1 << (bitnum))
- #define CLR_BIT(Reg, bitnum) (Reg) &= ~(1 << (bitnum))
- #define TOGGLE_BIT(Reg, bitnum) (Reg) ^= (1 << (bitnum))
- #define GET_BIT(Reg, bitnum) (((Reg)>>(bitnum)) & 1)

5.1.1 Detailed Description

5.1.2 Macro Definition Documentation

5.1.2.1 SET_BIT

```
#define SET_BIT(
              Reg,
              bitnum ) (Reg) |= (1 << (bitnum))
```

```
#include <src/COTS/LIB/LSTD_BITMATH.h>
```

Sets a certain bit's value

Definition at line 23 of file LSTD_BITMATH.h.

5.1.2.2 CLR_BIT

```
#define CLR_BIT(
              Reg,
              bitnum ) (Reg) &= ~(1 << (bitnum))
```

```
#include <src/COTS/LIB/LSTD_BITMATH.h>
```

Clears a certain bit's value to

Definition at line 30 of file LSTD_BITMATH.h.

5.1.2.3 TOGGLE_BIT

```
#define TOGGLE_BIT(
              Reg,
              bitnum ) (Reg) ^= (1 << (bitnum))
```

```
#include <src/COTS/LIB/LSTD_BITMATH.h>
```

Toggle a bit to `0` if it's `1`, `1` otherwise

Definition at line 37 of file LSTD_BITMATH.h.

5.1.2.4 GET_BIT

```
#define GET_BIT(
              Reg,
              bitnum ) (((Reg)>>(bitnum)) & 1)
```

```
#include <src/COTS/LIB/LSTD_BITMATH.h>
```

Returns the value of the bit whether it's `1` or `0`

Definition at line 44 of file LSTD_BITMATH.h.

5.2 Bit Group Manipulation Math Macros

Macros

- #define SET_BITs(Reg, bits, bitnum, factor) (Reg) |= ((bits) << (bitnum * factor))
- #define CLR_BITs(Reg, bits, bitnum, factor) (Reg) &= ~((bits) << (bitnum * factor))
- #define TOGGLE_BITs(Reg, bits, bitnum, factor) (Reg) ^= ((bits) << (bitnum * factor))
- #define GET_BITs(Reg, bits, bitnum, factor) (((Reg) >> (bitnum * factor)) & (bits))

5.2.1 Detailed Description

5.2.2 Macro Definition Documentation

5.2.2.1 SET_BITs

```
#define SET_BITs(
             Reg,
             bits,
             bitnum,
             factor ) (Reg) |= ((bits) << (bitnum * factor))
```

```
#include <src/COTS/LIB/LSTD_BITMATH.h>
```

Sets the values of a group of bits

Definition at line 59 of file LSTD_BITMATH.h.

5.2.2.2 CLR_BITs

```
#define CLR_BITs(
             Reg,
             bits,
             bitnum,
             factor ) (Reg) &= ~((bits) << (bitnum * factor))
```

```
#include <src/COTS/LIB/LSTD_BITMATH.h>
```

Clears the value of a group of bits

Definition at line 66 of file LSTD_BITMATH.h.

5.2.2.3 TOGGLE_BITs

```
#define TOGGLE_BITs(
             Reg,
             bits,
             bitnum,
             factor ) (Reg) ^= ((bits) << (bitnum * factor))
```

```
#include <src/COTS/LIB/LSTD_BITMATH.h>
```

Toggles the value of a group of bits

Definition at line 73 of file LSTD_BITMATH.h.

5.2.2.4 GET_BITs

```
#define GET_BITs(
              Reg,
              bits,
              bitnum,
              factor ) (((Reg) >> (bitnum * factor)) & (bits))
```

```
#include <src/COTS/LIB/LSTD_BITMATH.h>
```

Returns the value of a group of bits

Definition at line 80 of file LSTD_BITMATH.h.

5.3 Compiler standard macros

Macros

- #define VAR(vartype) vartype
- #define FUNC(rettype) rettype
- #define P2VAR(ptrtype) ptrtype *
- #define P2CONST(ptrtype) const ptrtype *
- #define CONSTP2VAR(ptrtype) ptrtype * const
- #define CONSTP2CONST(ptrtype) const ptrtype * const
- #define P2FUNC(rettype, fctname) rettype (* fctname)
- #define CONST(consttype) const consttype
- #define STATIC static

5.3.1 Detailed Description

5.3.2 Macro Definition Documentation

5.3.2.1 VAR

```
#define VAR(
              vartype ) vartype
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a variable with the specified type

Definition at line 23 of file LSTD_COMPILER.h.

5.3.2.2 FUNC

```
#define FUNC(
             rettype ) rettype
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a function's return type

Definition at line 30 of file LSTD_COMPILER.h.

5.3.2.3 P2VAR

```
#define P2VAR(
             ptrtype ) ptrtype *
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a pointer-to-variable with the specified type

Definition at line 37 of file LSTD_COMPILER.h.

5.3.2.4 P2CONST

```
#define P2CONST(
             ptrtype ) const ptrtype *
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a constant pointer-to-variable with the specified type

Definition at line 44 of file LSTD_COMPILER.h.

5.3.2.5 CONSTP2VAR

```
#define CONSTP2VAR(
             ptrtype ) ptrtype * const
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a pointer-to-variable constant with the specified type

Definition at line 51 of file LSTD_COMPILER.h.

5.3.2.6 CONSTP2CONST

```
#define CONSTP2CONST(
             ptrtype ) const ptrtype * const
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a constant pointer-to-variable constant with the specified type

Definition at line 58 of file LSTD_COMPILER.h.

5.3.2.7 P2FUNC

```
#define P2FUNC(
             rettype,
             fctname ) rettype (* fctname)
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a pointer-to-function with the specified type

Definition at line 65 of file LSTD_COMPILER.h.

5.3.2.8 CONST

```
#define CONST(
             consttype ) const consttype
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a standard constant variable with the specified type

Definition at line 73 of file LSTD_COMPILER.h.

5.3.2.9 STATIC

```
#define STATIC static
```

```
#include <src/COTS/LIB/LSTD_COMPILER.h>
```

Declare a standard static variable

Definition at line 82 of file LSTD_COMPILER.h.

5.4 Utilities macros

Macros

- #define MY_MS_DELAY(T) do{ u32 Timer = (T ∗ 500); while (Timer--) { asm ("nop"); } } while(0);

5.4.1 Detailed Description

5.4.2 Macro Definition Documentation

5.4.2.1 MY_MS_DELAY

```
#define MY_MS_DELAY(
              T ) do{ u32 Timer = (T * 500); while (Timer--) { asm ("nop"); } } while(0);
```

```
#include <src/COTS/LIB/LSTD_MCU_UTILITIES.h>
```

Declare a delay with a `T` microseconds with a NOP delay

Definition at line 23 of file LSTD_MCU_UTILITIES.h.

5.5 Standard types

Typedefs

- typedef unsigned char bool_t
- typedef unsigned char u8_t
- typedef unsigned short int u16_t
- typedef unsigned long int u32_t
- typedef signed char s8_t
- typedef signed short int s16_t
- typedef signed long int s32_t
- typedef float fl32_t
- typedef double fl64_t

5.5.1 Detailed Description

5.5.2 Typedef Documentation

5.5.2.1 bool_t

bool_t

#include <src/COTS/LIB/LSTD_TYPES.h>

Type definition for boolean

Definition at line 23 of file LSTD_TYPES.h.

5.5.2.2 u8_t

u8_t

#include <src/COTS/LIB/LSTD_TYPES.h>

Type definition for 8-bit unsigned INT

Definition at line 30 of file LSTD_TYPES.h.

5.5.2.3 u16_t

u16_t

#include <src/COTS/LIB/LSTD_TYPES.h>

Type definition for 16-bit unsigned INT

Definition at line 37 of file LSTD_TYPES.h.

5.5.2.4 u32_t

u32_t

#include <src/COTS/LIB/LSTD_TYPES.h>

Type definition for 32-bit unsigned INT

Definition at line 44 of file LSTD_TYPES.h.

5.5.2.5 s8_t

s8_t

#include <src/COTS/LIB/LSTD_TYPES.h>

Type definition for 8-bit signed INT

Definition at line 51 of file LSTD_TYPES.h.

5.5.2.6 s16_t

s16_t

#include <src/COTS/LIB/LSTD_TYPES.h>

Type definition for 16-bit signed INT

Definition at line 58 of file LSTD_TYPES.h.

5.5.2.7 s32_t

s32_t

#include <src/COTS/LIB/LSTD_TYPES.h>

Type definition for 32-bit signed INT

Definition at line 65 of file LSTD_TYPES.h.

5.5.2.8 fl32_t

fl32_t

#include <src/COTS/LIB/LSTD_TYPES.h>

Type definition for 32-bit float

Definition at line 72 of file LSTD_TYPES.h.

5.5.2.9 fl64_t

```
fl64_t
```

```
#include <src/COTS/LIB/LSTD_TYPES.h>
```

Type definition for 64-bit float

Definition at line 79 of file LSTD_TYPES.h.

5.6 Standard values

Macros

- #define TRUE (1)
- #define FALSE (0)
- #define NULL (P2VAR(void) 0)
- #define INITIAL_ZERO (0)
- #define FLAG_SET (1)
- #define FLAG_CLEARED (0)

5.6.1 Detailed Description

5.6.2 Macro Definition Documentation

5.6.2.1 TRUE

```
#define TRUE (1)
```

```
#include <src/COTS/LIB/LSTD_VALUES.h>
```

Type definition for TRUE

Definition at line 24 of file LSTD_VALUES.h.

5.6.2.2 FALSE

```
#define FALSE (0)
```

```
#include <src/COTS/LIB/LSTD_VALUES.h>
```

Type definition for FALSE

Definition at line 33 of file LSTD_VALUES.h.

5.6.2.3 NULL

```
#define NULL (P2VAR(void) 0)
```

```
#include <src/COTS/LIB/LSTD_VALUES.h>
```

Type definition for NULL

Definition at line 42 of file LSTD_VALUES.h.

5.6.2.4 INITIAL_ZERO

```
#define INITIAL_ZERO (0)
```

```
#include <src/COTS/LIB/LSTD_VALUES.h>
```

Type definition for INITIAL_ZERO

Definition at line 51 of file LSTD_VALUES.h.

5.6.2.5 FLAG_SET

```
#define FLAG_SET (1)
```

```
#include <src/COTS/LIB/LSTD_VALUES.h>
```

Type definition for FLAG_SET

Definition at line 60 of file LSTD_VALUES.h.

5.6.2.6 FLAG_CLEARED

```
#define FLAG_CLEARED (0)
```

```
#include <src/COTS/LIB/LSTD_VALUES.h>
```

Type definition for FLAG_CLEARED

Definition at line 69 of file LSTD_VALUES.h.

5.7 GPIO Modes

Macros

- #define GPIOx_MODE_INPUT (0b00)

  Control input mode.
- #define GPIOx_MODE_OUTPUT (0b01)

  Control output mode.
- #define GPIOx_MODE_AF (0b10)

  Control alternate function mode.
- #define GPIOx_MODE_ANALOG (0b11)

  Control analog mode.

5.7.1 Detailed Description

5.7.2 Macro Definition Documentation

5.7.2.1 GPIOx_MODE_INPUT

```
#define GPIOx_MODE_INPUT (0b00)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

Control input mode.

Definition at line 158 of file GPIO_interface.h.

5.7.2.2 GPIOx_MODE_OUTPUT

```
#define GPIOx_MODE_OUTPUT (0b01)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

Control output mode.

Definition at line 165 of file GPIO_interface.h.

5.7.2.3 GPIOx_MODE_AF

```
#define GPIOx_MODE_AF (0b10)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

Control alternate function mode.

Definition at line 172 of file GPIO_interface.h.

5.7.2.4 GPIOx_MODE_ANALOG

```
#define GPIOx_MODE_ANALOG (0b11)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

Control analog mode.

Definition at line 179 of file GPIO_interface.h.

5.8 GPIO Ports

Macros

- #define GPIO_PORTA (0)
  GPIO Port A.
- #define GPIO_PORTB (1)
  GPIO Port B.
- #define GPIO_PORTC (2)
  GPIO Port C.

5.8.1 Detailed Description

5.8.2 Macro Definition Documentation

5.8.2.1 GPIO_PORTA

```
#define GPIO_PORTA (0)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Port A.

Definition at line 194 of file GPIO_interface.h.

5.8.2.2 GPIO_PORTB

```
#define GPIO_PORTB (1)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Port B.

Definition at line 201 of file GPIO_interface.h.

5.8.2.3 GPIO_PORTC

```
#define GPIO_PORTC (2)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Port C.

Definition at line 208 of file GPIO_interface.h.

5.9 GPIO Output Types

Macros

- #define GPIOx_OPENDRAIN (1)
  GPIO open-drain.
- #define GPIOx_PUSHPULL (2)
  GPIO push-pull.

5.9.1 Detailed Description

5.9.2 Macro Definition Documentation

5.9.2.1 GPIOx_OPENDRAIN

```
#define GPIOx_OPENDRAIN (1)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO open-drain.

Definition at line 223 of file GPIO_interface.h.

5.9.2.2 GPIOx_PUSHPULL

```
#define GPIOx_PUSHPULL (2)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO push-pull.

Definition at line 230 of file GPIO_interface.h.

5.10 GPIO PIN Speed

Macros

- #define GPIOx_LowSpeed (0b00)
  GPIO low speed.
- #define GPIOx_MediumSpeed (0b01)
  GPIO medium speed.
- #define GPIOx_HighSpeed (0b10)
  GPIO high speed.
- #define GPIOx_VeryHighSpeed (0b11)
  GPIO very high speed.

5.10.1 Detailed Description

5.10.2 Macro Definition Documentation

5.10.2.1 GPIOx_LowSpeed

```
#define GPIOx_LowSpeed (0b00)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO low speed.

Definition at line 245 of file GPIO_interface.h.

5.10.2.2 GPIOx_MediumSpeed

```
#define GPIOx_MediumSpeed (0b01)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO medium speed.

Definition at line 252 of file GPIO_interface.h.

5.10.2.3 GPIOx_HighSpeed

```
#define GPIOx_HighSpeed (0b10)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO high speed.

Definition at line 259 of file GPIO_interface.h.

5.10.2.4 GPIOx_VeryHighSpeed

```
#define GPIOx_VeryHighSpeed (0b11)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO very high speed.

Definition at line 266 of file GPIO_interface.h.

5.11 GPIO Pull Types

Macros

- #define GPIOx_NoPull (0b00)
  GPIO No PULL.
- #define GPIOx_PullUp (0b01)
  GPIO Pull UP.
- #define GPIOx_PullDown (0b10)
  GPIO Pull Down.

5.11.1 Detailed Description

5.11.2 Macro Definition Documentation

5.11.2.1 GPIOx_NoPull

```
#define GPIOx_NoPull (0b00)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO No PULL.

Definition at line 281 of file GPIO_interface.h.

5.11.2.2 GPIOx_PullUp

```
#define GPIOx_PullUp (0b01)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Pull UP.

Definition at line 288 of file GPIO_interface.h.

5.11.2.3 GPIOx_PullDown

```
#define GPIOx_PullDown (0b10)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Pull Down.

Definition at line 295 of file GPIO_interface.h.

5.12 GPIO Output Values

Macros

- #define GPIOx_HIGH (1)
  GPIO output high.
- #define GPIOx_LOW (2)
  GPIO output low.

5.12.1 Detailed Description

5.12.2 Macro Definition Documentation

5.12.2.1 GPIOx_HIGH

```
#define GPIOx_HIGH (1)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO output high.

Definition at line 310 of file GPIO_interface.h.

5.12.2.2 GPIOx_LOW

```
#define GPIOx_LOW (2)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO output low.

Definition at line 317 of file GPIO_interface.h.

5.13 GPIO Output PINs

Macros

- #define GPIOx_PIN0 (0)
  GPIO PIN 0.
- #define GPIOx_PIN1 (1)
  GPIO PIN 1.
- #define GPIOx_PIN2 (2)
  GPIO PIN 2.
- #define GPIOx_PIN3 (3)
  GPIO PIN 3.
- #define GPIOx_PIN4 (4)
  GPIO PIN 4.
- #define GPIOx_PIN5 (5)
  GPIO PIN 5.
- #define GPIOx_PIN6 (6)
  GPIO PIN 6.
- #define GPIOx_PIN7 (7)
  GPIO PIN 7.
- #define GPIOx_PIN8 (8)
  GPIO PIN 8.
- #define GPIOx_PIN9 (9)
  brief GPIO PIN 9
- #define GPIOx_PIN10 (10)
  GPIO PIN 10.
- #define GPIOx_PIN11 (11)
  GPIO PIN 11.
- #define GPIOx_PIN12 (12)
  GPIO PIN 12.
- #define GPIOx_PIN13 (13)
  GPIO PIN 13.
- #define GPIOx_PIN14 (14)
  GPIO PIN 14.
- #define GPIOx_PIN15 (15)
  GPIO PIN 15.

5.13.1 Detailed Description

5.13.2 Macro Definition Documentation

5.13.2.1 GPIOx_PIN0

```
#define GPIOx_PIN0 (0)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 0.

Definition at line 332 of file GPIO_interface.h.

5.13.2.2 GPIOx_PIN1

```
#define GPIOx_PIN1 (1)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 1.

Definition at line 339 of file GPIO_interface.h.

5.13.2.3 GPIOx_PIN2

```
#define GPIOx_PIN2 (2)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 2.

Definition at line 346 of file GPIO_interface.h.

5.13.2.4 GPIOx_PIN3

```
#define GPIOx_PIN3 (3)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 3.

Definition at line 353 of file GPIO_interface.h.

5.13.2.5 GPIOx_PIN4

```
#define GPIOx_PIN4 (4)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 4.

Definition at line 360 of file GPIO_interface.h.

5.13.2.6 GPIOx_PIN5

```
#define GPIOx_PIN5 (5)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 5.

Definition at line 367 of file GPIO_interface.h.

5.13.2.7 GPIOx_PIN6

```
#define GPIOx_PIN6 (6)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 6.

Definition at line 374 of file GPIO_interface.h.

5.13.2.8 GPIOx_PIN7

```
#define GPIOx_PIN7 (7)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 7.

Definition at line 381 of file GPIO_interface.h.

5.13.2.9 GPIOx_PIN8

```
#define GPIOx_PIN8 (8)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 8.

Definition at line 388 of file GPIO_interface.h.

5.13.2.10 GPIOx_PIN9

```
#define GPIOx_PIN9 (9)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

brief GPIO PIN 9

Definition at line 395 of file GPIO_interface.h.

5.13.2.11 GPIOx_PIN10

```
#define GPIOx_PIN10 (10)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 10.

Definition at line 402 of file GPIO_interface.h.

5.13.2.12 GPIOx_PIN11

```
#define GPIOx_PIN11 (11)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 11.

Definition at line 409 of file GPIO_interface.h.

5.13.2.13 GPIOx_PIN12

```
#define GPIOx_PIN12 (12)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 12.

Definition at line 416 of file GPIO_interface.h.

5.13.2.14 GPIOx_PIN13

```
#define GPIOx_PIN13 (13)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 13.

Definition at line 423 of file GPIO_interface.h.

5.13.2.15 GPIOx_PIN14

```
#define GPIOx_PIN14 (14)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 14.

Definition at line 430 of file GPIO_interface.h.

5.13.2.16 GPIOx_PIN15

```
#define GPIOx_PIN15 (15)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 15.

Definition at line 437 of file GPIO_interface.h.

5.14 GPIO Alternate Functions

Macros

- #define GPIOx_AF0 (0)
  GPIO Alternate function 0.
- #define GPIOx_AF1 (1)
  GPIO Alternate function 1.
- #define GPIOx_AF2 (2)
  GPIO Alternate function 2.
- #define GPIOx_AF3 (3)
  GPIO Alternate function 3.
- #define GPIOx_AF4 (4)
  GPIO Alternate function 4.
- #define GPIOx_AF5 (5)
  GPIO Alternate function 5.

- #define GPIOx_AF6 (6)
  GPIO Alternate function 6.
- #define GPIOx_AF7 (7)
  GPIO Alternate function 7.
- #define GPIOx_AF8 (8)
  GPIO Alternate function 8.
- #define GPIOx_AF9 (9)
  GPIO Alternate function 9.
- #define GPIOx_AF10 (10)
  GPIO Alternate function 10.
- #define GPIOx_AF11 (11)
  GPIO Alternate function 11.
- #define GPIOx_AF12 (12)
  GPIO Alternate function 12.
- #define GPIOx_AF13 (13)
  GPIO Alternate function 13.
- #define GPIOx_AF14 (14)
  GPIO Alternate function 14.
- #define GPIOx_AF15 (15)
  GPIO Alternate function 15.

5.14.1 Detailed Description

5.14.2 Macro Definition Documentation

5.14.2.1 GPIOx_AF0

```
#define GPIOx_AF0 (0)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 0.

Definition at line 452 of file GPIO_interface.h.

5.14.2.2 GPIOx_AF1

```
#define GPIOx_AF1 (1)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 1.

Definition at line 459 of file GPIO_interface.h.

5.14.2.3 GPIOx_AF2

```
#define GPIOx_AF2 (2)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 2.

Definition at line 466 of file GPIO_interface.h.

5.14.2.4 GPIOx_AF3

```
#define GPIOx_AF3 (3)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 3.

Definition at line 473 of file GPIO_interface.h.

5.14.2.5 GPIOx_AF4

```
#define GPIOx_AF4 (4)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 4.

Definition at line 480 of file GPIO_interface.h.

5.14.2.6 GPIOx_AF5

```
#define GPIOx_AF5 (5)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 5.

Definition at line 487 of file GPIO_interface.h.

5.14.2.7 GPIOx_AF6

```
#define GPIOx_AF6 (6)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 6.

Definition at line 494 of file GPIO_interface.h.

5.14.2.8 GPIOx_AF7

```
#define GPIOx_AF7 (7)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 7.

Definition at line 501 of file GPIO_interface.h.

5.14.2.9 GPIOx_AF8

```
#define GPIOx_AF8 (8)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 8.

Definition at line 508 of file GPIO_interface.h.

5.14.2.10 GPIOx_AF9

```
#define GPIOx_AF9 (9)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 9.

Definition at line 515 of file GPIO_interface.h.

5.14.2.11 GPIOx_AF10

```
#define GPIOx_AF10 (10)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 10.

Definition at line 522 of file GPIO_interface.h.

5.14.2.12 GPIOx_AF11

```
#define GPIOx_AF11 (11)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 11.

Definition at line 529 of file GPIO_interface.h.

5.14.2.13 GPIOx_AF12

```
#define GPIOx_AF12 (12)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 12.

Definition at line 536 of file GPIO_interface.h.

5.14.2.14 GPIOx_AF13

```
#define GPIOx_AF13 (13)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 13.

Definition at line 543 of file GPIO_interface.h.

5.14.2.15 GPIOx_AF14

```
#define GPIOx_AF14 (14)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 14.

Definition at line 550 of file GPIO_interface.h.

5.14.2.16 GPIOx_AF15

```
#define GPIOx_AF15 (15)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 15.

Definition at line 557 of file GPIO_interface.h.

5.15 GPIO Addresses

Macros

- #define GPIOA_BASE_ADDRESS (0x40020000)
- #define GPIOB_BASE_ADDRESS (0x40020400)
- #define GPIOC_BASE_ADDRESS (0x40020800)

5.15.1 Detailed Description

5.15.2 Macro Definition Documentation

5.15.2.1 GPIOA_BASE_ADDRESS

```
#define GPIOA_BASE_ADDRESS (0x40020000)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_private.h>
```

Port A Base address

Definition at line 83 of file GPIO_private.h.

5.15.2.2 GPIOB_BASE_ADDRESS

```
#define GPIOB_BASE_ADDRESS (0x40020400)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_private.h>
```

Port B Base address

Definition at line 90 of file GPIO_private.h.

5.15.2.3 GPIOC_BASE_ADDRESS

```
#define GPIOC_BASE_ADDRESS (0x40020800)
```

```
#include <src/COTS/MCAL/GPIO/GPIO_private.h>
```

Port C Base address

Definition at line 97 of file GPIO_private.h.

5.16 GPIO Registers

Macros

- #define GPIOA ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOA_BASE_ADDRESS))
- #define GPIOB ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOB_BASE_ADDRESS))
- #define GPIOC ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOC_BASE_ADDRESS))

5.16.1 Detailed Description

5.16.2 Macro Definition Documentation

5.16.2.1 GPIOA

```
#define GPIOA ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOA_BASE_ADDRESS))
```

```
#include <src/COTS/MCAL/GPIO/GPIO_private.h>
```

GPIO register for port A

Definition at line 112 of file GPIO_private.h.

5.16.2.2 GPIOB

```
#define GPIOB ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOB_BASE_ADDRESS))
```

```
#include <src/COTS/MCAL/GPIO/GPIO_private.h>
```

GPIO register for port B

Definition at line 119 of file GPIO_private.h.

5.16.2.3 GPIOC

```
#define GPIOC ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOC_BASE_ADDRESS))
```

```
#include <src/COTS/MCAL/GPIO/GPIO_private.h>
```

GPIO register for port C

Definition at line 126 of file GPIO_private.h.

Chapter 6

Data Structure Documentation

6.1 GPIOx_MemoryMapType Struct Reference

MDIO Configuration structure for a speific PIN initialization.

`#include "GPIO_private.h"`

Data Fields

- u32_t MODERx
  *Control PIN mode whether **INPUT** or **OUTPUT***
- u32_t OTYPERx
  Select the output type between push-pull or open-drain.
- u32_t OSPEEDRx
  Configure the speed that the PIN is connected to.
- u32_t PUPDRx
  Control the pull-up or pull-down of the pin, despite its directon.
- u32_t IDRx
  Read the input data.
- u32_t ODRx
  Write the output data.
- u32_t BSRRx
  (Re)set each bit in the output data register
- u32_t LCKRx
  Lock the GPIO control registers.
- u32_t AFRLx
  *Control alternate function **LOW** register.*
- u32_t AFRHx
  *Control alternate function **HIGH** register.*

6.1.1 Detailed Description

MDIO Configuration structure for a speific PIN initialization.

Definition at line 18 of file GPIO_private.h.

6.1.2 Field Documentation

6.1.2.1 MODERx

`u32_t MODERx`

Control PIN mode whether **INPUT** or **OUTPUT**

Definition at line 23 of file GPIO_private.h.

6.1.2.2 OTYPERx

`u32_t OTYPERx`

Select the output type between push-pull or open-drain.

Definition at line 28 of file GPIO_private.h.

6.1.2.3 OSPEEDRx

`u32_t OSPEEDRx`

Configure the speed that the PIN is connected to.

Definition at line 33 of file GPIO_private.h.

6.1.2.4 PUPDRx

`u32_t PUPDRx`

Control the pull-up or pull-down of the pin, despite its directon.

Definition at line 38 of file GPIO_private.h.

6.1.2.5 IDRx

`u32_t IDRx`

Read the input data.

Definition at line 43 of file GPIO_private.h.

6.1.2.6 ODRx

```
u32_t ODRx
```

Write the output data.

Definition at line 48 of file GPIO_private.h.

6.1.2.7 BSRRx

```
u32_t BSRRx
```

(Re)set each bit in the output data register

Definition at line 53 of file GPIO_private.h.

6.1.2.8 LCKRx

```
u32_t LCKRx
```

Lock the GPIO control registers.

