Boatloader

AUTHOR
Version

Table of Contents

Table of contents

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

File Index

File List

Here is a list of all files with brief descriptions:

Data Structure Documentation

FDI_t Struct Reference

`#include <MFDI_private.h>`

Data Fields

- **u32 ACR**
- **u32 KEYR**
- **u32 OPTKEYR**
- **u32 SR**
- **u32 CR**
- **u32 OPTCR**

Field Documentation

u32 ACR

u32 CR

u32 KEYR

u32 OPTCR

u32 OPTKEYR

u32 SR

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/mfdi/**MFDI_private.h**

ST_MGPIO_altPinCfg_t Struct Reference

`#include <gpio_config.h>`

Data Fields

- **ST_MGPIOx_RegistersMap_t * PS_GPIOx**
- **EN_MGPIO_pinOptions_t copy_uddtPinNum**
- **EN_MGPIO_altfnOptions_t Copy_uddtAltFun**
- **EN_MGPIO_outputResistorOptions_t copy_uddtOutputResistor**
- **EN_MGPIO_outputSpeedOptions_t copy_uddtOutputSpeed**
- **EN_MGPIO_pushPullOptions_t copy_uddtPullState**

Field Documentation

EN_MGPIO_altfnOptions_t Copy_uddtAltFun

EN_MGPIO_outputResistorOptions_t copy_uddtOutputResistor

EN_MGPIO_outputSpeedOptions_t copy_uddtOutputSpeed

EN_MGPIO_pinOptions_t copy_uddtPinNum

EN_MGPIO_pushPullOptions_t copy_uddtPullState

ST_MGPIOx_RegistersMap_t* PS_GPIOx

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/gpio/**gpio_config.h**

ST_MGPIO_pinCfg_t Struct Reference

`#include <gpio_config.h>`

Data Fields

- **ST_MGPIOx_RegistersMap_t * PS_GPIOx**
- **EN_MGPIO_pinOptions_t copy_uddtPinNum**
- **EN_MGPIO_pinModeOptions_t copy_uddtPinMode**
- **EN_MGPIO_outputResistorOptions_t copy_uddtOutputResistor**
- **EN_MGPIO_outputSpeedOptions_t copy_uddtOutputSpeed**
- **EN_MGPIO_pinLogicOptions_t copy_uddtPtrRetOfPinLogic**
- **EN_MGPIO_pushPullOptions_t copy_uddtPullState**

Field Documentation

EN_MGPIO_outputResistorOptions_t copy_uddtOutputResistor

EN_MGPIO_outputSpeedOptions_t copy_uddtOutputSpeed

EN_MGPIO_pinModeOptions_t copy_uddtPinMode

EN_MGPIO_pinOptions_t copy_uddtPinNum

EN_MGPIO_pinLogicOptions_t copy_uddtPtrRetOfPinLogic

EN_MGPIO_pushPullOptions_t copy_uddtPullState

ST_MGPIOx_RegistersMap_t* PS_GPIOx

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/gpio/**gpio_config.h**

ST_MGPIOx_RegistersMap_t Struct Reference

`#include <gpio_private.h>`

Data Fields

- **vuint32_t MGPIOx_MODER**
- **vuint32_t MGPIOx_OTYPER**
- **vuint32_t MGPIOx_OSPEEDR**
- **vuint32_t MGPIOx_PUPDR**
- **vuint32_t MGPIOx_IDR**
- **vuint32_t MGPIOx_ODR**
- **vuint32_t MGPIOx_BSRR**
- **vuint32_t MGPIOx_LCKR**
- **vuint32_t MGPIOx_AFRL**
- **vuint32_t MGPIOx_AFRH**

Field Documentation

vuint32_t MGPIOx_AFRH

vuint32_t MGPIOx_AFRL

vuint32_t MGPIOx_BSRR

vuint32_t MGPIOx_IDR

vuint32_t MGPIOx_LCKR

vuint32_t MGPIOx_MODER

vuint32_t MGPIOx_ODR

vuint32_t MGPIOx_OSPEEDR

vuint32_t MGPIOx_OTYPER

vuint32_t MGPIOx_PUPDR

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/gpio/**gpio_private.h**

ST_MRCC_RegistersMap_t Struct Reference

`#include <rcc_private.h>`

Data Fields

- **vuint32_t RCC_CR_REG**
- **vuint32_t RCC_PLLCFGR_REG**
- **vuint32_t RCC_CFGR_REG**
- **vuint32_t RCC_CIR_REG**
- **vuint32_t RCC_AHB1RSTR_REG**
- **vuint32_t RCC_AHB2RSTR_REG**
- **vuint32_t RESERVED0_REG**
- **vuint32_t RESERVED1_REG**
- **vuint32_t RCC_APB1RSTR_REG**
- **vuint32_t RCC_APB2RSTR_REG**
- **vuint32_t RESERVED2_REG**
- **vuint32_t RESERVED3_REG**
- **vuint32_t RCC_AHB1ENR_REG**
- **vuint32_t RCC_AHB2ENR_REG**
- **vuint32_t Reserved5_REG**
- **vuint32_t Reserved6_REG**
- **vuint32_t RCC_APB1ENR_REG**
- **vuint32_t RCC_APB2ENR_REG**
- **vuint32_t RESERVED7_REG**
- **vuint32_t RESERVED8_REG**
- **vuint32_t RCC_AHB1LPENR_REG**
- **vuint32_t RCC_AHB2LPENR_REG**
- **vuint32_t RESERVED9_REG**
- **vuint32_t RESERVED10_REG**
- **vuint32_t RCC_APB1LPENR_REG**
- **vuint32_t RCC_APB2LPENR_REG**
- **vuint32_t RESERVED11_REG**
- **vuint32_t RESERVED12_REG**
- **vuint32_t RCC_BDCR_REG**
- **vuint32_t RCC_CSR_REG**
- **vuint32_t RESERVED13_REG**
- **vuint32_t RESERVED14_REG**
- **vuint32_t RCC_SSCGR_REG**
- **vuint32_t RCC_PLLI2SCFGR_REG**
- **vuint32_t RESERVED15_REG**
- **vuint32_t RCC_DCKCFGR_REG**

Field Documentation

vuint32_t RCC_AHB1ENR_REG

vuint32_t RCC_AHB1LPENR_REG

vuint32_t RCC_AHB1RSTR_REG

vuint32_t RCC_AHB2ENR_REG

vuint32_t RCC_AHB2LPENR_REG

vuint32_t RCC_AHB2RSTR_REG

vuint32_t RCC_APB1ENR_REG

vuint32_t RCC_APB1LPENR_REG

vuint32_t RCC_APB1RSTR_REG

vuint32_t RCC_APB2ENR_REG

vuint32_t RCC_APB2LPENR_REG

vuint32_t RCC_APB2RSTR_REG

vuint32_t RCC_BDCR_REG

vuint32_t RCC_CFGR_REG

vuint32_t RCC_CIR_REG

vuint32_t RCC_CR_REG

vuint32_t RCC_CSR_REG

vuint32_t RCC_DCKCFGR_REG

vuint32_t RCC_PLLCFGR_REG

vuint32_t RCC_PLLI2SCFGR_REG

vuint32_t RCC_SSCGR_REG

vuint32_t RESERVED0_REG

vuint32_t RESERVED10_REG

vuint32_t RESERVED11_REG

vuint32_t RESERVED12_REG

vuint32_t RESERVED13_REG

vuint32_t RESERVED14_REG

vuint32_t RESERVED15_REG

vuint32_t RESERVED1_REG

vuint32_t RESERVED2_REG

vuint32_t RESERVED3_REG

vuint32_t Reserved5_REG

vuint32_t Reserved6_REG

vuint32_t RESERVED7_REG

vuint32_t RESERVED8_REG

vuint32_t RESERVED9_REG

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/rcc/**rcc_private.h**

ST_MSTK_RegistersMap_t Struct Reference

`#include <systick_private.h>`

Data Fields

- **vuint32_t MSTK_STK_CTRL**
- **vuint32_t MSTK_STK_LOAD**
- **vuint32_t MSTK_STK_VAL**
- **vuint32_t MSTK_STK_CALIB**

Field Documentation

vuint32_t MSTK_STK_CALIB

vuint32_t MSTK_STK_CTRL

vuint32_t MSTK_STK_LOAD

vuint32_t MSTK_STK_VAL

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/systick/**systick_private.h**

ST_MUART_RegistersMap_t Struct Reference

`#include <usart_private.h>`

Data Fields

- **vuint32_t MUSART_SR**
- **vuint32_t MUSART_DR**
- **vuint32_t MUSART_BRR**
- **vuint32_t MUSART_CR1**
- **vuint32_t MUSART_CR2**
- **vuint32_t MUSART_CR3**
- **vuint32_t MUSART_GTPR**

Field Documentation

vuint32_t MUSART_BRR

vuint32_t MUSART_CR1

vuint32_t MUSART_CR2

vuint32_t MUSART_CR3

vuint32_t MUSART_DR

vuint32_t MUSART_GTPR

vuint32_t MUSART_SR

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/usart/**usart_private.h**

ST_MUSART_cfg_t Struct Reference

Structure for USART configuration.
`#include <usart_config.h>`

Data Fields

- **EN_MUSART_transferControl_t copy_uddtTransferDirection**
- **EN_MUSART_samplingModeOptions_t copy_uddtSamplingModeOption**
- **EN_MUSART_baudRateOptions_t copy_uddtBuadRateOption**
- **EN_MUSART_dataSizeOptions_t copy_uddtDataSizeOption**
- **EN_MUSART_parityControlOption_t copy_uddtParityControl**
- **EN_MUSART_paritySelectionOption_t copy_uddtParitySelection**
- **EN_MUSART_stopBitOption_t copy_uddtStopBitSelection**
- **uint8_t copy_HardwareFlowControl**
- **ST_MUSART_clockInit_t copy_uddtUartClockInit**

Detailed Description

Structure for USART configuration.

Field Documentation

uint8_t copy_HardwareFlowControl

Hardware flow control.

EN_MUSART_baudRateOptions_t copy_uddtBuadRateOption

Baud rate option.

EN_MUSART_dataSizeOptions_t copy_uddtDataSizeOption

Data size option.

EN_MUSART_parityControlOption_t copy_uddtParityControl

Parity control option.

EN_MUSART_paritySelectionOption_t copy_uddtParitySelection

Parity selection option.

EN_MUSART_samplingModeOptions_t copy_uddtSamplingModeOption

Sampling mode option.

EN_MUSART_stopBitOption_t copy_uddtStopBitSelection

Stop bit option.

EN_MUSART_transferControl_t copy_uddtTransferDirection

Transfer direction.

ST_MUSART_clockInit_t copy_uddtUartClockInit

USART clock initialization.

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/usart/**usart_config.h**

ST_MUSART_clockInit_t Struct Reference

Structure for USART clock initialization.
`#include <usart_config.h>`

Data Fields

- **uint8_t clockOutput**
- **uint8_t clockPolarity**
- **uint8_t clockPhase**
- **uint8_t lastBitClockPulse**

Detailed Description

Structure for USART clock initialization.

Field Documentation

uint8_t clockOutput

Clock output.

uint8_t clockPhase

Clock phase.

uint8_t clockPolarity

Clock polarity.

uint8_t lastBitClockPulse

Last bit clock pulse.

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

- D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/usart/**usart_config.h**

File Documentation

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/COMMON/bit_math.h File Reference

Macros

- #define **SET_BIT**(REG, BIT_NUMBER) (REG |= (1 << BIT_NUMBER))
- #define **GET_BIT**(REG, BIT_NUMBER) ((REG >> BIT_NUMBER) & 1)
- #define **CLR_BIT**(REG, BIT_NUMBER) (REG &= (~(1 << BIT_NUMBER)))
- #define **TOG_BIT**(REG, BIT_NUMBER) (REG ^= (1 << BIT_NUMBER))
- #define **SET_BITS**(REG, MSK) (REG) |= (MSK)
- #define **CLR_BITS**(REG, MSK) (REG) &= ~(MSK)
- #define **TOG_BITS**(REG, MSK) (REG) ^= (MSK)
- #define **SET_ALL_BITS**(REG) (REG) = (0xFFFFFFFF)
- #define **CLR_ALL_BITS**(REG) (REG) = (0x00000000)
- #define **TOG_ALL_BITS**(REG) (REG) ^= (0xFFFFFFFF)

Macro Definition Documentation

#define CLR_ALL_BITS(REG) (REG) = (0x00000000)

#define CLR_BIT(REG, BIT_NUMBER) (REG &= (~(1 << BIT_NUMBER)))

#define CLR_BITS(REG, MSK) (REG) &= ~(MSK)

#define GET_BIT(REG, BIT_NUMBER) ((REG >> BIT_NUMBER) & 1)

#define SET_ALL_BITS(REG) (REG) = (0xFFFFFFFF)

#define SET_BIT(REG, BIT_NUMBER) (REG |= (1 << BIT_NUMBER))

#define SET_BITS(REG, MSK) (REG) |= (MSK)

#define TOG_ALL_BITS(REG) (REG) ^= (0xFFFFFFFF)

#define TOG_BIT(REG, BIT_NUMBER) (REG ^= (1 << BIT_NUMBER))

#define TOG_BITS(REG, MSK) (REG) ^= (MSK)

bit_math.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC_Drivers
4 // File          : main.c
5 // Date          : Sep 8, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8
9 #ifndef COMMON_BIT_MATH_H_
10 #define COMMON_BIT_MATH_H_
11
12 #define SET_BIT(REG, BIT_NUMBER) (REG |= (1 << BIT_NUMBER))
13 #define GET_BIT(REG, BIT_NUMBER) ((REG >> BIT_NUMBER) & 1)
14 #define CLR_BIT(REG, BIT_NUMBER) (REG &= (~(1 << BIT_NUMBER)))
15 #define TOG_BIT(REG, BIT_NUMBER) (REG ^= (1 << BIT_NUMBER))
16
17
18  #define SET_BITS(REG,MSK)          (REG) |=  (MSK)
19  #define CLR_BITS(REG,MSK)          (REG) &= ~(MSK)
20  #define TOG_BITS(REG,MSK)          (REG) ^=  (MSK)
21
22  #define SET_ALL_BITS(REG)          (REG) =  (0xFFFFFFFF)
23  #define CLR_ALL_BITS(REG)          (REG) =  (0x00000000)
24  #define TOG_ALL_BITS(REG)          (REG) ^= (0xFFFFFFFF)
25
26 #endif /* COMMON_BIT_MATH_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/COMMON/std_types.h File Reference

Macros

- #define **TRUE** 1
- #define **FALSE** 0
- #define **STR_NULL** '\0'
- #define **PTR_NULL** (void*)0
- #define **NULL** (void*)0

Typedefs

- typedef unsigned char **u8**
- typedef unsigned short int **u16**
- typedef unsigned long int **u32**
- typedef unsigned char **uint8_t**
- typedef unsigned short int **uint16_t**
- typedef unsigned long int **uint32_t**
- typedef signed char **sint8_t**
- typedef signed short int **sint16_t**
- typedef signed long int **sint32_t**
- typedef float **float32_t**
- typedef double **float64_t**
- typedef long double **float96_t**
- typedef volatile unsigned char **vuint8_t**
- typedef volatile unsigned short int **vuint16_t**
- typedef volatile unsigned long int **vuint32_t**
- typedef volatile signed char **vsint8_t**
- typedef volatile signed short int **vsint16_t**
- typedef volatile signed long int **vsint32_t**
- typedef volatile float **vfloat32_t**
- typedef volatile double **vfloat64_t**
- typedef volatile long double **vfloat96_t**

Macro Definition Documentation

#define FALSE 0

#define NULL (void*)0

#define PTR_NULL (void*)0

#define STR_NULL '\0'

#define TRUE 1

Typedef Documentation

typedef float float32_t

typedef double float64_t

typedef long double float96_t

typedef signed short int sint16_t

typedef signed long int sint32_t

typedef signed char sint8_t

typedef unsigned short int u16

typedef unsigned long int u32

typedef unsigned char u8

typedef unsigned short int uint16_t

typedef unsigned long int uint32_t

typedef unsigned char uint8_t

typedef volatile float vfloat32_t

typedef volatile double vfloat64_t

typedef volatile long double vfloat96_t

typedef volatile signed short int vsint16_t

typedef volatile signed long int vsint32_t

typedef volatile signed char vsint8_t

typedef volatile unsigned short int vuint16_t

typedef volatile unsigned long int vuint32_t

typedef volatile unsigned char vuint8_t

std_types.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC_Drivers
4 // File          : main.c
5 // Date          : Sep 8, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8
9 #ifndef COMMON_STD_TYPES_H_
10 #define COMMON_STD_TYPES_H_
11
12 typedef unsigned char                       u8     ;
13 typedef unsigned short int                  u16    ;
14 typedef unsigned long  int                  u32    ;
15
16 typedef unsigned char                       uint8_t     ;
17 typedef unsigned short int                  uint16_t    ;
18 typedef unsigned long  int                  uint32_t    ;
19 typedef signed   char                       sint8_t     ;
20 typedef signed   short int                  sint16_t    ;
21 typedef signed   long  int                  sint32_t    ;
22 typedef float                              float32_t   ;
23 typedef double                             float64_t   ;
24 typedef long double                        float96_t   ;
25
26
27 typedef volatile unsigned char              vuint8_t    ;
28 typedef volatile unsigned short int         vuint16_t   ;
29 typedef volatile unsigned long  int         vuint32_t   ;
30 typedef volatile signed   char              vsint8_t    ;
31 typedef volatile signed   short int         vsint16_t   ;
32 typedef volatile signed   long  int         vsint32_t   ;
33 typedef volatile float                      vfloat32_t  ;
34 typedef volatile double                     vfloat64_t  ;
35 typedef volatile long double                vfloat96_t  ;
36
37
38 #ifndef TRUE
39 #define TRUE   1
40 #endif
41
42
43 #ifndef FALSE
44 #define FALSE   0
45 #endif
46
47
48 #ifndef STR_NULL
49 #define STR_NULL    '\0'
50 #endif
51
52 #ifndef PTR_NULL
53 #define PTR_NULL    (void*)0
54 #endif
55
56 #ifndef NULL
57 #define NULL        (void*)0
58 #endif
59
60 #endif /* COMMON_STD_TYPES_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/gpio/gpio_config.h File Reference

