

Auto Sense

AUTHOR
Version

Table of Contents

Table of contents

Topic Index

Topics

Here is a list of all topics with brief descriptions:

ACC Configuration	7
Accelerated Adaptive Cruise Control (AACC) Module	8
Automatic Emergency Brake (AEB) Configuration	9
Automatic Emergency Brake (AEB) Interface	10
Body Control Module (BCM) Configuration	11
Body Control Module (BCM) Interface	13
Lane Keep Assistant (LKA) Configuration	14
Lane Keep Assistant (LKA) Interface	15
Normal Cruise Control (NCC) Configuration	17
Normal Cruise Control (NCC) Interface	18
User Dashboard Interface (UDI) Interface	19

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

ST_DCM_cfg_t	21
ST_HIR_cfg_t (Configuration structure for the Human Interface Receiver (HIR) module)	22
ST_MEXTI_RegistersMap_t	23
ST_MGPIO_altPinCfg_t	24
ST_MGPIO_pinCfg_t	25
ST_MGPIOx_RegistersMap_t	26
ST_MNVIC_RegistersMap_t	27
ST_MRCC_RegistersMap_t	29
ST_MSTK_RegistersMap_t	32
ST_MSYSCFG_RegistersMap_t	33
ST_MTMRx_RegistersMap_t	34
ST_MUART_RegistersMap_t	36
ST_MUSART_cfg_t (Structure for USART configuration)	37
ST_MUSART_clockInit_t (Structure for USART clock initialization)	39

File Index

File List

Here is a list of all files with brief descriptions:

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Adaptive_Cruise_Control/acc_config.h (Configuration file for Adaptive Cruise Control (ACC) in the application)	40
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Adaptive_Cruise_Control/acc_interface. h (Interfacing file for Adaptive Cruise Control (ACC) in the application)	42
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Automatic_Emergency_Brake/aeb_conf g.h (Configuration file for Automatic Emergency Brake (AEB) application)	44
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Automatic_Emergency_Brake/aeb_interf ace.h (Interface for Automatic Emergency Brake (AEB) application)	46
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Body_Control_Module/bcm_config.h (Configuration file for Body Control Module (BCM) in the application)	48
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Body_Control_Module/bcm_interface.h (Interface for Body Control Module (BCM) in the application)	50
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Lane_Keep_Assistant/LKA_config.h (Configuration file for Lane Keep Assistant (LKA) application)	52
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Lane_Keep_Assistant/LKA_interface.h (Interface for Lane Keep Assistant (LKA) in the application)	54
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Normal_Cruise_Control/ncc_config.h (Configuration file for Normal Cruise Control (NCC) in the Adaptive Cruise Control project)	56
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Normal_Cruise_Control/ncc_interface.h (Interface for Normal Cruise Control (NCC) in the Adaptive Cruise Control project)	58
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/User_Dashboard_Interface/udi_interface .h (Interface for User Dashboard in the Adaptive Cruise Control project)	60
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/COMMON/bit_math.h	62
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/COMMON/std_types.h	64
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/dcm/dcm_config.h	67
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/dcm/dcm_interface.h (Header file for the Direct Current Motor (DCM) module)	70
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/hc05/hc05_config.h	74
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/hc05/hc05_interface.h (Header file for the Bluetooth Module (HHC05) module)	76

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ir/IR_config.h	82
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ir/IR_interface.h (Header file for the Infrared Sensor (HIR) module)	84
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ultrasonic/Delay.h	87
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ultrasonic/ult_config.h	89
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ultrasonic/ult_interface.h	91
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/exti/exti_config.h	93
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/exti/exti_interface.h (Header file for the EXTI (External Interrupt) module interface)	96
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/exti/exti_private.h	101
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/gpio/gpio_config.h	103
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/gpio/gpio_interface.h (Header file for GPIO (General Purpose I/O) module)	109
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/gpio/gpio_private.h	117
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/nvic/nvic_config.h	119
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/nvic/nvic_interface.h (Header file for the NVIC (Nested Vectored Interrupt Controller) module interface)	122
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/nvic/nvic_private.h	126
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/rcc/rcc_config.h	128
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/rcc/rcc_interface.h (Header file for RCC (Reset and Clock Control) module)	136
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/rcc/rcc_private.h	139
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/systick/systick_config.h	145
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/systick/systick_interface.h (Header file for the SysTick (System Timer) module interface)	148
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/systick/systick_private.h	152
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/tmr/tmr_config.h	154
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/tmr/tmr_interface.h (Header file for the Multi-Mode Timer (MTMR) module)	158
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/tmr/tmr_private.h	166
D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/usart/usart_config.h	171