Definition at line 58 of file GPIO_private.h.

6.1.2.9 AFRLx

```
u32_t AFRLx
```

Control alternate function **LOW** register.

Definition at line 63 of file GPIO_private.h.

6.1.2.10 AFRHx

```
u32_t AFRHx
```

Control alternate function **HIGH** register.

Definition at line 68 of file GPIO_private.h.

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

- src/COTS/MCAL/GPIO/GPIO_private.h

6.2 MGPIOx_ConfigType Struct Reference

MDIO Configuration structure for a speific PIN initialization.

`#include "GPIO_interface.h"`

Data Fields

- u8_t Port
  Configures a speific GPIO port.
- u8_t Pin
  Configures a speific GPIO pin.
- u8_t Mode
  Configures a speific GPIO mode.
- u8_t OutputType
  Configures a speific GPIO output's type.
- u8_t OutputSpeed
  Configures a speific GPIO output's speed.
- u8_t InputType
  Configures a speific GPIO input type.
- u8_t AF_Type
  Configures a speific GPIO Alternative function.

6.2.1 Detailed Description

MDIO Configuration structure for a speific PIN initialization.

Definition at line 18 of file GPIO_interface.h.

6.2.2 Field Documentation

6.2.2.1 Port

`u8_t Port`

Configures a speific GPIO port.

Definition at line 23 of file GPIO_interface.h.

Referenced by MGPIOx_vInit().

6.2.2.2 Pin

```
u8_t Pin
```

Configures a speific GPIO pin.

Definition at line 27 of file GPIO_interface.h.

Referenced by MGPIOx_vInit().

6.2.2.3 Mode

```
u8_t Mode
```

Configures a speific GPIO mode.

Definition at line 31 of file GPIO_interface.h.

Referenced by MGPIOx_vInit().

6.2.2.4 OutputType

```
u8_t OutputType
```

Configures a speific GPIO output's type.

Definition at line 35 of file GPIO_interface.h.

Referenced by MGPIOx_vInit().

6.2.2.5 OutputSpeed

```
u8_t OutputSpeed
```

Configures a speific GPIO output's speed.

Definition at line 39 of file GPIO_interface.h.

Referenced by MGPIOx_vInit().

6.2.2.6 InputType

```
u8_t InputType
```

Configures a speific GPIO input type.

Definition at line 43 of file GPIO_interface.h.

Referenced by MGPIOx_vInit().

6.2.2.7 AF_Type

```
u8_t AF_Type
```

Configures a speific GPIO Alternative function.

Definition at line 47 of file GPIO_interface.h.

Referenced by MGPIOx_vInit().

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

- src/COTS/MCAL/GPIO/GPIO_interface.h

6.3 RCC_MemoryMapType Struct Reference

```
#include "MRCC_private.h"
```

Data Fields

- u32_t CR
- u32_t PLLCFGR
- u32_t CFGR
- u32_t CIR
- u32_t AHB1RSTR
- u32_t AHB2RSTR
- u32_t Reserved1
- u32_t Reserved2
- u32_t APB1RSTR
- u32_t APB2RSTR
- u32_t Reserved3
- u32_t Reserved4
- u32_t AHB1ENR
- u32_t AHB2ENR
- u32_t Reserved5
- u32_t Reserved6
- u32_t APB1ENR
- u32_t APB2ENR
- u32_t Reserved7

- u32_t Reserved8
- u32_t AHB1LPENR
- u32_t AHB2LPENR
- u32_t Reserved9
- u32_t Reserved10
- u32_t APB1LPENR
- u32_t APB2LPENR
- u32_t Reserved11
- u32_t Reserved12
- u32_t BDCR
- u32_t CSR
- u32_t Reserved13
- u32_t Reserved14
- u32_t SSCGR
- u32_t PLLI2SCFGR
- u32_t DCKCFGR

6.3.1 Detailed Description

Definition at line 19 of file MRCC_private.h.

6.3.2 Field Documentation

6.3.2.1 CR

```
u32_t CR
```

Definition at line 22 of file MRCC_private.h.

6.3.2.2 PLLCFGR

```
u32_t PLLCFGR
```

Definition at line 23 of file MRCC_private.h.

6.3.2.3 CFGR

```
u32_t CFGR
```

Definition at line 24 of file MRCC_private.h.

6.3.2.4 CIR

```
u32_t CIR
```

Definition at line 25 of file MRCC_private.h.

6.3.2.5 AHB1RSTR

```
u32_t AHB1RSTR
```

Definition at line 26 of file MRCC_private.h.

6.3.2.6 AHB2RSTR

```
u32_t AHB2RSTR
```

Definition at line 27 of file MRCC_private.h.

6.3.2.7 Reserved1

```
u32_t Reserved1
```

Definition at line 29 of file MRCC_private.h.

6.3.2.8 Reserved2

```
u32_t Reserved2
```

Definition at line 30 of file MRCC_private.h.

6.3.2.9 APB1RSTR

```
u32_t APB1RSTR
```

Definition at line 32 of file MRCC_private.h.

6.3.2.10 APB2RSTR

```
u32_t APB2RSTR
```

Definition at line 33 of file MRCC_private.h.

6.3.2.11 Reserved3

```
u32_t Reserved3
```

Definition at line 35 of file MRCC_private.h.

6.3.2.12 Reserved4

```
u32_t Reserved4
```

Definition at line 36 of file MRCC_private.h.

6.3.2.13 AHB1ENR

```
u32_t AHB1ENR
```

Definition at line 38 of file MRCC_private.h.

6.3.2.14 AHB2ENR

```
u32_t AHB2ENR
```

Definition at line 39 of file MRCC_private.h.

6.3.2.15 Reserved5

```
u32_t Reserved5
```

Definition at line 41 of file MRCC_private.h.

6.3.2.16 Reserved6

```
u32_t Reserved6
```

Definition at line 42 of file MRCC_private.h.

6.3.2.17 APB1ENR

```
u32_t APB1ENR
```

Definition at line 44 of file MRCC_private.h.

6.3.2.18 APB2ENR

```
u32_t APB2ENR
```

Definition at line 45 of file MRCC_private.h.

6.3.2.19 Reserved7

```
u32_t Reserved7
```

Definition at line 47 of file MRCC_private.h.

6.3.2.20 Reserved8

```
u32_t Reserved8
```

Definition at line 48 of file MRCC_private.h.

6.3.2.21 AHB1LPENR

```
u32_t AHB1LPENR
```

Definition at line 50 of file MRCC_private.h.

6.3.2.22 AHB2LPENR

```
u32_t AHB2LPENR
```

Definition at line 51 of file MRCC_private.h.

6.3.2.23 Reserved9

```
u32_t Reserved9
```

Definition at line 53 of file MRCC_private.h.

6.3.2.24 Reserved10

```
u32_t Reserved10
```

Definition at line 54 of file MRCC_private.h.

6.3.2.25 APB1LPENR

```
u32_t APB1LPENR
```

Definition at line 56 of file MRCC_private.h.

6.3.2.26 APB2LPENR

```
u32_t APB2LPENR
```

Definition at line 57 of file MRCC_private.h.

6.3.2.27 Reserved11

```
u32_t Reserved11
```

Definition at line 59 of file MRCC_private.h.

6.3.2.28 Reserved12

`u32_t Reserved12`

Definition at line 60 of file MRCC_private.h.

6.3.2.29 BDCR

`u32_t BDCR`

Definition at line 62 of file MRCC_private.h.

6.3.2.30 CSR

`u32_t CSR`

Definition at line 63 of file MRCC_private.h.

6.3.2.31 Reserved13

`u32_t Reserved13`

Definition at line 65 of file MRCC_private.h.

6.3.2.32 Reserved14

`u32_t Reserved14`

Definition at line 66 of file MRCC_private.h.

6.3.2.33 SSCGR

`u32_t SSCGR`

Definition at line 68 of file MRCC_private.h.

6.3.2.34 PLLI2SCFGR

`u32_t PLLI2SCFGR`

Definition at line 69 of file MRCC_private.h.

6.3.2.35 DCKCFGR

`u32_t DCKCFGR`

Definition at line 70 of file MRCC_private.h.

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

- src/COTS/MCAL/RCC/MRCC_private.h

Chapter 7

File Documentation

7.1 README.md File Reference

7.2 src/COTS/LIB/LSTD_BITMATH.h File Reference

This file contains the bit math manipulation macro-functions.

Macros

- #define SET_BIT(Reg, bitnum) (Reg) |= (1 << (bitnum))
- #define CLR_BIT(Reg, bitnum) (Reg) &= ~(1 << (bitnum))
- #define TOGGLE_BIT(Reg, bitnum) (Reg) ^= (1 << (bitnum))
- #define GET_BIT(Reg, bitnum) (((Reg)>>(bitnum)) & 1)
- #define SET_BITs(Reg, bits, bitnum, factor) (Reg) |= ((bits) << (bitnum * factor))
- #define CLR_BITs(Reg, bits, bitnum, factor) (Reg) &= ~((bits) << (bitnum * factor))
- #define TOGGLE_BITs(Reg, bits, bitnum, factor) (Reg) ^= ((bits) << (bitnum * factor))
- #define GET_BITs(Reg, bits, bitnum, factor) (((Reg) >> (bitnum * factor)) & (bits))

7.2.1 Detailed Description

This file contains the bit math manipulation macro-functions.

Author

Ali El Bana

Version

1.0

Date

10/29/2022

Definition in file LSTD_BITMATH.h.

7.3 LSTD_BITMATH.h

Go to the documentation of this file.

```
00001
00009 #ifndef COTS_LIB_LSTD_BITMATH_H_
00010 #define COTS_LIB_LSTD_BITMATH_H_
00011
00023 #define SET_BIT(Reg, bitnum) (Reg) |= (1 « (bitnum))
00024
00030 #define CLR_BIT(Reg, bitnum) (Reg) &= ~(1 « (bitnum))
00031
00037 #define TOGGLE_BIT(Reg, bitnum) (Reg) ^= (1 « (bitnum))
00038
00044 #define GET_BIT(Reg, bitnum)        (((Reg)»(bitnum)) & 1)
00045
00059 #define SET_BITs(Reg, bits, bitnum, factor) (Reg) |= ((bits) « (bitnum * factor))
00060
00066 #define CLR_BITs(Reg, bits, bitnum, factor) (Reg) &= ~((bits) « (bitnum * factor))
00067
00073 #define TOGGLE_BITs(Reg, bits, bitnum, factor) (Reg) ^= ((bits) « (bitnum * factor))
00074
00080 #define GET_BITs(Reg, bits, bitnum, factor) (((Reg) » (bitnum * factor)) & (bits))
00081
00084 #endif /* COTS_LIB_LSTD_BITMATH_H_ */
```

7.4 src/COTS/LIB/LSTD_COMPILER.h File Reference

This file contains the compiler standard macros.

Macros

- #define VAR(vartype) vartype
- #define FUNC(rettype) rettype
- #define P2VAR(ptrtype) ptrtype ∗
- #define P2CONST(ptrtype) const ptrtype ∗
- #define CONSTP2VAR(ptrtype) ptrtype ∗ const
- #define CONSTP2CONST(ptrtype) const ptrtype ∗ const
- #define P2FUNC(rettype, fctname) rettype (∗ fctname)
- #define CONST(consttype) const consttype
- #define STATIC static

7.4.1 Detailed Description

This file contains the compiler standard macros.

Author

Mohamed Alaa

Version

1.0

Date

11/04/2022

Definition in file LSTD_COMPILER.h.

7.5 LSTD_COMPILER.h

Go to the documentation of this file.

```
00001
00009 #ifndef COTS_LIB_LSTD_COMPILER_H_
00010 #define COTS_LIB_LSTD_COMPILER_H_
00011
00023 #define VAR(vartype) vartype
00024
00030 #define FUNC(rettype) rettype
00031
00037 #define P2VAR(ptrtype) ptrtype *
00038
00044 #define P2CONST(ptrtype) const ptrtype *
00045
00051 #define CONSTP2VAR(ptrtype) ptrtype * const
00052
00058 #define CONSTP2CONST(ptrtype) const ptrtype * const
00059
00065 #define P2FUNC(rettype, fctname) rettype (* fctname)
00066
00072 #ifndef CONST
00073 #define CONST(consttype) const consttype
00074 #endif
00075
00081 #ifndef STATIC
00082 #define STATIC static
00083 #endif
00084
00087 #endif /* COTS_LIB_LSTD_COMPILER_H_ */
```

7.6 src/COTS/LIB/LSTD_MCU_UTILITIES.h File Reference

This file contains the MCU utility macro-functions.

Macros

- #define MY_MS_DELAY(T) do{ u32 Timer = (T * 500); while (Timer--) { asm ("nop"); } } while(0);

7.6.1 Detailed Description

This file contains the MCU utility macro-functions.

Author

Ali El Bana

Version

1.0

Date

10/29/2022

Definition in file LSTD_MCU_UTILITIES.h.

7.7 LSTD_MCU_UTILITIES.h

Go to the documentation of this file.

```
00001
00009 #ifndef COTS_LIB_LSTD_MCU_UTILITIES_H_
00010 #define COTS_LIB_LSTD_MCU_UTILITIES_H_
00023 #define MY_MS_DELAY(T)   do{ u32 Timer = (T * 500); while (Timer--) { asm ("nop"); } } while(0);
00024
00027 #endif /* COTS_LIB_LSTD_MCU_UTILITIES_H_ */
```

7.8 src/COTS/LIB/LSTD_TYPES.h File Reference

This file contains the standard types.

Typedefs

- typedef unsigned char bool_t
- typedef unsigned char u8_t
- typedef unsigned short int u16_t
- typedef unsigned long int u32_t
- typedef signed char s8_t
- typedef signed short int s16_t
- typedef signed long int s32_t
- typedef float fl32_t
- typedef double fl64_t

7.8.1 Detailed Description

This file contains the standard types.

Author

Ali El Bana

Version

1.0

Date

10/29/2022

Definition in file LSTD_TYPES.h.

7.9 LSTD_TYPES.h

Go to the documentation of this file.

```
00001
00009 #ifndef COTS_LIB_LSTD_TYPES_H_
00010 #define COTS_LIB_LSTD_TYPES_H_
00011
00023 typedef unsigned char bool_t;
00024
00030 typedef unsigned char u8_t;
00031
00037 typedef unsigned short int u16_t;
00038
00044 typedef unsigned long int u32_t;
00045
00051 typedef signed char s8_t;
00052
00058 typedef signed short int s16_t;
00059
00065 typedef signed long int s32_t;
00066
00072 typedef float fl32_t;
00073
00079 typedef double fl64_t;
00080
00083 #endif /* COTS_LIB_LSTD_TYPES_H_ */
```

7.10 src/COTS/LIB/LSTD_VALUES.h File Reference

This file contains the standard values.

Macros

- #define TRUE (1)
- #define FALSE (0)
- #define NULL (P2VAR(void) 0)
- #define INITIAL_ZERO (0)
- #define FLAG_SET (1)
- #define FLAG_CLEARED (0)

7.10.1 Detailed Description

This file contains the standard values.

Author

Ali El Bana

Version

1.0

Date

10/29/2022

Definition in file LSTD_VALUES.h.

7.11 LSTD_VALUES.h

Go to the documentation of this file.

```
00001
00009 #ifndef COTS_LIB_LSTD_VALUES_H_
00010 #define COTS_LIB_LSTD_VALUES_H_
00011
00023 #ifndef TRUE
00024 #define TRUE (1)
00025 #endif  /* ifndef TRUE */
00026
00032 #ifndef FALSE
00033 #define FALSE (0)
00034 #endif  /* ifndef FALSE */
00035
00041 #ifndef NULL
00042 #define NULL (P2VAR(void) 0)
00043 #endif  /* ifndef NULL */
00044
00050 #ifndef INITIAL_ZERO
00051 #define INITIAL_ZERO (0)
00052 #endif  /* ifndef INITIAL_ZERO */
00053
00059 #ifndef FLAG_SET
00060 #define FLAG_SET (1)
00061 #endif  /* ifndef FLAG_SET */
00062
00068 #ifndef FLAG_CLEARED
00069 #define FLAG_CLEARED (0)
00070 #endif  /* ifndef FLAG_CLEARED */
00071
00074 #endif /* COTS_LIB_LSTD_VALUES_H_ */
```

7.12 src/COTS/MCAL/GPIO/GPIO_config.h File Reference

This file contains the GPIO configurations.

Macros

- #define GPIOA_PIN_POS (0b1110000000000000)
- #define GPIOB_PIN_POS (0b0000000000011100)
- #define LCKK_BIT_POS (16U)

7.12.1 Detailed Description

This file contains the GPIO configurations.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

08/22/2022

Definition in file GPIO_config.h.

7.12.2 Macro Definition Documentation

7.12.2.1 GPIOA_PIN_POS

```
#define GPIOA_PIN_POS (0b1110000000000000)
```

Definition at line 17 of file GPIO_config.h.

7.12.2.2 GPIOB_PIN_POS

```
#define GPIOB_PIN_POS (0b0000000000011100)
```

Definition at line 18 of file GPIO_config.h.

7.12.2.3 LCKK_BIT_POS

```
#define LCKK_BIT_POS (16U)
```

Definition at line 19 of file GPIO_config.h.

7.13 GPIO_config.h

Go to the documentation of this file.

```
00001
00009 #ifndef _GPIO_config_H
00010 #define _GPIO_config_H
00011
00012
00013 /*******************************************************************************/
00014                             // GPIOx configurations //
00015 /*******************************************************************************/
00016
00017 #define GPIOA_PIN_POS (0b1110000000000000)  //I want to lock PA13,14,15.
00018 #define GPIOB_PIN_POS (0b0000000000011100)  //I want to lock PB2,3,4.
00019 #define LCKK_BIT_POS (16U)                  //Position of LCKK bit.
00020
00021
00022 #endif //_GPIO_config_H
```

7.14 src/COTS/MCAL/GPIO/GPIO_interface.h File Reference

This file contains the interfacing information for the GPIO module.

Data Structures

- struct MGPIOx_ConfigType

  MDIO Configuration structure for a speific PIN initialization.

Macros

- #define GPIOx_MODE_INPUT (0b00)

  Control input mode.
- #define GPIOx_MODE_OUTPUT (0b01)

  Control output mode.
- #define GPIOx_MODE_AF (0b10)

  Control alternate function mode.
- #define GPIOx_MODE_ANALOG (0b11)

  Control analog mode.
- #define GPIO_PORTA (0)

  GPIO Port A.
- #define GPIO_PORTB (1)

  GPIO Port B.
- #define GPIO_PORTC (2)

  GPIO Port C.
- #define GPIOx_OPENDRAIN (1)

  GPIO open-drain.
- #define GPIOx_PUSHPULL (2)

  GPIO push-pull.
- #define GPIOx_LowSpeed (0b00)

  GPIO low speed.
- #define GPIOx_MediumSpeed (0b01)

  GPIO medium speed.
- #define GPIOx_HighSpeed (0b10)

  GPIO high speed.
- #define GPIOx_VeryHighSpeed (0b11)

  GPIO very high speed.
- #define GPIOx_NoPull (0b00)

  GPIO No PULL.
- #define GPIOx_PullUp (0b01)

  GPIO Pull UP.
- #define GPIOx_PullDown (0b10)

  GPIO Pull Down.
- #define GPIOx_HIGH (1)

  GPIO output high.
- #define GPIOx_LOW (2)

  GPIO output low.
- #define GPIOx_PIN0 (0)

  GPIO PIN 0.
- #define GPIOx_PIN1 (1)

  GPIO PIN 1.
- #define GPIOx_PIN2 (2)

  GPIO PIN 2.
- #define GPIOx_PIN3 (3)

GPIO PIN 3.

- #define GPIOx_PIN4 (4)

  GPIO PIN 4.
- #define GPIOx_PIN5 (5)

  GPIO PIN 5.
- #define GPIOx_PIN6 (6)

  GPIO PIN 6.
- #define GPIOx_PIN7 (7)

  GPIO PIN 7.
- #define GPIOx_PIN8 (8)

  GPIO PIN 8.
- #define GPIOx_PIN9 (9)

  brief GPIO PIN 9
- #define GPIOx_PIN10 (10)

  GPIO PIN 10.
- #define GPIOx_PIN11 (11)

  GPIO PIN 11.
- #define GPIOx_PIN12 (12)

  GPIO PIN 12.
- #define GPIOx_PIN13 (13)

  GPIO PIN 13.
- #define GPIOx_PIN14 (14)

  GPIO PIN 14.
- #define GPIOx_PIN15 (15)

  GPIO PIN 15.
- #define GPIOx_AF0 (0)

  GPIO Alternate function 0.
- #define GPIOx_AF1 (1)

  GPIO Alternate function 1.
- #define GPIOx_AF2 (2)

  GPIO Alternate function 2.
- #define GPIOx_AF3 (3)

  GPIO Alternate function 3.
- #define GPIOx_AF4 (4)

  GPIO Alternate function 4.
- #define GPIOx_AF5 (5)

  GPIO Alternate function 5.
- #define GPIOx_AF6 (6)

  GPIO Alternate function 6.
- #define GPIOx_AF7 (7)

  GPIO Alternate function 7.
- #define GPIOx_AF8 (8)

  GPIO Alternate function 8.
- #define GPIOx_AF9 (9)

  GPIO Alternate function 9.
- #define GPIOx_AF10 (10)

  GPIO Alternate function 10.
- #define GPIOx_AF11 (11)

  GPIO Alternate function 11.
- #define GPIOx_AF12 (12)

  GPIO Alternate function 12.

- #define GPIOx_AF13 (13)
  GPIO Alternate function 13.
- #define GPIOx_AF14 (14)
  GPIO Alternate function 14.
- #define GPIOx_AF15 (15)
  GPIO Alternate function 15.

Functions

- void MGPIOx_vLockedPins (void)
- void MGPIOx_vSetPinMode (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8Mode)
  Sets a certain pin's mode on a speific port.
- void MGPIOx_vSetPinOutputType (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8Output↩Type)
  Sets a certain pin's output type on a speific port.
- void MGPIOx_vSetPinOutputSpeed (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8↩OutputSpeed)
  Sets a certain pin's output speed on a speific port.
- void MGPIOx_vSetPinInputPullType (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8Input↩PullType)
  Sets a certain pin's input pull type on a speific port.
- u8_t MGPIOx_u8GetPinValue (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID)
  Gets the value currently on a certain pin.
- void MGPIOx_vSetPinValue (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8PinValue)
  Sets a certain pin's output value on a speific port.
- void MGPIOx_vSetResetAtomic (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8SetReset↩PinValue)
  Resets a certain pin's output value on a speific port.
- void MGPIOx_vSetAlternateFunctionON (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8↩AFID)
  Applys an alternative function on a certain pin.
- void MGPIOx_vSetPortConfigLock (VAR(u8_t) A_u8PortID)
  Updates a port's configuration lock.
- void MGPIOx_vInit (P2VAR(MGPIOx_ConfigType) A_xPinConfig)
  Initialize the GPIO with a certain configuration.
- void MGPIOx_vTogglePinValue (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID)
  Toggles a certain's pin's value on a certain port.

7.14.1 Detailed Description

This file contains the interfacing information for the GPIO module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file GPIO_interface.h.

7.14.2 Function Documentation

7.14.2.1 MGPIOx_vLockedPins()

```
void MGPIOx_vLockedPins (
            void  )
```

Locks the prohibited GPIO PINs.

Definition at line 24 of file GPIO_program.c.

```
00025 {
00026
00027     VAR(volatile u32_t) L_u32LockGPIOA = INITIAL_ZERO ;
00028
00029     /* Lock key write sequence */
00030
00031     /* WR LCKR[16] = 1 + LCKR[13,14,15] = 1 */
00032     L_u32LockGPIOA = ( (1UL << LCKK_BIT_POS) | (GPIOA_PIN_POS) ) ;
00033
00034     GPIOA->LCKRx = L_u32LockGPIOA ;
00035
00036     /* WR LCKR[16] = 0 + LCKR[13,14,15] should not change*/
00037     GPIOA->LCKRx = (GPIOA_PIN_POS) ;
00038
00039     /* WR LCKR[16] = 1 + LCKR[13,14,15] should not change*/
00040     GPIOA->LCKRx = L_u32LockGPIOA ;
00041
00042     /* RD LCKR */
00043     L_u32LockGPIOA = GPIOA->LCKRx ;
00044
00046
00047     VAR(volatile u32_t) L_u32LockGPIOB = INITIAL_ZERO ;
00048
00049     /* Lock key write sequence */
00050
00051     /* WR LCKR[16] = 1 + LCKR[2,3,4] = 1 */
00052     L_u32LockGPIOB = ( (1UL << LCKK_BIT_POS) | (GPIOB_PIN_POS) ) ;
00053
00054     GPIOB->LCKRx = L_u32LockGPIOB ;
00055
00056     /* WR LCKR[16] = 0 + LCKR[2,3,4] should not change*/
00057     GPIOB->LCKRx = (GPIOB_PIN_POS) ;
00058
00059     /* WR LCKR[16] = 1 + LCKR[2,3,4] should not change*/
00060     GPIOB->LCKRx = L_u32LockGPIOB ;
00061
00062     /* RD LCKR */
00063     L_u32LockGPIOB = GPIOB->LCKRx ;
00064
00065 }
```

References GPIOA, GPIOA_PIN_POS, GPIOB, GPIOB_PIN_POS, INITIAL_ZERO, LCKK_BIT_POS, and VAR.

7.14.2.2 MGPIOx_vSetPinMode()

```
void MGPIOx_vSetPinMode (
            VAR(u8_t) A_u8PortID,
            VAR(u8_t) A_u8PinID,
            VAR(u8_t) A_u8Mode )
```

Sets a certain pin's mode on a speific port.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to update its mode
in	*A_u8Mode*	The mode to apply the pin

7.14.2.3 MGPIOx_vSetPinOutputType()

```
void MGPIOx_vSetPinOutputType (
            VAR(u8_t) A_u8PortID,
            VAR(u8_t) A_u8PinID,
            VAR(u8_t) A_u8OutputType )
```

Sets a certain pin's output type on a speific port.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to update its mode
in	*A_u8OutputType*	The output type to apply on the pin

7.14.2.4 MGPIOx_vSetPinOutputSpeed()

```
void MGPIOx_vSetPinOutputSpeed (
            VAR(u8_t) A_u8PortID,
            VAR(u8_t) A_u8PinID,
            VAR(u8_t) A_u8OutputSpeed )
```

Sets a certain pin's output speed on a speific port.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to update its mode
in	*A_u8OutputSpeed*	The output speed to apply on the pin

7.14.2.5 MGPIOx_vSetPinInputPullType()

```
void MGPIOx_vSetPinInputPullType (
            VAR(u8_t) A_u8PortID,
```

```
        VAR(u8_t) A_u8PinID,
        VAR(u8_t) A_u8InputPullType )
```

Sets a certain pin's input pull type on a speific port.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to update its mode
in	*A_u8InputPullType*	The input pull type to apply on the pin

7.14.2.6 MGPIOx_u8GetPinValue()

```
u8_t MGPIOx_u8GetPinValue (
            VAR(u8_t) A_u8PortID,
            VAR(u8_t) A_u8PinID )
```

Gets the value currently on a certain pin.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to update its mode

Returns

The current value on the pin

7.14.2.7 MGPIOx_vSetPinValue()

```
void MGPIOx_vSetPinValue (
            VAR(u8_t) A_u8PortID,
            VAR(u8_t) A_u8PinID,
            VAR(u8_t) A_u8PinValue )
```

Sets a certain pin's output value on a speific port.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to update its mode
in	*A_u8PinValue*	The set value to set on the pin

7.14.2.8 MGPIOx_vSetResetAtomic()

```
void MGPIOx_vSetResetAtomic (
            VAR(u8_t) A_u8PortID,
```

```
            VAR(u8_t) A_u8PinID,
            VAR(u8_t) A_u8SetResetPinValue )
```

Resets a certain pin's output value on a speific port.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to update its mode
in	*A_u8SetResetPinValue*	The reset value to set on the pin

7.14.2.9 MGPIOx_vSetAlternateFunctionON()

```
void MGPIOx_vSetAlternateFunctionON (
            VAR(u8_t) A_u8PortID,
            VAR(u8_t) A_u8PinID,
            VAR(u8_t) A_u8AFID )
```

Applys an alternative function on a certain pin.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to update its mode
in	*A_u8AFID*	The alternative function to apply on the pin

7.14.2.10 MGPIOx_vSetPortConfigLock()

```
void MGPIOx_vSetPortConfigLock (
            VAR(u8_t) A_u8PortID )
```

Updates a port's configuration lock.