Data Structures

- struct **ST_MGPIO_pinCfg_t**struct **ST_MGPIO_altPinCfg_t**

Macros

- #define **MIN_VAL_OF_U16** 0
- #define **MAX_VAL_OF_U16** 65536

Enumerations

- enum **EN_MGPIO_systemState_t** { **MGPIO_NOK** = 0, **MGPIO_OK**, **MGPIO_INVALID_PARAMTER**, **MGPIO_PTR_NULL** }
- enum **EN_MGPIO_pinOptions_t** { **MGPIO_PIN0** = 0, **MGPIO_PIN1**, **MGPIO_PIN2**, **MGPIO_PIN3**, **MGPIO_PIN4**, **MGPIO_PIN5**, **MGPIO_PIN6**, **MGPIO_PIN7**, **MGPIO_PIN8**, **MGPIO_PIN9**, **MGPIO_PIN10**, **MGPIO_PIN11**, **MGPIO_PIN12**, **MGPIO_PIN13**, **MGPIO_PIN14**, **MGPIO_PIN15** }
- enum **EN_MGPIO_pinModeOptions_t** { **MGPIO_MODE_INPUT** = 0, **MGPIO_MODE_OUTPUT**, **MGPIO_MODE_ALTF**, **MGPIO_MODE_ANALOG** }
- enum **EN_MGPIO_pinLogicOptions_t** { **MGPIO_LOGIC_LOW** = 0, **MGPIO_LOGIC_HIGH** }
- enum **EN_MGPIO_outputSpeedOptions_t** { **MGPIO_OUTPUT_SPEED_LOW** = 0, **MGPIO_OUTPUT_SPEED_MEDIUM**, **MGPIO_OUTPUT_SPEED_HIGH**, **MGPIO_OUTPUT_SPEED_VERY_HIGH** }
- enum **EN_MGPIO_outputResistorOptions_t** { **MGPIO_OUTPUT_RESISTOR_PUSH_PULL** = 0, **MGPIO_OUTPUT_RESISTOR_OPEN_DRAIN** }
- enum **EN_MGPIO_pushPullOptions_t** { **MGPIO_PULL_FLOATING** = 0, **MGPIO_PULL_PULL_UP**, **MGPIO_PULL_PULL_DOWN** }
- enum **EN_MGPIO_altfnOptions_t** { **MGPIO_ALTFN_0** = 0, **MGPIO_ALTFN_1**, **MGPIO_ALTFN_2**, **MGPIO_ALTFN_3**, **MGPIO_ALTFN_4**, **MGPIO_ALTFN_5**, **MGPIO_ALTFN_6**, **MGPIO_ALTFN_7**, **MGPIO_ALTFN_8**, **MGPIO_ALTFN_9**, **MGPIO_ALTFN_10**, **MGPIO_ALTFN_11**, **MGPIO_ALTFN_12**, **MGPIO_ALTFN_13**, **MGPIO_ALTFN_14**, **MGPIO_ALTFN_15** }
- enum **EN_MGPIO_setResetOptions_t** { **MGPIO_PIN_RESET** = 0, **MGPIO_PIN_SET** }

Macro Definition Documentation

#define MAX_VAL_OF_U16 65536

#define MIN_VAL_OF_U16 0

Enumeration Type Documentation

enum EN_MGPIO_altfnOptions_t

Enumerator:

MGPIO_ALTFN_0	
MGPIO_ALTFN_	

1	
MGPIO_ALTFN_2	
MGPIO_ALTFN_3	
MGPIO_ALTFN_4	
MGPIO_ALTFN_5	
MGPIO_ALTFN_6	
MGPIO_ALTFN_7	
MGPIO_ALTFN_8	
MGPIO_ALTFN_9	
MGPIO_ALTFN_10	
MGPIO_ALTFN_11	
MGPIO_ALTFN_12	
MGPIO_ALTFN_13	
MGPIO_ALTFN_14	
MGPIO_ALTFN_15	

enum EN_MGPIO_outputResistorOptions_t

Enumerator:

MGPIO_OUTPUT_RESISTOR_PUSH_PULL	
MGPIO_OUTPUT_RESISTOR_OPEN_DRAIN	

enum EN_MGPIO_outputSpeedOptions_t

Enumerator:

MGPIO_OUTPUT_SPEED_LOW	
MGPIO_OUTPUT_SPEED_MEDIUM	
MGPIO_OUTPUT_SPEED_HIGH	
MGPIO_OUTPUT_SPEED_VERY_HIGH	

enum EN_MGPIO_pinLogicOptions_t

Enumerator:

MGPIO_LOGIC_LOW	
MGPIO_LOGIC_HIGH	

enum EN_MGPIO_pinModeOptions_t

Enumerator:

MGPIO_MODE_INPUT	
MGPIO_MODE_OUTPUT	
MGPIO_MODE_ALTF	
MGPIO_MODE_ANALOG	

enum EN_MGPIO_pinOptions_t

Enumerator:

MGPIO_PIN0	
MGPIO_PIN1	
MGPIO_PIN2	
MGPIO_PIN3	
MGPIO_PIN4	
MGPIO_PIN5	
MGPIO_PIN6	
MGPIO_PIN7	
MGPIO_PIN8	
MGPIO_PIN9	
MGPIO_PIN10	
MGPIO_PIN11	
MGPIO_PIN12	
MGPIO_PIN13	
MGPIO_PIN14	
MGPIO_PIN15	

enum EN_MGPIO_pushPullOptions_t

Enumerator:

MGPIO_PULL_FLOATING	
MGPIO_PULL_PULL_UP	
MGPIO_PULL_PULL_DOWN	

enum EN_MGPIO_setResetOptions_t

Enumerator:

MGPIO_PIN_RESET	
MGPIO_PIN_SET	

enum EN_MGPIO_systemState_t

Enumerator:

MGPIO_NOK	
MGPIO_OK	
MGPIO_INVALID_PARAMTER	
MGPIO_PTR_NULL	

gpio_config.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC
4 // File          : gpio_config.h
5 // Date          : Sep 10, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8 #ifndef MCAL_GPIO_GPIO_CONFIG_H_
9 #define MCAL_GPIO_GPIO_CONFIG_H_
10
11
12 #define MIN_VAL_OF_U16              0
13 #define MAX_VAL_OF_U16              65536
14
15 typedef enum
16 {
17     MGPIO_NOK = 0,
18     MGPIO_OK,
19     MGPIO_INVALID_PARAMTER,
20     MGPIO_PTR_NULL
21 }EN_MGPIO_systemState_t;
22
23
24 typedef enum
25 {
26     MGPIO_PIN0 = 0,
27     MGPIO_PIN1,
28     MGPIO_PIN2,
29     MGPIO_PIN3,
30     MGPIO_PIN4,
31     MGPIO_PIN5,
32     MGPIO_PIN6,
33     MGPIO_PIN7,
34     MGPIO_PIN8,
35     MGPIO_PIN9,
36     MGPIO_PIN10,
37     MGPIO_PIN11,
38     MGPIO_PIN12,
39     MGPIO_PIN13,
40     MGPIO_PIN14,
41     MGPIO_PIN15,
42
43 }EN_MGPIO_pinOptions_t;
44
45
46 typedef enum
47 {
48     MGPIO_MODE_INPUT = 0,
49     MGPIO_MODE_OUTPUT,
50     MGPIO_MODE_ALTF,
51     MGPIO_MODE_ANALOG
52
53 }EN_MGPIO_pinModeOptions_t;
54
55
56 typedef enum
57 {
58     MGPIO_LOGIC_LOW = 0,
59     MGPIO_LOGIC_HIGH
60 }EN_MGPIO_pinLogicOptions_t;
61
62
63 typedef enum
64 {
65     MGPIO_OUTPUT_SPEED_LOW = 0,
66     MGPIO_OUTPUT_SPEED_MEDIUM,
67     MGPIO_OUTPUT_SPEED_HIGH,
68     MGPIO_OUTPUT_SPEED_VERY_HIGH
69 }EN_MGPIO_outputSpeedOptions_t;
70
71 typedef enum
72 {
```

```
73     MGPIO_OUTPUT_RESISTOR_PUSH_PULL = 0,
74     MGPIO_OUTPUT_RESISTOR_OPEN_DRAIN
75 }EN_MGPIO_outputResistorOptions_t;
76
77
78 typedef enum
79 {
80     MGPIO_PULL_FLOATING = 0,
81     MGPIO_PULL_PULL_UP,
82     MGPIO_PULL_PULL_DOWN
83 }EN_MGPIO_pushPullOptions_t;
84
85
86 typedef enum
87 {
88      MGPIO_ALTFN_0 = 0,
89      MGPIO_ALTFN_1 ,
90      MGPIO_ALTFN_2 ,
91      MGPIO_ALTFN_3 ,
92      MGPIO_ALTFN_4 ,
93      MGPIO_ALTFN_5 ,
94      MGPIO_ALTFN_6 ,
95      MGPIO_ALTFN_7 ,
96      MGPIO_ALTFN_8 ,
97      MGPIO_ALTFN_9 ,
98      MGPIO_ALTFN_10,
99      MGPIO_ALTFN_11,
100      MGPIO_ALTFN_12,
101      MGPIO_ALTFN_13,
102      MGPIO_ALTFN_14,
103      MGPIO_ALTFN_15
104 }EN_MGPIO_altfnOptions_t;
105
106
107 typedef enum
108 {
109     MGPIO_PIN_RESET = 0,
110     MGPIO_PIN_SET
111 }EN_MGPIO_setResetOptions_t;
112
113
114 typedef struct
115 {
116     ST_MGPIOx_RegistersMap_t  *PS_GPIOx;
117     EN_MGPIO_pinOptions_t copy_uddtPinNum;
118     EN_MGPIO_pinModeOptions_t copy_uddtPinMode;
119     EN_MGPIO_outputResistorOptions_t copy_uddtOutputResistor;
120     EN_MGPIO_outputSpeedOptions_t copy_uddtOutputSpeed;
121     EN_MGPIO_pinLogicOptions_t copy_uddtPtrRetOfPinLogic;
122     EN_MGPIO_pushPullOptions_t copy_uddtPullState;
123 }ST_MGPIO_pinCfg_t;
124
125 typedef struct
126 {
127     ST_MGPIOx_RegistersMap_t  *PS_GPIOx;
128     EN_MGPIO_pinOptions_t copy_uddtPinNum;
129     EN_MGPIO_altfnOptions_t Copy_uddtAltFun;
130     EN_MGPIO_outputResistorOptions_t copy_uddtOutputResistor;
131     EN_MGPIO_outputSpeedOptions_t copy_uddtOutputSpeed;
132     EN_MGPIO_pushPullOptions_t copy_uddtPullState;
133 }ST MGPIO altPinCfg t;
134
135 #endif /* MCAL_GPIO_GPIO_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/gpio/gpio_interface.h File Reference

Header file for GPIO (General Purpose I/O) module.

```
#include "../../COMMON/bit_math.h"
#include "../../COMMON/std_types.h"
#include "gpio_private.h"
#include "gpio_config.h"
```

Functions

- **EN_MGPIO_systemState_t MGPIO_uddtSetPinMode** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **EN_MGPIO_pinOptions_t** copy_uddtPinNum, **EN_MGPIO_pinModeOptions_t** copy_uddtPinMode)
  Set the mode of a GPIO pin.

- **EN_MGPIO_systemState_t MGPIO_uddtSetOutputMode** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **EN_MGPIO_pinOptions_t** copy_uddtPinNum, **EN_MGPIO_outputResistorOptions_t** copy_uddtOutputResistor)
  Set the output mode of a GPIO pin.

- **EN_MGPIO_systemState_t MGPIO_uddtSetOutputSpeed** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **EN_MGPIO_pinOptions_t** copy_uddtPinNum, **EN_MGPIO_outputSpeedOptions_t** copy_uddtOutputSpeed)
  Set the output speed of a GPIO pin.

- **EN_MGPIO_systemState_t MGPIO_uddtSetPullState** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **EN_MGPIO_pinOptions_t** copy_uddtPinNum, **EN_MGPIO_pushPullOptions_t** copy_uddtPullState)
  Set the pull state of a GPIO pin.

- **EN_MGPIO_systemState_t MGPIO_uddtGetPinVal** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **EN_MGPIO_pinOptions_t** copy_uddtPinNum, **EN_MGPIO_pinLogicOptions_t** *copy_uddtPtrRetOfPinLogic)
  Get the logic level of a GPIO pin.

- **EN_MGPIO_systemState_t MGPIO_uddtSetPinVal** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **EN_MGPIO_pinOptions_t** copy_uddtPinNum, **EN_MGPIO_pinLogicOptions_t** copy_uddtPinLogic)
  Set the logic level of a GPIO pin.

- **EN_MGPIO_systemState_t MGPIO_uddtDirectSetReset** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **EN_MGPIO_pinOptions_t** copy_uddtPinNum, **EN_MGPIO_setResetOptions_t** copy_uddtSetResetState)
  Perform a direct set/reset operation on a GPIO pin.

- **EN_MGPIO_systemState_t MGPIO_uddtSetPortVal** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **uint16_t** copy_u16OutputVal)
  Set the value of an entire GPIO port.

- **EN_MGPIO_systemState_t MGPIO_uddtSetAltFun** (**ST_MGPIOx_RegistersMap_t** *PS_GPIOx, **EN_MGPIO_pinOptions_t** copy_uddtPinNum, **EN_MGPIO_altfnOptions_t** Copy_uddtAltFun)
  Set the alternate function of a GPIO pin.

- **EN_MGPIO_systemState_t MGPIO_uddtInitPin** (**ST_MGPIO_pinCfg_t** *PS_pinInstance)
  Initialize a GPIO pin based on a configuration structure.

- **EN_MGPIO_systemState_t MGPIO_uddtInitAltPin** (**ST_MGPIO_altPinCfg_t** *PS_altPinInstance)
  Initialize an alternate GPIO pin based on a configuration structure.

Detailed Description

Header file for GPIO (General Purpose I/O) module.

Function Documentation

EN_MGPIO_systemState_t MGPIO_uddtDirectSetReset (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, EN_MGPIO_pinOptions_t *copy_uddtPinNum*, EN_MGPIO_setResetOptions_t *copy_uddtSetResetState*)

Perform a direct set/reset operation on a GPIO pin.

This function performs a direct set/reset operation on a specified GPIO pin.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.
copy_uddtPinNum	The pin number to configure. Possible values are: • **MGPIO_PIN0** • **MGPIO_PIN1** • ... • **MGPIO_PIN15**
copy_uddtSetReset State	The set/reset option. Possible values are: • **MGPIO_PIN_RESET** • **MGPIO_PIN_SET**

Returns

The system state after the set/reset operation.

- **MGPIO_OK**: Set/reset operation successful.
- **MGPIO_NOK**: Set/reset operation failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtGetPinVal (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, EN_MGPIO_pinOptions_t *copy_uddtPinNum*, EN_MGPIO_pinLogicOptions_t * *copy_uddtPtrRetOfPinLogic*)

Get the logic level of a GPIO pin.

This function retrieves the logic level of a specified GPIO pin.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.
copy_uddtPinNum	The pin number to read. Possible values are: • **MGPIO_PIN0** • **MGPIO_PIN1** • ... • **MGPIO_PIN15**
copy_uddtPtrRetOfPinLogic	Pointer to store the retrieved logic level.

Returns

The system state after getting the pin logic level.

- **MGPIO_OK**: Pin logic level retrieval successful.
- **MGPIO_NOK**: Pin logic level retrieval failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtInitAltPin (ST_MGPIO_altPinCfg_t * *PS_altPinInstance*)

Initialize an alternate GPIO pin based on a configuration structure.

This function initializes an alternate GPIO pin based on the provided configuration structure.

Parameters

PS_altPinInstance	Pointer to the alternate GPIO pin configuration structure.

Returns

The system state after initializing the alternate GPIO pin.

- **MGPIO_OK**: Alternate GPIO pin initialization successful.
- **MGPIO_NOK**: Alternate GPIO pin initialization failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtInitPin (ST_MGPIO_pinCfg_t * *PS_pinInstance*)

Initialize a GPIO pin based on a configuration structure.

This function initializes a GPIO pin based on the provided configuration structure.

Parameters

PS_pinInstance	Pointer to the GPIO pin configuration structure.

Returns

The system state after initializing the GPIO pin.