D:/Programing/Embedded System Diploma/ITI/grad	
doc/Adaptive_Cruise_Control/Inc/MCAL/usart/usart_interface.h (Header file for the	
Universal Synchronous/Asynchronous Receiver Transmitter (USART) module)	177
D:/Programing/Embedded System Diploma/ITI/grad	
doc/Adaptive_Cruise_Control/Inc/MCAL/usart/usart_private.h	184

Topic Documentation

ACC Configuration

Macros

- **#define RED_RANGE 0**
Red range definition for ACC.
- **#define BLUE_RANGE 20**
Blue range definition for ACC.
- **#define GREEN_RANGE 50**
Green range definition for ACC.

Detailed Description

Macro Definition Documentation

#define BLUE_RANGE 20

Blue range definition for ACC.

Define the blue range for the ACC system. This range may represent a specific condition or state.

#define GREEN_RANGE 50

Green range definition for ACC.

Define the green range for the ACC system. This range may represent a specific condition or state.

#define RED_RANGE 0

Red range definition for ACC.

Define the red range for the ACC system. This range may represent a specific condition or state.

Accelerated Adaptive Cruise Control (AACC) Module

Functions

- **void AACC_vSetSpeedLimit** (ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm, uint8_t copy_u8SpeedLimit)
Set speed limit for the AACC.
- **void AACC_vControllingCar** (ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm, uint32_t copy_u32CurrentDistance)
Control the car using AACC based on the current distance.
- **void AACC_vStopAcc** (ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm)
Stop the acceleration in the AACC.
- **uint8_t AACC_vChangeAccSpeedLimit** (uint8_t copy_u8SpeedAction)
Change the AACC acceleration speed limit.

Detailed Description

Function Documentation

uint8_t AACC_vChangeAccSpeedLimit (uint8_t copy_u8SpeedAction)

Change the AACC acceleration speed limit.

Parameters

<i>copy_u8SpeedAction</i>	Action to determine the speed limit change.
---------------------------	---------------------------------------------

Returns

New speed limit after the change.

void AACC_vControllingCar (ST_DCM_cfg_t * rightdcm, ST_DCM_cfg_t * leftdcm, uint32_t copy_u32CurrentDistance)

Control the car using AACC based on the current distance.

Parameters

<i>rightdcm</i>	Pointer to the configuration structure for the right DC motor.
<i>leftdcm</i>	Pointer to the configuration structure for the left DC motor.
<i>copy_u32CurrentDistance</i>	Current distance from the front obstacle.

```
void AACC_vSetSpeedLimit (ST_DCM_cfg_t * rightdcm, ST_DCM_cfg_t * leftdcm,
uint8_t copy_u8SpeedLimit)
```

Set speed limit for the AACC.

Parameters

<i>rightdcm</i>	Pointer to the configuration structure for the right DC motor.
<i>leftdcm</i>	Pointer to the configuration structure for the left DC motor.
<i>copy_u8SpeedLimit</i>	Speed limit to be set.

```
void AACC_vStopAcc (ST_DCM_cfg_t * rightdcm, ST_DCM_cfg_t * leftdcm)
```

Stop the acceleration in the AACC.

Parameters

<i>rightdcm</i>	Pointer to the configuration structure for the right DC motor.
<i>leftdcm</i>	Pointer to the configuration structure for the left DC motor.

Automatic Emergency Brake (AEB) Configuration

Macros

- `#define STOP_SPEED 1`
Speed value to indicate stopping in the AEB system.
- `#define DANGEROUS_ZONE 10`
Distance threshold for a dangerous zone in the AEB system.

Enumerations

- `enum EN_AAEB_zones_t { AAEB_SAFE_ZONE = 0, AAEB_DANGEROUS_ZONE }`
Enumeration representing different zones for AEB action.

Detailed Description

Macro Definition Documentation

```
#define DANGEROUS_ZONE 10
```

Distance threshold for a dangerous zone in the AEB system.

Define the distance at which the AEB system considers a zone as dangerous.