Parameters

in	*A_u8PortID*	The port to update the pin's mode

7.14.2.11 MGPIOx_vInit()

```
void MGPIOx_vInit (
            P2VAR(MGPIOx_ConfigType) A_xPinConfig )
```

Initialize the GPIO with a certain configuration.

Parameters

in	*A_xPinConfig*	The initialization configuration for the GPIO

7.14.2.12 MGPIOx_vTogglePinValue()

```
void MGPIOx_vTogglePinValue (
            VAR(u8_t) A_u8PortID,
            VAR(u8_t) A_u8PinID )
```

Toggles a certain's pin's value on a certain port.

Parameters

in	*A_u8PortID*	The port that the pin belongs to
in	*A_u8PinID*	The pin to toggle its value

7.15 GPIO_interface.h

Go to the documentation of this file.

```
00001
00009 /* Header file guard */
00010 #ifndef _GPIO_interface_H
00011 #define _GPIO_interface_H
00012
00018 typedef struct
00019 {
00023     u8_t Port;
00027     u8_t Pin;
00031     u8_t Mode;
00035     u8_t OutputType;
00039     u8_t OutputSpeed;
00043     u8_t InputType;
00047     u8_t AF_Type;
00048
00049 } MGPIOx_ConfigType;
00050
00051 /*****************************************************************************/
00052 /*                          Functions prototypes                             */
00053 /*****************************************************************************/
00054
00058 void MGPIOx_vLockedPins(void);
00059
00066 void MGPIOx_vSetPinMode(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8Mode);
00067
00074 void MGPIOx_vSetPinOutputType(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8OutputType);
00075
00082 void MGPIOx_vSetPinOutputSpeed(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8OutputSpeed);
00083
00090 void MGPIOx_vSetPinInputPullType(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
      A_u8InputPullType);
00091
00098 u8_t MGPIOx_u8GetPinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID);
00099
00106 void MGPIOx_vSetPinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8PinValue);
00107
00114 void MGPIOx_vSetResetAtomic(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
      A_u8SetResetPinValue);
00115
00122 void MGPIOx_vSetAlternateFunctionON(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8AFID);
00123
00128 void MGPIOx_vSetPortConfigLock(VAR(u8_t) A_u8PortID);
00129
```

```
00134 void MGPIOx_vInit(P2VAR(MGPIOx_ConfigType) A_xPinConfig);
00135
00141 void MGPIOx_vTogglePinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID);
00142
00143 /*******************************************************************************/
00144 /*                          Interfacing macros                                 */
00145 /*******************************************************************************/
00146
00158 #define GPIOx_MODE_INPUT (0b00)
00159
00165 #define GPIOx_MODE_OUTPUT (0b01)
00166
00172 #define GPIOx_MODE_AF (0b10)
00173
00179 #define GPIOx_MODE_ANALOG (0b11)
00180
00194 #define GPIO_PORTA (0)
00195
00201 #define GPIO_PORTB (1)
00202
00208 #define GPIO_PORTC (2)
00209
00223 #define GPIOx_OPENDRAIN (1)
00224
00230 #define GPIOx_PUSHPULL (2)
00231
00245 #define GPIOx_LowSpeed (0b00)
00246
00252 #define GPIOx_MediumSpeed (0b01)
00253
00259 #define GPIOx_HighSpeed (0b10)
00260
00266 #define GPIOx_VeryHighSpeed (0b11)
00267
00281 #define GPIOx_NoPull (0b00)
00282
00288 #define GPIOx_PullUp (0b01)
00289
00295 #define GPIOx_PullDown (0b10)
00296
00310 #define GPIOx_HIGH (1)
00311
00317 #define GPIOx_LOW (2)
00318
00332 #define GPIOx_PIN0 (0)
00333
00339 #define GPIOx_PIN1 (1)
00340
00346 #define GPIOx_PIN2 (2)
00347
00353 #define GPIOx_PIN3 (3)
00354
00360 #define GPIOx_PIN4 (4)
00361
00367 #define GPIOx_PIN5 (5)
00368
00374 #define GPIOx_PIN6 (6)
00375
00381 #define GPIOx_PIN7 (7)
00382
00388 #define GPIOx_PIN8 (8)
00389
00395 #define GPIOx_PIN9 (9)
00396
00402 #define GPIOx_PIN10 (10)
00403
00409 #define GPIOx_PIN11 (11)
00410
00416 #define GPIOx_PIN12 (12)
00417
00423 #define GPIOx_PIN13 (13)
00424
00430 #define GPIOx_PIN14 (14)
00431
00437 #define GPIOx_PIN15 (15)
00438
00452 #define GPIOx_AF0 (0)
00453
00459 #define GPIOx_AF1 (1)
00460
00466 #define GPIOx_AF2 (2)
00467
00473 #define GPIOx_AF3 (3)
00474
00480 #define GPIOx_AF4 (4)
00481
00487 #define GPIOx_AF5 (5)
```

```
00488
00494 #define GPIOx_AF6 (6)
00495
00501 #define GPIOx_AF7 (7)
00502
00508 #define GPIOx_AF8 (8)
00509
00515 #define GPIOx_AF9 (9)
00516
00522 #define GPIOx_AF10 (10)
00523
00529 #define GPIOx_AF11 (11)
00530
00536 #define GPIOx_AF12 (12)
00537
00543 #define GPIOx_AF13 (13)
00544
00550 #define GPIOx_AF14 (14)
00551
00557 #define GPIOx_AF15 (15)
00558
00561 #endif //_GPIO_interface_H
```

7.16 src/COTS/MCAL/GPIO/GPIO_private.h File Reference

This file contains the registers information and addresses for the GPIO module.

Data Structures

- struct GPIOx_MemoryMapType

  MDIO Configuration structure for a speific PIN initialization.

Macros

- #define GPIOA_BASE_ADDRESS (0x40020000)
- #define GPIOB_BASE_ADDRESS (0x40020400)
- #define GPIOC_BASE_ADDRESS (0x40020800)
- #define GPIOA ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOA_BASE_ADDRESS))
- #define GPIOB ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOB_BASE_ADDRESS))
- #define GPIOC ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOC_BASE_ADDRESS))

7.16.1 Detailed Description

This file contains the registers information and addresses for the GPIO module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

08/22/2022

Definition in file GPIO_private.h.

7.17 GPIO_private.h

Go to the documentation of this file.

```
00001
00009 /* Header file guard */
00010 #ifndef _GPIO_private_H
00011 #define _GPIO_private_H
00012
00018 typedef struct
00019 {
00023     u32_t MODERx;
00024
00028     u32_t OTYPERx;
00029
00033     u32_t OSPEEDRx;
00034
00038     u32_t PUPDRx;
00039
00043     u32_t IDRx;
00044
00048     u32_t ODRx;
00049
00053     u32_t BSRRx;
00054
00058     u32_t LCKRx;
00059
00063     u32_t AFRLx;
00064
00068     u32_t AFRHx;
00069
00070 } GPIOx_MemoryMapType;
00071
00083 #define GPIOA_BASE_ADDRESS (0x40020000)
00084
00090 #define GPIOB_BASE_ADDRESS (0x40020400)
00091
00097 #define GPIOC_BASE_ADDRESS (0x40020800)
00098
00112 #define GPIOA ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOA_BASE_ADDRESS))
00113
00119 #define GPIOB ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOB_BASE_ADDRESS))
00120
00126 #define GPIOC ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOC_BASE_ADDRESS))
00127
00130 #endif //_GPIO_private_H
```

7.18 src/COTS/MCAL/GPIO/GPIO_program.c File Reference

This file contains the source code of the interfacing for the GPIO modules.

```
#include "../../LIB/LSTD_TYPES.h"
#include "../../LIB/LSTD_COMPILER.h"
#include "../../LIB/LSTD_VALUES.h"
#include "../../LIB/LSTD_BITMATH.h"
#include "GPIO_interface.h"
#include "GPIO_private.h"
#include "GPIO_config.h"
```

Functions

- void MGPIOx_vLockedPins (void)
- void MGPIOx_vSetPinMode (u8_t A_u8PortID, u8_t A_u8PinID, u8_t A_u8Mode)
- void MGPIOx_vSetPinOutputType (u8_t A_u8PortID, u8_t A_u8PinID, u8_t A_u8OutputType)
- void MGPIOx_vSetPinOutputSpeed (u8_t A_u8PortID, u8_t A_u8PinID, u8_t A_u8OutputSpeed)
- void MGPIOx_vSetPinInputPullType (u8_t A_u8PortID, u8_t A_u8PinID, u8_t A_u8InputPullType)
- u8_t MGPIOx_u8GetPinValue (u8_t A_u8PortID, u8_t A_u8PinID)

- void MGPIOx_vSetPinValue (u8_t A_u8PortID, u8_t A_u8PinID, u8_t A_u8PinValue)
- void MGPIOx_vSetResetAtomic (u8_t A_u8PortID, u8_t A_u8PinID, u8_t A_u8SetResetPinValue)
- void MGPIOx_vSetAlternateFunctionON (u8_t A_u8PortID, u8_t A_u8PinID, u8_t A_u8AFID)
- void MGPIOx_vInit (MGPIOx_ConfigType *A_xPinConfig)
- void MGPIOx_vTogglePinValue (u8_t A_u8PortID, u8_t A_u8PinID)

7.18.1 Detailed Description

This file contains the source code of the interfacing for the GPIO modules.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

08/22/2022

Definition in file GPIO_program.c.

7.18.2 Function Documentation

7.18.2.1 MGPIOx_vLockedPins()

```
void MGPIOx_vLockedPins (
            void  )
```

Locks the prohibited GPIO PINs.

Definition at line 24 of file GPIO_program.c.

```
00025 {
00026
00027     VAR(volatile u32_t) L_u32LockGPIOA = INITIAL_ZERO ;
00028
00029     /* Lock key write sequence */
00030
00031     /* WR LCKR[16] = 1 + LCKR[13,14,15] = 1 */
00032     L_u32LockGPIOA = ( (1UL << LCKK_BIT_POS) | (GPIOA_PIN_POS) ) ;
00033
00034     GPIOA->LCKRx = L_u32LockGPIOA ;
00035
00036     /* WR LCKR[16] = 0 + LCKR[13,14,15] should not change*/
00037     GPIOA->LCKRx = (GPIOA_PIN_POS) ;
00038
00039     /* WR LCKR[16] = 1 + LCKR[13,14,15] should not change*/
00040     GPIOA->LCKRx = L_u32LockGPIOA ;
00041
00042     /* RD LCKR */
00043     L_u32LockGPIOA = GPIOA->LCKRx ;
00044
00046
00047     VAR(volatile u32_t) L_u32LockGPIOB = INITIAL_ZERO ;
```

```
00048
00049     /* Lock key write sequence */
00050
00051     /* WR LCKR[16] = 1 + LCKR[2,3,4] = 1 */
00052     L_u32LockGPIOB = ( (1UL << LCKK_BIT_POS) | (GPIOB_PIN_POS) ) ;
00053
00054     GPIOB->LCKRx = L_u32LockGPIOB ;
00055
00056     /* WR LCKR[16] = 0 + LCKR[2,3,4] should not change*/
00057     GPIOB->LCKRx = (GPIOB_PIN_POS) ;
00058
00059     /* WR LCKR[16] = 1 + LCKR[2,3,4] should not change*/
00060     GPIOB->LCKRx = L_u32LockGPIOB ;
00061
00062     /* RD LCKR */
00063     L_u32LockGPIOB = GPIOB->LCKRx ;
00064
00065 }
```

References GPIOA, GPIOA_PIN_POS, GPIOB, GPIOB_PIN_POS, INITIAL_ZERO, LCKK_BIT_POS, and VAR.

7.18.2.2 MGPIOx_vSetPinMode()

```
void MGPIOx_vSetPinMode (
              u8_t A_u8PortID,
              u8_t A_u8PinID,
              u8_t A_u8Mode )
```

Definition at line 70 of file GPIO_program.c.

```
00071 {
00072
00073     switch (A_u8PortID)
00074     {
00075         case GPIO_PORTA:
00076             CLR_BITs(GPIOA->MODERx, 0b11, A_u8PinID, 2);
00077             SET_BITs(GPIOA->MODERx, A_u8Mode, A_u8PinID, 2);
00078             break;
00079
00080         case GPIO_PORTB:
00081             CLR_BITs(GPIOB->MODERx, 0b11, A_u8PinID, 2);
00082             SET_BITs(GPIOB->MODERx, A_u8Mode, A_u8PinID, 2);
00083             break;
00084
00085         case GPIO_PORTC:
00086             CLR_BITs(GPIOC->MODERx, 0b11, A_u8PinID, 2);
00087             SET_BITs(GPIOC->MODERx, A_u8Mode, A_u8PinID, 2);
00088             break;
00089     }
00090
00091 }
```

References CLR_BITs, GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, and SET_BITs.

Referenced by MGPIOx_vInit().

7.18.2.3 MGPIOx_vSetPinOutputType()

```
void MGPIOx_vSetPinOutputType (
              u8_t A_u8PortID,
              u8_t A_u8PinID,
              u8_t A_u8OutputType )
```

Definition at line 96 of file GPIO_program.c.

```
00097 {
```

```
00098
00099     switch (A_u8OutputType)
00100     {
00101         case GPIOx_OPENDRAIN:
00102             switch (A_u8PortID)
00103             {
00104             case GPIO_PORTA:
00105                 SET_BIT(GPIOA->OTYPERx, A_u8PinID);
00106                 break;
00107
00108             case GPIO_PORTB:
00109                 SET_BIT(GPIOB->OTYPERx, A_u8PinID);
00110                 break;
00111
00112             case GPIO_PORTC:
00113                 SET_BIT(GPIOC->OTYPERx, A_u8PinID);
00114                 break;
00115             }
00116             break;
00117
00118         case GPIOx_PUSHPULL:
00119             switch(A_u8PortID)
00120             {
00121             case GPIO_PORTA:
00122                 CLR_BIT(GPIOA->OTYPERx, A_u8PinID);
00123                 break;
00124
00125             case GPIO_PORTB:
00126                 CLR_BIT(GPIOB->OTYPERx, A_u8PinID);
00127                 break;
00128
00129             case GPIO_PORTC:
00130                 CLR_BIT(GPIOC->OTYPERx, A_u8PinID);
00131                 break;
00132             }
00133             break;
00134     }
00135
00136 }
```

References CLR_BIT, GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, GPIOx_OPENDRAIN, GPIOx_PUSHPULL, and SET_BIT.

Referenced by MGPIOx_vInit().

7.18.2.4 MGPIOx_vSetPinOutputSpeed()

```
void MGPIOx_vSetPinOutputSpeed (
             u8_t A_u8PortID,
             u8_t A_u8PinID,
             u8_t A_u8OutputSpeed )
```

Definition at line 141 of file GPIO_program.c.

```
00142 {
00143
00144     switch(A_u8PortID)
00145     {
00146         case GPIO_PORTA:
00147             CLR_BITs(GPIOA->OSPEEDRx, 0b11, A_u8PinID, 2);
00148             SET_BITs(GPIOA->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00149             break;
00150
00151         case GPIO_PORTB:
00152             CLR_BITs(GPIOB->OSPEEDRx, 0b11, A_u8PinID, 2);
00153             SET_BITs(GPIOB->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00154             break;
00155
00156         case GPIO_PORTC:
00157             CLR_BITs(GPIOC->OSPEEDRx, 0b11, A_u8PinID, 2);
00158             SET_BITs(GPIOC->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00159             break;
00160     }
00161
00162 }
```

References CLR_BITs, GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, and SET_BITs.

Referenced by MGPIOx_vInit().

7.18.2.5 MGPIOx_vSetPinInputPullType()

```
void MGPIOx_vSetPinInputPullType (
              u8_t A_u8PortID,
              u8_t A_u8PinID,
              u8_t A_u8InputPullType )
```

Definition at line 167 of file GPIO_program.c.

```
00168 {
00169
00170     switch (A_u8PortID)
00171     {
00172         case GPIO_PORTA:
00173             CLR_BITs(GPIOA->PUPDRx, 0b11, A_u8PinID, 2);
00174             SET_BITs(GPIOA->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00175             break;
00176
00177         case GPIO_PORTB:
00178             CLR_BITs(GPIOB->PUPDRx, 0b11, A_u8PinID, 2);
00179             SET_BITs(GPIOB->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00180             break;
00181
00182         case GPIO_PORTC:
00183             CLR_BITs(GPIOC->PUPDRx, 0b11, A_u8PinID, 2);
00184             SET_BITs(GPIOC->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00185             break;
00186     }
00187
00188 }
```

References CLR_BITs, GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, and SET_BITs.

Referenced by MGPIOx_vInit().

7.18.2.6 MGPIOx_u8GetPinValue()

```
u8_t MGPIOx_u8GetPinValue (
              u8_t A_u8PortID,
              u8_t A_u8PinID )
```

Definition at line 193 of file GPIO_program.c.

```
00194 {
00195
00196     VAR(u8_t) L_u8PinValue = INITIAL_ZERO;
00197
00198     switch (A_u8PortID)
00199     {
00200         case GPIO_PORTA:
00201             L_u8PinValue = GET_BIT(GPIOA->IDRx, A_u8PinID);
00202             break;
00203
00204         case GPIO_PORTB:
00205             L_u8PinValue = GET_BIT(GPIOB->IDRx, A_u8PinID);
00206             break;
00207
00208         case GPIO_PORTC:
00209             L_u8PinValue = GET_BIT(GPIOC->IDRx, A_u8PinID);
00210             break;
00211     }
00212
00213     return L_u8PinValue;
00214
00215 }
```

References GET_BIT, GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, INITIAL_ZERO, and VAR.

7.18.2.7 MGPIOx_vSetPinValue()

```
void MGPIOx_vSetPinValue (
             u8_t A_u8PortID,
             u8_t A_u8PinID,
             u8_t A_u8PinValue )
```

Definition at line 220 of file GPIO_program.c.

```
00221 {
00222
00223     switch (A_u8PinValue)
00224     {
00225         case GPIOx_HIGH:
00226             switch (A_u8PortID)
00227             {
00228                 case GPIO_PORTA:
00229                     SET_BIT(GPIOA->ODRx, A_u8PinID);
00230                     break;
00231
00232                 case GPIO_PORTB:
00233                     SET_BIT(GPIOB->ODRx, A_u8PinID);
00234                     break;
00235
00236                 case GPIO_PORTC:
00237                     SET_BIT(GPIOC->ODRx, A_u8PinID);
00238                     break;
00239             }
00240             break;
00241
00242         case GPIOx_LOW:
00243             switch( A_u8PortID )
00244             {
00245                 case GPIO_PORTA:
00246                     CLR_BIT(GPIOA->ODRx, A_u8PinID);
00247                     break;
00248
00249                 case GPIO_PORTB:
00250                     CLR_BIT(GPIOB->ODRx, A_u8PinID);
00251                     break;
00252
00253                 case GPIO_PORTC:
00254                     CLR_BIT(GPIOC->ODRx, A_u8PinID);
00255                     break;
00256             }
00257             break;
00258     }
00259
00260 }
```

References CLR_BIT, GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, GPIOx_HIGH, GPIOx_LOW, and SET_BIT.

7.18.2.8 MGPIOx_vSetResetAtomic()

```
void MGPIOx_vSetResetAtomic (
             u8_t A_u8PortID,
             u8_t A_u8PinID,
             u8_t A_u8SetResetPinValue )
```

Definition at line 265 of file GPIO_program.c.

```
00266 {
00267
00268     switch (A_u8PortID)
00269     {
00270
00271         case GPIO_PORTA:
00272
00273         switch (A_u8SetResetPinValue)
00274         {
00275             case GPIOx_HIGH:
00276             GPIOA->BSRRx = (1 « A_u8PinID);
```

```
00277             break;
00278
00279             case GPIOx_LOW:
00280             GPIOA->BSRRx = (1 « (A_u8PinID + 16));
00281             break;
00282         }
00283
00284         break;
00285
00286         case GPIO_PORTB:
00287
00288             switch (A_u8SetResetPinValue)
00289             {
00290                 case GPIOx_HIGH:
00291                 GPIOB->BSRRx = (1 « A_u8PinID);
00292                 break;
00293
00294                 case GPIOx_LOW:
00295                 GPIOB->BSRRx = (1 « (A_u8PinID + 16));
00296                 break;
00297             }
00298
00299             break;
00300
00301         case GPIO_PORTC:
00302
00303                 switch (A_u8SetResetPinValue)
00304                 {
00305                     case GPIOx_HIGH:
00306                     GPIOC->BSRRx = (1 « A_u8PinID);
00307                     break;
00308
00309                     case GPIOx_LOW:
00310                     GPIOC->BSRRx = (1 « (A_u8PinID + 16));
00311                     break;
00312                 }
00313
00314                 break;
00315     }
00316
00317 }
```

References GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, GPIOx_HIGH, and GPIOx_LOW.

7.18.2.9 MGPIOx_vSetAlternateFunctionON()

```
void MGPIOx_vSetAlternateFunctionON (
             u8_t A_u8PortID,
             u8_t A_u8PinID,
             u8_t A_u8AFID )
```

Definition at line 322 of file GPIO_program.c.

```
00323 {
00324
00325     switch (A_u8PortID)
00326     {
00327         case GPIO_PORTA:
00328             switch (A_u8PinID)
00329             {
00330                 case GPIOx_PIN0:
00331                 case GPIOx_PIN1:
00332                 case GPIOx_PIN2:
00333                 case GPIOx_PIN3:
00334                 case GPIOx_PIN4:
00335                 case GPIOx_PIN5:
00336                 case GPIOx_PIN6:
00337                 case GPIOx_PIN7:
00338                     CLR_BITs(GPIOA->AFRLx, 0b1111, A_u8PinID, 4);
00339                     SET_BITs(GPIOA->AFRLx, A_u8AFID, A_u8PinID, 4);
00340                     break;
00341
00342                 case GPIOx_PIN8:
00343                 case GPIOx_PIN9:
00344                 case GPIOx_PIN10:
00345                 case GPIOx_PIN11:
```

```
00346                 case GPIOx_PIN12:
00347                 case GPIOx_PIN13:
00348                 case GPIOx_PIN14:
00349                 case GPIOx_PIN15:
00350                     CLR_BITs(GPIOA->AFRHx, 0b1111, A_u8PinID, 4);
00351                     SET_BITs(GPIOA->AFRHx, A_u8AFID, A_u8PinID, 4);
00352                     break;
00353             }
00354             break;
00355
00356         case GPIO_PORTB:
00357             switch (A_u8PinID)
00358             {
00359                 case GPIOx_PIN0:
00360                 case GPIOx_PIN1:
00361                 case GPIOx_PIN2:
00362                 case GPIOx_PIN3:
00363                 case GPIOx_PIN4:
00364                 case GPIOx_PIN5:
00365                 case GPIOx_PIN6:
00366                 case GPIOx_PIN7:
00367                     CLR_BITs(GPIOB->AFRLx, 0b1111, A_u8PinID, 4);
00368                     SET_BITs(GPIOB->AFRLx, A_u8AFID, A_u8PinID, 4);
00369                     break;
00370
00371                 case GPIOx_PIN8:
00372                 case GPIOx_PIN9:
00373                 case GPIOx_PIN10:
00374                 case GPIOx_PIN11:
00375                 case GPIOx_PIN12:
00376                 case GPIOx_PIN13:
00377                 case GPIOx_PIN14:
00378                 case GPIOx_PIN15:
00379                     CLR_BITs(GPIOB->AFRHx, 0b1111, A_u8PinID-8, 4);
00380                     SET_BITs(GPIOB->AFRHx, A_u8AFID, A_u8PinID-8, 4);
00381                     break;
00382             }
00383             break;
00384
00385         case GPIO_PORTC:
00386             switch (A_u8PinID)
00387             {
00388                 case GPIOx_PIN0:
00389                 case GPIOx_PIN1:
00390                 case GPIOx_PIN2:
00391                 case GPIOx_PIN3:
00392                 case GPIOx_PIN4:
00393                 case GPIOx_PIN5:
00394                 case GPIOx_PIN6:
00395                 case GPIOx_PIN7:
00396                     CLR_BITs(GPIOC->AFRLx, 0b1111, A_u8PinID, 4);
00397                     SET_BITs(GPIOC->AFRLx, A_u8AFID, A_u8PinID, 4);
00398                     break;
00399
00400                 case GPIOx_PIN8:
00401                 case GPIOx_PIN9:
00402                 case GPIOx_PIN10:
00403                 case GPIOx_PIN11:
00404                 case GPIOx_PIN12:
00405                 case GPIOx_PIN13:
00406                 case GPIOx_PIN14:
00407                 case GPIOx_PIN15:
00408                     CLR_BITs(GPIOC->AFRHx, 0b1111, A_u8PinID, 4);
00409                     SET_BITs(GPIOC->AFRHx, A_u8AFID, A_u8PinID, 4);
00410                     break;
00411             }
00412             break;
00413     }
00414
00415 }
```

References CLR_BITs, GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, GPIOx_PIN0, GPIOx_PIN1, GPIOx_PIN10, GPIOx_PIN11, GPIOx_PIN12, GPIOx_PIN13, GPIOx_PIN14, GPIOx_PIN15, GPIOx_PIN2, GPIOx_PIN3, GPIOx_PIN4, GPIOx_PIN5, GPIOx_PIN6, GPIOx_PIN7, GPIOx_PIN8, GPIOx_PIN9, and SET_BITs.

Referenced by MGPIOx_vInit().

7.18.2.10 MGPIOx_vInit()

```
void MGPIOx_vInit (
            MGPIOx_ConfigType * A_xPinConfig )
```

Definition at line 420 of file GPIO_program.c.

```
00421 {
00422
00423     MGPIOx_vSetPinMode               (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->Mode
      );
00424     MGPIOx_vSetPinOutputType        (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->OutputType
      );
00425     MGPIOx_vSetPinOutputSpeed      (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->OutputSpeed
      );
00426     MGPIOx_vSetPinInputPullType      (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->InputType
      );
00427     MGPIOx_vSetAlternateFunctionON    (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->AF_Type
      );
00428
00429 }
```

References MGPIOx_ConfigType::AF_Type, MGPIOx_ConfigType::InputType, MGPIOx_vSetAlternateFunctionON(), MGPIOx_vSetPinInputPullType(), MGPIOx_vSetPinMode(), MGPIOx_vSetPinOutputSpeed(), MGPIOx_vSetPinOutputType(), MGPIOx_ConfigType::Mode, MGPIOx_ConfigType::OutputSpeed, MGPIOx_ConfigType::OutputType, MGPIOx_ConfigType::Pin, and MGPIOx_ConfigType::Port.