- **MGPIO_OK**: GPIO pin initialization successful.
- **MGPIO_NOK**: GPIO pin initialization failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtSetAltFun (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, EN_MGPIO_pinOptions_t *copy_uddtPinNum*, EN_MGPIO_altfnOptions_t *Copy_uddtAltFun*)

Set the alternate function of a GPIO pin.

This function sets the alternate function of a specified GPIO pin.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.
copy_uddtPinNum	The pin number to configure. Possible values are: • **MGPIO_PIN0** • **MGPIO_PIN1** • ... • **MGPIO_PIN15**
Copy_uddtAltFun	The alternate function option. Possible values are: • **MGPIO_ALTFN_0** • **MGPIO_ALTFN_1** • ... • **MGPIO_ALTFN_15**

Returns

The system state after setting the alternate function.

- **MGPIO_OK**: Alternate function setting successful.
- **MGPIO_NOK**: Alternate function setting failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtSetOutputMode (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, EN_MGPIO_pinOptions_t *copy_uddtPinNum*, EN_MGPIO_outputResistorOptions_t *copy_uddtOutputResistor*)

Set the output mode of a GPIO pin.

This function sets the output mode of a specified GPIO pin.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.
copy_uddtPinNum	The pin number to configure. Possible values are: • **MGPIO_PIN0** • **MGPIO_PIN1** • ... • **MGPIO_PIN15**
copy_uddtOutputResistor	The output resistor option. Possible values are: • **MGPIO_OUTPUT_RESISTOR_PUSH_PULL** • **MGPIO_OUTPUT_RESISTOR_OPEN_DRAIN**

Returns

The system state after setting the pin output mode.

- **MGPIO_OK**: Pin output mode setting successful.
- **MGPIO_NOK**: Pin output mode setting failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtSetOutputSpeed (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, EN_MGPIO_pinOptions_t *copy_uddtPinNum*, EN_MGPIO_outputSpeedOptions_t *copy_uddtOutputSpeed*)

Set the output speed of a GPIO pin.

This function sets the output speed of a specified GPIO pin.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.
copy_uddtPinNum	The pin number to configure. Possible values are: • **MGPIO_PIN0** • **MGPIO_PIN1** • ... • **MGPIO_PIN15**
copy_uddtOutputSpeed	The output speed option. Possible values are: • **MGPIO_OUTPUT_SPEED_LOW** • **MGPIO_OUTPUT_SPEED_MEDIUM** • **MGPIO_OUTPUT_SPEED_HIGH** • **MGPIO_OUTPUT_SPEED_VERY_HIGH**

Returns

The system state after setting the pin output speed.

- **MGPIO_OK**: Pin output speed setting successful.
- **MGPIO_NOK**: Pin output speed setting failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtSetPinMode (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, EN_MGPIO_pinOptions_t *copy_uddtPinNum*, EN_MGPIO_pinModeOptions_t *copy_uddtPinMode*)

Set the mode of a GPIO pin.

This function sets the mode of a specified GPIO pin.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.
copy_uddtPinNum	The pin number to configure. Possible values are: • **MGPIO_PIN0** • **MGPIO_PIN1** • ... • **MGPIO_PIN15**
copy_uddtPinMode	The mode to set for the pin. Possible values are: • **MGPIO_MODE_INPUT** • **MGPIO_MODE_OUTPUT** • **MGPIO_MODE_ALTF** • **MGPIO_MODE_ANALOG**

Returns

The system state after setting the pin mode.

- **MGPIO_OK**: Pin mode setting successful.
- **MGPIO_NOK**: Pin mode setting failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.

- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtSetPinVal (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, EN_MGPIO_pinOptions_t *copy_uddtPinNum*, EN_MGPIO_pinLogicOptions_t *copy_uddtPinLogic*)

Set the logic level of a GPIO pin.

This function sets the logic level of a specified GPIO pin.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.
copy_uddtPinNum	The pin number to configure. Possible values are: • **MGPIO_PIN0** • **MGPIO_PIN1** • ... • **MGPIO_PIN15**
copy_uddtPinLogic	The logic level to set for the pin. Possible values are: • **MGPIO_LOGIC_LOW** • **MGPIO_LOGIC_HIGH**

Returns

The system state after setting the pin logic level.

- **MGPIO_OK**: Pin logic level setting successful.
- **MGPIO_NOK**: Pin logic level setting failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtSetPortVal (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, uint16_t *copy_u16OutputVal*)

Set the value of an entire GPIO port.

This function sets the value of an entire GPIO port.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.
copy_u16OutputVal	The value to set for the entire port.

Returns

The system state after setting the port value.

- **MGPIO_OK**: Port value setting successful.
- **MGPIO_NOK**: Port value setting failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

EN_MGPIO_systemState_t MGPIO_uddtSetPullState (ST_MGPIOx_RegistersMap_t * *PS_GPIOx*, EN_MGPIO_pinOptions_t *copy_uddtPinNum*, EN_MGPIO_pushPullOptions_t *copy_uddtPullState*)

Set the pull state of a GPIO pin.

This function sets the pull state of a specified GPIO pin.

Parameters

PS_GPIOx	Pointer to the GPIOx registers map.

copy_uddtPinNum	The pin number to configure. Possible values are: • **MGPIO_PIN0** • **MGPIO_PIN1** • ... • **MGPIO_PIN15**
copy_uddtPullState	The pull state option. Possible values are: • **MGPIO_PULL_FLOATING** • **MGPIO_PULL_PULL_UP** • **MGPIO_PULL_PULL_DOWN**

Returns

The system state after setting the pin pull state.

- **MGPIO_OK**: Pin pull state setting successful.
- **MGPIO_NOK**: Pin pull state setting failed.
- **MGPIO_INVALID_PARAMTER**: Invalid parameter detected during the operation.
- **MGPIO_PTR_NULL**: Null pointer encountered during the operation.

gpio_interface.h

Go to the documentation of this file.

```
1
6 #ifndef MCAL_GPIO_GPIO_INTERFACE_H_
7 #define MCAL_GPIO_GPIO_INTERFACE_H_
8
9 #include "../../COMMON/bit_math.h"
10 #include "../../COMMON/std_types.h"
11 #include "gpio_private.h"
12 #include "gpio_config.h"
13
39 EN_MGPIO_systemState_t MGPIO_uddtSetPinMode(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_pinModeOptions_t copy_uddtPinMode);
40
64 EN_MGPIO_systemState_t MGPIO_uddtSetOutputMode(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_outputResistorOptions_t
copy_uddtOutputResistor);
65
91 EN_MGPIO_systemState_t MGPIO_uddtSetOutputSpeed(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_outputSpeedOptions_t
copy_uddtOutputSpeed);
92
117 EN_MGPIO_systemState_t MGPIO_uddtSetPullState(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_pushPullOptions_t copy_uddtPullState);
118
139 EN_MGPIO_systemState_t MGPIO_uddtGetPinVal(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_pinLogicOptions_t
*copy_uddtPtrRetOfPinLogic);
140
164 EN_MGPIO_systemState_t MGPIO_uddtSetPinVal(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_pinLogicOptions_t copy_uddtPinLogic);
165
189 EN_MGPIO_systemState_t MGPIO_uddtDirectSetReset(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_setResetOptions_t
copy_uddtSetResetState);
190
205 EN_MGPIO_systemState_t MGPIO_uddtSetPortVal(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
uint16_t copy_u16OutputVal);
206
232 EN_MGPIO_systemState_t MGPIO_uddtSetAltFun(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_altfnOptions_t Copy_uddtAltFun);
233
247 EN_MGPIO_systemState_t MGPIO_uddtInitPin(ST_MGPIO_pinCfg_t *PS_pinInstance);
248
262 EN_MGPIO_systemState_t MGPIO_uddtInitAltPin(ST_MGPIO_altPinCfg_t
*PS_altPinInstance);
263
264
265 //EN_MGPIO_systemState_t MGPIO_PinLock      (u8 copy_u8PortName , u8 copy_u8PinNum );
266 //EN_MGPIO_systemState_t MGPIO_SetPortMode (u8 Copy_u8PortName , u8 Copy_u8Mode);
267
268
269 #endif /* MCAL_GPIO_GPIO_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/gpio/gpio_private.h File Reference

Data Structures

struct ST_MGPIOx_RegistersMap_t**Macros**

- #define **MGPIOA_PERIPHERAL_BASE_ADDR** (0x40020000)
- #define **MGPIOB_PERIPHERAL_BASE_ADDR** (0x40020400)
- #define **MGPIOC_PERIPHERAL_BASE_ADDR** (0x40020800)
- #define **MGPIOD_PERIPHERAL_BASE_ADDR** (0x40020C00)
- #define **MGPIOE_PERIPHERAL_BASE_ADDR** (0x40021000)
- #define **MGPIOH_PERIPHERAL_BASE_ADDR** (0x40021C00)
- #define **MGPIOA_PERIPHERAL** ((**ST_MGPIOx_RegistersMap_t** *)**MGPIOA_PERIPHERAL_BASE_ADDR**)
- #define **MGPIOB_PERIPHERAL** ((**ST_MGPIOx_RegistersMap_t** *)**MGPIOB_PERIPHERAL_BASE_ADDR**)

Macro Definition Documentation

#define MGPIOA_PERIPHERAL ((ST_MGPIOx_RegistersMap_t *)MGPIOA_PERIPHERAL_BASE_ADDR)

#define MGPIOA_PERIPHERAL_BASE_ADDR (0x40020000)

#define MGPIOB_PERIPHERAL ((ST_MGPIOx_RegistersMap_t *)MGPIOB_PERIPHERAL_BASE_ADDR)

#define MGPIOB_PERIPHERAL_BASE_ADDR (0x40020400)

#define MGPIOC_PERIPHERAL_BASE_ADDR (0x40020800)

#define MGPIOD_PERIPHERAL_BASE_ADDR (0x40020C00)

#define MGPIOE_PERIPHERAL_BASE_ADDR (0x40021000)

#define MGPIOH_PERIPHERAL_BASE_ADDR (0x40021C00)

gpio_private.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC
4 // File          : gpio_private.h
5 // Date          : Sep 10, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8 #ifndef MCAL_GPIO_GPIO_PRIVATE_H_
9 #define MCAL_GPIO_GPIO_PRIVATE_H_
10
11
12 #define  MGPIOA_PERIPHERAL_BASE_ADDR        (0x40020000)
13 #define  MGPIOB_PERIPHERAL_BASE_ADDR        (0x40020400)
14 #define  MGPIOC_PERIPHERAL_BASE_ADDR        (0x40020800)
15 #define  MGPIOD_PERIPHERAL_BASE_ADDR        (0x40020C00)
16 #define  MGPIOE_PERIPHERAL_BASE_ADDR        (0x40021000)
17 #define  MGPIOH_PERIPHERAL_BASE_ADDR        (0x40021C00)
18
19
20 typedef struct
21 {
22
23     vuint32_t   MGPIOx_MODER;
24     vuint32_t   MGPIOx_OTYPER;
25     vuint32_t   MGPIOx_OSPEEDR;
26     vuint32_t   MGPIOx_PUPDR;
27     vuint32_t   MGPIOx_IDR;
28     vuint32_t   MGPIOx_ODR;
29     vuint32_t   MGPIOx_BSRR;
30     vuint32_t   MGPIOx_LCKR;
31     vuint32_t   MGPIOx_AFRL;
32     vuint32_t   MGPIOx_AFRH;
33
34
35 }ST_MGPIOx_RegistersMap_t;
36
37
38
39
40 #define MGPIOA_PERIPHERAL (( ST_MGPIOx_RegistersMap_t *)MGPIOA_PERIPHERAL_BASE_ADDR)
41 #define MGPIOB_PERIPHERAL (( ST_MGPIOx_RegistersMap_t *)MGPIOB_PERIPHERAL_BASE_ADDR)
42
43
44
45
46 #endif /* MCAL_GPIO_GPIO_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/mfdi/MFDI_config.h File Reference

Macros

- #define **SIZE_OF_WRITE_WORD HALF_WORD**

Macro Definition Documentation

#define SIZE_OF_WRITE_WORD HALF_WORD

MFDI_config.h

Go to the documentation of this file.

```
1 /******************************************************************
2 * File Name     :   MFDI_config.h                                 *
3 * Author        :   Mahmoud Gamal                                 *
4 * Version       :   1.0.0                                         *
5 * Date          :   28/09/2023                                    *
6 * Description   :                                                 *
7 ******************************************************************/
8 #ifndef MFDI_CONFIG_H_
9 #define MFDI_CONFIG_H_
10
11
12
13
14 /* Select writing size :
15  *          HALF_WORD
16  *          ONE_WORD
17  *          DOUBLE_WORD
18  */
19 #define SIZE_OF_WRITE_WORD      HALF_WORD
20
21
22
23 #endif
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/mfdi/MFDI_interface.h File Reference

Header file for FDI (Flash Driver Interface) module.

```
#include "../../COMMON/bit_math.h"
#include "../../COMMON/std_types.h"
#include "MFDI_private.h"
#include "MFDI_config.h"
```

Macros

- #define **FLASH_START_ADDRESS** 0x08000000
- #define **FLASH_APP_ADDRESS** 0x08004000

Functions

- void **MFDI_voidEraseSector** (**u8** copy_u8SectorNumber)
  Erases a specified sector in the flash memory.

- void **MFDI_voidFlashWrite** (**u32** copy_u32Address, **u16** *copy_u16PtrData, **u8** copy_u8DataLength)
  Writes data to the flash memory at the specified address.

- void **MFDI_voidEraseAppArea** (**u8** Copy_u8AppId)
  Erases a specific application area in the flash memory.

Detailed Description

Header file for FDI (Flash Driver Interface) module.

Macro Definition Documentation

#define FLASH_APP_ADDRESS 0x08004000

#define FLASH_START_ADDRESS 0x08000000

Function Documentation

void MFDI_voidEraseAppArea (u8 *Copy_u8AppId*)

Erases a specific application area in the flash memory.

This function erases the specified application area in the flash memory.

Parameters

Copy_u8AppId	The application ID or identifier for the application area to be erased.

void MFDI_voidEraseSector (u8 *copy_u8SectorNumber*)

Erases a specified sector in the flash memory.

This function erases the specified sector in the flash memory.

Parameters

copy_u8SectorNumber	The sector number to be erased.

void MFDI_voidFlashWrite (u32 *copy_u32Address*, u16 * *copy_u16PtrData*, u8 *copy_u8DataLength*)

Writes data to the flash memory at the specified address.

This function writes data to the flash memory at the specified address.

Parameters

copy_u32Address	The address in the flash memory where the data should be written.
copy_u16PtrData	A pointer to an array of unsigned 16-bit integers representing the data to be written.
copy_u8DataLength	The length of the data to be written.

MFDI_interface.h

Go to the documentation of this file.

```
1
6 #ifndef MFDI_INTERFACE_H_
7 #define MFDI_INTERFACE_H_
8
9 #include "../../COMMON/bit_math.h"
10 #include "../../COMMON/std_types.h"
11 #include "MFDI_private.h"
12 #include "MFDI_config.h"
13
14
15 #define FLASH_START_ADDRESS   0x08000000
16 #define FLASH_APP_ADDRESS     0x08004000
17
25 void MFDI_voidEraseSector(u8 copy_u8SectorNumber);
26
36 void MFDI_voidFlashWrite(u32 copy_u32Address, u16 *copy_u16PtrData, u8
copy_u8DataLength);
37
45 void MFDI_voidEraseAppArea(u8 Copy_u8AppId);
46 #endif
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/mfdi/MFDI_private.h File Reference

Data Structures

struct FDI_tMacros

- #define **FDI_BASE_ADDRESS** (0x40023C00)
- #define **MFDI** ((volatile **FDI_t** *)**FDI_BASE_ADDRESS**)
- #define **CR_LOCK** 31
- #define **CR_ERRIE** 25
- #define **CR_EOPIE** 24
- #define **CR_STRT** 16
- #define **CR_PSIZE** 8
- #define **CR_SNB** 3
- #define **CR_MER** 2
- #define **CR_SER** 1
- #define **CR_PG** 0
- #define **SR_EOP** 0
- #define **SR_BSY** 16
- #define **OPTKEY1** (0x45670123)
- #define **OPTKEY2** (0xCDEF89AB)
- #define **HALF_WORD** 1
- #define **ONE_WORD** 2
- #define **DOUBLE_WORD** 4

Macro Definition Documentation

#define CR_EOPIE 24

#define CR_ERRIE 25

#define CR_LOCK 31

#define CR_MER 2

#define CR_PG 0

#define CR_PSIZE 8

#define CR_SER 1

#define CR_SNB 3

#define CR_STRT 16

#define DOUBLE_WORD 4

#define FDI_BASE_ADDRESS (0x40023C00)

#define HALF_WORD 1

#define MFDI ((volatile FDI_t *)FDI_BASE_ADDRESS)

#define ONE_WORD 2

#define OPTKEY1 (0x45670123)

#define OPTKEY2 (0xCDEF89AB)

#define SR_BSY 16

#define SR_EOP 0

MFDI_private.h

Go to the documentation of this file.