#define STOP_SPEED 1

Speed value to indicate stopping in the AEB system.

Define the speed at which the AEB system considers stopping.

Enumeration Type Documentation

enum EN_AAEB_zones_t

Enumeration representing different zones for AEB action.

Enumerator:

AAEB_SAFE_ZONE	Safe zone where no AEB action is required.
AAEB_DANGEROUS_ZONE	Dangerous zone where AEB action is needed.

Automatic Emergency Brake (AEB) Interface

Functions

- **void AAEB_vIsReady (void)**
Check if the AEB system is ready.
- **EN_AAEB_zones_t AAEB_uddtCheckForObstacles (ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm, uint32_t copy_u32CurrentDistance)**
Check for obstacles and determine the AEB action.

Detailed Description

Function Documentation

EN_AAEB_zones_t AAEB_uddtCheckForObstacles (ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm, uint32_t copy_u32CurrentDistance)

Check for obstacles and determine the AEB action.

This function checks for obstacles and determines the appropriate action to be taken by the AEB system.

Parameters

PS_uddtRightDcm	Pointer to the configuration structure for the right DC motor.
-----------------	----------------------------------------------------------------

<i>PS_uddtLeftDcm</i>	Pointer to the configuration structure for the left DC motor.
<i>copy_u32CurrentDistance</i>	Current distance from the front obstacle.

Returns

Enumeration representing the AEB action zones.

void AAEB_vIsReady (void)

Check if the AEB system is ready.

This function checks the readiness of the AEB system. It may perform any necessary initialization checks.

Body Control Module (BCM) Configuration

Enumerations

- enum **EN_ABCM_carStates_t** { **ABCM_CAR_STANDBY** = 0, **ABCM_CAR_ON**, **ABCM_CAR_NCC_ACTIVE**, **ABCM_CAR_ACC_SET**, **ABCM_CAR_ACC_ACTIVE**, **ABCM_CAR_GET_FAULT**, **ABCM_CAR_NCC_OFF**, **ABCM_CAR_ACC_OFF**, **ABCM_CAR_IDLE**, **ABCM_UPDATE_FIRMWARE**, **ABCM_CHANGE_SPEED_LIMIT** }
Enumeration representing different states of the car in BCM.
- enum **EN_ABCM_faultCodes_t** { **ABCM_FAULT_CAR_IS_ALREADY_ON** = 1, **ABCM_FAULT_NCC_IS_ALREADY_ACTIVE**, **ABCM_FAULT_ACC_IS_ALREADY_ACTIVE**, **ABCM_FAULT_CAR_IS_ALREADY_OFF**, **ABCM_FAULT_ACC_IS_ALREADY_OFF**, **ABCM_FAULT_NCC_IS_ALREADY_OFF**, **ABCM_FAULT_SPEED_RANGE_INVALID**, **ABCM_FAULT_ACC_NOR_NCC_IS_WORKING**, **ABCM_NO_FIRMWARE** }
Enumeration representing different fault codes in BCM.

Detailed Description

Enumeration Type Documentation

enum EN_ABCM_carStates_t

Enumeration representing different states of the car in BCM.

Enumerator:

ABCM_CAR_STANDBY	Car in standby state.
ABCM_CAR_ON	Car turned on.
ABCM_CAR_NCC_ACTIVE	Car with NCC (Non-Collision Control) active.
ABCM_CAR_ACC	Car with ACC (Adaptive Cruise Control) speed set.

C_SET	
ABCM_CAR_ACC_ACTIVE	Car with ACC (Adaptive Cruise Control) active.
ABCM_CAR_GET_FAULT	Car checking for faults.
ABCM_CAR_NCC_OFF	Car with NCC (Non-Collision Control) turned off.
ABCM_CAR_ACC_OFF	Car with ACC (Adaptive Cruise Control) turned off.
ABCM_CAR_IDLE	Car in idle state.
ABCM_UPDATE_FIRMWARE	Car updating firmware.
ABCM_CHANGE_SPEED_LIMIT	Car changing speed limit.

enum EN_ABCM_faultCodes_t

Enumeration representing different fault codes in BCM.