7.18.2.11 MGPIOx_vTogglePinValue()

```
void MGPIOx_vTogglePinValue (
            u8_t A_u8PortID,
            u8_t A_u8PinID )
```

Definition at line 434 of file GPIO_program.c.

```
00435 {
00436
00437     switch (A_u8PortID)
00438     {
00439         case GPIO_PORTA:
00440             TOGGLE_BIT(GPIOA->ODRx, A_u8PinID);
00441             break;
00442
00443         case GPIO_PORTB:
00444             TOGGLE_BIT(GPIOB->ODRx, A_u8PinID);
00445             break;
00446
00447         case GPIO_PORTC:
00448             TOGGLE_BIT(GPIOC->ODRx, A_u8PinID);
00449             break;
00450     }
00451
00452 }
```

References GPIO_PORTA, GPIO_PORTB, GPIO_PORTC, GPIOA, GPIOB, GPIOC, and TOGGLE_BIT.

7.19 GPIO_program.c

Go to the documentation of this file.

```
00001
00009 /*****************************************************************************/
00010 /*                          Include headers                                  */
00011 /*****************************************************************************/
00012 #include "../../LIB/LSTD_TYPES.h"
00013 #include "../../LIB/LSTD_COMPILER.h"
00014 #include "../../LIB/LSTD_VALUES.h"
00015 #include "../../LIB/LSTD_BITMATH.h"
```

```
00016 #include "GPIO_interface.h"
00017 #include "GPIO_private.h"
00018 #include "GPIO_config.h"
00019
00020 /*****************************************************************************/
00021 /*                      Functions' implementations                           */
00022 /*****************************************************************************/
00023
00024 FUNC(void) MGPIOx_vLockedPins(void)
00025 {
00026
00027     VAR(volatile u32_t) L_u32LockGPIOA = INITIAL_ZERO ;
00028
00029     /* Lock key write sequence */
00030
00031     /* WR LCKR[16] = 1 + LCKR[13,14,15] = 1 */
00032     L_u32LockGPIOA = ( (1UL << LCKK_BIT_POS) | (GPIOA_PIN_POS) ) ;
00033
00034     GPIOA->LCKRx = L_u32LockGPIOA ;
00035
00036     /* WR LCKR[16] = 0 + LCKR[13,14,15] should not change*/
00037     GPIOA->LCKRx = (GPIOA_PIN_POS) ;
00038
00039     /* WR LCKR[16] = 1 + LCKR[13,14,15] should not change*/
00040     GPIOA->LCKRx = L_u32LockGPIOA ;
00041
00042     /* RD LCKR */
00043     L_u32LockGPIOA = GPIOA->LCKRx ;
00044
00046
00047     VAR(volatile u32_t) L_u32LockGPIOB = INITIAL_ZERO ;
00048
00049     /* Lock key write sequence */
00050
00051     /* WR LCKR[16] = 1 + LCKR[2,3,4] = 1 */
00052     L_u32LockGPIOB = ( (1UL << LCKK_BIT_POS) | (GPIOB_PIN_POS) ) ;
00053
00054     GPIOB->LCKRx = L_u32LockGPIOB ;
00055
00056     /* WR LCKR[16] = 0 + LCKR[2,3,4] should not change*/
00057     GPIOB->LCKRx = (GPIOB_PIN_POS) ;
00058
00059     /* WR LCKR[16] = 1 + LCKR[2,3,4] should not change*/
00060     GPIOB->LCKRx = L_u32LockGPIOB ;
00061
00062     /* RD LCKR */
00063     L_u32LockGPIOB = GPIOB->LCKRx ;
00064
00065 }
00066
00067 /*****************************************************************************************/
00068 /*****************************************************************************************/
00069
00070 FUNC(void) MGPIOx_vSetPinMode(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8Mode)
00071 {
00072
00073     switch (A_u8PortID)
00074     {
00075         case GPIO_PORTA:
00076             CLR_BITs(GPIOA->MODERx, 0b11, A_u8PinID, 2);
00077             SET_BITs(GPIOA->MODERx, A_u8Mode, A_u8PinID, 2);
00078             break;
00079
00080         case GPIO_PORTB:
00081             CLR_BITs(GPIOB->MODERx, 0b11, A_u8PinID, 2);
00082             SET_BITs(GPIOB->MODERx, A_u8Mode, A_u8PinID, 2);
00083             break;
00084
00085         case GPIO_PORTC:
00086             CLR_BITs(GPIOC->MODERx, 0b11, A_u8PinID, 2);
00087             SET_BITs(GPIOC->MODERx, A_u8Mode, A_u8PinID, 2);
00088             break;
00089     }
00090
00091 }
00092
00093 /*****************************************************************************************/
00094 /*****************************************************************************************/
00095
00096 FUNC(void) MGPIOx_vSetPinOutputType(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
      A_u8OutputType)
00097 {
00098
00099     switch (A_u8OutputType)
00100     {
00101         case GPIOx_OPENDRAIN:
00102             switch (A_u8PortID)
```

```
00103             {
00104             case GPIO_PORTA:
00105                 SET_BIT(GPIOA->OTYPERx, A_u8PinID);
00106                 break;
00107 
00108             case GPIO_PORTB:
00109                 SET_BIT(GPIOB->OTYPERx, A_u8PinID);
00110                 break;
00111 
00112             case GPIO_PORTC:
00113                 SET_BIT(GPIOC->OTYPERx, A_u8PinID);
00114                 break;
00115             }
00116             break;
00117 
00118         case GPIOx_PUSHPULL:
00119             switch(A_u8PortID)
00120             {
00121             case GPIO_PORTA:
00122                 CLR_BIT(GPIOA->OTYPERx, A_u8PinID);
00123                 break;
00124 
00125             case GPIO_PORTB:
00126                 CLR_BIT(GPIOB->OTYPERx, A_u8PinID);
00127                 break;
00128 
00129             case GPIO_PORTC:
00130                 CLR_BIT(GPIOC->OTYPERx, A_u8PinID);
00131                 break;
00132             }
00133             break;
00134     }
00135 
00136 }
00137 
00138 /*********************************************************************************************/
00139 /*********************************************************************************************/
00140 
00141 FUNC(void) MGPIOx_vSetPinOutputSpeed(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
    A_u8OutputSpeed)
00142 {
00143 
00144     switch(A_u8PortID)
00145     {
00146         case GPIO_PORTA:
00147             CLR_BITs(GPIOA->OSPEEDRx, 0b11, A_u8PinID, 2);
00148             SET_BITs(GPIOA->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00149             break;
00150 
00151         case GPIO_PORTB:
00152             CLR_BITs(GPIOB->OSPEEDRx, 0b11, A_u8PinID, 2);
00153             SET_BITs(GPIOB->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00154             break;
00155 
00156         case GPIO_PORTC:
00157             CLR_BITs(GPIOC->OSPEEDRx, 0b11, A_u8PinID, 2);
00158             SET_BITs(GPIOC->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00159             break;
00160     }
00161 
00162 }
00163 
00164 /*********************************************************************************************/
00165 /*********************************************************************************************/
00166 
00167 FUNC(void) MGPIOx_vSetPinInputPullType(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
    A_u8InputPullType)
00168 {
00169 
00170     switch (A_u8PortID)
00171     {
00172         case GPIO_PORTA:
00173             CLR_BITs(GPIOA->PUPDRx, 0b11, A_u8PinID, 2);
00174             SET_BITs(GPIOA->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00175             break;
00176 
00177         case GPIO_PORTB:
00178             CLR_BITs(GPIOB->PUPDRx, 0b11, A_u8PinID, 2);
00179             SET_BITs(GPIOB->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00180             break;
00181 
00182         case GPIO_PORTC:
00183             CLR_BITs(GPIOC->PUPDRx, 0b11, A_u8PinID, 2);
00184             SET_BITs(GPIOC->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00185             break;
00186     }
00187 
```

```
00188 }
00189
00190 /*********************************************************************************************/
00191 /*********************************************************************************************/
00192
00193 FUNC(u8_t) MGPIOx_u8GetPinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID)
00194 {
00195
00196     VAR(u8_t) L_u8PinValue = INITIAL_ZERO;
00197
00198     switch (A_u8PortID)
00199     {
00200         case GPIO_PORTA:
00201             L_u8PinValue = GET_BIT(GPIOA->IDRx, A_u8PinID);
00202             break;
00203
00204         case GPIO_PORTB:
00205             L_u8PinValue = GET_BIT(GPIOB->IDRx, A_u8PinID);
00206             break;
00207
00208         case GPIO_PORTC:
00209             L_u8PinValue = GET_BIT(GPIOC->IDRx, A_u8PinID);
00210             break;
00211     }
00212
00213     return L_u8PinValue;
00214
00215 }
00216
00217 /*********************************************************************************************/
00218 /*********************************************************************************************/
00219
00220 FUNC(void) MGPIOx_vSetPinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8PinValue)
00221 {
00222
00223     switch (A_u8PinValue)
00224     {
00225         case GPIOx_HIGH:
00226             switch (A_u8PortID)
00227             {
00228                 case GPIO_PORTA:
00229                     SET_BIT(GPIOA->ODRx, A_u8PinID);
00230                     break;
00231
00232                 case GPIO_PORTB:
00233                     SET_BIT(GPIOB->ODRx, A_u8PinID);
00234                     break;
00235
00236                 case GPIO_PORTC:
00237                     SET_BIT(GPIOC->ODRx, A_u8PinID);
00238                     break;
00239             }
00240             break;
00241
00242         case GPIOx_LOW:
00243             switch( A_u8PortID )
00244             {
00245                 case GPIO_PORTA:
00246                     CLR_BIT(GPIOA->ODRx, A_u8PinID);
00247                     break;
00248
00249                 case GPIO_PORTB:
00250                     CLR_BIT(GPIOB->ODRx, A_u8PinID);
00251                     break;
00252
00253                 case GPIO_PORTC:
00254                     CLR_BIT(GPIOC->ODRx, A_u8PinID);
00255                     break;
00256             }
00257             break;
00258     }
00259
00260 }
00261
00262 /*********************************************************************************************/
00263 /*********************************************************************************************/
00264
00265 FUNC(void) MGPIOx_vSetResetAtomic(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
      A_u8SetResetPinValue)
00266 {
00267
00268     switch (A_u8PortID)
00269     {
00270
00271         case GPIO_PORTA:
00272
00273         switch (A_u8SetResetPinValue)
```

```
00274         {
00275             case GPIOx_HIGH:
00276             GPIOA->BSRRx = (1 « A_u8PinID);
00277             break;
00278 
00279             case GPIOx_LOW:
00280             GPIOA->BSRRx = (1 « (A_u8PinID + 16));
00281             break;
00282         }
00283 
00284         break;
00285 
00286         case GPIO_PORTB:
00287 
00288             switch (A_u8SetResetPinValue)
00289             {
00290                 case GPIOx_HIGH:
00291                 GPIOB->BSRRx = (1 « A_u8PinID);
00292                 break;
00293 
00294                 case GPIOx_LOW:
00295                 GPIOB->BSRRx = (1 « (A_u8PinID + 16));
00296                 break;
00297             }
00298 
00299             break;
00300 
00301         case GPIO_PORTC:
00302 
00303                 switch (A_u8SetResetPinValue)
00304                 {
00305                     case GPIOx_HIGH:
00306                     GPIOC->BSRRx = (1 « A_u8PinID);
00307                     break;
00308 
00309                     case GPIOx_LOW:
00310                     GPIOC->BSRRx = (1 « (A_u8PinID + 16));
00311                     break;
00312                 }
00313 
00314                 break;
00315     }
00316 
00317 }
00318 
00319 /*********************************************************************************************/
00320 /*********************************************************************************************/
00321 
00322 FUNC(void) MGPIOx_vSetAlternateFunctionON(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
      A_u8AFID)
00323 {
00324 
00325     switch (A_u8PortID)
00326     {
00327         case GPIO_PORTA:
00328             switch (A_u8PinID)
00329             {
00330                 case GPIOx_PIN0:
00331                 case GPIOx_PIN1:
00332                 case GPIOx_PIN2:
00333                 case GPIOx_PIN3:
00334                 case GPIOx_PIN4:
00335                 case GPIOx_PIN5:
00336                 case GPIOx_PIN6:
00337                 case GPIOx_PIN7:
00338                     CLR_BITs(GPIOA->AFRLx, 0b1111, A_u8PinID, 4);
00339                     SET_BITs(GPIOA->AFRLx, A_u8AFID, A_u8PinID, 4);
00340                     break;
00341 
00342                 case GPIOx_PIN8:
00343                 case GPIOx_PIN9:
00344                 case GPIOx_PIN10:
00345                 case GPIOx_PIN11:
00346                 case GPIOx_PIN12:
00347                 case GPIOx_PIN13:
00348                 case GPIOx_PIN14:
00349                 case GPIOx_PIN15:
00350                     CLR_BITs(GPIOA->AFRHx, 0b1111, A_u8PinID, 4);
00351                     SET_BITs(GPIOA->AFRHx, A_u8AFID, A_u8PinID, 4);
00352                     break;
00353             }
00354             break;
00355 
00356         case GPIO_PORTB:
00357             switch (A_u8PinID)
00358             {
00359                 case GPIOx_PIN0:
```

```
00360                 case GPIOx_PIN1:
00361                 case GPIOx_PIN2:
00362                 case GPIOx_PIN3:
00363                 case GPIOx_PIN4:
00364                 case GPIOx_PIN5:
00365                 case GPIOx_PIN6:
00366                 case GPIOx_PIN7:
00367                     CLR_BITs(GPIOB->AFRLx, 0b1111, A_u8PinID, 4);
00368                     SET_BITs(GPIOB->AFRLx, A_u8AFID, A_u8PinID, 4);
00369                     break;
00370 
00371                 case GPIOx_PIN8:
00372                 case GPIOx_PIN9:
00373                 case GPIOx_PIN10:
00374                 case GPIOx_PIN11:
00375                 case GPIOx_PIN12:
00376                 case GPIOx_PIN13:
00377                 case GPIOx_PIN14:
00378                 case GPIOx_PIN15:
00379                     CLR_BITs(GPIOB->AFRHx, 0b1111, A_u8PinID-8, 4);
00380                     SET_BITs(GPIOB->AFRHx, A_u8AFID, A_u8PinID-8, 4);
00381                     break;
00382             }
00383             break;
00384 
00385         case GPIO_PORTC:
00386             switch (A_u8PinID)
00387             {
00388                 case GPIOx_PIN0:
00389                 case GPIOx_PIN1:
00390                 case GPIOx_PIN2:
00391                 case GPIOx_PIN3:
00392                 case GPIOx_PIN4:
00393                 case GPIOx_PIN5:
00394                 case GPIOx_PIN6:
00395                 case GPIOx_PIN7:
00396                     CLR_BITs(GPIOC->AFRLx, 0b1111, A_u8PinID, 4);
00397                     SET_BITs(GPIOC->AFRLx, A_u8AFID, A_u8PinID, 4);
00398                     break;
00399 
00400                 case GPIOx_PIN8:
00401                 case GPIOx_PIN9:
00402                 case GPIOx_PIN10:
00403                 case GPIOx_PIN11:
00404                 case GPIOx_PIN12:
00405                 case GPIOx_PIN13:
00406                 case GPIOx_PIN14:
00407                 case GPIOx_PIN15:
00408                     CLR_BITs(GPIOC->AFRHx, 0b1111, A_u8PinID, 4);
00409                     SET_BITs(GPIOC->AFRHx, A_u8AFID, A_u8PinID, 4);
00410                     break;
00411             }
00412             break;
00413     }
00414 
00415 }
00416 
00417 /*********************************************************************************************/
00418 /*********************************************************************************************/
00419 
00420 FUNC(void) MGPIOx_vInit(P2VAR(MGPIOx_ConfigType) A_xPinConfig)
00421 {
00422 
00423     MGPIOx_vSetPinMode                (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->Mode
      );
00424     MGPIOx_vSetPinOutputType         (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->OutputType
      );
00425     MGPIOx_vSetPinOutputSpeed       (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->OutputSpeed
      );
00426     MGPIOx_vSetPinInputPullType       (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->InputType
      );
00427     MGPIOx_vSetAlternateFunctionON     (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->AF_Type
      );
00428 
00429 }
00430 
00431 /*********************************************************************************************/
00432 /*********************************************************************************************/
00433 
00434 FUNC(void) MGPIOx_vTogglePinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID)
00435 {
00436 
00437     switch (A_u8PortID)
00438     {
00439         case GPIO_PORTA:
00440             TOGGLE_BIT(GPIOA->ODRx, A_u8PinID);
00441             break;
```

```
00442
00443         case GPIO_PORTB:
00444             TOGGLE_BIT(GPIOB->ODRx, A_u8PinID);
00445             break;
00446
00447         case GPIO_PORTC:
00448             TOGGLE_BIT(GPIOC->ODRx, A_u8PinID);
00449             break;
00450     }
00451
00452 }
00453
00454 /*********************************************************************************************/
00455 /*********************************************************************************************/
00456
00457
00458
```

7.20 src/COTS/MCAL/RCC/MRCC_config.h File Reference

This file contains the RCC configurations.

Macros

- #define PLLI2S DISABLE
- #define PLL DISABLE
- #define CSS ENABLE
- #define HSEBYP NOTBYBASED
- #define HSE_EN DISABLE
- #define HSI_EN ENABLE
- #define PLLQ Equal_2
- #define PLLSRC HSE
- #define PLLP PLLCLK
- #define PLLN Equal_200
- #define PLLM Equal_2
- #define MCO2 SYSCLK
- #define MCO2PRE NoDivision
- #define MCO1PRE NoDivision
- #define I2SSRC PLLI2SCLK
- #define MCO1 HSE
- #define RTCPRE HSEby2
- #define PPRE2 NoDivision
- #define PPRE1 NoDivision
- #define HPRE SYSCLKby2
- #define SWS HSI
- #define SW HSI
- #define DMA2CLK DISABLE
- #define DMA1CLK DISABLE
- #define CRCCLK DISABLE
- #define GPIOHCLK DISABLE
- #define GPIOECLK DISABLE
- #define GPIODCLK DISABLE
- #define GPIOCCLK DISABLE
- #define GPIOBCLK DISABLE
- #define GPIOACLK DISABLE
- #define OTGFS DISABLE
- #define PWREN DISABLE

- #define I2C3EN DISABLE
- #define I2C2EN DISABLE
- #define I2C1EN DISABLE
- #define USART2EN DISABLE
- #define SPI3EN DISABLE
- #define SPI2EN DISABLE
- #define WWDGEN DISABLE
- #define TIM5EN DISABLE
- #define TIM4EN DISABLE
- #define TIM3EN DISABLE
- #define TIM2EN DISABLE
- #define TIM11EN DISABLE
- #define TIM10EN DISABLE
- #define TIM9EN DISABLE
- #define SYSCFGEN DISABLE
- #define SPI4EN DISABLE
- #define SPI1EN DISABLE
- #define SDIOEN DISABLE
- #define ADC1EN DISABLE
- #define USART6EN DISABLE
- #define USART1EN DISABLE
- #define TIM1EN DISABLE

7.20.1 Detailed Description

This file contains the RCC configurations.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file MRCC_config.h.

7.20.2 Macro Definition Documentation

7.20.2.1 PLLI2S

```
#define PLLI2S DISABLE
```

Definition at line 21 of file MRCC_config.h.

7.20.2.2 PLL

```
#define PLL DISABLE
```

Definition at line 29 of file MRCC_config.h.

7.20.2.3 CSS

```
#define CSS ENABLE
```

Definition at line 37 of file MRCC_config.h.

7.20.2.4 HSEBYP

```
#define HSEBYP NOTBYBASED
```

Definition at line 45 of file MRCC_config.h.

7.20.2.5 HSE_EN

```
#define HSE_EN DISABLE
```

Definition at line 53 of file MRCC_config.h.

7.20.2.6 HSI_EN

```
#define HSI_EN ENABLE
```

Definition at line 61 of file MRCC_config.h.

7.20.2.7 PLLQ

```
#define PLLQ Equal_2
```

Definition at line 84 of file MRCC_config.h.

7.20.2.8 PLLSRC

```
#define PLLSRC HSE
```

Definition at line 92 of file MRCC_config.h.

7.20.2.9 PLLP

```
#define PLLP PLLCLK
```

Definition at line 102 of file MRCC_config.h.

7.20.2.10 PLLN

```
#define PLLN Equal_200
```

Definition at line 173 of file MRCC_config.h.

7.20.2.11 PLLM

```
#define PLLM Equal_2
```

Definition at line 240 of file MRCC_config.h.

7.20.2.12 MCO2

```
#define MCO2 SYSCLK
```

Definition at line 253 of file MRCC_config.h.

7.20.2.13 MCO2PRE

```
#define MCO2PRE NoDivision
```

Definition at line 264 of file MRCC_config.h.

7.20.2.14 MCO1PRE

```
#define MCO1PRE NoDivision
```

Definition at line 275 of file MRCC_config.h.

7.20.2.15 I2SSRC

```
#define I2SSRC PLLI2SCLK
```

Definition at line 283 of file MRCC_config.h.

7.20.2.16 MCO1

```
#define MCO1 HSE
```

Definition at line 293 of file MRCC_config.h.

7.20.2.17 RTCPRE

```
#define RTCPRE HSEby2
```

Definition at line 332 of file MRCC_config.h.

7.20.2.18 PPRE2

```
#define PPRE2 NoDivision
```

Definition at line 343 of file MRCC_config.h.

7.20.2.19 PPRE1

```
#define PPRE1 NoDivision
```

Definition at line 354 of file MRCC_config.h.

7.20.2.20 HPRE

```
#define HPRE SYSCLKby2
```

Definition at line 369 of file MRCC_config.h.

7.20.2.21 SWS

```
#define SWS HSI
```

Definition at line 378 of file MRCC_config.h.

7.20.2.22 SW

```
#define SW HSI
```

Definition at line 387 of file MRCC_config.h.

7.20.2.23 DMA2CLK

```
#define DMA2CLK DISABLE
```

Definition at line 398 of file MRCC_config.h.

7.20.2.24 DMA1CLK

```
#define DMA1CLK DISABLE
```

Definition at line 406 of file MRCC_config.h.

7.20.2.25 CRCCLK

```
#define CRCCLK DISABLE
```

Definition at line 414 of file MRCC_config.h.

7.20.2.26 GPIOHCLK

```
#define GPIOHCLK DISABLE
```

Definition at line 422 of file MRCC_config.h.

7.20.2.27 GPIOECLK

```
#define GPIOECLK DISABLE
```

Definition at line 430 of file MRCC_config.h.

7.20.2.28 GPIODCLK

```
#define GPIODCLK DISABLE
```

Definition at line 438 of file MRCC_config.h.

7.20.2.29 GPIOCCLK

```
#define GPIOCCLK DISABLE
```

Definition at line 446 of file MRCC_config.h.

7.20.2.30 GPIOBCLK

```
#define GPIOBCLK DISABLE
```

Definition at line 454 of file MRCC_config.h.

7.20.2.31 GPIOACLK

```
#define GPIOACLK DISABLE
```

Definition at line 462 of file MRCC_config.h.

7.20.2.32 OTGFS

```
#define OTGFS DISABLE
```

Definition at line 473 of file MRCC_config.h.

7.20.2.33 PWREN

```
#define PWREN DISABLE
```

Definition at line 484 of file MRCC_config.h.

7.20.2.34 I2C3EN

```
#define I2C3EN DISABLE
```

Definition at line 492 of file MRCC_config.h.

7.20.2.35 I2C2EN

```
#define I2C2EN DISABLE
```

Definition at line 500 of file MRCC_config.h.

7.20.2.36 I2C1EN

```
#define I2C1EN DISABLE
```

Definition at line 508 of file MRCC_config.h.

7.20.2.37 USART2EN

```
#define USART2EN DISABLE
```

Definition at line 516 of file MRCC_config.h.

7.20.2.38 SPI3EN

```
#define SPI3EN DISABLE
```

Definition at line 524 of file MRCC_config.h.

7.20.2.39 SPI2EN

```
#define SPI2EN DISABLE
```

Definition at line 532 of file MRCC_config.h.

7.20.2.40 WWDGEN

```
#define WWDGEN DISABLE
```

Definition at line 540 of file MRCC_config.h.

7.20.2.41 TIM5EN

```
#define TIM5EN DISABLE
```

Definition at line 548 of file MRCC_config.h.

7.20.2.42 TIM4EN

```
#define TIM4EN DISABLE
```

Definition at line 556 of file MRCC_config.h.

7.20.2.43 TIM3EN

```
#define TIM3EN DISABLE
```

Definition at line 564 of file MRCC_config.h.

7.20.2.44 TIM2EN

```
#define TIM2EN DISABLE
```

Definition at line 572 of file MRCC_config.h.

7.20.2.45 TIM11EN

```
#define TIM11EN DISABLE
```

Definition at line 583 of file MRCC_config.h.

7.20.2.46 TIM10EN

```
#define TIM10EN DISABLE
```

Definition at line 591 of file MRCC_config.h.

7.20.2.47 TIM9EN

```
#define TIM9EN DISABLE
```

Definition at line 599 of file MRCC_config.h.

7.20.2.48 SYSCFGEN

```
#define SYSCFGEN DISABLE
```

Definition at line 607 of file MRCC_config.h.

7.20.2.49 SPI4EN

```
#define SPI4EN DISABLE
```

Definition at line 615 of file MRCC_config.h.

7.20.2.50 SPI1EN

```
#define SPI1EN DISABLE
```

Definition at line 623 of file MRCC_config.h.

7.20.2.51 SDIOEN

```
#define SDIOEN DISABLE
```

Definition at line 631 of file MRCC_config.h.

7.20.2.52 ADC1EN

```
#define ADC1EN DISABLE
```

Definition at line 639 of file MRCC_config.h.

7.20.2.53 USART6EN

```
#define USART6EN DISABLE
```

Definition at line 647 of file MRCC_config.h.

7.20.2.54 USART1EN

```
#define USART1EN DISABLE
```

Definition at line 655 of file MRCC_config.h.

7.20.2.55 TIM1EN

```
#define TIM1EN DISABLE
```

Definition at line 663 of file MRCC_config.h.