```
1 /******************************************************************
2 * File Name     :   MFDI_private.h                                *
3 * Author        :   Mahmoud Gamal                                 *
4 * Version       :   1.0.0                                         *
5 * Date          :   28/09/2023                                    *
6 * Description   :                                                 *
7 ******************************************************************/
8 #ifndef MFDI_PRIVATE_H_
9 #define MFDI_PRIVATE_H_
10
11 #define FDI_BASE_ADDRESS        (0x40023C00)
12
13 typedef struct
14 {
15
16     u32 ACR;
17     u32 KEYR;
18     u32 OPTKEYR;
19     u32 SR;
20     u32 CR;
21     u32 OPTCR;
22
23 }FDI_t;
24
25 #define MFDI ((volatile FDI_t *)FDI_BASE_ADDRESS)
26
27 /* FLASH_CR Pins Difinitions */
28 #define CR_LOCK            31
29 #define CR_ERRIE           25
30 #define CR_EOPIE           24
31 #define CR_STRT            16
32 #define CR_PSIZE           8
33 #define CR_SNB             3
34 #define CR_MER             2
35 #define CR_SER             1
36 #define CR_PG              0
37
38 #define SR_EOP             0
39 #define SR_BSY             16
40
41
42 #define OPTKEY1  (0x45670123)
43 #define OPTKEY2  (0xCDEF89AB)
44
45
46 #define HALF_WORD       1
47 #define ONE_WORD        2
48 #define DOUBLE_WORD     4
49
50
51 #endif /* MCAL_FDI_FDI_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/rcc/rcc_config.h File Reference

Macros

- #define **sysClkSelect** **MRCC_SYS_CLK_HSI**
- #define **pllStatus** **MRCC_PLL_DISABLE**
- #define **pllSourceOfEntryClk** **MRCC_PLL_ENTRY_CLK_HSI**
- #define **pllpDivisionFactor** **MRCC_PLL_DIVISION_FACTOR_2**
- #define **pllmDivisionFactor** 0
- #define **pllnMulFactor** 0
- #define **apbHighSpeedPrescaler** **MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED**
- #define **apbLowSpeedPrescaler** **MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED**
- #define **ahbPrescaler** **MRCC_AHB_PRESCALER_2**

Enumerations

- enum **EN_MRCC_systemState_t** { **MRCC_OK** = 0, **MRCC_NOK**, **MRCC_PTR_NULL**, **MRCC_INVALID_PARAMTER** }
- enum **EN_MRCC_sysClkSelect_t** { **MRCC_SYS_CLK_HSI** = 0, **MRCC_SYS_CLK_HSE_BYPASS**, **MRCC_SYS_CLK_HSE_NOT_BYPASS** = 1, **MRCC_SYS_CLK_PLL**, **MRCC_SYS_CLK_NOT_ALLOWED** }
  Enumeration for the system clock selection.
- enum **EN_MRCC_pllClkSourceEntry_t** { **MRCC_PLL_ENTRY_CLK_HSI** = 0, **MRCC_PLL_ENTRY_CLK_HSE** }
  Enumeration for the PLL entry clock source.
- enum **EN_MRCC_pllDivisionFactor_t** { **MRCC_PLL_DIVISION_FACTOR_2** = 0, **MRCC_PLL_DIVISION_FACTOR_4**, **MRCC_PLL_DIVISION_FACTOR_6**, **MRCC_PLL_DIVISION_FACTOR_8** }
  Enumeration for the PLL division factor.
- enum **EN_MRCC_apbPrescalerSpeed_t** { **MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED** = 0, **MRCC_APB_PRESCALER_SPEED_2** = 4, **MRCC_APB_PRESCALER_SPEED_4**, **MRCC_APB_PRESCALER_SPEED_8**, **MRCC_APB_PRESCALER_SPEED_16** }
  Enumeration for APB (Advanced Peripheral Bus) prescaler speed.
- enum **EN_MRCC_ahbPrescaler_t** { **MRCC_AHB_PRESCALER_NOT_DIVIDED** = 0, **MRCC_AHB_PRESCALER_2** = 8, **MRCC_AHB_PRESCALER_4**, **MRCC_AHB_PRESCALER_8**, **MRCC_AHB_PRESCALER_16**, **MRCC_AHB_PRESCALER_64**, **MRCC_AHB_PRESCALER_128**, **MRCC_AHB_PRESCALER_256**, **MRCC_AHB_PRESCALER_512** }
  Enumeration for AHB (Advanced High-Performance Bus) prescaler.
- enum **EN_MRCC_pllStatus_t** { **MRCC_PLL_DISABLE** = 0, **MRCC_PLL_ENABLE** }
  Enumeration for PLL status.
- enum **EN_MRCC_busOptions_t** { **MRCC_AHP1_BUS** = 0, **MRCC_AHP2_BUS**, **MRCC_APB1_BUS**, **MRCC_APB2_BUS** }
  Enumeration for different buses in MRCC.
- enum **EN_MRCC_peripheralOptions_t** { **MRCC_GPIOA_PERIPHERAL** = 0, **MRCC_GPIOB_PERIPHERAL**, **MRCC_CRC_PERIPHERAL** = 12, **MRCC_DMA1_PERIPHERAL** = 21, **MRCC_DMA2_PERIPHERAL**, **MRCC_OTGFS_PERIPHERAL** = 7, **MRCC_TIM2_PERIPHERAL** = 0, **MRCC_TIM3_PERIPHERAL**, **MRCC_TIM4_PERIPHERAL**, **MRCC_TIM5_PERIPHERAL**, **MRCC_WWDG_PERIPHERAL** = 11, **MRCC_SPI2_PERIPHERAL** = 14, **MRCC_SPI3_PERIPHERAL** = 15, **MRCC_USART2_PERIPHERAL** = 17, **MRCC_I2C1_PERIPHERAL** = 21, **MRCC_I2C2_PERIPHERAL**, **MRCC_I2C3_PERIPHERAL**, **MRCC_PWR_PERIPHERAL**

= 28, **MRCC_TIM1_PERIPHERAL** = 0, **MRCC_USART1_PERIPHERAL** = 4, **MRCC_USART6_PERIPHERAL**, **MRCC_ADC1_PERIPHERAL** = 8, **MRCC_SDIO_PERIPHERAL** = 11, **MRCC_SPI1_PERIPHERAL**, **MRCC_SPI4_PERIPHERAL**, **MRCC_SYSCFG_PERIPHERAL**, **MRCC_TIM9_PERIPHERAL** = 16, **MRCC_TIM10_PERIPHERAL**, **MRCC_TIM11_PERIPHERAL** }

Enumeration for different peripheral options in MRCC.

Macro Definition Documentation

#define ahbPrescaler MRCC_AHB_PRESCALER_2

#define apbHighSpeedPrescaler MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED

#define apbLowSpeedPrescaler MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED

#define pllmDivisionFactor 0

#define pllnMulFactor 0

#define pllpDivisionFactor MRCC_PLL_DIVISION_FACTOR_2

#define pllSourceOfEntryClk MRCC_PLL_ENTRY_CLK_HSI

#define pllStatus MRCC_PLL_DISABLE

#define sysClkSelect MRCC_SYS_CLK_HSI

Enumeration Type Documentation

enum EN_MRCC_ahbPrescaler_t

Enumeration for AHB (Advanced High-Performance Bus) prescaler.

Enumerator:

MRCC_AHB_PRESCALER_NOT_DIVIDED	AHB prescaler: Not divided.
MRCC_AHB_PRESCALER_2	AHB prescaler: 2.
MRCC_AHB_PRESCALER_4	AHB prescaler: 4.
MRCC_AHB_PRESCALER_8	AHB prescaler: 8.
MRCC_AHB_PRESCALER_16	AHB prescaler: 16.
MRCC_AHB_PRESCALER_64	AHB prescaler: 64.

MRCC_AHB_PRESCALER_128	AHB prescaler: 128.
MRCC_AHB_PRESCALER_256	AHB prescaler: 256.
MRCC_AHB_PRESCALER_512	AHB prescaler: 512.

enum EN_MRCC_apbPrescalerSpeed_t

Enumeration for APB (Advanced Peripheral Bus) prescaler speed.

Enumerator:

MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED	APB prescaler speed: Not divided.
MRCC_APB_PRESCALER_SPEED_2	APB prescaler speed: 2.
MRCC_APB_PRESCALER_SPEED_4	APB prescaler speed: 4.
MRCC_APB_PRESCALER_SPEED_8	APB prescaler speed: 8.
MRCC_APB_PRESCALER_SPEED_16	APB prescaler speed: 16.

enum EN_MRCC_busOptions_t

Enumeration for different buses in MRCC.

Enumerator:

MRCC_AHP1_BUS	AHP1 bus.
MRCC_AHP2_BUS	AHP2 bus.
MRCC_APB1_BUS	APB1 bus.
MRCC_APB2_BUS	APB2 bus.

enum EN_MRCC_peripheralOptions_t

Enumeration for different peripheral options in MRCC.

Enumerator:

MRCC_GPIOA_PERIPHERAL	
MRCC_GPIOB_P	

ERIPHERAL	
MRCC_CRC_PERIPHERAL	
MRCC_DMA1_PERIPHERAL	
MRCC_DMA2_PERIPHERAL	
MRCC_OTGFS_PERIPHERAL	
MRCC_TIM2_PERIPHERAL	
MRCC_TIM3_PERIPHERAL	
MRCC_TIM4_PERIPHERAL	
MRCC_TIM5_PERIPHERAL	
MRCC_WWDG_PERIPHERAL	
MRCC_SPI2_PERIPHERAL	
MRCC_SPI3_PERIPHERAL	
MRCC_USART2_PERIPHERAL	
MRCC_I2C1_PERIPHERAL	
MRCC_I2C2_PERIPHERAL	
MRCC_I2C3_PERIPHERAL	
MRCC_PWR_PERIPHERAL	
MRCC_TIM1_PERIPHERAL	
MRCC_USART1_PERIPHERAL	
MRCC_USART6_PERIPHERAL	
MRCC_ADC1_PERIPHERAL	
MRCC_SDIO_PERIPHERAL	
MRCC_SPI1_PERIPHERAL	
MRCC_SPI4_PERIPHERAL	
MRCC_SYSCFG_PERIPHERAL	
MRCC_TIM9_PERIPHERAL	
MRCC_TIM10_PERIPHERAL	
MRCC_TIM11_PERIPHERAL	

enum EN_MRCC_pllClkSourceEntry_t

Enumeration for the PLL entry clock source.

Enumerator:

MRCC_PLL_ENTRY_CLK_HSI	PLL entry clock source: HSI.
MRCC_PLL_ENTRY_CLK_HSE	PLL entry clock source: HSE.

enum EN_MRCC_pllDivisionFactor_t

Enumeration for the PLL division factor.

Enumerator:

MRCC_PLL_DIVISION_FACTOR_2	PLL division factor: 2.
MRCC_PLL_DIVISION_FACTOR_4	PLL division factor: 4.
MRCC_PLL_DIVISION_FACTOR_6	PLL division factor: 6.
MRCC_PLL_DIVISION_FACTOR_8	PLL division factor: 8.

enum EN_MRCC_pllStatus_t

Enumeration for PLL status.

Enumerator:

MRCC_PLL_DISABLE	PLL is disabled.
MRCC_PLL_ENABLE	PLL is enabled.

enum EN_MRCC_sysClkSelect_t

Enumeration for the system clock selection.

Enumerator:

MRCC_SYS_CLK_HSI	HSI (High-Speed Internal) oscillator.
MRCC_SYS_CLK_HSE_BYPASS	HSE (High-Speed External) oscillator with bypass.
MRCC_SYS_CLK_HSE_NOT_BYPASS	HSE oscillator without bypass.
MRCC_SYS_CLK	PLL (Phase-Locked Loop).

_PLL	
MRCC_SYS_CLK_NOT_ALLOWED	Not allowed system clock source.

enum EN_MRCC_systemState_t

Enumerator:

MRCC_OK	Operation successful.
MRCC_NOK	Operation failed.
MRCC_PTR_NULL	Null pointer encountered.
MRCC_INVALID_PARAMTER	Invalid parameter detected.

rcc_config.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC_Drivers
4 // File          : rcc_config.h
5 // Date          : Sep 8, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8 #ifndef MCAL_RCC_RCC_CONFIG_H_
9 #define MCAL_RCC_RCC_CONFIG_H_
10
11 #define sysClkSelect                MRCC_SYS_CLK_HSI
12 #define pllStatus                   MRCC_PLL_DISABLE
13 #define pllSourceOfEntryClk         MRCC_PLL_ENTRY_CLK_HSI
14 #define pllpDivisionFactor          MRCC_PLL_DIVISION_FACTOR_2
15 #define pllmDivisionFactor          0
16 #define pllnMulFactor               0
17 #define apbHighSpeedPrescaler       MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED
18 #define apbLowSpeedPrescaler        MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED
19 #define ahbPrescaler                MRCC_AHB_PRESCALER_2
20
21 typedef enum
22 {
23     MRCC_OK = 0,
24     MRCC_NOK,
25     MRCC_PTR_NULL,
26     MRCC_INVALID_PARAMTER
27 } EN_MRCC_systemState_t;
28
32 typedef enum
33 {
34     MRCC_SYS_CLK_HSI = 0,
35     MRCC_SYS_CLK_HSE_BYPASS,
36     MRCC_SYS_CLK_HSE_NOT_BYPASS = 1,
37     MRCC SYS CLK PLL,
38     MRCC_SYS_CLK_NOT_ALLOWED
39 } EN_MRCC_sysClkSelect_t;
40
44 typedef enum
45 {
46     MRCC PLL ENTRY CLK HSI = 0,
47     MRCC_PLL_ENTRY_CLK_HSE
48 } EN_MRCC_pllClkSourceEntry_t;
49
53 typedef enum
54 {
55     MRCC_PLL_DIVISION_FACTOR_2 = 0,
56     MRCC_PLL_DIVISION_FACTOR_4,
57     MRCC_PLL_DIVISION_FACTOR_6,
58     MRCC_PLL_DIVISION_FACTOR_8
59 } EN_MRCC_pllDivisionFactor_t;
60
64 typedef enum
65 {
66     MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED = 0,
67     MRCC_APB_PRESCALER_SPEED_2 = 4,
68     MRCC_APB_PRESCALER_SPEED_4,
69     MRCC APB PRESCALER SPEED 8,
70     MRCC_APB_PRESCALER_SPEED_16
71 } EN_MRCC_apbPrescalerSpeed_t;
72
76 typedef enum
77 {
78     MRCC AHB PRESCALER NOT DIVIDED = 0,
79     MRCC_AHB_PRESCALER_2 = 8,
80     MRCC_AHB_PRESCALER_4,
81     MRCC_AHB_PRESCALER_8,
82     MRCC_AHB_PRESCALER_16,
83     MRCC AHB PRESCALER 64,
84     MRCC_AHB_PRESCALER_128,
85     MRCC_AHB_PRESCALER_256,
86     MRCC_AHB_PRESCALER_512
87 } EN_MRCC_ahbPrescaler_t;
```

```
88
92 typedef enum
93 {
94     MRCC_PLL_DISABLE = 0,
95     MRCC_PLL_ENABLE
96 } EN_MRCC_pllStatus_t;
97
101 typedef enum
102 {
103     MRCC_AHP1_BUS = 0,
104     MRCC_AHP2_BUS,
105     MRCC_APB1_BUS,
106     MRCC_APB2_BUS
107 } EN_MRCC_busOptions_t;
108
112 typedef enum
113 {
114     MRCC_GPIOA_PERIPHERAL = 0,
115     MRCC_GPIOB_PERIPHERAL,
116     MRCC_CRC_PERIPHERAL = 12,
117     MRCC_DMA1_PERIPHERAL = 21,
118     MRCC_DMA2_PERIPHERAL,
119     MRCC_OTGFS_PERIPHERAL = 7,
120     MRCC_TIM2_PERIPHERAL = 0,
121     MRCC_TIM3_PERIPHERAL,
122     MRCC_TIM4_PERIPHERAL,
123     MRCC_TIM5_PERIPHERAL,
124     MRCC_WWDG_PERIPHERAL = 11,
125     MRCC_SPI2_PERIPHERAL = 14,
126     MRCC_SPI3_PERIPHERAL = 15,
127     MRCC_USART2_PERIPHERAL = 17,
128     MRCC_I2C1_PERIPHERAL = 21,
129     MRCC_I2C2_PERIPHERAL,
130     MRCC_I2C3_PERIPHERAL,
131     MRCC_PWR_PERIPHERAL = 28,
132     MRCC_TIM1_PERIPHERAL = 0,
133     MRCC_USART1_PERIPHERAL = 4,
134     MRCC_USART6_PERIPHERAL,
135     MRCC_ADC1_PERIPHERAL = 8,
136     MRCC_SDIO_PERIPHERAL = 11,
137     MRCC_SPI1_PERIPHERAL,
138     MRCC_SPI4_PERIPHERAL,
139     MRCC_SYSCFG_PERIPHERAL,
140     MRCC_TIM9_PERIPHERAL = 16,
141     MRCC_TIM10_PERIPHERAL,
142     MRCC_TIM11_PERIPHERAL
143
144 }EN_MRCC_peripheralOptions_t;
145
146
147 #endif /* MCAL_RCC_RCC_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/rcc/rcc_interface.h File Reference

Header file for RCC (Reset and Clock Control) module.

```
#include "../../COMMON/bit_math.h"
#include "../../COMMON/std_types.h"
#include "rcc_private.h"
#include "rcc_config.h"
```

Functions

- **EN_MRCC_systemState_t MRCC_Init** (void)
  Initialize the MRCC (Reset and Clock Control) module.