Enumerator:

ABCM_FAULT_CAR_IS_ALREADY_ON	
ABCM_FAULT_NCC_IS_ALREADY_ACTIVE	
ABCM_FAULT_ACC_IS_ALREADY_ACTIVE	
ABCM_FAULT_CAR_IS_ALREADY_OFF	
ABCM_FAULT_ACC_IS_ALREADY_OFF	
ABCM_FAULT_NCC_IS_ALREADY_OFF	
ABCM_FAULT_SPEED_RANGE_INVALID	
ABCM_FAULT_ACC_NOR_NCC_IS_WORKING	
ABCM_NO_FIRMWARE	

Body Control Module (BCM) Interface

Functions

- void **ABCM_vSysInit** (void)
Initialize the Body Control Module (BCM) system.
- void **ABCM_vSysMangement** (void)
Manage the Body Control Module (BCM) system.
- void **ABCM_vThreadMode** (void)
Execute the Body Control Module (BCM) in thread mode.
- **EN_ABCM_carStates_t** **ABCM_uddtDetermineCarState** (uint8_t copy_u8Action)
Determine the car state based on the given action.
- **EN_ABCM_carStates_t** **ABCM_uddtFaultDetection** (EN_ABCM_faultCodes_t copy_uddtFaultCode)
Detect faults in the Body Control Module (BCM) system.

Detailed Description

Function Documentation

EN_ABCM_carStates_t **ABCM_uddtDetermineCarState** (uint8_t *copy_u8Action*)

Determine the car state based on the given action.

This function determines the car state based on the provided action.

Parameters

<i>copy_u8Action</i>	Action to be considered for determining the car state.
----------------------	--------------------------------------------------------

Returns

Enumeration representing the determined car state.

EN_ABCM_carStates_t **ABCM_uddtFaultDetection** (EN_ABCM_faultCodes_t *copy_uddtFaultCode*)

Detect faults in the Body Control Module (BCM) system.

This function detects faults in the BCM system based on the provided fault code.

Parameters

<i>copy_uddtFaultCode</i>	Fault code to be used for fault detection.
---------------------------	--------------------------------------------

Returns

Enumeration representing the detected car state after fault detection.

void ABCM_vSysInit (void)

Initialize the Body Control Module (BCM) system.

This function initializes the necessary components and modules for the BCM system.

void ABCM_vSysMangment (void)

Manage the Body Control Module (BCM) system.

This function manages the overall operation of the BCM system. It includes handling different states and modes of the car.

void ABCM_vThreadMode (void)

Execute the Body Control Module (BCM) in thread mode.

This function represents the thread mode execution of the BCM system. It may perform tasks related to the thread-based functionality of the BCM.

Lane Keep Assistant (LKA) Configuration

Enumerations

- enum EN_ALKA_systeamState_t { ALKA_IN_LANE = 0, ALKA_OUT_LEFT_LANE, ALKA_OUT_RIGHT_LANE, ALKA_OUT_BOTH_LANE, ALKA_PTR_NULL, ALKA_POS_SET }

Enumeration representing different system states for Lane Keep Assistant (LKA).

Detailed Description

Enumeration Type Documentation

enum EN_ALKA_systeamState_t

Enumeration representing different system states for Lane Keep Assistant (LKA).

Enumerator:

ALKA_IN_LANE	Car is in the lane.
ALKA_OUT_LEF T_LANE	Car is out of the left lane.
ALKA_OUT_RIG HT_LANE	Car is out of the right lane.

ALKA_OUT_BOTH_LANE	Car is out of both lanes.
ALKA_PTR_NULL	Pointer is null (used for error handling).
ALKA_POS_SET	Lane position is set.

Lane Keep Assistant (LKA) Interface

Functions

- EN_ALKA_systemState_t ALKA_uddtGetLanePosition (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr)**
Get the lane position using IR sensors.
- EN_ALKA_systemState_t ALKA_uddtSetCarInLanes (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr, ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm)**
Set the car in lanes based on IR sensor readings and DC motor configurations.
- EN_ALKA_systemState_t ALKA_forward (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr, ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm)**
Move the car forward based on IR sensor readings and DC motor configurations.
- EN_ALKA_systemState_t ALKA_right (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr, ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm)**
Move the car to the right based on IR sensor readings and DC motor configurations.
- EN_ALKA_systemState_t ALKA_left (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr, ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm)**
Move the car to the left based on IR sensor readings and DC motor configurations.