7.21 MRCC_config.h

Go to the documentation of this file.

```
00001
00009 #ifndef MCAL_RCC_MRCC_CONFIG_H_
00010 #define MCAL_RCC_MRCC_CONFIG_H_
00011
00012
00013 /*****************************************************************************************/
00014 /*                              RCC_CR configurations                                    */
00015 /*****************************************************************************************/
00016
00017 /*options:
00018 *ENABLE
00019 *DISABLE
00020 */
00021 #define PLLI2S DISABLE
00022
00023 /*****************************************************************************************/
00024
00025 /*options:
00026 *ENABLE
00027 *DISABLE
00028 */
00029 #define PLL DISABLE
00030
00031 /*****************************************************************************************/
00032
00033 /*options:
00034 *ENABLE
00035 *DISABLE
00036 */
00037 #define CSS ENABLE
00038
00039 /*****************************************************************************************/
00040
00041 /*options:
00042 *BYBASED
00043 *NOTBYBASED
00044 */
00045 #define HSEBYP NOTBYBASED
00046
00047 /*****************************************************************************************/
00048
00049 /*options:
00050 *ENABLE
00051 *DISABLE
00052 */
00053 #define HSE_EN DISABLE
00054
00055 /*****************************************************************************************/
00056
00057 /*options:
00058 *ENABLE
00059 *DISABLE
00060 */
00061 #define HSI_EN ENABLE
00062
00063
00064 /*****************************************************************************************/
00065                               // RCC_PLLCFGR configurations //
00066 /*****************************************************************************************/
00067
00068 /*options:
00069 *Equal_2
00070 *Equal_3
00071 *Equal_4
00072 *Equal_5
00073 *Equal_6
00074 *Equal_7
00075 *Equal_8
00076 *Equal_9
00077 *Equal_10
00078 *Equal_11
00079 *Equal_12
00080 *Equal_13
00081 *Equal_14
00082 *Equal_15
00083 */
00084 #define PLLQ Equal_2
00085
00086 /*****************************************************************************************/
00087
00088 /*options:
00089 *HSI
```

```
00090 *HSE
00091 */
00092 #define PLLSRC HSE
00093
00094 /*******************************************************************************/
00095
00096 /*options:
00097 *Equal_2
00098 *Equal_4
00099 *Equal_6
00100 *Equal_8
00101 */
00102 #define PLLP PLLCLK
00103
00104 /*******************************************************************************/
00105
00106 /*options:
00107 *Equal_192
00108 *Equal_193
00109 *Equal_194
00110 *Equal_195
00111 *Equal_196
00112 *Equal_197
00113 *Equal_198
00114 *Equal_199
00115 *Equal_200
00116 *Equal_201
00117 *Equal_202
00118 *Equal_203
00119 *Equal_204
00120 *Equal_205
00121 *Equal_206
00122 *Equal_207
00123 *Equal_208
00124 *Equal_209
00125 *Equal_210
00126
00127 *Equal_300
00128 *Equal_301
00129 *Equal_302
00130 *Equal_303
00131 *Equal_304
00132 *Equal_305
00133 *Equal_306
00134 *Equal_307
00135 *Equal_308
00136 *Equal_309
00137 *Equal_310
00138
00139 *Equal_400
00140 *Equal_401
00141 *Equal_402
00142 *Equal_403
00143 *Equal_404
00144 *Equal_405
00145 *Equal_406
00146 *Equal_407
00147 *Equal_408
00148 *Equal_409
00149 *Equal_410
00150 *Equal_411
00151 *Equal_412
00152 *Equal_413
00153 *Equal_414
00154 *Equal_415
00155 *Equal_416
00156 *Equal_417
00157 *Equal_418
00158 *Equal_419
00159 *Equal_420
00160 *Equal_421
00161 *Equal_422
00162 *Equal_423
00163 *Equal_424
00164 *Equal_425
00165 *Equal_426
00166 *Equal_427
00167 *Equal_428
00168 *Equal_429
00169 *Equal_430
00170 *Equal_431
00171 *Equal_432
00172 */
00173 #define PLLN Equal_200
00174
00175 /*******************************************************************************/
00176
```

```
00177 /*options:
00178 *Equal_2
00179 *Equal_4
00180 *Equal_5
00181 *Equal_6
00182 *Equal_7
00183 *Equal_8
00184 *Equal_9
00185 *Equal_10
00186 *Equal_11
00187 *Equal_12
00188 *Equal_13
00189 *Equal_14
00190 *Equal_15
00191 *Equal_16
00192 *Equal_17
00193 *Equal_18
00194 *Equal_19
00195 *Equal_20
00196 *Equal_21
00197 *Equal_22
00198 *Equal_23
00199 *Equal_24
00200 *Equal_25
00201 *Equal_26
00202 *Equal_27
00203 *Equal_28
00204 *Equal_29
00205 *Equal_30
00206 *Equal_31
00207 *Equal_32
00208 *Equal_33
00209 *Equal_34
00210 *Equal_35
00211 *Equal_36
00212 *Equal_37
00213 *Equal_38
00214 *Equal_39
00215 *Equal_40
00216 *Equal_41
00217 *Equal_42
00218 *Equal_43
00219 *Equal_44
00220 *Equal_45
00221 *Equal_46
00222 *Equal_47
00223 *Equal_48
00224 *Equal_49
00225 *Equal_50
00226 *Equal_51
00227 *Equal_52
00228 *Equal_53
00229 *Equal_54
00230 *Equal_55
00231 *Equal_56
00232 *Equal_57
00233 *Equal_58
00234 *Equal_59
00235 *Equal_60
00236 *Equal_61
00237 *Equal_62
00238 *Equal_63
00239 */
00240 #define PLLM Equal_2
00241
00242
00243 /*******************************************************************************/
00244                             // RCC_CFGR configurations //
00245 /*******************************************************************************/
00246
00247 /*options:
00248 *SYSCLK
00249 *PLLI2SCLK
00250 *HSE
00251 *PLLCLK
00252 */
00253 #define MCO2 SYSCLK
00254
00255 /*******************************************************************************/
00256
00257 /*options:
00258 *NoDivision
00259 *DivisionBy2
00260 *DivisionBy3
00261 *DivisionBy4
00262 *DivisionBy5
00263 */
```

```
00264 #define MCO2PRE NoDivision
00265
00266 /*******************************************************************************/
00267
00268 /*options:
00269 *NoDivision
00270 *DivisionBy2
00271 *DivisionBy3
00272 *DivisionBy4
00273 *DivisionBy5
00274 */
00275 #define MCO1PRE NoDivision
00276
00277 /*******************************************************************************/
00278
00279 /*options:
00280 *PLLI2SCLK
00281 *I2S_CKIN
00282 */
00283 #define I2SSRC PLLI2SCLK
00284
00285 /*******************************************************************************/
00286
00287 /*options:
00288 *HSI
00289 *LSE
00290 *HSE
00291 *PLLCLK
00292 */
00293 #define MCO1 HSE
00294
00295 /*******************************************************************************/
00296
00297 /*options:
00298 *NoCLK0
00299 *NoCLK1
00300 *HSEby2
00301 *HSEby3
00302 *HSEby4
00303 *HSEby5
00304 *HSEby6
00305 *HSEby7
00306 *HSEby8
00307 *HSEby9
00308 *HSEby10
00309 *HSEby11
00310 *HSEby12
00311 *HSEby13
00312 *HSEby14
00313 *HSEby15
00314 *HSEby16
00315 *HSEby17
00316 *HSEby18
00317 *HSEby19
00318 *HSEby20
00319 *HSEby21
00320 *HSEby22
00321 *HSEby22
00322 *HSEby23
00323 *HSEby24
00324 *HSEby25
00325 *HSEby26
00326 *HSEby27
00327 *HSEby28
00328 *HSEby29
00329 *HSEby30
00330 *HSEby31
00331 */
00332 #define RTCPRE HSEby2
00333
00334 /*******************************************************************************/
00335
00336 /*options:
00337 *NoDivision
00338 *AHBby2
00339 *AHBby4
00340 *AHBBby8
00341 *AHBby16
00342 */
00343 #define PPRE2 NoDivision
00344
00345 /*******************************************************************************/
00346
00347 /*options:
00348 *NoDivision
00349 *AHBby2
00350 *AHBby4
```

```
00351 *AHBby8
00352 *AHBby16
00353 */
00354 #define PPRE1 NoDivision
00355
00356 /*****************************************************************************/
00357
00358 /*options:
00359 *NoDivision
00360 *SYSCLKby2
00361 *SYSCLKby4
00362 *SYSCLKby8
00363 *SYSCLKby16
00364 *SYSCLKby64
00365 *SYSCLKby128
00366 *SYSCLKby256
00367 *SYSCLKby512
00368 */
00369 #define HPRE SYSCLKby2
00370
00371 /*****************************************************************************/
00372
00373 /*options:
00374 *HSI
00375 *HSE
00376 *PLLCLK
00377 */
00378 #define SWS HSI
00379
00380 /*****************************************************************************/
00381
00382 /*options:
00383 *HSI
00384 *HSE
00385 *PLLCLK
00386 */
00387 #define SW HSI
00388
00389
00390 /*****************************************************************************/
00391                             // RCC_AHB1ENR configurations //
00392 /*****************************************************************************/
00393
00394 /*options:
00395 *ENABLE
00396 *DISABLE
00397 */
00398 #define DMA2CLK DISABLE
00399
00400 /*****************************************************************************/
00401
00402 /*options:
00403 *ENABLE
00404 *DISABLE
00405 */
00406 #define DMA1CLK DISABLE
00407
00408 /*****************************************************************************/
00409
00410 /*options:
00411 *ENABLE
00412 *DISABLE
00413 */
00414 #define CRCCLK DISABLE
00415
00416 /*****************************************************************************/
00417
00418 /*options:
00419 *ENABLE
00420 *DISABLE
00421 */
00422 #define GPIOHCLK DISABLE
00423
00424 /*****************************************************************************/
00425
00426 /*options:
00427 *ENABLE
00428 *DISABLE
00429 */
00430 #define GPIOECLK DISABLE
00431
00432 /*****************************************************************************/
00433
00434 /*options:
00435 *ENABLE
00436 *DISABLE
00437 */
```

```
00438 #define GPIODCLK DISABLE
00439
00440 /*****************************************************************************/
00441
00442 /*options:
00443 *ENABLE
00444 *DISABLE
00445 */
00446 #define GPIOCCLK DISABLE
00447
00448 /*****************************************************************************/
00449
00450 /*options:
00451 *ENABLE
00452 *DISABLE
00453 */
00454 #define GPIOBCLK DISABLE
00455
00456 /*****************************************************************************/
00457
00458 /*options:
00459 *ENABLE
00460 *DISABLE
00461 */
00462 #define GPIOACLK DISABLE
00463
00464
00465 /*****************************************************************************/
00466                             // RCC_AHB2ENR configurations //
00467 /*****************************************************************************/
00468
00469 /*options:
00470 *ENABLE
00471 *DISABLE
00472 */
00473 #define OTGFS DISABLE
00474
00475
00476 /*****************************************************************************/
00477                            // RCC_APB1ENR configurations //
00478 /*****************************************************************************/
00479
00480 /*options:
00481 *ENABLE
00482 *DISABLE
00483 */
00484 #define PWREN DISABLE
00485
00486 /*****************************************************************************/
00487
00488 /*options:
00489 *ENABLE
00490 *DISABLE
00491 */
00492 #define I2C3EN DISABLE
00493
00494 /*****************************************************************************/
00495
00496 /*options:
00497 *ENABLE
00498 *DISABLE
00499 */
00500 #define I2C2EN DISABLE
00501
00502 /*****************************************************************************/
00503
00504 /*options:
00505 *ENABLE
00506 *DISABLE
00507 */
00508 #define I2C1EN DISABLE
00509
00510 /*****************************************************************************/
00511
00512 /*options:
00513 *ENABLE
00514 *DISABLE
00515 */
00516 #define USART2EN DISABLE
00517
00518 /*****************************************************************************/
00519
00520 /*options:
00521 *ENABLE
00522 *DISABLE
00523 */
00524 #define SPI3EN DISABLE
```

```
00525
00526 /*****************************************************************************/
00527
00528 /*options:
00529 *ENABLE
00530 *DISABLE
00531 */
00532 #define SPI2EN DISABLE
00533
00534 /*****************************************************************************/
00535
00536 /*options:
00537 *ENABLE
00538 *DISABLE
00539 */
00540 #define WWDGEN DISABLE
00541
00542 /*****************************************************************************/
00543
00544 /*options:
00545 *ENABLE
00546 *DISABLE
00547 */
00548 #define TIM5EN DISABLE
00549
00550 /*****************************************************************************/
00551
00552 /*options:
00553 *ENABLE
00554 *DISABLE
00555 */
00556 #define TIM4EN DISABLE
00557
00558 /*****************************************************************************/
00559
00560 /*options:
00561 *ENABLE
00562 *DISABLE
00563 */
00564 #define TIM3EN DISABLE
00565
00566 /*****************************************************************************/
00567
00568 /*options:
00569 *ENABLE
00570 *DISABLE
00571 */
00572 #define TIM2EN DISABLE
00573
00574
00575 /*****************************************************************************/
00576                             // RCC_APB2ENR configurations //
00577 /*****************************************************************************/
00578
00579 /*options:
00580 *ENABLE
00581 *DISABLE
00582 */
00583 #define TIM11EN DISABLE
00584
00585 /*****************************************************************************/
00586
00587 /*options:
00588 *ENABLE
00589 *DISABLE
00590 */
00591 #define TIM10EN DISABLE
00592
00593 /*****************************************************************************/
00594
00595 /*options:
00596 *ENABLE
00597 *DISABLE
00598 */
00599 #define TIM9EN DISABLE
00600
00601 /*****************************************************************************/
00602
00603 /*options:
00604 *ENABLE
00605 *DISABLE
00606 */
00607 #define SYSCFGEN DISABLE
00608
00609 /*****************************************************************************/
00610
00611 /*options:
```

```
00612 *ENABLE
00613 *DISABLE
00614 */
00615 #define SPI4EN DISABLE
00616
00617 /*******************************************************************************************/
00618
00619 /*options:
00620 *ENABLE
00621 *DISABLE
00622 */
00623 #define SPI1EN DISABLE
00624
00625 /*******************************************************************************************/
00626
00627 /*options:
00628 *ENABLE
00629 *DISABLE
00630 */
00631 #define SDIOEN DISABLE
00632
00633 /*******************************************************************************************/
00634
00635 /*options:
00636 *ENABLE
00637 *DISABLE
00638 */
00639 #define ADC1EN DISABLE
00640
00641 /*******************************************************************************************/
00642
00643 /*options:
00644 *ENABLE
00645 *DISABLE
00646 */
00647 #define USART6EN DISABLE
00648
00649 /*******************************************************************************************/
00650
00651 /*options:
00652 *ENABLE
00653 *DISABLE
00654 */
00655 #define USART1EN DISABLE
00656
00657 /*******************************************************************************************/
00658
00659 /*options:
00660 *ENABLE
00661 *DISABLE
00662 */
00663 #define TIM1EN DISABLE
00664
00665 /*******************************************************************************************/
00666
00667
00668
00669
00670
00671
00672
00673
00674
00675
00676
00677
00678
00679
00680
00681
00682 #endif /* MCAL_RCC_MRCC_CONFIG_H_ */
```

7.22 src/COTS/MCAL/RCC/MRCC_interface.h File Reference

This file contains the interfacing information for the RCC module.

Macros

- #define RCC_AHB1 1

- #define RCC_AHB2 2
- #define RCC_APB1 3
- #define RCC_APB2 4
- #define RCC_AHB1LPENR 5
- #define AHB1ENR_DMA2EN 22
- #define AHB1ENR_DMA1EN 21
- #define AHB1ENR_CRCEN 12
- #define AHB1ENR_GPIOHEN 7
- #define AHB1ENR_GPIOEEN 4
- #define AHB1ENR_GPIODEN 3
- #define AHB1ENR_GPIOCEN 2
- #define AHB1ENR_GPIOBEN 1
- #define AHB1ENR_GPIOAEN 0
- #define AHB2ENR_OTGFSEN 7
- #define APB1ENR_PWREN 28
- #define APB1ENR_I2C3EN 23
- #define APB1ENR_I2C2EN 22
- #define APB1ENR_I2C1EN 21
- #define APB1ENR_USART2EN 17
- #define APB1ENR_SPI3EN 15
- #define APB1ENR_SPI2EN 14
- #define APB1ENR_WWDGEN 11
- #define APB1ENR_TIM5EN 3
- #define APB1ENR_TIM4EN 2
- #define APB1ENR_TIM3EN 1
- #define APB1ENR_TIM2EN 0
- #define APB2ENR_TIM11EN 18
- #define APB2ENR_TIM10EN 17
- #define APB2ENR_TIM9EN 16
- #define APB2ENR_SYSCFGEN 14
- #define APB2ENR_SPI4EN 13
- #define APB2ENR_SPI1EN 12
- #define APB2ENR_SDIOEN 11
- #define APB2ENR_ADC1EN 8
- #define APB2ENR_USART6EN 5
- #define APB2ENR_USART1EN 4
- #define APB2ENR_TIM1EN 0
- #define AHB1LPENR_FLITFLPEN 15

Functions

- void MRCC_vInit (void)
- void MRCC_vEnablePeriphralCLK (u32_t A_u32BusID, u32_t A_u32PeriphralID)
- void MRCC_vDisablePeriphralCLK (u32_t A_u32BusID, u32_t A_u32PeriphralID)

7.22.1 Detailed Description

This file contains the interfacing information for the RCC module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file MRCC_interface.h.

7.22.2 Macro Definition Documentation

7.22.2.1 RCC_AHB1

```
#define RCC_AHB1 1
```

Definition at line 31 of file MRCC_interface.h.

7.22.2.2 RCC_AHB2

```
#define RCC_AHB2 2
```

Definition at line 32 of file MRCC_interface.h.

7.22.2.3 RCC_APB1

```
#define RCC_APB1 3
```

Definition at line 33 of file MRCC_interface.h.

7.22.2.4 RCC_APB2

```
#define RCC_APB2 4
```

Definition at line 34 of file MRCC_interface.h.

7.22.2.5 RCC_AHB1LPENR

```
#define RCC_AHB1LPENR 5
```

Definition at line 35 of file MRCC_interface.h.

7.22.2.6 AHB1ENR_DMA2EN

```
#define AHB1ENR_DMA2EN 22
```

Definition at line 38 of file MRCC_interface.h.

7.22.2.7 AHB1ENR_DMA1EN

```
#define AHB1ENR_DMA1EN 21
```

Definition at line 39 of file MRCC_interface.h.

7.22.2.8 AHB1ENR_CRCEN

```
#define AHB1ENR_CRCEN 12
```

Definition at line 40 of file MRCC_interface.h.

7.22.2.9 AHB1ENR_GPIOHEN

```
#define AHB1ENR_GPIOHEN 7
```

Definition at line 41 of file MRCC_interface.h.

7.22.2.10 AHB1ENR_GPIOEEN

```
#define AHB1ENR_GPIOEEN 4
```

Definition at line 42 of file MRCC_interface.h.

7.22.2.11 AHB1ENR_GPIODEN

```
#define AHB1ENR_GPIODEN 3
```

Definition at line 43 of file MRCC_interface.h.

7.22.2.12 AHB1ENR_GPIOCEN

```
#define AHB1ENR_GPIOCEN 2
```

Definition at line 44 of file MRCC_interface.h.

7.22.2.13 AHB1ENR_GPIOBEN

```
#define AHB1ENR_GPIOBEN 1
```

Definition at line 45 of file MRCC_interface.h.

7.22.2.14 AHB1ENR_GPIOAEN

```
#define AHB1ENR_GPIOAEN 0
```

Definition at line 46 of file MRCC_interface.h.

7.22.2.15 AHB2ENR_OTGFSEN

```
#define AHB2ENR_OTGFSEN 7
```

Definition at line 48 of file MRCC_interface.h.

7.22.2.16 APB1ENR_PWREN

```
#define APB1ENR_PWREN 28
```

Definition at line 50 of file MRCC_interface.h.

7.22.2.17 APB1ENR_I2C3EN

```
#define APB1ENR_I2C3EN 23
```

Definition at line 51 of file MRCC_interface.h.

7.22.2.18 APB1ENR_I2C2EN

```
#define APB1ENR_I2C2EN 22
```

Definition at line 52 of file MRCC_interface.h.

7.22.2.19 APB1ENR_I2C1EN

```
#define APB1ENR_I2C1EN 21
```

Definition at line 53 of file MRCC_interface.h.

7.22.2.20 APB1ENR_USART2EN

```
#define APB1ENR_USART2EN 17
```

Definition at line 54 of file MRCC_interface.h.

7.22.2.21 APB1ENR_SPI3EN

```
#define APB1ENR_SPI3EN 15
```

Definition at line 55 of file MRCC_interface.h.

7.22.2.22 APB1ENR_SPI2EN

```
#define APB1ENR_SPI2EN 14
```

Definition at line 56 of file MRCC_interface.h.

7.22.2.23 APB1ENR_WWDGEN

```
#define APB1ENR_WWDGEN 11
```

Definition at line 57 of file MRCC_interface.h.

7.22.2.24 APB1ENR_TIM5EN

```
#define APB1ENR_TIM5EN 3
```

Definition at line 58 of file MRCC_interface.h.

7.22.2.25 APB1ENR_TIM4EN

```
#define APB1ENR_TIM4EN 2
```

Definition at line 59 of file MRCC_interface.h.

7.22.2.26 APB1ENR_TIM3EN

```
#define APB1ENR_TIM3EN 1
```

Definition at line 60 of file MRCC_interface.h.

7.22.2.27 APB1ENR_TIM2EN

```
#define APB1ENR_TIM2EN 0
```

Definition at line 61 of file MRCC_interface.h.

7.22.2.28 APB2ENR_TIM11EN

```
#define APB2ENR_TIM11EN 18
```

Definition at line 63 of file MRCC_interface.h.

7.22.2.29 APB2ENR_TIM10EN

```
#define APB2ENR_TIM10EN 17
```

Definition at line 64 of file MRCC_interface.h.

7.22.2.30 APB2ENR_TIM9EN

```
#define APB2ENR_TIM9EN 16
```

Definition at line 65 of file MRCC_interface.h.

7.22.2.31 APB2ENR_SYSCFGEN

```
#define APB2ENR_SYSCFGEN 14
```

Definition at line 66 of file MRCC_interface.h.

7.22.2.32 APB2ENR_SPI4EN

```
#define APB2ENR_SPI4EN 13
```

Definition at line 67 of file MRCC_interface.h.

7.22.2.33 APB2ENR_SPI1EN

```
#define APB2ENR_SPI1EN 12
```

Definition at line 68 of file MRCC_interface.h.

7.22.2.34 APB2ENR_SDIOEN

```
#define APB2ENR_SDIOEN 11
```

Definition at line 69 of file MRCC_interface.h.

7.22.2.35 APB2ENR_ADC1EN

```
#define APB2ENR_ADC1EN 8
```

Definition at line 70 of file MRCC_interface.h.

7.22.2.36 APB2ENR_USART6EN

```
#define APB2ENR_USART6EN 5
```

Definition at line 71 of file MRCC_interface.h.

7.22.2.37 APB2ENR_USART1EN

```
#define APB2ENR_USART1EN 4
```

Definition at line 72 of file MRCC_interface.h.

7.22.2.38 APB2ENR_TIM1EN

```
#define APB2ENR_TIM1EN 0
```

Definition at line 73 of file MRCC_interface.h.

7.22.2.39 AHB1LPENR_FLITFLPEN

```
#define AHB1LPENR_FLITFLPEN 15
```

Definition at line 77 of file MRCC_interface.h.

7.22.3 Function Documentation

7.22.3.1 MRCC_vInit()

```
void MRCC_vInit (
             void  )
```

Definition at line 29 of file MRCC_program.c.

```
00030 {
00031
00032     // PLLI2S (ON/OFF).
00033 #if PLLI2S == ENABLE
00034
00035         SET_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00036
00037 #elif PLLI2S == DISABLE
00038
00039         CLR_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00040
00041 #endif
00042
00043
00044     // PLL (ON/OFF).
00045 #if PLL == ENABLE
00046
00047         SET_BIT( RCC->CR, RCC_CR_PLLON ) ;
00048
00049 #elif PLL == DISABLE
00050
00051         CLR_BIT( RCC->CR, RCC_CR_PLLON ) ;
00052
00053 #endif
00054
00055
00056     // CSS (ON/OFF).
00057 #if CSS == ENABLE
00058
00059         SET_BIT( RCC->CR, RCC_CR_CSSON ) ;
00060
00061 #elif CSS == DISABLE
00062
00063         CLR_BIT( RCC->CR, RCC_CR_CSSON ) ;
00064
00065 #endif
00066
00067
00068     // HSEBYP.
00069 #if HSEBYP == BYBASED
00070
00071         SET_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00072
00073 #elif HSEBYP == NOTBYBASED
00074
00075         CLR_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00076
00077 #endif
00078
00079
00080     // Select CLK switch (HSI/HSE/PLL).
00081 #if SW == HSI
00082
00083         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00084         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00085
00086 #elif SW == HSE
00087
00088         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00089         SET_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00090
00091 #elif SW == PLLCLK
00092
00093         SET_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00094         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00095
00096 #endif
00097
00098
```

```
00099     // MCO2 selection:
00100 #if MCO2 == SYSCLK
00101
00102         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00103         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00104
00105 #elif MCO2 == PLLI2SCLK
00106
00107         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00108         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00109
00110 #elif MCO2 == HSE
00111
00112         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00113         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00114
00115 #elif MCO2 == PLLCLK
00116
00117         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00118         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00119
00120 #endif
00121
00122
00123     // MCO2 prescaler:
00124 #if MCO2PRE == NoDivision
00125
00126         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00127         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00128         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00129
00130 #elif MCO2PRE == DivisionBy2
00131
00132         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00133         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00134         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00135
00136 #elif MCO2PRE == DivisionBy3
00137
00138         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00139         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00140         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00141
00142 #elif MCO2PRE == DivisionBy4
00143
00144         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00145         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00146         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00147
00148 #elif MCO2PRE == DivisionBy5
00149
00150         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00151         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00152         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00153
00154 #endif
00155
00156
00157     // MCO1 selection:
00158 #if MCO1 == HSI
00159
00160         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00161         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00162
00163 #elif MCO1 == LSE
00164
00165         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00166         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00167
00168 #elif MCO1 == HSE
00169
00170         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00171         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00172
00173 #elif MCO1 == PLLCLK
00174
00175         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00176         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00177
00178 #endif
00179
00180
00181     // MCO1 prescaler:
00182 #if MCO1PRE == NoDivision
00183
00184         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00185         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
```

```
00186         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00187
00188 #elif MCO1PRE == DivisionBy2
00189
00190         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00191         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00192         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00193
00194 #elif MCO1PRE == DivisionBy3
00195
00196         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00197         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00198         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00199
00200 #elif MCO1PRE == DivisionBy4
00201
00202         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00203         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00204         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00205
00206 #elif MCO1PRE == DivisionBy5
00207
00208         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00209         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00210         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00211
00212 #endif
00213
00214
00215     // AHB prescalers:
00216 #if HPRE == NoDivision
00217
00218         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00219         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00220         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00221         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00222
00223 #elif HPRE == SYSCLKby2
00224
00225         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00226         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00227         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00228         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00229
00230 #elif HPRE == SYSCLKby4
00231
00232         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00233         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00234         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00235         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00236
00237 #elif HPRE == SYSCLKby8
00238
00239         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00240         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00241         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00242         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00243
00244 #elif HPRE == SYSCLKby16
00245
00246         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00247         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00248         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00249         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00250
00251 #elif HPRE == SYSCLKby64
00252
00253         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00254         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00255         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00256         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00257
00258 #elif HPRE == SYSCLKby128
00259
00260         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00261         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00262         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00263         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00264
00265 #elif HPRE == SYSCLKby256
00266
00267         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00268         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00269         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00270         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00271
00272 #elif HPRE == SYSCLKby512
```

```
00273
00274         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00275         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00276         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00277         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00278
00279 #endif
00280
00281
00282     // APB1 prescalers:
00283 #if PPRE1 == NoDivision
00284
00285         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00286         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00287         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00288
00289 #elif PPRE1 == AHBby2
00290
00291         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00292         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00293         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00294
00295 #elif PPRE1 == AHBby4
00296
00297         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00298         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00299         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00300
00301 #elif PPRE1 == AHBby8
00302
00303         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00304         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00305         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00306
00307 #elif PPRE1 == AHBby16
00308
00309         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00310         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00311         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00312
00313 #endif
00314
00315     // APB2 prescalers:
00316 #if PPRE2 == NoDivision
00317
00318         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00319         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00320         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00321
00322 #elif PPRE2 == AHBby2
00323
00324         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00325         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00326         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00327
00328 #elif PPRE2 == AHBby4
00329
00330         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00331         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00332         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00333
00334 #elif PPRE2 == AHBby8
00335
00336         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00337         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00338         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00339
00340 #elif PPRE2 == AHBby16
00341
00342         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00343         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00344         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00345
00346 #endif
00347
00348
00349     // PLL configurations:
00350
00351
00352
00353     // Enable the selected CLK (HSI ON/HSE ON/PLL ON): //
00354     // HSE (ON/OFF).
00355 #if HSE_EN == ENABLE
00356
00357         SET_BIT( RCC->CR, RCC_CR_HSEON ) ;
00358
00359 #elif HSE_EN == DISABLE
```

```
00360
00361         CLR_BIT( RCC->CR, RCC_CR_HSEON ) ;
00362
00363 #endif
00364
00365
00366     // HSI (ON/OFF).
00367 #if HSI_EN == ENABLE
00368
00369         SET_BIT( RCC->CR, RCC_CR_HSION ) ;
00370
00371 #elif HSI_EN == DISABLE
00372
00373         CLR_BIT( RCC->CR, RCC_CR_HSION ) ;
00374
00375 #endif
00376
00377 }
```

References CLR_BIT, RCC, RCC_CFGR_HPRE_b0, RCC_CFGR_HPRE_b1, RCC_CFGR_HPRE_b2, RCC_CFGR_HPRE_b3, RCC_CFGR_MOC1_b0, RCC_CFGR_MOC1_b1, RCC_CFGR_MOC1PRE_b0, RCC_CFGR_MOC1PRE_b1, RCC_CFGR_MOC1PRE_b2, RCC_CFGR_MOC2_b0, RCC_CFGR_MOC2_b1, RCC_CFGR_MOC2PRE_b0, RCC_CFGR_MOC2PRE_b1, RCC_CFGR_MOC2PRE_b2, RCC_CFGR_PPRE1_b0, RCC_CFGR_PPRE1_b1, RCC_CFGR_PPRE1_b2, RCC_CFGR_PPRE2_b0, RCC_CFGR_PPRE2_b1, RCC_CFGR_PPRE2_b2, RCC_CFGR_SW_b0, RCC_CFGR_SW_b1, RCC_CR_CSSON, RCC_CR_HSEBYP, RCC_CR_HSEON, RCC_CR_HSION, RCC_CR_PLLI2SON, RCC_CR_PLLON, and SET_BIT.