- **EN_MRCC_systemState_t MRCC_enablePeripheral** (**EN_MRCC_busOptions_t** busSelection, **EN_MRCC_peripheralOptions_t** PeripheralNumber)
  Enable a specific peripheral on a selected bus.

- **EN_MRCC_systemState_t MRCC_disablePeripheral** (**EN_MRCC_busOptions_t** busSelection, **EN_MRCC_peripheralOptions_t** PeripheralNumber)
  Disable a specific peripheral on a selected bus.

- void **HAL_DeInit** (void)

Detailed Description

Header file for RCC (Reset and Clock Control) module.

Function Documentation

void HAL_DeInit (void)

EN_MRCC_systemState_t MRCC_disablePeripheral (EN_MRCC_busOptions_t *busSelection*, EN_MRCC_peripheralOptions_t *PeripheralNumber*)

Disable a specific peripheral on a selected bus.

This function disables a peripheral on the specified bus.

Parameters

busSelection	The bus on which the peripheral is located. Possible values are: • #EN_MRCC_AHP1_BUS • #EN_MRCC_AHP2_BUS • #EN_MRCC_APB1_BUS • #EN_MRCC_APB2_BUS
PeripheralNumber	The specific peripheral to disable. Refer to the enumeration **EN_MRCC_peripheralOptions_t** for available options.

Returns

The state of peripheral disabling. Possible values are:

- #EN_MRCC_OK: Peripheral disabling successful.
- #EN_MRCC_NOK: Peripheral disabling failed.
- #EN_MRCC_PTR_NULL: Null pointer encountered during the operation.
- #EN_MRCC_INVALID_PARAMTER: Invalid parameter detected during the operation.

EN_MRCC_systemState_t MRCC_enablePeripheral (EN_MRCC_busOptions_t *busSelection*, EN_MRCC_peripheralOptions_t *PeripheralNumber*)

Enable a specific peripheral on a selected bus.

This function enables a peripheral on the specified bus.

Parameters

busSelection	The bus on which the peripheral is located. Possible values are: • #EN_MRCC_AHP1_BUS • #EN_MRCC_AHP2_BUS • #EN_MRCC_APB1_BUS • #EN_MRCC_APB2_BUS
PeripheralNumber	The specific peripheral to enable. Refer to the enumeration **EN_MRCC_peripheralOptions_t** for available options.

Returns

The state of peripheral enabling. Possible values are:

- #EN_MRCC_OK: Peripheral enabling successful.
- #EN_MRCC_NOK: Peripheral enabling failed.
- #EN_MRCC_PTR_NULL: Null pointer encountered during the operation.
- #EN_MRCC_INVALID_PARAMTER: Invalid parameter detected during the operation.

EN_MRCC_systemState_t MRCC_Init (void)

Initialize the MRCC (Reset and Clock Control) module.

This function initializes the MRCC module, configuring the system clocks and other essential settings.

Returns

The system initialization state. Possible values are:

- #EN_MRCC_OK: Initialization successful.
- #EN_MRCC_NOK: Initialization failed.
- #EN_MRCC_PTR_NULL: Null pointer encountered during initialization.
- #EN_MRCC_INVALID_PARAMTER: Invalid parameter detected during initialization.

rcc_interface.h

Go to the documentation of this file.

```
1
6 #ifndef MCAL_RCC_RCC_INTERFACE_H_
7 #define MCAL_RCC_RCC_INTERFACE_H_
8
9
10 #include "../../COMMON/bit_math.h"
11 #include "../../COMMON/std_types.h"
12 #include "rcc_private.h"
13 #include "rcc_config.h"
14
28 EN_MRCC_systemState_t MRCC_Init(void);
29
51 EN_MRCC_systemState_t MRCC_enablePeripheral(EN_MRCC_busOptions_t busSelection,
EN_MRCC_peripheralOptions_t PeripheralNumber);
52
74 EN_MRCC_systemState_t MRCC_disablePeripheral(EN_MRCC_busOptions_t busSelection,
EN_MRCC_peripheralOptions_t PeripheralNumber);
75
76 void HAL_DeInit(void);
77
78 #endif /* MCAL_RCC_RCC_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/rcc/rcc_private.h File Reference

Data Structures

struct ST_MRCC_RegistersMap_t**Macros**

- #define **RCC_PERIPHERAL_BASE_ADDR** (0x40023800)
- #define **MRCC_PERIPHERAL** ((volatile **ST_MRCC_RegistersMap_t** *)**RCC_PERIPHERAL_BASE_ADDR**)
- #define **HSION_BIT** 0
- #define **HSIRDY_BIT** 1
- #define **HSEON_BIT** 16
- #define **HSERDY_BIT** 17
- #define **HSEBYP_BIT** 18
- #define **CSSON_BIT** 19
- #define **PLLON_BIT** 24
- #define **PLLRDY_BIT** 25
- #define **PLLI2SON_BIT** 26
- #define **PLLI2SRDY_BIT** 27
- #define **PLLM0_BIT** 0
- #define **PLLM1_BIT** 1
- #define **PLLM2_BIT** 2
- #define **PLLM3_BIT** 3
- #define **PLLM4_BIT** 4
- #define **PLLM5_BIT** 5
- #define **PLLN0_BIT** 6
- #define **PLLP0_BIT** 16
- #define **PLLSRC_BIT** 22
- #define **SW0_BIT** 0
- #define **SW1_BIT** 1
- #define **SWS0_BIT** 2
- #define **SWS1_BIT** 3
- #define **HPRE0_BIT** 4
- #define **PPRE10_BIT** 10
- #define **PPRE20_BIT** 13
- #define **__HAL_RCC_APB1_FORCE_RESET**() (**MRCC_PERIPHERAL**->RCC_APB1RSTR_REG = 0xFFFFFFFFU)
- #define **__HAL_RCC_APB1_RELEASE_RESET**() (**MRCC_PERIPHERAL**->RCC_APB1RSTR_REG = 0x00U)
- #define **__HAL_RCC_APB2_FORCE_RESET**() (**MRCC_PERIPHERAL**->RCC_APB2RSTR_REG = 0xFFFFFFFFU)
- #define **__HAL_RCC_APB2_RELEASE_RESET**() (**MRCC_PERIPHERAL**->RCC_APB2RSTR_REG = 0x00U)
- #define **__HAL_RCC_AHB1_FORCE_RESET**() (**MRCC_PERIPHERAL**->RCC_AHB1RSTR_REG = 0xFFFFFFFFU)
- #define **__HAL_RCC_AHB1_RELEASE_RESET**() (**MRCC_PERIPHERAL**->RCC_AHB1RSTR_REG = 0x00U)

Macro Definition Documentation

#define __HAL_RCC_AHB1_FORCE_RESET() (MRCC_PERIPHERAL->RCC_AHB1RSTR_REG = 0xFFFFFFFFU)

#define __HAL_RCC_AHB1_RELEASE_RESET() (MRCC_PERIPHERAL->RCC_AHB1RSTR_REG = 0x00U)

#define __HAL_RCC_APB1_FORCE_RESET() (MRCC_PERIPHERAL->RCC_APB1RSTR_REG = 0xFFFFFFFFU)

#define __HAL_RCC_APB1_RELEASE_RESET() (MRCC_PERIPHERAL->RCC_APB1RSTR_REG = 0x00U)

#define __HAL_RCC_APB2_FORCE_RESET() (MRCC_PERIPHERAL->RCC_APB2RSTR_REG = 0xFFFFFFFFU)

#define __HAL_RCC_APB2_RELEASE_RESET() (MRCC_PERIPHERAL->RCC_APB2RSTR_REG = 0x00U)

#define CSSON_BIT 19

#define HPRE0_BIT 4

#define HSEBYP_BIT 18

#define HSEON_BIT 16

#define HSERDY_BIT 17

#define HSION_BIT 0

#define HSIRDY_BIT 1

#define MRCC_PERIPHERAL ((volatile ST_MRCC_RegistersMap_t *)RCC_PERIPHERAL_BASE_ADDR)

#define PLLI2SON_BIT 26

#define PLLI2SRDY_BIT 27

#define PLLM0_BIT 0

#define PLLM1_BIT 1

#define PLLM2_BIT 2

#define PLLM3_BIT 3

```
#define PLLM4_BIT   4

#define PLLM5_BIT   5

#define PLLN0_BIT   6

#define PLLON_BIT   24

#define PLLP0_BIT   16

#define PLLRDY_BIT   25

#define PLLSRC_BIT   22

#define PPRE10_BIT   10

#define PPRE20_BIT   13

#define RCC_PERIPHERAL_BASE_ADDR   (0x40023800)

#define SW0_BIT   0

#define SW1_BIT   1

#define SWS0_BIT   2

#define SWS1_BIT   3
```

rcc_private.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC_Drivers
4 // File          : rcc_private.h
5 // Date          : Sep 8, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8 #ifndef MCAL_RCC_RCC_PRIVATE_H_
9 #define MCAL_RCC_RCC_PRIVATE_H_
10
11
12 #define  RCC_PERIPHERAL_BASE_ADDR       (0x40023800)
13
14
15 typedef struct
16 {
17     vuint32_t RCC_CR_REG;
18     vuint32_t RCC_PLLCFGR_REG;
19     vuint32_t RCC_CFGR_REG;
20     vuint32_t RCC_CIR_REG;
21     vuint32_t RCC_AHB1RSTR_REG;
22     vuint32_t RCC_AHB2RSTR_REG;
23     vuint32_t RESERVED0_REG;
24     vuint32_t RESERVED1_REG;
25     vuint32_t RCC_APB1RSTR_REG;
26     vuint32_t RCC_APB2RSTR_REG;
27     vuint32_t RESERVED2_REG;
28     vuint32_t RESERVED3_REG;
29     vuint32_t RCC_AHB1ENR_REG;
30     vuint32_t RCC_AHB2ENR_REG;
31     vuint32_t Reserved5_REG;
32     vuint32_t Reserved6_REG;
33     vuint32_t RCC_APB1ENR_REG;
34     vuint32_t RCC_APB2ENR_REG;
35     vuint32_t RESERVED7_REG;
36     vuint32_t RESERVED8_REG;
37     vuint32_t RCC_AHB1LPENR_REG;
38     vuint32_t RCC_AHB2LPENR_REG;
39     vuint32_t RESERVED9_REG;
40     vuint32_t RESERVED10_REG;
41     vuint32_t RCC_APB1LPENR_REG;
42     vuint32_t RCC_APB2LPENR_REG;
43     vuint32_t RESERVED11_REG;
44     vuint32_t RESERVED12_REG;
45     vuint32_t RCC_BDCR_REG;
46     vuint32_t RCC_CSR_REG;
47     vuint32_t RESERVED13_REG;
48     vuint32_t RESERVED14_REG;
49     vuint32_t RCC_SSCGR_REG;
50     vuint32_t RCC_PLLI2SCFGR_REG;
51     vuint32_t RESERVED15_REG;
52     vuint32_t RCC_DCKCFGR_REG;
53 }ST_MRCC_RegistersMap_t;
54
55 #define MRCC_PERIPHERAL ((volatile ST_MRCC_RegistersMap_t *)RCC_PERIPHERAL_BASE_ADDR)
56
57 /* RCC CR REG Bits */
58
59 #define HSION_BIT           0
60 #define HSIRDY_BIT          1
61 #define HSEON_BIT           16
62 #define HSERDY_BIT          17
63 #define HSEBYP_BIT          18
64 #define CSSON_BIT           19
65 #define PLLON_BIT           24
66 #define PLLRDY_BIT          25
67 #define PLLI2SON_BIT        26
68 #define PLLI2SRDY_BIT       27
69
70 /* RCC_PLLCFGR_REG Bits */
71
72 #define PLLM0_BIT   0
```

```
73 #define PLLM1_BIT   1
74 #define PLLM2_BIT   2
75 #define PLLM3_BIT   3
76 #define PLLM4_BIT   4
77 #define PLLM5_BIT   5
78 #define PLLN0_BIT   6
79 #define PLLP0_BIT   16
80 #define PLLSRC_BIT  22
81
82 /* RCC_CFGR_REG Bits */
83
84 #define SW0_BIT     0
85 #define SW1_BIT     1
86 #define SWS0_BIT    2
87 #define SWS1_BIT    3
88 #define HPRE0_BIT   4
89 #define PPRE10_BIT  10
90 #define PPRE20_BIT  13
91
92
93 #define __HAL_RCC_APB1_FORCE_RESET()     (MRCC_PERIPHERAL->RCC_APB1RSTR_REG =
0xFFFFFFFFU)
94 #define __HAL_RCC_APB1_RELEASE_RESET()   (MRCC_PERIPHERAL->RCC_APB1RSTR_REG = 0x00U)
95 #define __HAL_RCC_APB2_FORCE_RESET()     (MRCC_PERIPHERAL->RCC_APB2RSTR_REG =
0xFFFFFFFFU)
96 #define __HAL_RCC_APB2_RELEASE_RESET()   (MRCC_PERIPHERAL->RCC_APB2RSTR_REG = 0x00U)
97 #define __HAL_RCC_AHB1_FORCE_RESET()     (MRCC_PERIPHERAL->RCC_AHB1RSTR_REG =
0xFFFFFFFFU)
98 #define __HAL_RCC_AHB1_RELEASE_RESET()   (MRCC_PERIPHERAL->RCC_AHB1RSTR_REG = 0x00U)
99 #endif /* MCAL_RCC_RCC_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/systick/systick_config.h File Reference

Macros

- #define **MIN_VAL_OF_U32** 0
- #define **MAX_VAL_OF_U32** 0xFFFFFFFF
- #define **MSTK_IntervalSingle** 0
- #define **MSTK_IntervalPeriodic** 1

Enumerations

- enum **EN_MSTK_systemState_t** { **MSTK_OK** = 0, **MSTK_NOK**, **MSTK_INVALID_PARAMTER** }
  Enumeration for the system state of SysTick functions.
- enum **EN_MSTK_clkSourceOptions_t** { **MSTK_CLK_AHB_8** = 0, **MSTK_CLK_PROCESSOR_AHB** }
  Enumeration for the clock source options in SysTick.
- enum **EN_MSTK_interruptStates_t** { **MSTK_INTERRUPT_ENABLED** = 0, **MSTK_INTERRUPT_DISABLED** }
  Enumeration for the interrupt states in SysTick.

Macro Definition Documentation

#define MAX_VAL_OF_U32 0xFFFFFFFF

#define MIN_VAL_OF_U32 0

#define MSTK_IntervalPeriodic 1

#define MSTK_IntervalSingle 0

Enumeration Type Documentation

enum EN_MSTK_clkSourceOptions_t

Enumeration for the clock source options in SysTick.

Enumerator:

MSTK_CLK_AHB_8	SysTick clock source is AHB/8.
MSTK_CLK_PROCESSOR_AHB	SysTick clock source is the processor clock (AHB).

enum EN_MSTK_interruptStates_t

Enumeration for the interrupt states in SysTick.

Enumerator:

MSTK_INTERRUPT_ENABLED	SysTick interrupt is enabled.
MSTK_INTERRUPT_DISABLED	SysTick interrupt is disabled.

enum EN_MSTK_systemState_t

Enumeration for the system state of SysTick functions.

Enumerator:

MSTK_OK	Operation successful.
MSTK_NOK	Operation failed.
MSTK_INVALID_PARAMTER	Invalid parameter detected.

systick_config.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC
4 // File          : systick_config.h
5 // Date          : Sep 12, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8 #ifndef MCAL_SYSTICK_SYSTICK_CONFIG_H_
9 #define MCAL_SYSTICK_SYSTICK_CONFIG_H_
10
11
12 #define MIN_VAL_OF_U32              0
13 #define MAX_VAL_OF_U32              0xFFFFFFFF
14
15 #define MSTK_IntervalSingle         0
16 #define MSTK_IntervalPeriodic       1
17
18
22 typedef enum
23 {
24     MSTK_OK = 0,
25     MSTK_NOK,
26     MSTK_INVALID_PARAMTER
27 } EN_MSTK_systemState_t;
28
32 typedef enum
33 {
34     MSTK_CLK_AHB_8 = 0,
35     MSTK_CLK_PROCESSOR_AHB
36 } EN_MSTK_clkSourceOptions_t;
37
41 typedef enum
42 {
43     MSTK INTERRUPT ENABLED = 0,
44     MSTK_INTERRUPT_DISABLED
45 } EN_MSTK_interruptStates_t;
46
47 #endif /* MCAL_SYSTICK_SYSTICK_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/systick/systick_interface.h File Reference

Header file for the SysTick (System Timer) module interface.