Detailed Description

Function Documentation

EN_ALKA_systemState_t ALKA_forward (ST_HIR_cfg_t * PS_uddtRightIr, ST_HIR_cfg_t * PS_uddtLeftIr, ST_DCM_cfg_t * PS_uddtRightDcm, ST_DCM_cfg_t * PS_uddtLeftDcm)

Move the car forward based on IR sensor readings and DC motor configurations.

This function moves the car forward based on IR sensor readings and DC motor configurations.

Parameters

<i>PS_uddtRightIr</i>	Pointer to the configuration structure for the right IR sensor.
<i>PS_uddtLeftIr</i>	Pointer to the configuration structure for the left IR sensor.
<i>PS_uddtRightDcm</i>	Pointer to the configuration structure for the right DC motor.
<i>PS_uddtLeftDcm</i>	Pointer to the configuration structure for the left DC motor.

Returns

Enumeration representing the system state after moving the car forward.

EN_ALKA_systemState_t ALKA_left (ST_HIR_cfg_t * *PS_uddtRightIr*, ST_HIR_cfg_t * *PS_uddtLeftIr*, ST_DCM_cfg_t * *PS_uddtRightDcm*, ST_DCM_cfg_t * *PS_uddtLeftDcm*)

Move the car to the left based on IR sensor readings and DC motor configurations.

This function moves the car to the left based on IR sensor readings and DC motor configurations.

Parameters

<i>PS_uddtRightIr</i>	Pointer to the configuration structure for the right IR sensor.
<i>PS_uddtLeftIr</i>	Pointer to the configuration structure for the left IR sensor.
<i>PS_uddtRightDcm</i>	Pointer to the configuration structure for the right DC motor.
<i>PS_uddtLeftDcm</i>	Pointer to the configuration structure for the left DC motor.

Returns

Enumeration representing the system state after moving the car to the left.

EN_ALKA_systemState_t ALKA_right (ST_HIR_cfg_t * *PS_uddtRightIr*, ST_HIR_cfg_t * *PS_uddtLeftIr*, ST_DCM_cfg_t * *PS_uddtRightDcm*, ST_DCM_cfg_t * *PS_uddtLeftDcm*)

Move the car to the right based on IR sensor readings and DC motor configurations.

This function moves the car to the right based on IR sensor readings and DC motor configurations.

Parameters

<i>PS_uddtRightIr</i>	Pointer to the configuration structure for the right IR sensor.
<i>PS_uddtLeftIr</i>	Pointer to the configuration structure for the left IR sensor.
<i>PS_uddtRightDcm</i>	Pointer to the configuration structure for the right DC motor.
<i>PS_uddtLeftDcm</i>	Pointer to the configuration structure for the left DC motor.

Returns

Enumeration representing the system state after moving the car to the right.

EN_ALKA_systemState_t ALKA_uddtGetLanePosition (ST_HIR_cfg_t * *PS_uddtRightIr*, ST_HIR_cfg_t * *PS_uddtLeftIr*)

Get the lane position using IR sensors.

This function retrieves the lane position based on IR sensor readings.

Parameters

<i>PS_uddtRightIr</i>	Pointer to the configuration structure for the right IR sensor.
<i>PS_uddtLeftIr</i>	Pointer to the configuration structure for the left IR sensor.

Returns

Enumeration representing the system state based on lane position.

EN_ALKA_systemState_t ALKA_uddtSetCarInLanes (ST_HIR_cfg_t * PS_uddtRightIr, ST_HIR_cfg_t * PS_uddtLeftIr, ST_DCM_cfg_t * PS_uddtRightDcm, ST_DCM_cfg_t * PS_uddtLeftDcm)

Set the car in lanes based on IR sensor readings and DC motor configurations.

This function sets the car within lanes based on IR sensor readings and DC motor configurations.

Parameters

<i>PS_uddtRightIr</i>	Pointer to the configuration structure for the right IR sensor.
<i>PS_uddtLeftIr</i>	Pointer to the configuration structure for the left IR sensor.
<i>PS_uddtRightDcm</i>	Pointer to the configuration structure for the right DC motor.
<i>PS_uddtLeftDcm</i>	Pointer to the configuration structure for the left DC motor.

Returns

Enumeration representing the system state after setting the car in lanes.

Normal Cruise Control (NCC) Configuration

Macros

- **#define NCC_INCREMENT_SPEED +**
Macro to represent the speed increment operation in NCC.
- **#define NCC_DECREMENT_SPEED -**
Macro to represent the speed decrement operation in NCC.