7.22.3.2 MRCC_vEnablePeriphralCLK()

```
void MRCC_vEnablePeriphralCLK (
             u32_t A_u32BusID,
             u32_t A_u32PeriphralID )
```

Definition at line 382 of file MRCC_program.c.

```
00383 {
00384
00385     switch( A_u32BusID )
00386     {
00387
00388         case RCC_AHB1 :
00389
00390             SET_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00391
00392         break ;
00393
00394
00395         case RCC_AHB2 :
00396
00397             SET_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00398
00399
00400         break ;
00401
00402
00403         case RCC_APB1 :
00404
00405             SET_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00406
00407         break ;
00408
00409
00410         case RCC_APB2 :
00411
00412             SET_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00413
00414         break ;
00415
00416         case RCC_AHB1LPENR:
00417
00418             SET_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00419
00420         break ;
00421
00422         default:
```

```
00423
00424             // Error wrong Bus ID
00425
00426         break ;
00427
00428     }
00429
00430 }
```

References RCC, RCC_AHB1, RCC_AHB1LPENR, RCC_AHB2, RCC_APB1, RCC_APB2, and SET_BIT.

7.22.3.3 MRCC_vDisablePeriphralCLK()

```
void MRCC_vDisablePeriphralCLK (
             u32_t A_u32BusID,
             u32_t A_u32PeriphralID )
```

Definition at line 435 of file MRCC_program.c.

```
00436 {
00437
00438     switch( A_u32BusID )
00439     {
00440
00441         case RCC_AHB1 :
00442
00443             CLR_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00444
00445         break ;
00446
00447
00448         case RCC_AHB2 :
00449
00450             CLR_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00451
00452
00453         break ;
00454
00455
00456         case RCC_APB1 :
00457
00458             CLR_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00459
00460         break ;
00461
00462
00463         case RCC_APB2 :
00464
00465             CLR_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00466
00467         break ;
00468
00469
00470         case RCC_AHB1LPENR:
00471
00472             CLR_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00473
00474         break ;
00475
00476
00477         default:
00478
00479             // Error wrong Bus ID
00480
00481         break ;
00482
00483     }
00484
00485 }
```

References CLR_BIT, RCC, RCC_AHB1, RCC_AHB1LPENR, RCC_AHB2, RCC_APB1, and RCC_APB2.

7.23 MRCC_interface.h

Go to the documentation of this file.

```
00001
00010 #ifndef MCAL_RCC_MRCC_INTERFACE_H_
00011 #define MCAL_RCC_MRCC_INTERFACE_H_
00012
00013
00014
00015 /*****************************************************************************/
00016 /*                        Functions prototypes                               */
00017 /*****************************************************************************/
00018
00019 void MRCC_vInit( void ) ;
00020
00021 void MRCC_vEnablePeriphralCLK( u32_t A_u32BusID, u32_t A_u32PeriphralID ) ;
00022
00023 void MRCC_vDisablePeriphralCLK( u32_t A_u32BusID, u32_t A_u32PeriphralID ) ;
00024
00025
00026 /*****************************************************************************/
00027 /*                           Interfacing macros                              */
00028 /*****************************************************************************/
00029
00030 // A_u32BusID options:
00031 #define RCC_AHB1        1
00032 #define RCC_AHB2        2
00033 #define RCC_APB1        3
00034 #define RCC_APB2        4
00035 #define RCC_AHB1LPENR   5
00036
00037 // A_u32PeriphralID options:
00038 #define AHB1ENR_DMA2EN    22
00039 #define AHB1ENR_DMA1EN    21
00040 #define AHB1ENR_CRCEN     12
00041 #define AHB1ENR_GPIOHEN   7
00042 #define AHB1ENR_GPIOEEN   4
00043 #define AHB1ENR_GPIODEN   3
00044 #define AHB1ENR_GPIOCEN   2
00045 #define AHB1ENR_GPIOBEN   1
00046 #define AHB1ENR_GPIOAEN   0
00047
00048 #define AHB2ENR_OTGFSEN   7
00049
00050 #define APB1ENR_PWREN       28
00051 #define APB1ENR_I2C3EN      23
00052 #define APB1ENR_I2C2EN      22
00053 #define APB1ENR_I2C1EN      21
00054 #define APB1ENR_USART2EN    17
00055 #define APB1ENR_SPI3EN      15
00056 #define APB1ENR_SPI2EN      14
00057 #define APB1ENR_WWDGEN      11
00058 #define APB1ENR_TIM5EN      3
00059 #define APB1ENR_TIM4EN      2
00060 #define APB1ENR_TIM3EN      1
00061 #define APB1ENR_TIM2EN      0
00062
00063 #define APB2ENR_TIM11EN     18
00064 #define APB2ENR_TIM10EN     17
00065 #define APB2ENR_TIM9EN      16
00066 #define APB2ENR_SYSCFGEN    14
00067 #define APB2ENR_SPI4EN      13
00068 #define APB2ENR_SPI1EN      12
00069 #define APB2ENR_SDIOEN      11
00070 #define APB2ENR_ADC1EN      8
00071 #define APB2ENR_USART6EN    5
00072 #define APB2ENR_USART1EN    4
00073 #define APB2ENR_TIM1EN      0
00074
00075
00076
00077 #define AHB1LPENR_FLITFLPEN 15
00078
00079
00080
00081
00082 #endif /* MCAL_RCC_MRCC_INTERFACE_H_ */
```

7.24 src/COTS/MCAL/RCC/MRCC_private.h File Reference

This file contains the registers information and addresses for the RCC module.

Data Structures

- struct RCC_MemoryMapType

Macros

- #define RCC_BASE_ADDRESS 0x40023800
- #define RCC ((volatile P2VAR(RCC_MemoryMapType)) (RCC_BASE_ADDRESS))
- #define RCC_CR_PLLI2SRDY 27
- #define RCC_CR_PLLI2SON 26
- #define RCC_CR_PLLRDY 25
- #define RCC_CR_PLLON 24
- #define RCC_CR_CSSON 19
- #define RCC_CR_HSEBYP 18
- #define RCC_CR_HSERDY 17
- #define RCC_CR_HSEON 16
- #define RCC_CR_HSIRDY 1
- #define RCC_CR_HSION 0
- #define RCC_PLLCFGR_PLLQ_b0 24
- #define RCC_PLLCFGR_PLLQ_b1 25
- #define RCC_PLLCFGR_PLLQ_b2 26
- #define RCC_PLLCFGR_PLLQ_b3 27
- #define RCC_PLLCFGR_PLLSRC 22
- #define RCC_PLLCFGR_PLLP_b0 16
- #define RCC_PLLCFGR_PLLP_b1 17
- #define RCC_PLLCFGR_PLLN_b0 6
- #define RCC_PLLCFGR_PLLN_b1 7
- #define RCC_PLLCFGR_PLLN_b2 8
- #define RCC_PLLCFGR_PLLN_b3 9
- #define RCC_PLLCFGR_PLLN_b4 10
- #define RCC_PLLCFGR_PLLN_b5 11
- #define RCC_PLLCFGR_PLLN_b6 12
- #define RCC_PLLCFGR_PLLN_b7 13
- #define RCC_PLLCFGR_PLLN_b8 14
- #define RCC_PLLCFGR_PLLM_b0 0
- #define RCC_PLLCFGR_PLLM_b1 1
- #define RCC_PLLCFGR_PLLM_b2 2
- #define RCC_PLLCFGR_PLLM_b3 3
- #define RCC_PLLCFGR_PLLM_b4 4
- #define RCC_PLLCFGR_PLLM_b5 5
- #define RCC_CFGR_MOC2_b0 30
- #define RCC_CFGR_MOC2_b1 31
- #define RCC_CFGR_MOC2PRE_b0 27
- #define RCC_CFGR_MOC2PRE_b1 28
- #define RCC_CFGR_MOC2PRE_b2 29
- #define RCC_CFGR_MOC1PRE_b0 24
- #define RCC_CFGR_MOC1PRE_b1 25
- #define RCC_CFGR_MOC1PRE_b2 26
- #define RCC_CFGR_I2SSRC 23
- #define RCC_CFGR_MOC1_b0 21
- #define RCC_CFGR_MOC1_b1 22
- #define RCC_CFGR_RTCPRE_b0 16
- #define RCC_CFGR_RTCPRE_b1 17
- #define RCC_CFGR_RTCPRE_b2 18

- #define RCC_CFGR_RTCPRE_b3 19
- #define RCC_CFGR_RTCPRE_b4 20
- #define RCC_CFGR_PPRE2_b0 13
- #define RCC_CFGR_PPRE2_b1 14
- #define RCC_CFGR_PPRE2_b2 15
- #define RCC_CFGR_PPRE1_b0 10
- #define RCC_CFGR_PPRE1_b1 11
- #define RCC_CFGR_PPRE1_b2 12
- #define RCC_CFGR_HPRE_b0 4
- #define RCC_CFGR_HPRE_b1 5
- #define RCC_CFGR_HPRE_b2 6
- #define RCC_CFGR_HPRE_b3 7
- #define RCC_CFGR_SWS_b0 2
- #define RCC_CFGR_SWS_b1 3
- #define RCC_CFGR_SW_b0 0
- #define RCC_CFGR_SW_b1 1
- #define RCC_AHB1ENR_DMA2EN 22
- #define RCC_AHB1ENR_DMA1EN 21
- #define RCC_AHB1ENR_CRCEN 12
- #define RCC_AHB1ENR_GPIOHEN 7
- #define RCC_AHB1ENR_GPIOEEN 4
- #define RCC_AHB1ENR_GPIODEN 3
- #define RCC_AHB1ENR_GPIOCEN 2
- #define RCC_AHB1ENR_GPIOBEN 1
- #define RCC_AHB1ENR_GPIOAEN 0
- #define RCC_AHB2ENR_OTGFSEN 7
- #define RCC_APB1ENR_PWREN 28
- #define RCC_APB1ENR_I2C3EN 23
- #define RCC_APB1ENR_I2C2EN 22
- #define RCC_APB1ENR_I2C1EN 21
- #define RCC_APB1ENR_USART2EN 17
- #define RCC_APB1ENR_SPI3EN 15
- #define RCC_APB1ENR_SPI2EN 14
- #define RCC_APB1ENR_WWDGEN 11
- #define RCC_APB1ENR_TIM5EN 3
- #define RCC_APB1ENR_TIM4EN 2
- #define RCC_APB1ENR_TIM3EN 1
- #define RCC_APB1ENR_TIM2EN 0
- #define RCC_APB2ENR_TIM11EN 18
- #define RCC_APB2ENR_TIM10EN 17
- #define RCC_APB2ENR_TIM9EN 16
- #define RCC_APB2ENR_SYSCFGEN 14
- #define RCC_APB2ENR_SPI4EN 13
- #define RCC_APB2ENR_SPI1EN 12
- #define RCC_APB2ENR_SDIOEN 11
- #define RCC_APB2ENR_ADC1EN 8
- #define RCC_APB2ENR_USART6EN 5
- #define RCC_APB2ENR_USART1EN 4
- #define RCC_APB2ENR_TIM1EN 0
- #define RCC_AHB1LPENR_FLITFLPEN 15
- #define ENABLE 1
- #define DISABLE 2
- #define BYBASED 1
- #define NOTBYBASED 2
- #define SYSCLK 1

- #define PLLI2SCLK 5
- #define HSE 3
- #define PLLCLK 4
- #define NoDivision 1
- #define DivisionBy2 2
- #define DivisionBy3 3
- #define DivisionBy4 4
- #define DivisionBy5 5
- #define I2S_CKIN 2
- #define HSI 1
- #define LSE 2
- #define PLLCLK 4
- #define SYSCLKby2 2
- #define SYSCLKby4 3
- #define SYSCLKby8 4
- #define SYSCLKby16 5
- #define SYSCLKby64 6
- #define SYSCLKby128 7
- #define SYSCLKby256 8
- #define SYSCLKby512 9
- #define NoCLK0 1
- #define NoCLK1 2
- #define HSEby2 3
- #define HSEby3 4
- #define HSEby4 5
- #define HSEby5 6
- #define HSEby6 7
- #define HSEby7 8
- #define HSEby8 9
- #define HSEby9 10
- #define HSEby10 11
- #define HSEby11 12
- #define HSEby12 13
- #define HSEby13 14
- #define HSEby14 15
- #define HSEby15 16
- #define HSEby16 17
- #define HSEby17 18
- #define HSEby18 19
- #define HSEby19 20
- #define HSEby20 21
- #define HSEby21 22
- #define HSEby22 23
- #define HSEby23 24
- #define HSEby24 25
- #define HSEby25 26
- #define HSEby26 27
- #define HSEby27 28
- #define HSEby28 39
- #define HSEby29 30
- #define HSEby30 31
- #define HSEby31 32
- #define AHBby2 1
- #define AHBby4 2
- #define AHBby8 3

- #define AHBby16 4
- #define Equal_0 0
- #define Equal_1 1
- #define Equal_2 2
- #define Equal_3 3
- #define Equal_4 4
- #define Equal_5 5
- #define Equal_6 6
- #define Equal_7 7
- #define Equal_8 8
- #define Equal_9 9
- #define Equal_10 10
- #define Equal_11 11
- #define Equal_12 12
- #define Equal_13 13
- #define Equal_14 14
- #define Equal_15 15
- #define Equal_16 16
- #define Equal_17 17
- #define Equal_18 18
- #define Equal_19 19
- #define Equal_20 20
- #define Equal_21 21
- #define Equal_22 22
- #define Equal_23 23
- #define Equal_24 24
- #define Equal_25 25
- #define Equal_26 26
- #define Equal_27 27
- #define Equal_28 28
- #define Equal_29 29
- #define Equal_30 30
- #define Equal_31 31
- #define Equal_32 32
- #define Equal_33 33
- #define Equal_34 34
- #define Equal_35 35
- #define Equal_36 36
- #define Equal_37 37
- #define Equal_38 38
- #define Equal_39 39
- #define Equal_40 40
- #define Equal_41 41
- #define Equal_42 42
- #define Equal_43 43
- #define Equal_44 44
- #define Equal_45 45
- #define Equal_46 46
- #define Equal_47 47
- #define Equal_48 48
- #define Equal_49 49
- #define Equal_50 50
- #define Equal_51 51
- #define Equal_52 52
- #define Equal_53 53

- #define Equal_54 54
- #define Equal_55 55
- #define Equal_56 56
- #define Equal_57 57
- #define Equal_58 58
- #define Equal_59 59
- #define Equal_60 60
- #define Equal_61 61
- #define Equal_62 62
- #define Equal_63 63

7.24.1 Detailed Description

This file contains the registers information and addresses for the RCC module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file MRCC_private.h.

7.24.2 Macro Definition Documentation

7.24.2.1 RCC_BASE_ADDRESS

```
#define RCC_BASE_ADDRESS 0x40023800
```

Definition at line 76 of file MRCC_private.h.

7.24.2.2 RCC

```
#define RCC ( (volatile P2VAR(RCC_MemoryMapType) ) (RCC_BASE_ADDRESS) )
```

Definition at line 78 of file MRCC_private.h.

7.24.2.3 RCC_CR_PLLI2SRDY

```
#define RCC_CR_PLLI2SRDY 27
```

Definition at line 85 of file MRCC_private.h.

7.24.2.4 RCC_CR_PLLI2SON

```
#define RCC_CR_PLLI2SON 26
```

Definition at line 86 of file MRCC_private.h.

7.24.2.5 RCC_CR_PLLRDY

```
#define RCC_CR_PLLRDY 25
```

Definition at line 87 of file MRCC_private.h.

7.24.2.6 RCC_CR_PLLON

```
#define RCC_CR_PLLON 24
```

Definition at line 88 of file MRCC_private.h.

7.24.2.7 RCC_CR_CSSON

```
#define RCC_CR_CSSON 19
```

Definition at line 89 of file MRCC_private.h.

7.24.2.8 RCC_CR_HSEBYP

```
#define RCC_CR_HSEBYP 18
```

Definition at line 90 of file MRCC_private.h.

7.24.2.9 RCC_CR_HSERDY

```
#define RCC_CR_HSERDY 17
```

Definition at line 91 of file MRCC_private.h.

7.24.2.10 RCC_CR_HSEON

```
#define RCC_CR_HSEON 16
```

Definition at line 92 of file MRCC_private.h.

7.24.2.11 RCC_CR_HSIRDY

```
#define RCC_CR_HSIRDY 1
```

Definition at line 93 of file MRCC_private.h.

7.24.2.12 RCC_CR_HSION

```
#define RCC_CR_HSION 0
```

Definition at line 94 of file MRCC_private.h.

7.24.2.13 RCC_PLLCFGR_PLLQ_b0

```
#define RCC_PLLCFGR_PLLQ_b0 24
```

Definition at line 97 of file MRCC_private.h.

7.24.2.14 RCC_PLLCFGR_PLLQ_b1

```
#define RCC_PLLCFGR_PLLQ_b1 25
```

Definition at line 98 of file MRCC_private.h.

7.24.2.15 RCC_PLLCFGR_PLLQ_b2

```
#define RCC_PLLCFGR_PLLQ_b2 26
```

Definition at line 99 of file MRCC_private.h.

7.24.2.16 RCC_PLLCFGR_PLLQ_b3

```
#define RCC_PLLCFGR_PLLQ_b3 27
```

Definition at line 100 of file MRCC_private.h.

7.24.2.17 RCC_PLLCFGR_PLLSRC

```
#define RCC_PLLCFGR_PLLSRC 22
```

Definition at line 101 of file MRCC_private.h.

7.24.2.18 RCC_PLLCFGR_PLLP_b0

```
#define RCC_PLLCFGR_PLLP_b0 16
```

Definition at line 102 of file MRCC_private.h.

7.24.2.19 RCC_PLLCFGR_PLLP_b1

```
#define RCC_PLLCFGR_PLLP_b1 17
```

Definition at line 103 of file MRCC_private.h.

7.24.2.20 RCC_PLLCFGR_PLLN_b0

```
#define RCC_PLLCFGR_PLLN_b0 6
```

Definition at line 104 of file MRCC_private.h.

7.24.2.21 RCC_PLLCFGR_PLLN_b1

```
#define RCC_PLLCFGR_PLLN_b1 7
```

Definition at line 105 of file MRCC_private.h.

7.24.2.22 RCC_PLLCFGR_PLLN_b2

```
#define RCC_PLLCFGR_PLLN_b2 8
```

Definition at line 106 of file MRCC_private.h.

7.24.2.23 RCC_PLLCFGR_PLLN_b3

```
#define RCC_PLLCFGR_PLLN_b3 9
```

Definition at line 107 of file MRCC_private.h.

7.24.2.24 RCC_PLLCFGR_PLLN_b4

```
#define RCC_PLLCFGR_PLLN_b4 10
```

Definition at line 108 of file MRCC_private.h.

7.24.2.25 RCC_PLLCFGR_PLLN_b5

```
#define RCC_PLLCFGR_PLLN_b5 11
```

Definition at line 109 of file MRCC_private.h.

7.24.2.26 RCC_PLLCFGR_PLLN_b6

```
#define RCC_PLLCFGR_PLLN_b6 12
```

Definition at line 110 of file MRCC_private.h.

7.24.2.27 RCC_PLLCFGR_PLLN_b7

```
#define RCC_PLLCFGR_PLLN_b7 13
```

Definition at line 111 of file MRCC_private.h.

7.24.2.28 RCC_PLLCFGR_PLLN_b8

```
#define RCC_PLLCFGR_PLLN_b8 14
```

Definition at line 112 of file MRCC_private.h.

7.24.2.29 RCC_PLLCFGR_PLLM_b0

```
#define RCC_PLLCFGR_PLLM_b0 0
```

Definition at line 113 of file MRCC_private.h.

7.24.2.30 RCC_PLLCFGR_PLLM_b1

```
#define RCC_PLLCFGR_PLLM_b1 1
```

Definition at line 114 of file MRCC_private.h.

7.24.2.31 RCC_PLLCFGR_PLLM_b2

```
#define RCC_PLLCFGR_PLLM_b2 2
```

Definition at line 115 of file MRCC_private.h.

7.24.2.32 RCC_PLLCFGR_PLLM_b3

```
#define RCC_PLLCFGR_PLLM_b3 3
```

Definition at line 116 of file MRCC_private.h.

7.24.2.33 RCC_PLLCFGR_PLLM_b4

```
#define RCC_PLLCFGR_PLLM_b4 4
```

Definition at line 117 of file MRCC_private.h.

7.24.2.34 RCC_PLLCFGR_PLLM_b5

```
#define RCC_PLLCFGR_PLLM_b5 5
```

Definition at line 118 of file MRCC_private.h.

7.24.2.35 RCC_CFGR_MOC2_b0

```
#define RCC_CFGR_MOC2_b0 30
```

Definition at line 121 of file MRCC_private.h.

7.24.2.36 RCC_CFGR_MOC2_b1

```
#define RCC_CFGR_MOC2_b1 31
```

Definition at line 122 of file MRCC_private.h.

7.24.2.37 RCC_CFGR_MOC2PRE_b0

```
#define RCC_CFGR_MOC2PRE_b0 27
```

Definition at line 123 of file MRCC_private.h.

7.24.2.38 RCC_CFGR_MOC2PRE_b1

```
#define RCC_CFGR_MOC2PRE_b1 28
```

Definition at line 124 of file MRCC_private.h.

7.24.2.39 RCC_CFGR_MOC2PRE_b2

```
#define RCC_CFGR_MOC2PRE_b2 29
```

Definition at line 125 of file MRCC_private.h.

7.24.2.40 RCC_CFGR_MOC1PRE_b0

```
#define RCC_CFGR_MOC1PRE_b0 24
```

Definition at line 126 of file MRCC_private.h.

7.24.2.41 RCC_CFGR_MOC1PRE_b1

```
#define RCC_CFGR_MOC1PRE_b1 25
```

Definition at line 127 of file MRCC_private.h.

7.24.2.42 RCC_CFGR_MOC1PRE_b2

```
#define RCC_CFGR_MOC1PRE_b2 26
```

Definition at line 128 of file MRCC_private.h.

7.24.2.43 RCC_CFGR_I2SSRC

```
#define RCC_CFGR_I2SSRC 23
```

Definition at line 129 of file MRCC_private.h.

7.24.2.44 RCC_CFGR_MOC1_b0

```
#define RCC_CFGR_MOC1_b0 21
```

Definition at line 130 of file MRCC_private.h.

7.24.2.45 RCC_CFGR_MOC1_b1

```
#define RCC_CFGR_MOC1_b1 22
```

Definition at line 131 of file MRCC_private.h.

7.24.2.46 RCC_CFGR_RTCPRE_b0

```
#define RCC_CFGR_RTCPRE_b0 16
```

Definition at line 132 of file MRCC_private.h.

7.24.2.47 RCC_CFGR_RTCPRE_b1

```
#define RCC_CFGR_RTCPRE_b1 17
```

Definition at line 133 of file MRCC_private.h.

7.24.2.48 RCC_CFGR_RTCPRE_b2

```
#define RCC_CFGR_RTCPRE_b2 18
```

Definition at line 134 of file MRCC_private.h.

7.24.2.49 RCC_CFGR_RTCPRE_b3

```
#define RCC_CFGR_RTCPRE_b3 19
```

Definition at line 135 of file MRCC_private.h.

7.24.2.50 RCC_CFGR_RTCPRE_b4

```
#define RCC_CFGR_RTCPRE_b4 20
```

Definition at line 136 of file MRCC_private.h.

7.24.2.51 RCC_CFGR_PPRE2_b0

```
#define RCC_CFGR_PPRE2_b0 13
```

Definition at line 137 of file MRCC_private.h.

7.24.2.52 RCC_CFGR_PPRE2_b1

```
#define RCC_CFGR_PPRE2_b1 14
```

Definition at line 138 of file MRCC_private.h.

7.24.2.53 RCC_CFGR_PPRE2_b2

```
#define RCC_CFGR_PPRE2_b2 15
```

Definition at line 139 of file MRCC_private.h.

7.24.2.54 RCC_CFGR_PPRE1_b0

```
#define RCC_CFGR_PPRE1_b0 10
```

Definition at line 140 of file MRCC_private.h.

7.24.2.55 RCC_CFGR_PPRE1_b1

```
#define RCC_CFGR_PPRE1_b1 11
```

Definition at line 141 of file MRCC_private.h.