```
#include "../../COMMON/std_types.h"
#include "../../COMMON/bit_math.h"
#include "systick_private.h"
#include "systick_config.h"
```

Functions

- **EN_MSTK_systemState_t MSTK_init** (**EN_MSTK_clkSourceOptions_t** copy_uddtClkSource, **EN_MSTK_interruptStates_t** copy_uddtIntStates)
  Initialize the SysTick timer.

- **EN_MSTK_systemState_t MSTK_setBusyWait** (**uint32_t** copy_u32NumberOfTicks)
  Set a busy-wait delay using the SysTick timer.

- **EN_MSTK_systemState_t MSTK_SetIntervalSingle** (**uint32_t** copy_u32NumberOfTicks, void(*Pf)(void))
  Set a single-shot interval using the SysTick timer.

- **EN_MSTK_systemState_t MSTK_SetIntervalPeriodic** (**uint32_t** copy_u32NumberOfTicks, void(*Pf)(void))
  Set a periodic interval using the SysTick timer.

- **EN_MSTK_systemState_t MSTK_StopInterval** (void)
  Stop the current interval in the SysTick timer.

- **EN_MSTK_systemState_t MSTK_getElapsedTime** (**uint32_t** *copy_u32PtrRetOfElapsedTicks)
  Get the elapsed time since the last SysTick timer initialization.

Detailed Description

Header file for the SysTick (System Timer) module interface.

Function Documentation

EN_MSTK_systemState_t MSTK_getElapsedTime (uint32_t * *copy_u32PtrRetOfElapsedTicks*)

Get the elapsed time since the last SysTick timer initialization.

This function retrieves the elapsed time since the last SysTick timer initialization.

Parameters

copy_u32PtrRetOfElapsedTicks	Pointer to store the result of the elapsed ticks.

Returns

The system state after getting the elapsed time.

- MSTK_OK: Operation successful.
- MSTK_NOK: Operation failed.
- MSTK_INVALID_PARAMTER: Invalid parameter detected.

EN_MSTK_systemState_t MSTK_init (EN_MSTK_clkSourceOptions_t *copy_uddtClkSource*, EN_MSTK_interruptStates_t *copy_uddtIntStates*)

Initialize the SysTick timer.

This function initializes the SysTick timer with the specified clock source and interrupt state.

Parameters

copy_uddtClkSource	The clock source option (MSTK_CLK_AHB_8, MSTK_CLK_PROCESSOR_AHB).
copy_uddtIntStates	The interrupt state option (MSTK_INTERRUPT_ENABLED, MSTK_INTERRUPT_DISABLED).

Returns

The system state after initializing the SysTick timer.

- MSTK_OK: Operation successful.
- MSTK_NOK: Operation failed.
- MSTK_INVALID_PARAMTER: Invalid parameter detected.

EN_MSTK_systemState_t MSTK_setBusyWait (uint32_t *copy_u32NumberOfTicks*)

Set a busy-wait delay using the SysTick timer.

This function sets a busy-wait delay using the SysTick timer for the specified number of ticks.

Parameters

copy_u32NumberOfTicks	The number of ticks for the busy-wait delay.

Returns

The system state after setting the busy-wait delay.

- MSTK_OK: Operation successful.
- MSTK_NOK: Operation failed.
- MSTK_INVALID_PARAMTER: Invalid parameter detected.

EN_MSTK_systemState_t MSTK_SetIntervalPeriodic (uint32_t *copy_u32NumberOfTicks*, void(*)(void) *Pf*)

Set a periodic interval using the SysTick timer.

This function sets a periodic interval using the SysTick timer for the specified number of ticks and associates a callback function.

Parameters

copy_u32NumberOfTicks	The number of ticks for the periodic interval.
Pf	Pointer to the callback function to be executed after each interval elapses.

Returns

The system state after setting the periodic interval.

- MSTK_OK: Operation successful.
- MSTK_NOK: Operation failed.
- MSTK_INVALID_PARAMTER: Invalid parameter detected.

EN_MSTK_systemState_t MSTK_SetIntervalSingle (uint32_t *copy_u32NumberOfTicks*, void(*)(void) *Pf*)

Set a single-shot interval using the SysTick timer.

This function sets a single-shot interval using the SysTick timer for the specified number of ticks and associates a callback function.

Parameters

copy_u32Number OfTicks	The number of ticks for the single-shot interval.
Pf	Pointer to the callback function to be executed after the interval elapses.

Returns

The system state after setting the single-shot interval.

- MSTK_OK: Operation successful.
- MSTK_NOK: Operation failed.
- MSTK_INVALID_PARAMTER: Invalid parameter detected.

EN_MSTK_systemState_t MSTK_StopInterval (void)

Stop the current interval in the SysTick timer.

This function stops the current interval in the SysTick timer.

Returns

The system state after stopping the interval.

- MSTK_OK: Operation successful.
- MSTK_NOK: Operation failed.
- MSTK_INVALID_PARAMTER: Invalid parameter detected.

systick_interface.h

Go to the documentation of this file.

```
1
6 #ifndef MCAL_SYSTICK_SYSTICK_INTERFACE_H_
7 #define MCAL_SYSTICK_SYSTICK_INTERFACE_H_
8
9 #include "../../COMMON/std_types.h"
10 #include "../../COMMON/bit_math.h"
11 #include "systick_private.h"
12 #include "systick_config.h"
13
27 EN_MSTK_systemState_t MSTK_init(EN_MSTK_clkSourceOptions_t copy_uddtClkSource,
EN_MSTK_interruptStates_t copy_uddtIntStates);
28
41 EN_MSTK_systemState_t MSTK_setBusyWait(uint32_t copy_u32NumberOfTicks);
42
56 EN_MSTK_systemState_t MSTK_SetIntervalSingle(uint32_t copy_u32NumberOfTicks, void
(*Pf)(void));
57
71 EN_MSTK_systemState_t MSTK_SetIntervalPeriodic(uint32_t copy_u32NumberOfTicks, void
(*Pf)(void));
72
83 EN_MSTK_systemState_t MSTK_StopInterval(void);
84
97 EN_MSTK_systemState_t MSTK_getElapsedTime(uint32_t *copy_u32PtrRetOfElapsedTicks);
98
109 #endif /* MCAL_SYSTICK_SYSTICK_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/systick/systick_private.h File Reference

Data Structures

struct ST_MSTK_RegistersMap_t**Macros**

- #define **MSTK_PERIPHERAL_BASE_ADDR** (0xE000E010)
- #define **MSTK_ENABLE_BIT** 0
- #define **MSTK_TICKINT_BIT** 1
- #define **MSTK_CLKSOURCE_BIT** 2
- #define **MSTK_COUNTFLAG_BIT** 16
- #define **MSTK_PERIPHERAL** ((volatile **ST_MSTK_RegistersMap_t** *)**MSTK_PERIPHERAL_BASE_ADDR**)

Macro Definition Documentation

#define MSTK_CLKSOURCE_BIT 2

#define MSTK_COUNTFLAG_BIT 16

#define MSTK_ENABLE_BIT 0

#define MSTK_PERIPHERAL ((volatile ST_MSTK_RegistersMap_t *)MSTK_PERIPHERAL_BASE_ADDR)

#define MSTK_PERIPHERAL_BASE_ADDR (0xE000E010)

#define MSTK_TICKINT_BIT 1

systick_private.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC
4 // File          : systick_private.h
5 // Date          : Sep 12, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8 #ifndef MCAL_SYSTICK_SYSTICK_PRIVATE_H_
9 #define MCAL_SYSTICK_SYSTICK_PRIVATE_H_
10
11 #define  MSTK_PERIPHERAL_BASE_ADDR          (0xE000E010)
12
13
14 typedef struct
15 {
16
17     vuint32_t MSTK_STK_CTRL;
18     vuint32_t MSTK_STK_LOAD;
19     vuint32_t MSTK_STK_VAL;
20     vuint32_t MSTK_STK_CALIB;
21
22 }ST_MSTK_RegistersMap_t;
23
24 #define MSTK_ENABLE_BIT         0
25 #define MSTK_TICKINT_BIT        1
26 #define MSTK_CLKSOURCE_BIT      2
27 #define MSTK_COUNTFLAG_BIT      16
28
29 #define MSTK_PERIPHERAL     ((volatile ST_MSTK_RegistersMap_t
*)MSTK_PERIPHERAL_BASE_ADDR)
30
31
32 #endif /* MCAL_SYSTICK_SYSTICK_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/usart/usart_config.h File Reference

Data Structures

struct **ST_MUSART_clockInit_t***Structure for USART clock initialization.*

struct **ST_MUSART_cfg_t***Structure for USART configuration.*

Macros

- #define **THRESHOLD_VALUE** 50000
- #define **__PCLK__** 8000000UL
- #define **MUSART_ENABLE** 1
- #define **MUSART_DISABLE** 0

Enumerations

- enum **EN_MUSART_systeamState_t** { **MUSART_NOK** = 0, **MUSART_OK**, **MUSART_PTR_NULL** }
  Enumeration for USART system states.
- enum **EN_MUSART_samplingModeOptions_t** { **MUSART_SAMPLING_BY_16** = 0, **MUSART_SAMPLING_BY_8** }
  Enumeration for USART sampling mode options.
- enum **EN_MUSART_baudRateOptions_t** { **MUSART_BUAD_RATE_1200_bps** = 1200, **MUSART_BUAD_RATE_2400_bps** = 2400, **MUSART_BUAD_RATE_9600_bps** = 9600, **MUSART_BUAD_RATE_19200_bps** = 19200, **MUSART_BUAD_RATE_38400_bps** = 38400, **MUSART_BUAD_RATE_57600_bps** = 57600, **MUSART_BUAD_RATE_115200_bps** = 115200, **MUSART_BUAD_RATE_230400_bps** = 230400, **MUSART_BUAD_RATE_460800_bps** = 460800, **MUSART_BUAD_RATE_921600_bps** = 921600, **MUSART_BUAD_RATE_1792000_bps** = 1792000, **MUSART_BUAD_RATE_1843200_bps** = 1843200, **MUSART_BUAD_RATE_3584000_bps** = 3584000, **MUSART_BUAD_RATE_3686400_bps** = 3686400, **MUSART_BUAD_RATE_7168000_bps** = 7168000, **MUSART_BUAD_RATE_7372800_bps** = 7372800, **MUSART_BUAD_RATE_9000000_bps** = 9000000, **MUSART_BUAD_RATE_10500000_bps** = 10500000 }
  Enumeration for USART baud rate options.
- enum **EN_MUSART_transferControl_t** { **MUSART_TX_ONLY** = 0, **MUSART_RX_ONLY**, **MUSART_TX_RX** }
  Enumeration for USART transfer control options.
- enum **EN_MUSART_stopBitOption_t** { **MUSART_ONE_STOP_BIT** = 0, **MUSART_HALF_STOP_BIT**, **MUSART_TWO_STOP_BIT**, **MUSART_ONE_AND_HALF_BIT** }
  Enumeration for USART stop bit options.
- enum **EN_MUSART_parityControlOption_t** { **MUSART_PARITY_DISABLED** = 0, **MUSART_PARITY_ENABLED** }
  Enumeration for USART parity control options.
- enum **EN_MUSART_paritySelectionOption_t** { **MUSART_EVEN_PARITY** = 0, **MUSART_ODD_PARITY** }
  Enumeration for USART parity selection options.
- enum **EN_MUSART_dataSizeOptions_t** { **MUSART_DATA_SIZE_8_BIT** = 0, **MUSART_DATA_SIZE_9_BIT** }
  Enumeration for USART data size options.

Macro Definition Documentation

#define __PCLK__ 8000000UL

#define MUSART_DISABLE 0

#define MUSART_ENABLE 1

#define THRESHOLD_VALUE 50000

Enumeration Type Documentation

enum EN_MUSART_baudRateOptions_t

Enumeration for USART baud rate options.

Enumerator:

MUSART_BUAD_RATE_1200_bps	
MUSART_BUAD_RATE_2400_bps	
MUSART_BUAD_RATE_9600_bps	
MUSART_BUAD_RATE_19200_bps	
MUSART_BUAD_RATE_38400_bps	
MUSART_BUAD_RATE_57600_bps	
MUSART_BUAD_RATE_115200_bps	
MUSART_BUAD_RATE_230400_bps	
MUSART_BUAD_RATE_460800_bps	
MUSART_BUAD_RATE_921600_bps	
MUSART_BUAD_RATE_1792000_bps	
MUSART_BUAD_RATE_1843200_bps	
MUSART_BUAD_RATE_3584000_bps	
MUSART_BUAD	

_RATE_3686400_bps	
MUSART_BUAD_RATE_7168000_bps	
MUSART_BUAD_RATE_7372800_bps	
MUSART_BUAD_RATE_9000000_bps	
MUSART_BUAD_RATE_10500000_bps	

enum EN_MUSART_dataSizeOptions_t

Enumeration for USART data size options.

Enumerator:

MUSART_DATA_SIZE_8_BIT	USART data size: 8 bits.
MUSART_DATA_SIZE_9_BIT	USART data size: 9 bits.

enum EN_MUSART_parityControlOption_t

Enumeration for USART parity control options.

Enumerator:

MUSART_PARITY_DISABLED	USART parity control disabled.
MUSART_PARITY_ENABLED	USART parity control enabled.

enum EN_MUSART_paritySelectionOption_t

Enumeration for USART parity selection options.

Enumerator:

MUSART_EVEN_PARITY	USART even parity.
MUSART_ODD_PARITY	USART odd parity.

enum EN_MUSART_samplingModeOptions_t

Enumeration for USART sampling mode options.

Enumerator:

MUSART_SAMPLING_BY_16	USART sampling by 16.
MUSART_SAMPLING_BY_8	USART sampling by 8.

enum EN_MUSART_stopBitOption_t

Enumeration for USART stop bit options.

Enumerator:

MUSART_ONE_STOP_BIT	USART one stop bit.
MUSART_HALF_STOP_BIT	USART half stop bit.
MUSART_TWO_STOP_BIT	USART two stop bits.
MUSART_ONE_AND_HALF_BIT	USART one and a half stop bits.

enum EN_MUSART_systeamState_t

Enumeration for USART system states.

Enumerator:

MUSART_NOK	USART operation unsuccessful.
MUSART_OK	USART operation successful.
MUSART_PTR_NULL	Null pointer encountered during the operation.

enum EN_MUSART_transferControl_t

Enumeration for USART transfer control options.

Enumerator:

MUSART_TX_ONLY	USART transmit only.
MUSART_RX_ONLY	USART receive only.
MUSART_TX_RX	USART transmit and receive.

usart_config.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC
4 // File          : usart_config.h
5 // Date          : Sep 19, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8 #ifndef MCAL_USART_USART_CONFIG_H_
9 #define MCAL_USART_USART_CONFIG_H_
10
11 #define THRESHOLD_VALUE     50000
12 #define __PCLK__            8000000UL
13
14 #define MUSART_ENABLE               1
15 #define MUSART_DISABLE              0
16
20 typedef enum
21 {
22     MUSART_NOK = 0,
23     MUSART_OK,
24     MUSART_PTR_NULL
25 } EN_MUSART_systeamState_t;
26
30 typedef enum
31 {
32     MUSART_SAMPLING_BY_16 = 0,
33     MUSART_SAMPLING_BY_8
34 } EN_MUSART_samplingModeOptions_t;
35
39 typedef enum
40 {
41     MUSART_BUAD_RATE_1200_bps = 1200,
42     MUSART_BUAD_RATE_2400_bps = 2400,
43     MUSART_BUAD_RATE_9600_bps = 9600,
44     MUSART_BUAD_RATE_19200_bps = 19200,
45     MUSART_BUAD_RATE_38400_bps = 38400,
46     MUSART_BUAD_RATE_57600_bps = 57600,
47     MUSART_BUAD_RATE_115200_bps = 115200,
48     MUSART_BUAD_RATE_230400_bps = 230400,
49     MUSART_BUAD_RATE_460800_bps = 460800,
50     MUSART_BUAD_RATE_921600_bps = 921600,
51     MUSART_BUAD_RATE_1792000_bps = 1792000,
52     MUSART_BUAD_RATE_1843200_bps = 1843200,
53     MUSART_BUAD_RATE_3584000_bps = 3584000,
54     MUSART_BUAD_RATE_3686400_bps = 3686400,
55     MUSART_BUAD_RATE_7168000_bps = 7168000,
56     MUSART_BUAD_RATE_7372800_bps = 7372800,
57     MUSART_BUAD_RATE_9000000_bps = 9000000,
58     MUSART_BUAD_RATE_10500000_bps = 10500000,
59
60 }EN_MUSART_baudRateOptions_t;
61
65 typedef enum
66 {
67     MUSART_TX_ONLY = 0,
68     MUSART_RX_ONLY,
69     MUSART_TX_RX
70 } EN_MUSART_transferControl_t;
71
75 typedef enum
76 {
77     MUSART_ONE_STOP_BIT = 0,
78     MUSART_HALF_STOP_BIT,
79     MUSART_TWO_STOP_BIT,
80     MUSART_ONE_AND_HALF_BIT
81 } EN_MUSART_stopBitOption_t;
82
86 typedef enum
87 {
88     MUSART_PARITY_DISABLED = 0,
89     MUSART_PARITY_ENABLED
90 } EN_MUSART_parityControlOption_t;
```

```
91
95 typedef enum
96 {
97     MUSART_EVEN_PARITY = 0,
98     MUSART_ODD_PARITY
99 } EN_MUSART_paritySelectionOption_t;
100
104 typedef enum
105 {
106     MUSART_DATA_SIZE_8_BIT = 0,
107     MUSART_DATA_SIZE_9_BIT
108 } EN_MUSART_dataSizeOptions_t;
109
113 typedef struct
114 {
115     uint8_t clockOutput;
116     uint8_t clockPolarity;
117     uint8_t clockPhase;
118     uint8_t lastBitClockPulse;
119 } ST_MUSART_clockInit_t;
120
124 typedef struct
125 {
126     EN_MUSART_transferControl_t copy_uddtTransferDirection;
127     EN_MUSART_samplingModeOptions_t copy_uddtSamplingModeOption;
128     EN_MUSART_baudRateOptions_t copy_uddtBuadRateOption;
129     EN_MUSART_dataSizeOptions_t copy_uddtDataSizeOption;
130     EN_MUSART_parityControlOption_t copy_uddtParityControl;
131     EN_MUSART_paritySelectionOption_t copy_uddtParitySelection;
132     EN_MUSART_stopBitOption_t copy_uddtStopBitSelection;
133     uint8_t copy_HardwareFlowControl;
134     ST_MUSART_clockInit_t copy_uddtUartClockInit;
135 } ST_MUSART_cfg_t;
136
137 #endif /* MCAL_USART_USART_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/usart/usart_interface.h File Reference

Header file for the Universal Synchronous/Asynchronous Receiver Transmitter (MUSART) module.

```
#include "../../COMMON/std_types.h"
#include "../../COMMON/bit_math.h"
#include "../../MCAL/gpio/gpio_interface.h"
#include "usart_private.h"
#include "usart_config.h"
```

Functions

- **EN_MUSART_systeamState_t MUSART_uddtInit** (**ST_MUART_RegistersMap_t** *PS_USARTx, **ST_MUSART_cfg_t** const *PS_uddtUartCfg)
  Initialize the USART module with the given configuration.