Detailed Description

Macro Definition Documentation

#define NCC_DECREMENT_SPEED -

Macro to represent the speed decrement operation in NCC.

This macro defines the operation used for decrementing the speed in the NCC system.

#define NCC_INCREMENT_SPEED +

Macro to represent the speed increment operation in NCC.

This macro defines the operation used for incrementing the speed in the NCC system.

Normal Cruise Control (NCC) Interface

Macros

- `#define STOP_SPEED 1`
Speed value to indicate stopping in the NCC system.

Functions

- `void ANCC_vStartNcc (ST_DCM_cfg_t *leftdcm, ST_DCM_cfg_t *rightdcm, uint8_t copy_u8Speed)`
Start the Normal Cruise Control (NCC) system.
- `void ANCC_vStopNcc (ST_DCM_cfg_t *leftdcm, ST_DCM_cfg_t *rightdcm)`
Stop the Normal Cruise Control (NCC) system.
- `uint8_t ANCC_vChangeNccSpeedLimit (uint8_t copy_u8SpeedAction)`
Change the Normal Cruise Control (NCC) speed limit.

Detailed Description

Macro Definition Documentation

#define STOP_SPEED 1

Speed value to indicate stopping in the NCC system.

Define the speed at which the NCC system considers stopping.

Function Documentation

uint8_t ANCC_vChangeNccSpeedLimit (uint8_t *copy_u8SpeedAction*)

Change the Normal Cruise Control (NCC) speed limit.

Parameters

<i>copy_u8SpeedAction</i>	Action to determine the speed limit change.
---------------------------	---------------------------------------------

Returns

New speed limit after the change.

void ANCC_vStartNcc (ST_DCM_cfg_t * *lefthdcm*, ST_DCM_cfg_t * *rightdcm*, uint8_t *copy_u8Speed*)

Start the Normal Cruise Control (NCC) system.

Parameters

<i>lefthdcm</i>	Pointer to the configuration structure for the left DC motor.
<i>rightdcm</i>	Pointer to the configuration structure for the right DC motor.
<i>copy_u8Speed</i>	Speed to be set for the NCC system.

void ANCC_vStopNcc (ST_DCM_cfg_t * *lefthdcm*, ST_DCM_cfg_t * *rightdcm*)

Stop the Normal Cruise Control (NCC) system.

Parameters

<i>lefthdcm</i>	Pointer to the configuration structure for the left DC motor.
<i>rightdcm</i>	Pointer to the configuration structure for the right DC motor.

User Dashboard Interface (UDI) Interface

Functions

- void **AUDI_vInitInterface** (ST_MUART_RegistersMap_t *PS_USARTx, ST_MUSART_cfg_t const *PS_uddtUartCfg, void(*ptr)(void))
Initialize the User Dashboard Interface (UDI).
- void **AUDI_vStandByDashboard** (ST_MUART_RegistersMap_t *PS_USARTx)
Put the User Dashboard Interface (UDI) in standby mode.
- void **AUDI_vOnDashboard** (ST_MUART_RegistersMap_t *PS_USARTx)
Turn on the User Dashboard Interface (UDI).

Detailed Description

Function Documentation

void AUDI_vInitInterface (ST_MUART_RegistersMap_t * *PS_USARTx*, ST_MUSART_cfg_t const * *PS_uddtUartCfg*, void(*) (void) *ptr*)

Initialize the User Dashboard Interface (UDI).

Parameters

<i>PS_USARTx</i>	Pointer to the USART registers map.
<i>PS_uddtUartCfg</i>	Pointer to the USART configuration structure.
<i>ptr</i>	Pointer to the callback function.

void AUDI_vOnDashboard (ST_MUART_RegistersMap_t * *PS_USARTx*)

Turn on the User Dashboard Interface (UDI).

Parameters

<i>PS_USARTx</i>	Pointer to the USART registers map.
------------------	-------------------------------------

void AUDI_vStandByDashboard (ST_MUART_RegistersMap_t * *PS_USARTx*)

Put the User Dashboard Interface (UDI) in standby mode.