7.24.2.56 RCC_CFGR_PPRE1_b2

```
#define RCC_CFGR_PPRE1_b2 12
```

Definition at line 142 of file MRCC_private.h.

7.24.2.57 RCC_CFGR_HPRE_b0

```
#define RCC_CFGR_HPRE_b0 4
```

Definition at line 143 of file MRCC_private.h.

7.24.2.58 RCC_CFGR_HPRE_b1

```
#define RCC_CFGR_HPRE_b1 5
```

Definition at line 144 of file MRCC_private.h.

7.24.2.59 RCC_CFGR_HPRE_b2

```
#define RCC_CFGR_HPRE_b2 6
```

Definition at line 145 of file MRCC_private.h.

7.24.2.60 RCC_CFGR_HPRE_b3

```
#define RCC_CFGR_HPRE_b3 7
```

Definition at line 146 of file MRCC_private.h.

7.24.2.61 RCC_CFGR_SWS_b0

```
#define RCC_CFGR_SWS_b0 2
```

Definition at line 147 of file MRCC_private.h.

7.24.2.62 RCC_CFGR_SWS_b1

```
#define RCC_CFGR_SWS_b1 3
```

Definition at line 148 of file MRCC_private.h.

7.24.2.63 RCC_CFGR_SW_b0

```
#define RCC_CFGR_SW_b0 0
```

Definition at line 149 of file MRCC_private.h.

7.24.2.64 RCC_CFGR_SW_b1

```
#define RCC_CFGR_SW_b1 1
```

Definition at line 150 of file MRCC_private.h.

7.24.2.65 RCC_AHB1ENR_DMA2EN

```
#define RCC_AHB1ENR_DMA2EN 22
```

Definition at line 153 of file MRCC_private.h.

7.24.2.66 RCC_AHB1ENR_DMA1EN

```
#define RCC_AHB1ENR_DMA1EN 21
```

Definition at line 154 of file MRCC_private.h.

7.24.2.67 RCC_AHB1ENR_CRCEN

```
#define RCC_AHB1ENR_CRCEN 12
```

Definition at line 155 of file MRCC_private.h.

7.24.2.68 RCC_AHB1ENR_GPIOHEN

```
#define RCC_AHB1ENR_GPIOHEN 7
```

Definition at line 156 of file MRCC_private.h.

7.24.2.69 RCC_AHB1ENR_GPIOEEN

```
#define RCC_AHB1ENR_GPIOEEN 4
```

Definition at line 157 of file MRCC_private.h.

7.24.2.70 RCC_AHB1ENR_GPIODEN

```
#define RCC_AHB1ENR_GPIODEN 3
```

Definition at line 158 of file MRCC_private.h.

7.24.2.71 RCC_AHB1ENR_GPIOCEN

```
#define RCC_AHB1ENR_GPIOCEN 2
```

Definition at line 159 of file MRCC_private.h.

7.24.2.72 RCC_AHB1ENR_GPIOBEN

```
#define RCC_AHB1ENR_GPIOBEN 1
```

Definition at line 160 of file MRCC_private.h.

7.24.2.73 RCC_AHB1ENR_GPIOAEN

```
#define RCC_AHB1ENR_GPIOAEN 0
```

Definition at line 161 of file MRCC_private.h.

7.24.2.74 RCC_AHB2ENR_OTGFSEN

```
#define RCC_AHB2ENR_OTGFSEN 7
```

Definition at line 164 of file MRCC_private.h.

7.24.2.75 RCC_APB1ENR_PWREN

```
#define RCC_APB1ENR_PWREN 28
```

Definition at line 167 of file MRCC_private.h.

7.24.2.76 RCC_APB1ENR_I2C3EN

```
#define RCC_APB1ENR_I2C3EN 23
```

Definition at line 168 of file MRCC_private.h.

7.24.2.77 RCC_APB1ENR_I2C2EN

```
#define RCC_APB1ENR_I2C2EN 22
```

Definition at line 169 of file MRCC_private.h.

7.24.2.78 RCC_APB1ENR_I2C1EN

```
#define RCC_APB1ENR_I2C1EN 21
```

Definition at line 170 of file MRCC_private.h.

7.24.2.79 RCC_APB1ENR_USART2EN

```
#define RCC_APB1ENR_USART2EN 17
```

Definition at line 171 of file MRCC_private.h.

7.24.2.80 RCC_APB1ENR_SPI3EN

```
#define RCC_APB1ENR_SPI3EN 15
```

Definition at line 172 of file MRCC_private.h.

7.24.2.81 RCC_APB1ENR_SPI2EN

```
#define RCC_APB1ENR_SPI2EN 14
```

Definition at line 173 of file MRCC_private.h.

7.24.2.82 RCC_APB1ENR_WWDGEN

```
#define RCC_APB1ENR_WWDGEN 11
```

Definition at line 174 of file MRCC_private.h.

7.24.2.83 RCC_APB1ENR_TIM5EN

```
#define RCC_APB1ENR_TIM5EN 3
```

Definition at line 175 of file MRCC_private.h.

7.24.2.84 RCC_APB1ENR_TIM4EN

```
#define RCC_APB1ENR_TIM4EN 2
```

Definition at line 176 of file MRCC_private.h.

7.24.2.85 RCC_APB1ENR_TIM3EN

```
#define RCC_APB1ENR_TIM3EN 1
```

Definition at line 177 of file MRCC_private.h.

7.24.2.86 RCC_APB1ENR_TIM2EN

```
#define RCC_APB1ENR_TIM2EN 0
```

Definition at line 178 of file MRCC_private.h.

7.24.2.87 RCC_APB2ENR_TIM11EN

```
#define RCC_APB2ENR_TIM11EN 18
```

Definition at line 181 of file MRCC_private.h.

7.24.2.88 RCC_APB2ENR_TIM10EN

```
#define RCC_APB2ENR_TIM10EN 17
```

Definition at line 182 of file MRCC_private.h.

7.24.2.89 RCC_APB2ENR_TIM9EN

```
#define RCC_APB2ENR_TIM9EN 16
```

Definition at line 183 of file MRCC_private.h.

7.24.2.90 RCC_APB2ENR_SYSCFGEN

```
#define RCC_APB2ENR_SYSCFGEN 14
```

Definition at line 184 of file MRCC_private.h.

7.24.2.91 RCC_APB2ENR_SPI4EN

```
#define RCC_APB2ENR_SPI4EN 13
```

Definition at line 185 of file MRCC_private.h.

7.24.2.92 RCC_APB2ENR_SPI1EN

```
#define RCC_APB2ENR_SPI1EN 12
```

Definition at line 186 of file MRCC_private.h.

7.24.2.93 RCC_APB2ENR_SDIOEN

```
#define RCC_APB2ENR_SDIOEN 11
```

Definition at line 187 of file MRCC_private.h.

7.24.2.94 RCC_APB2ENR_ADC1EN

```
#define RCC_APB2ENR_ADC1EN 8
```

Definition at line 188 of file MRCC_private.h.

7.24.2.95 RCC_APB2ENR_USART6EN

```
#define RCC_APB2ENR_USART6EN 5
```

Definition at line 189 of file MRCC_private.h.

7.24.2.96 RCC_APB2ENR_USART1EN

```
#define RCC_APB2ENR_USART1EN 4
```

Definition at line 190 of file MRCC_private.h.

7.24.2.97 RCC_APB2ENR_TIM1EN

```
#define RCC_APB2ENR_TIM1EN 0
```

Definition at line 191 of file MRCC_private.h.

7.24.2.98 RCC_AHB1LPENR_FLITFLPEN

```
#define RCC_AHB1LPENR_FLITFLPEN 15
```

Definition at line 193 of file MRCC_private.h.

7.24.2.99 ENABLE

```
#define ENABLE 1
```

Definition at line 200 of file MRCC_private.h.

7.24.2.100 DISABLE

```
#define DISABLE 2
```

Definition at line 201 of file MRCC_private.h.

7.24.2.101 BYBASED

```
#define BYBASED 1
```

Definition at line 203 of file MRCC_private.h.

7.24.2.102 NOTBYBASED

```
#define NOTBYBASED 2
```

Definition at line 204 of file MRCC_private.h.

7.24.2.103 SYSCLK

```
#define SYSCLK 1
```

Definition at line 206 of file MRCC_private.h.

7.24.2.104 PLLI2SCLK

```
#define PLLI2SCLK 5
```

Definition at line 207 of file MRCC_private.h.

7.24.2.105 HSE

```
#define HSE 3
```

Definition at line 208 of file MRCC_private.h.

7.24.2.106 PLLCLK [1/2]

```
#define PLLCLK 4
```

Definition at line 222 of file MRCC_private.h.

7.24.2.107 NoDivision

```
#define NoDivision 1
```

Definition at line 211 of file MRCC_private.h.

7.24.2.108 DivisionBy2

```
#define DivisionBy2 2
```

Definition at line 212 of file MRCC_private.h.

7.24.2.109 DivisionBy3

```
#define DivisionBy3 3
```

Definition at line 213 of file MRCC_private.h.

7.24.2.110 DivisionBy4

```
#define DivisionBy4 4
```

Definition at line 214 of file MRCC_private.h.

7.24.2.111 DivisionBy5

```
#define DivisionBy5 5
```

Definition at line 215 of file MRCC_private.h.

7.24.2.112 I2S_CKIN

```
#define I2S_CKIN 2
```

Definition at line 218 of file MRCC_private.h.

7.24.2.113 HSI

```
#define HSI 1
```

Definition at line 220 of file MRCC_private.h.

7.24.2.114 LSE

```
#define LSE 2
```

Definition at line 221 of file MRCC_private.h.

7.24.2.115 PLLCLK [2/2]

```
#define PLLCLK 4
```

Definition at line 222 of file MRCC_private.h.

7.24.2.116 SYSCLKby2

```
#define SYSCLKby2 2
```

Definition at line 225 of file MRCC_private.h.

7.24.2.117 SYSCLKby4

```
#define SYSCLKby4 3
```

Definition at line 226 of file MRCC_private.h.

7.24.2.118 SYSCLKby8

```
#define SYSCLKby8 4
```

Definition at line 227 of file MRCC_private.h.

7.24.2.119 SYSCLKby16

```
#define SYSCLKby16 5
```

Definition at line 228 of file MRCC_private.h.

7.24.2.120 SYSCLKby64

```
#define SYSCLKby64 6
```

Definition at line 229 of file MRCC_private.h.

7.24.2.121 SYSCLKby128

```
#define SYSCLKby128 7
```

Definition at line 230 of file MRCC_private.h.

7.24.2.122 SYSCLKby256

```
#define SYSCLKby256 8
```

Definition at line 231 of file MRCC_private.h.

7.24.2.123 SYSCLKby512

```
#define SYSCLKby512 9
```

Definition at line 232 of file MRCC_private.h.

7.24.2.124 NoCLK0

```
#define NoCLK0 1
```

Definition at line 235 of file MRCC_private.h.

7.24.2.125 NoCLK1

```
#define NoCLK1 2
```

Definition at line 236 of file MRCC_private.h.

7.24.2.126 HSEby2

```
#define HSEby2 3
```

Definition at line 237 of file MRCC_private.h.

7.24.2.127 HSEby3

```
#define HSEby3 4
```

Definition at line 238 of file MRCC_private.h.

7.24.2.128 HSEby4

```
#define HSEby4 5
```

Definition at line 239 of file MRCC_private.h.

7.24.2.129 HSEby5

```
#define HSEby5 6
```

Definition at line 240 of file MRCC_private.h.

7.24.2.130 HSEby6

```
#define HSEby6 7
```

Definition at line 241 of file MRCC_private.h.

7.24.2.131 HSEby7

```
#define HSEby7 8
```

Definition at line 242 of file MRCC_private.h.

7.24.2.132 HSEby8

```
#define HSEby8 9
```

Definition at line 243 of file MRCC_private.h.

7.24.2.133 HSEby9

```
#define HSEby9 10
```

Definition at line 244 of file MRCC_private.h.

7.24.2.134 HSEby10

```
#define HSEby10 11
```

Definition at line 245 of file MRCC_private.h.

7.24.2.135 HSEby11

```
#define HSEby11 12
```

Definition at line 246 of file MRCC_private.h.

7.24.2.136 HSEby12

```
#define HSEby12 13
```

Definition at line 247 of file MRCC_private.h.

7.24.2.137 HSEby13

```
#define HSEby13 14
```

Definition at line 248 of file MRCC_private.h.

7.24.2.138 HSEby14

```
#define HSEby14 15
```

Definition at line 249 of file MRCC_private.h.

7.24.2.139 HSEby15

```
#define HSEby15 16
```

Definition at line 250 of file MRCC_private.h.

7.24.2.140 HSEby16

```
#define HSEby16 17
```

Definition at line 251 of file MRCC_private.h.

7.24.2.141 HSEby17

```
#define HSEby17 18
```

Definition at line 252 of file MRCC_private.h.

7.24.2.142 HSEby18

```
#define HSEby18 19
```

Definition at line 253 of file MRCC_private.h.

7.24.2.143 HSEby19

```
#define HSEby19 20
```

Definition at line 254 of file MRCC_private.h.

7.24.2.144 HSEby20

```
#define HSEby20 21
```

Definition at line 255 of file MRCC_private.h.

7.24.2.145 HSEby21

```
#define HSEby21 22
```

Definition at line 256 of file MRCC_private.h.

7.24.2.146 HSEby22

```
#define HSEby22 23
```

Definition at line 257 of file MRCC_private.h.

7.24.2.147 HSEby23

```
#define HSEby23 24
```

Definition at line 258 of file MRCC_private.h.

7.24.2.148 HSEby24

```
#define HSEby24 25
```

Definition at line 259 of file MRCC_private.h.

7.24.2.149 HSEby25

```
#define HSEby25 26
```

Definition at line 260 of file MRCC_private.h.

7.24.2.150 HSEby26

```
#define HSEby26 27
```

Definition at line 261 of file MRCC_private.h.

7.24.2.151 HSEby27

```
#define HSEby27 28
```

Definition at line 262 of file MRCC_private.h.

7.24.2.152 HSEby28

```
#define HSEby28 39
```

Definition at line 263 of file MRCC_private.h.

7.24.2.153 HSEby29

```
#define HSEby29 30
```

Definition at line 264 of file MRCC_private.h.

7.24.2.154 HSEby30

```
#define HSEby30 31
```

Definition at line 265 of file MRCC_private.h.

7.24.2.155 HSEby31

```
#define HSEby31 32
```

Definition at line 266 of file MRCC_private.h.

7.24.2.156 AHBby2

```
#define AHBby2 1
```

Definition at line 269 of file MRCC_private.h.

7.24.2.157 AHBby4

```
#define AHBby4 2
```

Definition at line 270 of file MRCC_private.h.

7.24.2.158 AHBby8

```
#define AHBby8 3
```

Definition at line 271 of file MRCC_private.h.

7.24.2.159 AHBby16

```
#define AHBby16 4
```

Definition at line 272 of file MRCC_private.h.

7.24.2.160 Equal_0

```
#define Equal_0 0
```

Definition at line 275 of file MRCC_private.h.

7.24.2.161 Equal_1

```
#define Equal_1 1
```

Definition at line 276 of file MRCC_private.h.

7.24.2.162 Equal_2

```
#define Equal_2 2
```

Definition at line 277 of file MRCC_private.h.

7.24.2.163 Equal_3

```
#define Equal_3 3
```

Definition at line 278 of file MRCC_private.h.

7.24.2.164 Equal_4

```
#define Equal_4 4
```

Definition at line 279 of file MRCC_private.h.

7.24.2.165 Equal_5

```
#define Equal_5 5
```

Definition at line 280 of file MRCC_private.h.

7.24.2.166 Equal_6

```
#define Equal_6 6
```

Definition at line 281 of file MRCC_private.h.

7.24.2.167 Equal_7

```
#define Equal_7 7
```

Definition at line 282 of file MRCC_private.h.

7.24.2.168 Equal_8

```
#define Equal_8 8
```

Definition at line 283 of file MRCC_private.h.

7.24.2.169 Equal_9

```
#define Equal_9 9
```

Definition at line 284 of file MRCC_private.h.

7.24.2.170 Equal_10

```
#define Equal_10 10
```

Definition at line 285 of file MRCC_private.h.

7.24.2.171 Equal_11

```
#define Equal_11 11
```

Definition at line 286 of file MRCC_private.h.

7.24.2.172 Equal_12

```
#define Equal_12 12
```

Definition at line 287 of file MRCC_private.h.

7.24.2.173 Equal_13

```
#define Equal_13 13
```

Definition at line 288 of file MRCC_private.h.

7.24.2.174 Equal_14

```
#define Equal_14 14
```

Definition at line 289 of file MRCC_private.h.

7.24.2.175 Equal_15

```
#define Equal_15 15
```

Definition at line 290 of file MRCC_private.h.

7.24.2.176 Equal_16

```
#define Equal_16 16
```

Definition at line 291 of file MRCC_private.h.

7.24.2.177 Equal_17

```
#define Equal_17 17
```

Definition at line 292 of file MRCC_private.h.

7.24.2.178 Equal_18

```
#define Equal_18 18
```

Definition at line 293 of file MRCC_private.h.

7.24.2.179 Equal_19

```
#define Equal_19 19
```

Definition at line 294 of file MRCC_private.h.

7.24.2.180 Equal_20

```
#define Equal_20 20
```

Definition at line 295 of file MRCC_private.h.

7.24.2.181 Equal_21

```
#define Equal_21 21
```

Definition at line 296 of file MRCC_private.h.

7.24.2.182 Equal_22

```
#define Equal_22 22
```

Definition at line 297 of file MRCC_private.h.

7.24.2.183 Equal_23

```
#define Equal_23 23
```

Definition at line 298 of file MRCC_private.h.

7.24.2.184 Equal_24

```
#define Equal_24 24
```

Definition at line 299 of file MRCC_private.h.

7.24.2.185 Equal_25

```
#define Equal_25 25
```

Definition at line 300 of file MRCC_private.h.

7.24.2.186 Equal_26

```
#define Equal_26 26
```

Definition at line 301 of file MRCC_private.h.

7.24.2.187 Equal_27

```
#define Equal_27 27
```

Definition at line 302 of file MRCC_private.h.

7.24.2.188 Equal_28

```
#define Equal_28 28
```

Definition at line 303 of file MRCC_private.h.

7.24.2.189 Equal_29

```
#define Equal_29 29
```

Definition at line 304 of file MRCC_private.h.

7.24.2.190 Equal_30

```
#define Equal_30 30
```

Definition at line 305 of file MRCC_private.h.

7.24.2.191 Equal_31

```
#define Equal_31 31
```

Definition at line 306 of file MRCC_private.h.

7.24.2.192 Equal_32

```
#define Equal_32 32
```

Definition at line 307 of file MRCC_private.h.

7.24.2.193 Equal_33

```
#define Equal_33 33
```

Definition at line 308 of file MRCC_private.h.

7.24.2.194 Equal_34

```
#define Equal_34 34
```

Definition at line 309 of file MRCC_private.h.

7.24.2.195 Equal_35

```
#define Equal_35 35
```

Definition at line 310 of file MRCC_private.h.

7.24.2.196 Equal_36

```
#define Equal_36 36
```

Definition at line 311 of file MRCC_private.h.

7.24.2.197 Equal_37

```
#define Equal_37 37
```

Definition at line 312 of file MRCC_private.h.

7.24.2.198 Equal_38

```
#define Equal_38 38
```

Definition at line 313 of file MRCC_private.h.

7.24.2.199 Equal_39

```
#define Equal_39 39
```

Definition at line 314 of file MRCC_private.h.

7.24.2.200 Equal_40

```
#define Equal_40 40
```

Definition at line 315 of file MRCC_private.h.

7.24.2.201 Equal_41

```
#define Equal_41 41
```

Definition at line 316 of file MRCC_private.h.

7.24.2.202 Equal_42

```
#define Equal_42 42
```

Definition at line 317 of file MRCC_private.h.

7.24.2.203 Equal_43

```
#define Equal_43 43
```

Definition at line 318 of file MRCC_private.h.

7.24.2.204 Equal_44

```
#define Equal_44 44
```

Definition at line 319 of file MRCC_private.h.

7.24.2.205 Equal_45

```
#define Equal_45 45
```

Definition at line 320 of file MRCC_private.h.

7.24.2.206 Equal_46

```
#define Equal_46 46
```

Definition at line 321 of file MRCC_private.h.

7.24.2.207 Equal_47

```
#define Equal_47 47
```

Definition at line 322 of file MRCC_private.h.

7.24.2.208 Equal_48

```
#define Equal_48 48
```

Definition at line 323 of file MRCC_private.h.

7.24.2.209 Equal_49

```
#define Equal_49 49
```

Definition at line 324 of file MRCC_private.h.

7.24.2.210 Equal_50

```
#define Equal_50 50
```

Definition at line 325 of file MRCC_private.h.

7.24.2.211 Equal_51

```
#define Equal_51 51
```

Definition at line 326 of file MRCC_private.h.

7.24.2.212 Equal_52

```
#define Equal_52 52
```

Definition at line 327 of file MRCC_private.h.

7.24.2.213 Equal_53

```
#define Equal_53 53
```

Definition at line 328 of file MRCC_private.h.

7.24.2.214 Equal_54

```
#define Equal_54 54
```

Definition at line 329 of file MRCC_private.h.

7.24.2.215 Equal_55

```
#define Equal_55 55
```

Definition at line 330 of file MRCC_private.h.

7.24.2.216 Equal_56

```
#define Equal_56 56
```

Definition at line 331 of file MRCC_private.h.

7.24.2.217 Equal_57

```
#define Equal_57 57
```

Definition at line 332 of file MRCC_private.h.

7.24.2.218 Equal_58

```
#define Equal_58 58
```

Definition at line 333 of file MRCC_private.h.

7.24.2.219 Equal_59

```
#define Equal_59 59
```

Definition at line 334 of file MRCC_private.h.

7.24.2.220 Equal_60

```
#define Equal_60 60
```

Definition at line 335 of file MRCC_private.h.

7.24.2.221 Equal_61

```
#define Equal_61 61
```

Definition at line 336 of file MRCC_private.h.

7.24.2.222 Equal_62

```
#define Equal_62 62
```

Definition at line 337 of file MRCC_private.h.

7.24.2.223 Equal_63

```
#define Equal_63 63
```

Definition at line 338 of file MRCC_private.h.