- **EN_MUSART_systeamState_t MUSART_uddtEnable** (**ST_MUART_RegistersMap_t** *PS_USARTx)
  Enable the USART module.

- **EN_MUSART_systeamState_t MUSART_uddtDisable** (**ST_MUART_RegistersMap_t** *PS_USARTx)
  Disable the USART module.

- **EN_MUSART_systeamState_t MUSART_uddtTransmitByte** (**ST_MUART_RegistersMap_t** *PS_USARTx, **uint8_t** copy_u8ByteToSend)
  Transmit a byte through the USART module.

- **EN_MUSART_systeamState_t MUSART_uddtTransmitString** (**ST_MUART_RegistersMap_t** *PS_USARTx, **uint8_t** *copy_u8StringToSend)
  Transmit a string through the USART module.

- **EN_MUSART_systeamState_t MUSART_uddtReadDataRegister** (**ST_MUART_RegistersMap_t** *PS_USARTx, **uint8_t** *copy_u8ByteToReceive)
  Read data from the USART data register.

- **EN_MUSART_systeamState_t MUSART_uddtClearFlags** (**ST_MUART_RegistersMap_t** *PS_USARTx)
  Clear the USART flags.

- **EN_MUSART_systeamState_t MUSART_uddtReceiveByteSynchNonBlocking** (**ST_MUART_RegistersMap_t** *PS_USARTx, **uint8_t** *copy_u8ByteToReceive)
  Receive a byte asynchronously in a non-blocking manner.

- **EN_MUSART_systeamState_t MUSART_uddtReceiveStringAsynchBlocking** (**ST_MUART_RegistersMap_t** *PS_USARTx, **uint8_t** *copy_u8ByteToReceive)
  Receive a string asynchronously in a blocking manner.

- **EN_MUSART_systeamState_t MUSART_uddtReceiveStringSynchNonBlocking** (**ST_MUART_RegistersMap_t** *PS_USARTx, **uint8_t** *copy_u8ByteToReceive)
  Receive a string asynchronously in a non-blocking manner.

- **EN_MUSART_systeamState_t MUSART_RxIntSetStatus** (**ST_MUART_RegistersMap_t** *PS_USARTx, **uint8_t** copy_u8Status)
  Set the receive interrupt status for the USART module.

- **EN_MUSART_systeamState_t MUSART1_uddtSetCallBack** (void(*ptr)(void))
  Set the callback function for USART1.

- **EN_MUSART_systeamState_t MUSART2_uddtSetCallBack** (void(*ptr)(void))
  Set the callback function for USART2.

- **EN_MUSART_systeamState_t MUSART6_uddtSetCallBack** (void(*ptr)(void))
  Set the callback function for USART6.

Detailed Description

Header file for the Universal Synchronous/Asynchronous Receiver Transmitter (MUSART) module.

Function Documentation

EN_MUSART_systeamState_t MUSART1_uddtSetCallBack (void(*)(void) *ptr*)

Set the callback function for USART1.

This function sets the callback function for USART1.

Parameters

ptr	Pointer to the callback function.

Returns

The system state after setting the callback function.

- #EN_MUSART_OK: Callback setting successful.
- #EN_MUSART_NOK: Callback setting failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART2_uddtSetCallBack (void(*)(void) *ptr*)

Set the callback function for USART2.

This function sets the callback function for USART2.

Parameters

ptr	Pointer to the callback function.

Returns

The system state after setting the callback function.

- #EN_MUSART_OK: Callback setting successful.
- #EN_MUSART_NOK: Callback setting failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART6_uddtSetCallBack (void(*)(void) *ptr*)

Set the callback function for USART6.

This function sets the callback function for USART6.

Parameters

ptr	Pointer to the callback function.

Returns

The system state after setting the callback function.

- #EN_MUSART_OK: Callback setting successful.
- #EN_MUSART_NOK: Callback setting failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_RxIntSetStatus (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t *copy_u8Status*)

Set the receive interrupt status for the USART module.

This function sets the receive interrupt status for the USART module.

Parameters

PS_USARTx	Pointer to the USARTx registers map.
copy_u8Status	The status to set.

Returns

The system state after setting the receive interrupt status.

- #EN_MUSART_OK: Status setting successful.
- #EN_MUSART_NOK: Status setting failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtClearFlags (ST_MUART_RegistersMap_t * *PS_USARTx*)

Clear the USART flags.

This function clears the USART flags.

Parameters

PS_USARTx	Pointer to the USARTx registers map.

Returns

The system state after clearing the USART flags.

- #EN_MUSART_OK: Flag clearing successful.
- #EN_MUSART_NOK: Flag clearing failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtDisable (ST_MUART_RegistersMap_t * *PS_USARTx*)

Disable the USART module.

This function disables the USART module.

Parameters

PS_USARTx	Pointer to the USARTx registers map.

Returns

The system state after disabling the USART module.

- #EN_MUSART_OK: USART disabling successful.
- #EN_MUSART_NOK: USART disabling failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtEnable (ST_MUART_RegistersMap_t * *PS_USARTx*)

Enable the USART module.

This function enables the USART module.

Parameters

PS_USARTx	Pointer to the USARTx registers map.

Returns

The system state after enabling the USART module.

- #EN_MUSART_OK: USART enabling successful.
- #EN_MUSART_NOK: USART enabling failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtInit (ST_MUART_RegistersMap_t * *PS_USARTx*, ST_MUSART_cfg_t const * *PS_uddtUartCfg*)

Initialize the USART module with the given configuration.

This function initializes the USART module with the provided configuration.

Parameters

PS_USARTx	Pointer to the USARTx registers map.
PS_uddtUartCfg	Pointer to the USART configuration structure.

Returns

The system state after initializing the USART module.

- #EN_MUSART_OK: Initialization successful.
- #EN_MUSART_NOK: Initialization failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtReadDataRegister (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t * *copy_u8ByteToReceive*)

Read data from the USART data register.

This function reads data from the USART data register.

Parameters

PS_USARTx	Pointer to the USARTx registers map.
copy_u8ByteToReceive	Pointer to store the received byte.

Returns

The system state after reading the data register.

- #EN_MUSART_OK: Data read successful.
- #EN_MUSART_NOK: Data read failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtReceiveByteSynchNonBlocking (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t * *copy_u8ByteToReceive*)

Receive a byte asynchronously in a non-blocking manner.

This function receives a byte asynchronously in a non-blocking manner.

Parameters

PS_USARTx	Pointer to the USARTx registers map.
copy_u8ByteToReceive	Pointer to store the received byte.

Returns

The system state after receiving the byte.

- #EN_MUSART_OK: Byte reception successful.
- #EN_MUSART_NOK: Byte reception failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtReceiveStringAsynchBlocking (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t * *copy_u8ByteToReceive*)

Receive a string asynchronously in a blocking manner.

This function receives a string asynchronously in a blocking manner.

Parameters

PS_USARTx	Pointer to the USARTx registers map.
copy_u8ByteToReceive	Pointer to store the received string.

Returns

The system state after receiving the string.

- #EN_MUSART_OK: String reception successful.
- #EN_MUSART_NOK: String reception failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtReceiveStringSynchNonBlocking (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t * *copy_u8ByteToReceive*)

Receive a string asynchronously in a non-blocking manner.

This function receives a string asynchronously in a non-blocking manner.

Parameters

PS_USARTx	Pointer to the USARTx registers map.
copy_u8ByteToReceive	Pointer to store the received string.

Returns

The system state after receiving the string.

- #EN_MUSART_OK: String reception successful.
- #EN_MUSART_NOK: String reception failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtTransmitByte (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t *copy_u8ByteToSend*)

Transmit a byte through the USART module.

This function transmits a byte through the USART module.

Parameters

PS_USARTx	Pointer to the USARTx registers map.
copy_u8ByteToSend	The byte to transmit.

Returns

The system state after transmitting the byte.

- #EN_MUSART_OK: Byte transmission successful.
- #EN_MUSART_NOK: Byte transmission failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

EN_MUSART_systeamState_t MUSART_uddtTransmitString (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t * *copy_u8StringToSend*)

Transmit a string through the USART module.

This function transmits a string through the USART module.

Parameters

PS_USARTx	Pointer to the USARTx registers map.
copy_u8StringToSend	Pointer to the string to transmit.

Returns

The system state after transmitting the string.

- #EN_MUSART_OK: String transmission successful.
- #EN_MUSART_NOK: String transmission failed.
- #EN_MUSART_PTR_NULL: Null pointer encountered during the operation.

usart_interface.h

Go to the documentation of this file.

```
1
6 #ifndef MCAL_USART_USART_INTERFACE_H_
7 #define MCAL_USART_USART_INTERFACE_H_
8
9 #include "../../COMMON/std_types.h"
10 #include "../../COMMON/bit_math.h"
11 #include "../../MCAL/gpio/gpio_interface.h"
12 #include "usart_private.h"
13 #include "usart_config.h"
14
15
29 EN_MUSART_systeamState_t MUSART_uddtInit(ST_MUART_RegistersMap_t *PS_USARTx,
ST_MUSART_cfg_t const *PS_uddtUartCfg);
30
43 EN_MUSART_systeamState_t MUSART_uddtEnable(ST_MUART_RegistersMap_t *PS_USARTx);
44
57 EN_MUSART_systeamState_t MUSART_uddtDisable(ST_MUART_RegistersMap_t *PS_USARTx);
58
72 EN_MUSART_systeamState_t MUSART_uddtTransmitByte(ST_MUART_RegistersMap_t *PS_USARTx,
uint8_t copy_u8ByteToSend);
73
87 EN_MUSART_systeamState_t MUSART_uddtTransmitString(ST_MUART_RegistersMap_t
*PS_USARTx, uint8_t *copy_u8StringToSend);
88
102 EN_MUSART_systeamState_t MUSART_uddtReadDataRegister(ST_MUART_RegistersMap_t
*PS_USARTx, uint8_t *copy_u8ByteToReceive);
103
116 EN_MUSART_systeamState_t MUSART_uddtClearFlags(ST_MUART_RegistersMap_t *PS_USARTx);
117
131 EN_MUSART_systeamState_t
MUSART_uddtReceiveByteSynchNonBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
*copy_u8ByteToReceive);
132
146 EN MUSART systeamState t
MUSART_uddtReceiveStringAsynchBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
*copy_u8ByteToReceive);
147
161 EN_MUSART_systeamState_t
MUSART_uddtReceiveStringSynchNonBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
*copy u8ByteToReceive);
162
176 EN_MUSART_systeamState_t MUSART_RxIntSetStatus(ST_MUART_RegistersMap_t *PS_USARTx,
uint8_t copy_u8Status);
177
190 EN_MUSART_systeamState_t MUSART1_uddtSetCallBack(void (*ptr)(void));
191
204 EN_MUSART_systeamState_t MUSART2_uddtSetCallBack(void (*ptr)(void));
205
218 EN_MUSART_systeamState_t MUSART6_uddtSetCallBack(void (*ptr)(void));
219
220
221 #endif /* MCAL_USART_USART_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/MCAL/usart/usart_private.h File Reference

Data Structures

struct ST_MUART_RegistersMap_t**Macros**

- #define **MUART1_PERIPHERAL_BASE_ADDR** (0x40011000)
- #define **MUART2_PERIPHERAL_BASE_ADDR** (0x40004400)
- #define **MUART6_PERIPHERAL_BASE_ADDR** (0x40011400)
- #define **MUART1_PERIPHERAL** ((**ST_MUART_RegistersMap_t** *)**MUART1_PERIPHERAL_BASE_ADDR**)
- #define **MUART2_PERIPHERAL** ((**ST_MUART_RegistersMap_t** *)**MUART2_PERIPHERAL_BASE_ADDR**)
- #define **MUART6_PERIPHERAL** ((**ST_MUART_RegistersMap_t** *)**MUART6_PERIPHERAL_BASE_ADDR**)
- #define **UART_DIV_SAMPLING16**(_PCLK_, _BAUD_) ((**uint32_t**)((((**float64_t**)(_PCLK_))*25U)/(4U*((**float64_t**)(_BAUD_)))))
- #define **UART_DIVMANT_SAMPLING16**(_PCLK_, _BAUD_) (**UART_DIV_SAMPLING16**((_PCLK_), (_BAUD_))/100U)
- #define **UART_DIVFRAQ_SAMPLING16**(_PCLK_, _BAUD_) ((((**UART_DIV_SAMPLING16**((_PCLK_), (_BAUD_)) - (**UART_DIVMANT_SAMPLING16**((_PCLK_), (_BAUD_)) * 100U)) * 16U) + 50U) / 100U)
- #define **UART_BRR_SAMPLING16**(_PCLK_, _BAUD_)
- #define **UART_DIV_SAMPLING8**(_PCLK_, _BAUD_) ((**uint32_t**)((((**float64_t**)(_PCLK_))*25U)/(2U*((**float64_t**)(_BAUD_)))))
- #define **UART_DIVMANT_SAMPLING8**(_PCLK_, _BAUD_) (**UART_DIV_SAMPLING8**((_PCLK_), (_BAUD_))/100U)
- #define **UART_DIVFRAQ_SAMPLING8**(_PCLK_, _BAUD_) ((((**UART_DIV_SAMPLING8**((_PCLK_), (_BAUD_)) - (**UART_DIVMANT_SAMPLING8**((_PCLK_), (_BAUD_)) * 100U)) * 8U) + 50U) / 100U)
- #define **UART_BRR_SAMPLING8**(_PCLK_, _BAUD_)
- #define **MUSART_SR_PE_BIT** 0
- #define **MUSART_SR_FE_BIT** 1
- #define **MUSART_SR_NE_BIT** 2
- #define **MUSART_SR_ORE_BIT** 3
- #define **MUSART_SR_IDLE_BIT** 4
- #define **MUSART_SR_RXNE_BIT** 5
- #define **MUSART_SR_TC_BIT** 6
- #define **MUSART_SR_TXE_BIT** 7
- #define **MUSART_SR_LBD_BIT** 8
- #define **MUSART_SR_CTS_BIT** 9
- #define **MUSART_CR1_SBK_BIT** 0
- #define **MUSART_CR1_RWU_BIT** 1
- #define **MUSART_CR1_RE_BIT** 2
- #define **MUSART_CR1_TE_BIT** 3
- #define **MUSART_CR1_IDLEIE_BIT** 4
- #define **MUSART_CR1_RXNEIE_BIT** 5
- #define **MUSART_CR1_TCIE_BIT** 6
- #define **MUSART_CR1_TXEIE_BIT** 7
- #define **MUSART_CR1_PEIE_BIT** 8
- #define **MUSART_CR1_PS_BIT** 9
- #define **MUSART_CR1_PCE_BIT** 10
- #define **MUSART_CR1_WAKE_BIT** 11
- #define **MUSART_CR1_M_BIT** 12
- #define **MUSART_CR1_UE_BIT** 13

- #define **MUSART_CR1_OVER8_BIT** 15
- #define **MUSART_CR2_ADD0_BIT** 0
- #define **MUSART_CR2_ADD1_BIT** 1
- #define **MUSART_CR2_ADD2_BIT** 2
- #define **MUSART_CR2_ADD3_BIT** 3
- #define **MUSART_CR2_LBDL_BIT** 5
- #define **MUSART_CR2_LBDIE_BIT** 6
- #define **MUSART_CR2_LBCL_BIT** 8
- #define **MUSART_CR2_CPHA_BIT** 9
- #define **MUSART_CR2_CPOL_BIT** 10
- #define **MUSART_CR2_CLKEN_BIT** 11
- #define **MUSART_CR2_STOP_BIT** 12
- #define **MUSART_CR2_STOP0_BIT** 12
- #define **MUSART_CR2_STOP1_BIT** 13
- #define **MUSART_CR2_LINEN_BIT** 14
- #define **MUSART_CR3_CTSIE_BIT** 10
- #define **MUSART_CR3_CTSE_BIT** 9
- #define **MUSART_CR3_RTSE_BIT** 8
- #define **MUSART_CR3_DMAT_BIT** 7
- #define **MUSART_CR3_DMAR_BIT** 6
- #define **MUSART_CR3_SCEN_BIT** 5
- #define **MUSART_CR3_NACK_BIT** 4
- #define **MUSART_CR3_HDSEL_BIT** 3
- #define **MUSART_CR3_IRLP_BIT** 2
- #define **MUSART_CR3_IREN_BIT** 1
- #define **MUSART_CR3_EIE_BIT** 0

Macro Definition Documentation

#define MUART1_PERIPHERAL ((ST_MUART_RegistersMap_t *)MUART1_PERIPHERAL_BASE_ADDR)

#define MUART1_PERIPHERAL_BASE_ADDR (0x40011000)

#define MUART2_PERIPHERAL ((ST_MUART_RegistersMap_t *)MUART2_PERIPHERAL_BASE_ADDR)

#define MUART2_PERIPHERAL_BASE_ADDR (0x40004400)

#define MUART6_PERIPHERAL ((ST_MUART_RegistersMap_t *)MUART6_PERIPHERAL_BASE_ADDR)

#define MUART6_PERIPHERAL_BASE_ADDR (0x40011400)

#define MUSART_CR1_IDLEIE_BIT 4

#define MUSART_CR1_M_BIT 12

#define MUSART_CR1_OVER8_BIT 15

#define MUSART_CR1_PCE_BIT 10

#define MUSART_CR1_PEIE_BIT 8

#define MUSART_CR1_PS_BIT 9

#define MUSART_CR1_RE_BIT 2

#define MUSART_CR1_RWU_BIT 1

#define MUSART_CR1_RXNEIE_BIT 5

#define MUSART_CR1_SBK_BIT 0

#define MUSART_CR1_TCIE_BIT 6

#define MUSART_CR1_TE_BIT 3

#define MUSART_CR1_TXEIE_BIT 7

#define MUSART_CR1_UE_BIT 13

#define MUSART_CR1_WAKE_BIT 11

#define MUSART_CR2_ADD0_BIT 0

#define MUSART_CR2_ADD1_BIT 1

#define MUSART_CR2_ADD2_BIT 2