Parameters

<i>PS_USARTx</i>	Pointer to the USART registers map.
------------------	-------------------------------------

Data Structure Documentation

ST_DCM_cfg_t Struct Reference

```
#include <dcm_config.h>
```

Data Fields

- **ST_MGPIox_RegistersMap_t * DCM_5vPort**
 - **EN_MGPIIO_pinOptions_t DCM_5vPin**
 - **ST_MGPIox_RegistersMap_t * DCM_gndPort**
 - **EN_MGPIIO_pinOptions_t DCM_gndPin**
 - **EN_DCM_states_t DCM_intialState**
 - **EN_DCM_direction_t DCM_defaultDirection**
-

Field Documentation

EN_MGPIIO_pinOptions_t DCM_5vPin

ST_MGPIox_RegistersMap_t* DCM_5vPort

EN_DCM_direction_t DCM_defaultDirection

EN_MGPIIO_pinOptions_t DCM_gndPin

ST_MGPIox_RegistersMap_t* DCM_gndPort

EN_DCM_states_t DCM_intialState

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

- D:/Programing/Embedded System Diploma/ITI/grad
doc/Adaptive_Cruise_Control/Inc/HAL/dcm/**dcm_config.h**

ST_HIR_cfg_t Struct Reference

Configuration structure for the Human Interface Receiver (HIR) module.

```
#include <IR_config.h>
```

Data Fields

- **ST_MGPIOx_RegistersMap_t * HIR_port**
 - **EN_MGPIO_pinOptions_t HIR_pin**
-

Detailed Description

Configuration structure for the Human Interface Receiver (HIR) module.

This structure holds the configuration parameters for the HIR module.

Field Documentation

EN_MGPIO_pinOptions_t HIR_pin

Pin option for the HIR module.

ST_MGPIOx_RegistersMap_t* HIR_port

Pointer to the GPIO port for the HIR module.

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

- D:/Programing/Embedded System Diploma/ITI/grad
doc/Adaptive_Cruise_Control/Inc/HAL/ir/**IR_config.h**

ST_MEXTI_RegistersMap_t Struct Reference

```
#include <exti_private.h>
```

Data Fields

- `vuint32_t MEXTI_IMR`
 - `vuint32_t MEXTI_EMR`
 - `vuint32_t MEXTI_RTISR`
 - `vuint32_t MEXTI_FTSR`
 - `vuint32_t MEXTI_SWIER`
 - `vuint32_t MEXTI_PR`
-

Field Documentation

`vuint32_t MEXTI_EMR`

`vuint32_t MEXTI_FTSR`

`vuint32_t MEXTI_IMR`

`vuint32_t MEXTI_PR`

`vuint32_t MEXTI_RTISR`

`vuint32_t MEXTI_SWIER`

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

- `D:/Programing/Embedded System Diploma/ITI/grad
doc/Adaptive_Cruise_Control/Inc/MCAL/exti/exti_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/Adaptive_Cruise_Control/Inc/MCAL/gpio/gpio_config.h

ST_MGPIOPinCfg_t Struct Reference

```
#include <gpio_config.h>
```

Data Fields

- ST_MGPIOPinRegistersMap_t * PS_GPIOx
 - EN_MGPIOPinOptions_t copy_uddtPinNum
 - EN_MGPIOPinModeOptions_t copy_uddtPinMode
 - EN_MGPIOPinOutputResistorOptions_t copy_uddtOutputResistor
 - EN_MGPIOPinOutputSpeedOptions_t copy_uddtOutputSpeed
 - EN_MGPIOPinLogicOptions_t copy_uddtPtrRetOfPinLogic
 - EN_MGPIOPinPushPullOptions_t copy_uddtPullState
-

Field Documentation

EN_MGPIOPinOutputResistorOptions_t copy_uddtOutputResistor

EN_MGPIOPinOutputSpeedOptions_t copy_uddtOutputSpeed

EN_MGPIOPinModeOptions_t copy_uddtPinMode

EN_MGPIOPinOptions_t copy_uddtPinNum

EN_MGPIOPinLogicOptions_t copy_uddtPtrRetOfPinLogic

EN_MGPIOPinPushPullOptions_t copy_uddtPullState

ST_MGPIOPinRegistersMap_t* PS_GPIOx

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

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

ST_MGPIORx_RegistersMap_t Struct Reference

```
#include <gpio_private.h>
```

Data Fields

- `vuint32_t MGPIORx_MODER`
- `vuint32_t MGPIORx_OTYPER`
- `vuint32_t MGPIORx_OSPEEDR`
- `vuint32_t MGPIORx_PUPDR`
- `vuint32_t MGPIORx_IDR`
- `vuint32_t MGPIORx_ODR`
- `vuint32_t MGPIORx_BSRR`
- `vuint32_t MGPIORx_LCKR`
- `vuint32_t MGPIORx_AFRH`
- `vuint32_t MGPIORx_AFRH`