7.25 MRCC_private.h

Go to the documentation of this file.

```
00001
00010 #ifndef MCAL_RCC_MRCC_PRIVATE_H_
00011 #define MCAL_RCC_MRCC_PRIVATE_H_
00012
00013
00014
00015 /*****************************************************************************/
00016 /*                          Peripherals declaration                          */
00017 /*****************************************************************************/
00018
00019 typedef struct
00020 {
00021
00022     u32_t CR             ;
00023     u32_t PLLCFGR       ;
00024     u32_t CFGR          ;
00025     u32_t CIR            ;
00026     u32_t AHB1RSTR      ;
00027     u32_t AHB2RSTR      ;
00028
00029     u32_t Reserved1    ;
00030     u32_t Reserved2    ;
00031
00032     u32_t APB1RSTR      ;
00033     u32_t APB2RSTR      ;
00034
00035     u32_t Reserved3    ;
00036     u32_t Reserved4    ;
00037
00038     u32_t AHB1ENR       ;
00039     u32_t AHB2ENR       ;
00040
00041     u32_t Reserved5    ;
00042     u32_t Reserved6    ;
00043
00044     u32_t APB1ENR       ;
00045     u32_t APB2ENR       ;
00046
00047     u32_t Reserved7    ;
00048     u32_t Reserved8    ;
00049
00050     u32_t AHB1LPENR    ;
00051     u32_t AHB2LPENR    ;
00052
00053     u32_t Reserved9    ;
00054     u32_t Reserved10     ;
00055
00056     u32_t APB1LPENR    ;
00057     u32_t APB2LPENR    ;
00058
00059     u32_t Reserved11     ;
00060     u32_t Reserved12     ;
00061
00062     u32_t BDCR          ;
00063     u32_t CSR            ;
00064
00065     u32_t Reserved13     ;
00066     u32_t Reserved14     ;
00067
00068     u32_t SSCGR        ;
00069     u32_t PLLI2SCFGR     ;
00070     u32_t DCKCFGR       ;
00071
00072
00073 } RCC_MemoryMapType ;
00074
00075
00076 #define RCC_BASE_ADDRESS 0x40023800
00077
00078 #define RCC ( (volatile P2VAR(RCC_MemoryMapType) ) (RCC_BASE_ADDRESS) ) // Is a pointer to the struct.
00079
00080
00081 /*****************************************************************************/
00082 /*                          Peripheral registers                             */
00083 /*****************************************************************************/
00084
00085 #define RCC_CR_PLLI2SRDY  27
00086 #define RCC_CR_PLLI2SON   26
00087 #define RCC_CR_PLLRDY     25
00088 #define RCC_CR_PLLON      24
00089 #define RCC_CR_CSSON      19
00090 #define RCC_CR_HSEBYP     18
```

```
00091 #define RCC_CR_HSERDY     17
00092 #define RCC_CR_HSEON      16
00093 #define RCC_CR_HSIRDY     1
00094 #define RCC_CR_HSION      0
00095
00096
00097 #define RCC_PLLCFGR_PLLQ_b0 24
00098 #define RCC_PLLCFGR_PLLQ_b1 25
00099 #define RCC_PLLCFGR_PLLQ_b2 26
00100 #define RCC_PLLCFGR_PLLQ_b3 27
00101 #define RCC_PLLCFGR_PLLSRC  22
00102 #define RCC_PLLCFGR_PLLP_b0 16
00103 #define RCC_PLLCFGR_PLLP_b1 17
00104 #define RCC_PLLCFGR_PLLN_b0 6
00105 #define RCC_PLLCFGR_PLLN_b1 7
00106 #define RCC_PLLCFGR_PLLN_b2 8
00107 #define RCC_PLLCFGR_PLLN_b3 9
00108 #define RCC_PLLCFGR_PLLN_b4 10
00109 #define RCC_PLLCFGR_PLLN_b5 11
00110 #define RCC_PLLCFGR_PLLN_b6 12
00111 #define RCC_PLLCFGR_PLLN_b7 13
00112 #define RCC_PLLCFGR_PLLN_b8 14
00113 #define RCC_PLLCFGR_PLLM_b0 0
00114 #define RCC_PLLCFGR_PLLM_b1 1
00115 #define RCC_PLLCFGR_PLLM_b2 2
00116 #define RCC_PLLCFGR_PLLM_b3 3
00117 #define RCC_PLLCFGR_PLLM_b4 4
00118 #define RCC_PLLCFGR_PLLM_b5 5
00119
00120
00121 #define RCC_CFGR_MOC2_b0        30
00122 #define RCC_CFGR_MOC2_b1        31
00123 #define RCC_CFGR_MOC2PRE_b0     27
00124 #define RCC_CFGR_MOC2PRE_b1     28
00125 #define RCC_CFGR_MOC2PRE_b2     29
00126 #define RCC_CFGR_MOC1PRE_b0     24
00127 #define RCC_CFGR_MOC1PRE_b1     25
00128 #define RCC_CFGR_MOC1PRE_b2     26
00129 #define RCC_CFGR_I2SSRC         23
00130 #define RCC_CFGR_MOC1_b0        21
00131 #define RCC_CFGR_MOC1_b1        22
00132 #define RCC_CFGR_RTCPRE_b0      16
00133 #define RCC_CFGR_RTCPRE_b1      17
00134 #define RCC_CFGR_RTCPRE_b2      18
00135 #define RCC_CFGR_RTCPRE_b3      19
00136 #define RCC_CFGR_RTCPRE_b4      20
00137 #define RCC_CFGR_PPRE2_b0       13
00138 #define RCC_CFGR_PPRE2_b1       14
00139 #define RCC_CFGR_PPRE2_b2       15
00140 #define RCC_CFGR_PPRE1_b0       10
00141 #define RCC_CFGR_PPRE1_b1       11
00142 #define RCC_CFGR_PPRE1_b2       12
00143 #define RCC_CFGR_HPRE_b0        4
00144 #define RCC_CFGR_HPRE_b1        5
00145 #define RCC_CFGR_HPRE_b2        6
00146 #define RCC_CFGR_HPRE_b3        7
00147 #define RCC_CFGR_SWS_b0         2
00148 #define RCC_CFGR_SWS_b1         3
00149 #define RCC_CFGR_SW_b0          0
00150 #define RCC_CFGR_SW_b1          1
00151
00152
00153 #define RCC_AHB1ENR_DMA2EN    22
00154 #define RCC_AHB1ENR_DMA1EN    21
00155 #define RCC_AHB1ENR_CRCEN     12
00156 #define RCC_AHB1ENR_GPIOHEN   7
00157 #define RCC_AHB1ENR_GPIOEEN   4
00158 #define RCC_AHB1ENR_GPIODEN   3
00159 #define RCC_AHB1ENR_GPIOCEN   2
00160 #define RCC_AHB1ENR_GPIOBEN   1
00161 #define RCC_AHB1ENR_GPIOAEN   0
00162
00163
00164 #define RCC_AHB2ENR_OTGFSEN   7
00165
00166
00167 #define RCC_APB1ENR_PWREN       28
00168 #define RCC_APB1ENR_I2C3EN      23
00169 #define RCC_APB1ENR_I2C2EN      22
00170 #define RCC_APB1ENR_I2C1EN      21
00171 #define RCC_APB1ENR_USART2EN    17
00172 #define RCC_APB1ENR_SPI3EN      15
00173 #define RCC_APB1ENR_SPI2EN      14
00174 #define RCC_APB1ENR_WWDGEN      11
00175 #define RCC_APB1ENR_TIM5EN      3
00176 #define RCC_APB1ENR_TIM4EN      2
00177 #define RCC_APB1ENR_TIM3EN      1
```

```
00178 #define RCC_APB1ENR_TIM2EN      0
00179
00180
00181 #define RCC_APB2ENR_TIM11EN     18
00182 #define RCC_APB2ENR_TIM10EN     17
00183 #define RCC_APB2ENR_TIM9EN      16
00184 #define RCC_APB2ENR_SYSCFGEN    14
00185 #define RCC_APB2ENR_SPI4EN      13
00186 #define RCC_APB2ENR_SPI1EN      12
00187 #define RCC_APB2ENR_SDIOEN      11
00188 #define RCC_APB2ENR_ADC1EN      8
00189 #define RCC_APB2ENR_USART6EN    5
00190 #define RCC_APB2ENR_USART1EN    4
00191 #define RCC_APB2ENR_TIM1EN      0
00192
00193 #define RCC_AHB1LPENR_FLITFLPEN 15
00194
00195
00196 /*****************************************************************************/
00197 /*                          Config.h Macros                                  */
00198 /*****************************************************************************/
00199
00200 #define ENABLE  1
00201 #define DISABLE 2
00202
00203 #define BYBASED     1
00204 #define NOTBYBASED  2
00205
00206 #define SYSCLK      1
00207 #define PLLI2SCLK   5
00208 #define HSE         3
00209 #define PLLCLK      4
00210
00211 #define NoDivision  1
00212 #define DivisionBy2 2
00213 #define DivisionBy3 3
00214 #define DivisionBy4 4
00215 #define DivisionBy5 5
00216
00217
00218 #define I2S_CKIN  2
00219
00220 #define HSI     1
00221 #define LSE     2
00222 #define PLLCLK  4
00223
00224 // AHB prescalers:
00225 #define SYSCLKby2   2   //(0b1000)
00226 #define SYSCLKby4   3   //(0b1001)
00227 #define SYSCLKby8   4   //(0b1010)
00228 #define SYSCLKby16  5   //(0b1011)
00229 #define SYSCLKby64  6   //(0b1100)
00230 #define SYSCLKby128 7   //(0b1101)
00231 #define SYSCLKby256 8   //(0b1110)
00232 #define SYSCLKby512 9   //(0b1111)
00233
00234
00235 #define NoCLK0  1
00236 #define NoCLK1  2
00237 #define HSEby2  3
00238 #define HSEby3  4
00239 #define HSEby4  5
00240 #define HSEby5  6
00241 #define HSEby6  7
00242 #define HSEby7  8
00243 #define HSEby8  9
00244 #define HSEby9  10
00245 #define HSEby10 11
00246 #define HSEby11 12
00247 #define HSEby12 13
00248 #define HSEby13 14
00249 #define HSEby14 15
00250 #define HSEby15 16
00251 #define HSEby16 17
00252 #define HSEby17 18
00253 #define HSEby18 19
00254 #define HSEby19 20
00255 #define HSEby20 21
00256 #define HSEby21 22
00257 #define HSEby22 23
00258 #define HSEby23 24
00259 #define HSEby24 25
00260 #define HSEby25 26
00261 #define HSEby26 27
00262 #define HSEby27 28
00263 #define HSEby28 39
00264 #define HSEby29 30
```

```
00265 #define HSEby30 31
00266 #define HSEby31 32
00267
00268
00269 #define AHBby2  1
00270 #define AHBby4  2
00271 #define AHBby8  3
00272 #define AHBby16 4
00273
00274
00275 #define Equal_0     0
00276 #define Equal_1     1
00277 #define Equal_2     2
00278 #define Equal_3     3
00279 #define Equal_4     4
00280 #define Equal_5     5
00281 #define Equal_6     6
00282 #define Equal_7     7
00283 #define Equal_8     8
00284 #define Equal_9     9
00285 #define Equal_10    10
00286 #define Equal_11    11
00287 #define Equal_12    12
00288 #define Equal_13    13
00289 #define Equal_14    14
00290 #define Equal_15    15
00291 #define Equal_16    16
00292 #define Equal_17    17
00293 #define Equal_18    18
00294 #define Equal_19    19
00295 #define Equal_20    20
00296 #define Equal_21    21
00297 #define Equal_22    22
00298 #define Equal_23    23
00299 #define Equal_24    24
00300 #define Equal_25    25
00301 #define Equal_26    26
00302 #define Equal_27    27
00303 #define Equal_28    28
00304 #define Equal_29    29
00305 #define Equal_30    30
00306 #define Equal_31    31
00307 #define Equal_32    32
00308 #define Equal_33    33
00309 #define Equal_34    34
00310 #define Equal_35    35
00311 #define Equal_36    36
00312 #define Equal_37    37
00313 #define Equal_38    38
00314 #define Equal_39    39
00315 #define Equal_40    40
00316 #define Equal_41    41
00317 #define Equal_42    42
00318 #define Equal_43    43
00319 #define Equal_44    44
00320 #define Equal_45    45
00321 #define Equal_46    46
00322 #define Equal_47    47
00323 #define Equal_48    48
00324 #define Equal_49    49
00325 #define Equal_50    50
00326 #define Equal_51    51
00327 #define Equal_52    52
00328 #define Equal_53    53
00329 #define Equal_54    54
00330 #define Equal_55    55
00331 #define Equal_56    56
00332 #define Equal_57    57
00333 #define Equal_58    58
00334 #define Equal_59    59
00335 #define Equal_60    60
00336 #define Equal_61    61
00337 #define Equal_62    62
00338 #define Equal_63    63
00339
00340
00341
00342
00343
00344 #endif /* MCAL_RCC_MRCC_PRIVATE_H_ */
```

7.26 src/COTS/MCAL/RCC/MRCC_program.c File Reference

This file contains the source code of the interfacing for the RCC module.

```
#include "../../LIB/LSTD_TYPES.h"
#include "../../LIB/LSTD_COMPILER.h"
#include "../../LIB/LSTD_VALUES.h"
#include "../../LIB/LSTD_BITMATH.h"
#include "MRCC_private.h"
#include "MRCC_interface.h"
#include "MRCC_config.h"
```

Functions

- void MRCC_vInit (void)
- void MRCC_vEnablePeriphralCLK (u32_t A_u32BusID, u32_t A_u32PeriphralID)
- void MRCC_vDisablePeriphralCLK (u32_t A_u32BusID, u32_t A_u32PeriphralID)

7.26.1 Detailed Description

This file contains the source code of the interfacing for the RCC module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file MRCC_program.c.

7.26.2 Function Documentation

7.26.2.1 MRCC_vInit()

```
void MRCC_vInit (
             void  )
```

Definition at line 29 of file MRCC_program.c.

```
00030 {
00031
00032     // PLLI2S (ON/OFF).
00033 #if PLLI2S == ENABLE
00034
00035         SET_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00036
00037 #elif PLLI2S == DISABLE
00038
00039         CLR_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00040
00041 #endif
00042
00043
00044     // PLL (ON/OFF).
00045 #if PLL == ENABLE
00046
00047         SET_BIT( RCC->CR, RCC_CR_PLLON ) ;
00048
00049 #elif PLL == DISABLE
00050
00051         CLR_BIT( RCC->CR, RCC_CR_PLLON ) ;
00052
00053 #endif
00054
00055
00056     // CSS (ON/OFF).
00057 #if CSS == ENABLE
00058
00059         SET_BIT( RCC->CR, RCC_CR_CSSON ) ;
00060
00061 #elif CSS == DISABLE
00062
00063         CLR_BIT( RCC->CR, RCC_CR_CSSON ) ;
00064
00065 #endif
00066
00067
00068     // HSEBYP.
00069 #if HSEBYP == BYBASED
00070
00071         SET_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00072
00073 #elif HSEBYP == NOTBYBASED
00074
00075         CLR_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00076
00077 #endif
00078
00079
00080     // Select CLK switch (HSI/HSE/PLL).
00081 #if SW == HSI
00082
00083         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00084         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00085
00086 #elif SW == HSE
00087
00088         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00089         SET_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00090
00091 #elif SW == PLLCLK
00092
00093         SET_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00094         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00095
00096 #endif
00097
00098
00099     // MCO2 selection:
00100 #if MCO2 == SYSCLK
00101
00102         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00103         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00104
00105 #elif MCO2 == PLLI2SCLK
00106
00107         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
```

```
00108         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00109
00110 #elif MCO2 == HSE
00111
00112         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00113         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00114
00115 #elif MCO2 == PLLCLK
00116
00117         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00118         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00119
00120 #endif
00121
00122
00123     // MCO2 prescaler:
00124 #if MCO2PRE == NoDivision
00125
00126         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00127         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00128         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00129
00130 #elif MCO2PRE == DivisionBy2
00131
00132         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00133         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00134         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00135
00136 #elif MCO2PRE == DivisionBy3
00137
00138         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00139         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00140         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00141
00142 #elif MCO2PRE == DivisionBy4
00143
00144         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00145         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00146         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00147
00148 #elif MCO2PRE == DivisionBy5
00149
00150         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00151         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00152         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00153
00154 #endif
00155
00156
00157     // MCO1 selection:
00158 #if MCO1 == HSI
00159
00160         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00161         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00162
00163 #elif MCO1 == LSE
00164
00165         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00166         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00167
00168 #elif MCO1 == HSE
00169
00170         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00171         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00172
00173 #elif MCO1 == PLLCLK
00174
00175         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00176         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00177
00178 #endif
00179
00180
00181     // MCO1 prescaler:
00182 #if MCO1PRE == NoDivision
00183
00184         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00185         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00186         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00187
00188 #elif MCO1PRE == DivisionBy2
00189
00190         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00191         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00192         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00193
00194 #elif MCO1PRE == DivisionBy3
```

```
00195
00196         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00197         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00198         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00199
00200 #elif MCO1PRE == DivisionBy4
00201
00202         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00203         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00204         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00205
00206 #elif MCO1PRE == DivisionBy5
00207
00208         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00209         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00210         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00211
00212 #endif
00213
00214
00215     // AHB prescalers:
00216 #if HPRE == NoDivision
00217
00218         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00219         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00220         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00221         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00222
00223 #elif HPRE == SYSCLKby2
00224
00225         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00226         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00227         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00228         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00229
00230 #elif HPRE == SYSCLKby4
00231
00232         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00233         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00234         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00235         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00236
00237 #elif HPRE == SYSCLKby8
00238
00239         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00240         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00241         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00242         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00243
00244 #elif HPRE == SYSCLKby16
00245
00246         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00247         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00248         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00249         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00250
00251 #elif HPRE == SYSCLKby64
00252
00253         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00254         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00255         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00256         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00257
00258 #elif HPRE == SYSCLKby128
00259
00260         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00261         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00262         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00263         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00264
00265 #elif HPRE == SYSCLKby256
00266
00267         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00268         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00269         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00270         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00271
00272 #elif HPRE == SYSCLKby512
00273
00274         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00275         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00276         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00277         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00278
00279 #endif
00280
00281
```

```
00282     // APB1 prescalers:
00283 #if PPRE1 == NoDivision
00284
00285         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00286         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00287         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00288
00289 #elif PPRE1 == AHBby2
00290
00291         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00292         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00293         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00294
00295 #elif PPRE1 == AHBby4
00296
00297         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00298         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00299         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00300
00301 #elif PPRE1 == AHBby8
00302
00303         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00304         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00305         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00306
00307 #elif PPRE1 == AHBby16
00308
00309         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00310         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00311         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00312
00313 #endif
00314
00315     // APB2 prescalers:
00316 #if PPRE2 == NoDivision
00317
00318         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00319         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00320         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00321
00322 #elif PPRE2 == AHBby2
00323
00324         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00325         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00326         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00327
00328 #elif PPRE2 == AHBby4
00329
00330         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00331         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00332         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00333
00334 #elif PPRE2 == AHBby8
00335
00336         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00337         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00338         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00339
00340 #elif PPRE2 == AHBby16
00341
00342         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00343         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00344         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00345
00346 #endif
00347
00348
00349     // PLL configurations:
00350
00351
00352
00353     // Enable the selected CLK (HSI ON/HSE ON/PLL ON): //
00354     // HSE (ON/OFF).
00355 #if HSE_EN == ENABLE
00356
00357         SET_BIT( RCC->CR, RCC_CR_HSEON ) ;
00358
00359 #elif HSE_EN == DISABLE
00360
00361         CLR_BIT( RCC->CR, RCC_CR_HSEON ) ;
00362
00363 #endif
00364
00365
00366     // HSI (ON/OFF).
00367 #if HSI_EN == ENABLE
00368
```

```
00369         SET_BIT( RCC->CR, RCC_CR_HSION ) ;
00370
00371 #elif HSI_EN == DISABLE
00372
00373         CLR_BIT( RCC->CR, RCC_CR_HSION ) ;
00374
00375 #endif
00376
00377 }
```

References CLR_BIT, RCC, RCC_CFGR_HPRE_b0, RCC_CFGR_HPRE_b1, RCC_CFGR_HPRE_b2, RCC_CFGR_HPRE_b3, RCC_CFGR_MOC1_b0, RCC_CFGR_MOC1_b1, RCC_CFGR_MOC1PRE_b0, RCC_CFGR_MOC1PRE_b1, RCC_CFGR_MOC1PRE_b2, RCC_CFGR_MOC2_b0, RCC_CFGR_MOC2_b1, RCC_CFGR_MOC2PRE_b0, RCC_CFGR_MOC2PRE_b1, RCC_CFGR_MOC2PRE_b2, RCC_CFGR_PPRE1_b0, RCC_CFGR_PPRE1_b1, RCC_CFGR_PPRE1_b2, RCC_CFGR_PPRE2_b0, RCC_CFGR_PPRE2_b1, RCC_CFGR_PPRE2_b2, RCC_CFGR_SW_b0, RCC_CFGR_SW_b1, RCC_CR_CSSON, RCC_CR_HSEBYP, RCC_CR_HSEON, RCC_CR_HSION, RCC_CR_PLLI2SON, RCC_CR_PLLON, and SET_BIT.

7.26.2.2 MRCC_vEnablePeriphralCLK()

```
void MRCC_vEnablePeriphralCLK (
             u32_t A_u32BusID,
             u32_t A_u32PeriphralID )
```

Definition at line 382 of file MRCC_program.c.

```
00383 {
00384
00385     switch( A_u32BusID )
00386     {
00387
00388         case RCC_AHB1 :
00389
00390             SET_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00391
00392         break ;
00393
00394
00395         case RCC_AHB2 :
00396
00397             SET_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00398
00399
00400         break ;
00401
00402
00403         case RCC_APB1 :
00404
00405             SET_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00406
00407         break ;
00408
00409
00410         case RCC_APB2 :
00411
00412             SET_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00413
00414         break ;
00415
00416         case RCC_AHB1LPENR:
00417
00418             SET_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00419
00420         break ;
00421
00422         default:
00423
00424             // Error wrong Bus ID
00425
00426         break ;
00427
00428     }
00429
00430 }
```

References RCC, RCC_AHB1, RCC_AHB1LPENR, RCC_AHB2, RCC_APB1, RCC_APB2, and SET_BIT.

7.26.2.3 MRCC_vDisablePeriphralCLK()

```
void MRCC_vDisablePeriphralCLK (
             u32_t A_u32BusID,
             u32_t A_u32PeriphralID )
```

Definition at line 435 of file MRCC_program.c.

```
00436 {
00437
00438     switch( A_u32BusID )
00439     {
00440
00441         case RCC_AHB1 :
00442
00443             CLR_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00444
00445         break ;
00446
00447
00448         case RCC_AHB2 :
00449
00450             CLR_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00451
00452
00453         break ;
00454
00455
00456         case RCC_APB1 :
00457
00458             CLR_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00459
00460         break ;
00461
00462
00463         case RCC_APB2 :
00464
00465             CLR_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00466
00467         break ;
00468
00469
00470         case RCC_AHB1LPENR:
00471
00472             CLR_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00473
00474         break ;
00475
00476
00477         default:
00478
00479             // Error wrong Bus ID
00480
00481         break ;
00482
00483     }
00484
00485 }
```

References CLR_BIT, RCC, RCC_AHB1, RCC_AHB1LPENR, RCC_AHB2, RCC_APB1, and RCC_APB2.

7.27 MRCC_program.c

Go to the documentation of this file.

```
00001
00011 /*****************************************************************************/
00012 /*                          Include headers                                  */
00013 /*****************************************************************************/
00014 #include "../../LIB/LSTD_TYPES.h"
00015 #include "../../LIB/LSTD_COMPILER.h"
00016 #include "../../LIB/LSTD_VALUES.h"
00017 #include "../../LIB/LSTD_BITMATH.h"
00018
00019 #include "MRCC_private.h"
00020 #include "MRCC_interface.h"
00021 #include "MRCC_config.h"
00022
```

```
00023
00024
00025 /*****************************************************************************/
00026 /*                      Functions  implementations                           */
00027 /*****************************************************************************/
00028
00029 FUNC(void) MRCC_vInit( void )
00030 {
00031
00032     // PLLI2S (ON/OFF).
00033 #if PLLI2S == ENABLE
00034
00035         SET_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00036
00037 #elif PLLI2S == DISABLE
00038
00039         CLR_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00040
00041 #endif
00042
00043
00044     // PLL (ON/OFF).
00045 #if PLL == ENABLE
00046
00047         SET_BIT( RCC->CR, RCC_CR_PLLON ) ;
00048
00049 #elif PLL == DISABLE
00050
00051         CLR_BIT( RCC->CR, RCC_CR_PLLON ) ;
00052
00053 #endif
00054
00055
00056     // CSS (ON/OFF).
00057 #if CSS == ENABLE
00058
00059         SET_BIT( RCC->CR, RCC_CR_CSSON ) ;
00060
00061 #elif CSS == DISABLE
00062
00063         CLR_BIT( RCC->CR, RCC_CR_CSSON ) ;
00064
00065 #endif
00066
00067
00068     // HSEBYP.
00069 #if HSEBYP == BYBASED
00070
00071         SET_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00072
00073 #elif HSEBYP == NOTBYBASED
00074
00075         CLR_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00076
00077 #endif
00078
00079
00080     // Select CLK switch (HSI/HSE/PLL).
00081 #if SW == HSI
00082
00083         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00084         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00085
00086 #elif SW == HSE
00087
00088         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00089         SET_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00090
00091 #elif SW == PLLCLK
00092
00093         SET_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00094         CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00095
00096 #endif
00097
00098
00099     // MCO2 selection:
00100 #if MCO2 == SYSCLK
00101
00102         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00103         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00104
00105 #elif MCO2 == PLLI2SCLK
00106
00107         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00108         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00109
```

```
00110 #elif MCO2 == HSE
00111
00112         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00113         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00114
00115 #elif MCO2 == PLLCLK
00116
00117         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00118         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00119
00120 #endif
00121
00122
00123     // MCO2 prescaler:
00124 #if MCO2PRE == NoDivision
00125
00126         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00127         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00128         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00129
00130 #elif MCO2PRE == DivisionBy2
00131
00132         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00133         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00134         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00135
00136 #elif MCO2PRE == DivisionBy3
00137
00138         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00139         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00140         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00141
00142 #elif MCO2PRE == DivisionBy4
00143
00144         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00145         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00146         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00147
00148 #elif MCO2PRE == DivisionBy5
00149
00150         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00151         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00152         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00153
00154 #endif
00155
00156
00157     // MCO1 selection:
00158 #if MCO1 == HSI
00159
00160         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00161         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00162
00163 #elif MCO1 == LSE
00164
00165         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00166         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00167
00168 #elif MCO1 == HSE
00169
00170         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00171         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00172
00173 #elif MCO1 == PLLCLK
00174
00175         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00176         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00177
00178 #endif
00179
00180
00181     // MCO1 prescaler:
00182 #if MCO1PRE == NoDivision
00183
00184         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00185         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00186         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00187
00188 #elif MCO1PRE == DivisionBy2
00189
00190         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00191         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00192         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00193
00194 #elif MCO1PRE == DivisionBy3
00195
00196         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
```

```
00197         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00198         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00199
00200 #elif MCO1PRE == DivisionBy4
00201
00202         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00203         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00204         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00205
00206 #elif MCO1PRE == DivisionBy5
00207
00208         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00209         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00210         SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00211
00212 #endif
00213
00214
00215     // AHB prescalers:
00216 #if HPRE == NoDivision
00217
00218         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00219         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00220         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00221         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00222
00223 #elif HPRE == SYSCLKby2
00224
00225         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00226         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00227         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00228         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00229
00230 #elif HPRE == SYSCLKby4
00231
00232         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00233         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00234         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00235         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00236
00237 #elif HPRE == SYSCLKby8
00238
00239         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00240         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00241         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00242         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00243
00244 #elif HPRE == SYSCLKby16
00245
00246         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00247         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00248         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00249         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00250
00251 #elif HPRE == SYSCLKby64
00252
00253         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00254         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00255         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00256         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00257
00258 #elif HPRE == SYSCLKby128
00259
00260         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00261         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00262         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00263         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00264
00265 #elif HPRE == SYSCLKby256
00266
00267         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00268         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00269         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00270         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00271
00272 #elif HPRE == SYSCLKby512
00273
00274         SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00275         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00276         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00277         CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00278
00279 #endif
00280
00281
00282     // APB1 prescalers:
00283 #if PPRE1 == NoDivision
```

```
00284
00285         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00286         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00287         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00288
00289 #elif PPRE1 == AHBby2
00290
00291         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00292         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00293         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00294
00295 #elif PPRE1 == AHBby4
00296
00297         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00298         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00299         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00300
00301 #elif PPRE1 == AHBby8
00302
00303         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00304         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00305         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00306
00307 #elif PPRE1 == AHBby16
00308
00309         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b0 ) ;
00310         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b1 ) ;
00311         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE1_b2 ) ;
00312
00313 #endif
00314
00315     // APB2 prescalers:
00316 #if PPRE2 == NoDivision
00317
00318         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00319         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00320         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00321
00322 #elif PPRE2 == AHBby2
00323
00324         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00325         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00326         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00327
00328 #elif PPRE2 == AHBby4
00329
00330         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00331         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00332         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00333
00334 #elif PPRE2 == AHBby8
00335
00336         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00337         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00338         CLR_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00339
00340 #elif PPRE2 == AHBby16
00341
00342         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00343         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00344         SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00345
00346 #endif
00347
00348
00349     // PLL configurations:
00350
00351
00352
00353     // Enable the selected CLK (HSI ON/HSE ON/PLL ON): //
00354     // HSE (ON/OFF).
00355 #if HSE_EN == ENABLE
00356
00357         SET_BIT( RCC->CR, RCC_CR_HSEON ) ;
00358
00359 #elif HSE_EN == DISABLE
00360
00361         CLR_BIT( RCC->CR, RCC_CR_HSEON ) ;
00362
00363 #endif
00364
00365
00366     // HSI (ON/OFF).
00367 #if HSI_EN == ENABLE
00368
00369         SET_BIT( RCC->CR, RCC_CR_HSION ) ;
00370
```

```
00371 #elif HSI_EN == DISABLE
00372
00373         CLR_BIT( RCC->CR, RCC_CR_HSION ) ;
00374
00375 #endif
00376
00377 }
00378
00379 /*****************************************************************/
00380 /*****************************************************************/
00381
00382 FUNC(void) MRCC_vEnablePeriphralCLK( VAR(u32_t) A_u32BusID, VAR(u32_t) A_u32PeriphralID )
00383 {
00384
00385     switch( A_u32BusID )
00386     {
00387
00388         case RCC_AHB1 :
00389
00390             SET_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00391
00392         break ;
00393
00394
00395         case RCC_AHB2 :
00396
00397             SET_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00398
00399
00400         break ;
00401
00402
00403         case RCC_APB1 :
00404
00405             SET_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00406
00407         break ;
00408
00409
00410         case RCC_APB2 :
00411
00412             SET_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00413
00414         break ;
00415
00416         case RCC_AHB1LPENR:
00417
00418             SET_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00419
00420         break ;
00421
00422         default:
00423
00424             // Error wrong Bus ID
00425
00426         break ;
00427
00428     }
00429
00430 }
00431
00432 /*****************************************************************/
00433 /*****************************************************************/
00434
00435 FUNC(void) MRCC_vDisablePeriphralCLK(  VAR(u32_t) A_u32BusID, VAR(u32_t) A_u32PeriphralID )
00436 {
00437
00438     switch( A_u32BusID )
00439     {
00440
00441         case RCC_AHB1 :
00442
00443             CLR_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00444
00445         break ;
00446
00447
00448         case RCC_AHB2 :
00449
00450             CLR_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00451
00452
00453         break ;
00454
00455
00456         case RCC_APB1 :
00457
```

```
00458             CLR_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00459
00460         break ;
00461
00462
00463         case RCC_APB2 :
00464
00465             CLR_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00466
00467         break ;
00468
00469
00470         case RCC_AHB1LPENR:
00471
00472             CLR_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00473
00474         break ;
00475
00476
00477         default:
00478
00479             // Error wrong Bus ID
00480
00481         break ;
00482
00483     }
00484
00485 }
00486
00487 /*****************************************************************/
00488 /*****************************************************************/
00489
00490
00491
00492
00493
00494
00495
00496
00497
```

Index