```
#define MUSART_CR2_ADD3_BIT   3

#define MUSART_CR2_CLKEN_BIT   11

#define MUSART_CR2_CPHA_BIT   9

#define MUSART_CR2_CPOL_BIT   10

#define MUSART_CR2_LBCL_BIT   8

#define MUSART_CR2_LBDIE_BIT   6

#define MUSART_CR2_LBDL_BIT   5

#define MUSART_CR2_LINEN_BIT   14

#define MUSART_CR2_STOP0_BIT   12

#define MUSART_CR2_STOP1_BIT   13

#define MUSART_CR2_STOP_BIT   12

#define MUSART_CR3_CTSE_BIT   9

#define MUSART_CR3_CTSIE_BIT   10

#define MUSART_CR3_DMAR_BIT   6

#define MUSART_CR3_DMAT_BIT   7

#define MUSART_CR3_EIE_BIT   0

#define MUSART_CR3_HDSEL_BIT   3

#define MUSART_CR3_IREN_BIT   1

#define MUSART_CR3_IRLP_BIT   2

#define MUSART_CR3_NACK_BIT   4

#define MUSART_CR3_RTSE_BIT   8

#define MUSART_CR3_SCEN_BIT   5

#define MUSART_SR_CTS_BIT   9

#define MUSART_SR_FE_BIT   1

#define MUSART_SR_IDLE_BIT   4

#define MUSART_SR_LBD_BIT   8
```

#define MUSART_SR_NE_BIT 2

#define MUSART_SR_ORE_BIT 3

#define MUSART_SR_PE_BIT 0

#define MUSART_SR_RXNE_BIT 5

#define MUSART_SR_TC_BIT 6

#define MUSART_SR_TXE_BIT 7

#define UART_BRR_SAMPLING16(_PCLK_, _BAUD_)

Value:

```
((UART_DIVMANT_SAMPLING16((_PCLK_), (_BAUD_)) << 4U) + \

(UART_DIVFRAQ_SAMPLING16((_PCLK_), (_BAUD_)) & 0xF0U) + \

(UART_DIVFRAQ_SAMPLING16((_PCLK_), (_BAUD_)) & 0x0FU))
```

#define UART_BRR_SAMPLING8(_PCLK_, _BAUD_)

Value:

```
((UART_DIVMANT_SAMPLING8((_PCLK_), (_BAUD_)) << 4U) + \

((UART_DIVFRAQ_SAMPLING8((_PCLK_), (_BAUD_)) & 0xF8U) << 1U) + \

(UART_DIVFRAQ_SAMPLING8((_PCLK_), (_BAUD_)) & 0x07U))
```

#define UART_DIV_SAMPLING16(_PCLK_, _BAUD_) ((uint32_t)((((float64_t)(_PCLK_))*25U)/(4U*((float64_t)(_BAUD_)))))

#define UART_DIV_SAMPLING8(_PCLK_, _BAUD_) ((uint32_t)((((float64_t)(_PCLK_))*25U)/(2U*((float64_t)(_BAUD_)))))

#define UART_DIVFRAQ_SAMPLING16(_PCLK_, _BAUD_) ((((UART_DIV_SAMPLING16((_PCLK_), (_BAUD_)) - (UART_DIVMANT_SAMPLING16((_PCLK_), (_BAUD_)) * 100U)) * 16U) + 50U) / 100U)

#define UART_DIVFRAQ_SAMPLING8(_PCLK_, _BAUD_) ((((UART_DIV_SAMPLING8((_PCLK_), (_BAUD_)) - (UART_DIVMANT_SAMPLING8((_PCLK_), (_BAUD_)) * 100U)) * 8U) + 50U) / 100U)

#define UART_DIVMANT_SAMPLING16(_PCLK_, _BAUD_) (UART_DIV_SAMPLING16((_PCLK_), (_BAUD_))/100U)

#define UART_DIVMANT_SAMPLING8(_PCLK_, _BAUD_) (UART_DIV_SAMPLING8((_PCLK_), (_BAUD_))/100U)

usart_private.h

Go to the documentation of this file.

```
1 /******************************************************************************/
2 // Author        : Sherif Ashraf Khadr
3 // Project       : STM32F401xC
4 // File          : uart_private.h
5 // Date          : Sep 19, 2023
6 // GitHub        : https://github.com/sherifkhadr
7 /******************************************************************************/
8 #ifndef MCAL_USART_USART_PRIVATE_H_
9 #define MCAL_USART_USART_PRIVATE_H_
10
11 #define  MUART1_PERIPHERAL_BASE_ADDR            (0x40011000)
12 #define  MUART2_PERIPHERAL_BASE_ADDR            (0x40004400)
13 #define  MUART6_PERIPHERAL_BASE_ADDR            (0x40011400)
14
15
16
17 typedef struct
18 {
19
20     vuint32_t MUSART_SR;
21     vuint32_t MUSART_DR;
22     vuint32_t MUSART_BRR;
23     vuint32_t MUSART_CR1;
24     vuint32_t MUSART_CR2;
25     vuint32_t MUSART_CR3;
26     vuint32_t MUSART_GTPR;
27
28 }ST_MUART_RegistersMap_t;
29
30
31 #define MUART1_PERIPHERAL   ((ST_MUART_RegistersMap_t *)MUART1_PERIPHERAL_BASE_ADDR)
32 #define MUART2_PERIPHERAL   ((ST_MUART_RegistersMap_t *)MUART2_PERIPHERAL_BASE_ADDR)
33 #define MUART6_PERIPHERAL   ((ST_MUART_RegistersMap_t *)MUART6_PERIPHERAL_BASE_ADDR)
34
35
36
37 #define UART_DIV_SAMPLING16(_PCLK_, _BAUD_)
((uint32_t)((((float64_t)(_PCLK_))*25U)/(4U*((float64_t)(_BAUD_)))))
38 #define UART_DIVMANT_SAMPLING16(_PCLK_, _BAUD_)
(UART DIV SAMPLING16(( PCLK ), ( BAUD ))/100U)
39 #define UART_DIVFRAQ_SAMPLING16(_PCLK_, _BAUD_)
((((UART_DIV_SAMPLING16((_PCLK_), (_BAUD_)) - (UART_DIVMANT_SAMPLING16((_PCLK_),
(_BAUD_)) * 100U)) * 16U) + 50U) / 100U)
40 /* UART BRR = mantissa + overflow + fraction
41             = (UART DIVMANT << 4) + (UART DIVFRAQ & 0xF0) + (UART DIVFRAQ & 0x0FU) */
42 #define UART_BRR_SAMPLING16(_PCLK_, _BAUD_)
((UART_DIVMANT_SAMPLING16((_PCLK_), (_BAUD_)) << 4U) + \
43
(UART_DIVFRAQ_SAMPLING16((_PCLK_), (_BAUD_)) & 0xF0U) + \
44
(UART_DIVFRAQ_SAMPLING16((_PCLK_), (_BAUD_)) & 0x0FU))
45
46 #define UART_DIV_SAMPLING8(_PCLK_, _BAUD_)
((uint32_t)((((float64_t)(_PCLK_))*25U)/(2U*((float64_t)(_BAUD_)))))
47 #define UART_DIVMANT_SAMPLING8(_PCLK_, _BAUD_)          (UART_DIV_SAMPLING8((_PCLK_),
(_BAUD_))/100U)
48 #define UART DIVFRAQ SAMPLING8( PCLK ,  BAUD )
((((UART_DIV_SAMPLING8((_PCLK_), (_BAUD_)) - (UART_DIVMANT_SAMPLING8((_PCLK_), (_BAUD_))
* 100U)) * 8U) + 50U) / 100U)
49 /* UART BRR = mantissa + overflow + fraction
50             = (UART DIVMANT << 4) + ((UART DIVFRAQ & 0xF8) << 1) + (UART DIVFRAQ & 0x07U)
*/
51 #define UART BRR SAMPLING8( PCLK ,  BAUD )
((UART_DIVMANT_SAMPLING8((_PCLK_), (_BAUD_)) << 4U) + \
52
((UART_DIVFRAQ_SAMPLING8((_PCLK_), (_BAUD_)) & 0xF8U) << 1U) + \
53
(UART DIVFRAQ SAMPLING8(( PCLK ), ( BAUD )) & 0x07U))
54
55
56
57 /* Registers Bits  */
```

```
58
59 /*********************************************/
60 /*              SR BITS Mapping              */
61 /*********************************************/
62 /*  Parity error                    */
63 #define MUSART_SR_PE_BIT    0
64 /*  Framing error                   */
65 #define MUSART_SR_FE_BIT    1
66 /*  Noise error flag                */
67 #define MUSART_SR_NE_BIT    2
68 /*  Overrun error                   */
69 #define MUSART_SR_ORE_BIT   3
70 /*  IDLE line detected              */
71 #define MUSART_SR_IDLE_BIT  4
72 /*  Read data register not empty    */
73 #define MUSART_SR_RXNE_BIT  5
74 /*  Transmission complete           */
75 #define MUSART_SR_TC_BIT    6
76 /*  Transmit data register empty    */
77 #define MUSART_SR_TXE_BIT   7
78 /*  LIN break detection flag        */
79 #define MUSART_SR_LBD_BIT   8
80 /*  CTS flag                        */
81 #define MUSART_SR_CTS_BIT   9
82
83
84
85 /*********************************************/
86 /*              CR1 BITS Mapping             */
87 /*********************************************/
88 /*  Send break  bit */
89 #define MUSART_CR1_SBK_BIT  0
90 /*  Recevier Wakeup bit */
91 #define MUSART_CR1_RWU_BIT  1
92 /*  Recevier Enable bit */
93 #define MUSART_CR1_RE_BIT   2
94 /*  Transmitter Enable bit  */
95 #define MUSART_CR1_TE_BIT   3
96 /*  IDLE interrupt enable bit   */
97 #define MUSART_CR1_IDLEIE_BIT   4
98 /*  RXNEIE interrupt enable bit */
99 #define MUSART_CR1_RXNEIE_BIT   5
100 /*  Transmission complete interrupt enable bit  */
101 #define MUSART_CR1_TCIE_BIT     6
102 /*  TXE interrupt enable bit    */
103 #define MUSART_CR1_TXEIE_BIT    7
104 /*  PE interrupt enable bit */
105 #define MUSART_CR1_PEIE_BIT     8
106 /*  Parity selection bit    */
107 #define MUSART_CR1_PS_BIT       9
108 /*  Parity control enable bit   */
109 #define MUSART_CR1_PCE_BIT      10
110 /*  Wakeup method bit   */
111 #define MUSART_CR1_WAKE_BIT     11
112 /*  Word length bit */
113 #define MUSART_CR1_M_BIT        12
114 /*  USART enable bit    */
115 #define MUSART_CR1_UE_BIT       13
116 /*  USART Oversampling bit  */
117 #define MUSART_CR1_OVER8_BIT    15
118
119 /*********************************************/
120 /*              CR2 BITS Mapping             */
121 /*********************************************/
122 /*  Address of the USART node bits  */
123 #define MUSART_CR2_ADD0_BIT     0
124 #define MUSART_CR2_ADD1_BIT     1
125 #define MUSART_CR2_ADD2_BIT     2
126 #define MUSART_CR2_ADD3_BIT     3
127 /*  lin break detection length bit  */
128 #define MUSART_CR2_LBDL_BIT 5
129 /*  LIN break detection interrupt enable bit    */
130 #define MUSART_CR2_LBDIE_BIT        6
131 /*  Last bit clock pulse bit    */
132 #define MUSART_CR2_LBCL_BIT     8
133 /*  Clock phase bit */
134 #define MUSART_CR2_CPHA_BIT     9
```

```
135 /*  Clock polarity bit  */
136 #define MUSART_CR2_CPOL_BIT     10
137 /*  Clock enable bit    */
138 #define MUSART_CR2_CLKEN_BIT    11
139 /*  STOP bit start */
140 #define MUSART_CR2_STOP_BIT     12
141 /*  STOP bits   */
142 #define MUSART_CR2_STOP0_BIT    12
143 #define MUSART_CR2_STOP1_BIT    13
144 /*  LIN mode enable bit */
145 #define MUSART_CR2_LINEN_BIT    14
146
147
148 /*********************************************/
149 /*              CR3 BITS Mapping             */
150 /*********************************************/
151 /*  CTS interrupt enable bit    */
152 #define MUSART_CR3_CTSIE_BIT    10
153 /*  CTS enable bit  */
154 #define MUSART_CR3_CTSE_BIT     9
155 /*  RTS enable bit  */
156 #define MUSART_CR3_RTSE_BIT     8
157 /*  DMA enable transmitter bit  */
158 #define MUSART_CR3_DMAT_BIT     7
159 /*  DMA enable receiver bit */
160 #define MUSART_CR3_DMAR_BIT     6
161 /*  Smartcard mode enable bit   */
162 #define MUSART_CR3_SCEN_BIT     5
163 /*  Smartcard NACK enable bit   */
164 #define MUSART_CR3_NACK_BIT     4
165 /*  Half-duplex selection bit   */
166 #define MUSART_CR3_HDSEL_BIT    3
167 /*  IrDA low-power bit  */
168 #define MUSART_CR3_IRLP_BIT     2
169 /*  IrDA mode enable bit    */
170 #define MUSART_CR3_IREN_BIT     1
171 /*  Error interrupt enable bit  */
172 #define MUSART_CR3_EIE_BIT      0
173
174 #endif /* MCAL_USART_USART_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Boatloader/Inc/SERVICE/SPARSE_interface.h File Reference

Functions

- **u8 PARSE_u8AsciToHex** (**u8** Copy_u8Asci)
  Converts an ASCII character to its hexadecimal equivalent.

- void **PARSE_voidParseData** (**u8** *Copy_u8BufferData)
  Parses data in a buffer and performs a specific action.

- void **PARSE_voidParseRecord** (**u8** *Copy_u8BufferData)
  Parses a record in a buffer and performs a specific action.

Function Documentation

u8 PARSE_u8AsciToHex (u8 *Copy_u8Asci*)

Converts an ASCII character to its hexadecimal equivalent.

This function takes an ASCII character as input and returns its hexadecimal equivalent as an unsigned 8-bit integer.

Parameters

Copy_u8Asci	The ASCII character to be converted.

Returns

The hexadecimal equivalent as an unsigned 8-bit integer.

void PARSE_voidParseData (u8 * *Copy_u8BufferData*)

Parses data in a buffer and performs a specific action.

This function parses the data in the provided buffer and performs a specific action based on the content of the data.

Parameters

Copy_u8BufferData	Pointer to the buffer containing the data to be parsed.

void PARSE_voidParseRecord (u8 * *Copy_u8BufferData*)

Parses a record in a buffer and performs a specific action.

This function parses a record in the provided buffer and performs a specific action based on the content of the record.

Parameters

Copy_u8BufferData	Pointer to the buffer containing the record to be parsed.

SPARSE_interface.h

Go to the documentation of this file.

```
1
6 #ifndef MPARSE_INTERFACE_H_
7 #define MPARSE_INTERFACE_H_
8
18 u8 PARSE_u8AsciToHex(u8 Copy_u8Asci);
19
28 void PARSE_voidParseData(u8* Copy_u8BufferData);
29
38 void PARSE_voidParseRecord(u8* Copy_u8BufferData);
39
40 #endif
```

Index

INDEX