Field Documentation

`vuint32_t MGPIORx_AFRH`

`vuint32_t MGPIORx_AFRH`

`vuint32_t MGPIORx_BSRR`

`vuint32_t MGPIORx_IDR`

`vuint32_t MGPIORx_LCKR`

`vuint32_t MGPIORx_MODER`

`vuint32_t MGPIORx_ODR`

`vuint32_t MGPIORx_OSPEEDR`

`vuint32_t MGPIORx_OTYPER`

`vuint32_t MGPIORx_PUPDR`

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

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

ST_MNVIC_RegistersMap_t Struct Reference

```
#include <nvic_private.h>
```

Data Fields

- `vuint32_t MNVIC_ISERx` [8]
- `vuint32_t MNVIC_RESERVED0` [24]
- `vuint32_t MNVIC_ICERx` [8]
- `vuint32_t MNVIC_RESERVED1` [24]
- `vuint32_t MNVIC_ISPRx` [8]
- `vuint32_t MNVIC_RESERVED2` [24]
- `vuint32_t MNVIC_ICPRx` [8]
- `vuint32_t MNVIC_RESERVED3` [24]
- `vuint32_t MNVIC_IABRx` [8]
- `vuint32_t MNVIC_RESERVED4` [56]
- `vuint8_t MNVIC_IPRx` [240]
- `vuint32_t MNVIC_RESERVED5` [580]
- `vuint32_t MNVIC_STIR`

Field Documentation

`vuint32_t MNVIC_IABRx`[8]

`vuint32_t MNVIC_ICERx`[8]

`vuint32_t MNVIC_ICPRx`[8]

`vuint8_t MNVIC_IPRx`[240]

`vuint32_t MNVIC_ISERx`[8]

`vuint32_t MNVIC_ISPRx`[8]

`vuint32_t MNVIC_RESERVED0`[24]

`vuint32_t MNVIC_RESERVED1`[24]

`vuint32_t MNVIC_RESERVED2`[24]

`vuint32_t MNVIC_RESERVED3`[24]

`vuint32_t MNVIC_RESERVED4`[56]

`vuint32_t MNVIC_RESERVED5`[580]

`vuint32_t MNVIC_STIR`

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

- `D:/Programing/Embedded System Diploma/ITI/grad
doc/Adaptive_Cruise_Control/Inc/MCAL/nvic/nvic_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/Adaptive_Cruise_Control/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/Adaptive_Cruise_Control/Inc/MCAL/systick/systick_private.h`

ST_MSYS_CFG_RegistersMap_t Struct Reference

```
#include <exti_private.h>
```

Data Fields

- `vuint32_t MSYS_CFG_MEMRMP`
 - `vuint32_t MSYS_CFG_PMC`
 - `vuint32_t MSYS_CFG_EXTICR1`
 - `vuint32_t MSYS_CFG_EXTICR2`
 - `vuint32_t MSYS_CFG_EXTICR3`
 - `vuint32_t MSYS_CFG_EXTICR4`
 - `vuint32_t MSYS_CFG_CMPCR`
-

Field Documentation

`vuint32_t MSYS_CFG_CMPCR`

`vuint32_t MSYS_CFG_EXTICR1`

`vuint32_t MSYS_CFG_EXTICR2`

`vuint32_t MSYS_CFG_EXTICR3`

`vuint32_t MSYS_CFG_EXTICR4`

`vuint32_t MSYS_CFG_MEMRMP`

`vuint32_t MSYS_CFG_PMC`

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

- `D:/Programing/Embedded System Diploma/ITI/grad
doc/Adaptive_Cruise_Control/Inc/MCAL/exti/exti_private.h`

ST_MTMRx_RegistersMap_t Struct Reference

```
#include <tmr_private.h>
```

Data Fields

- **vuint32_t MTMRx_CR1**
 - **vuint32_t MTMRx_CR2**
 - **vuint32_t MTMRx_SMCR**
 - **vuint32_t MTMRx_DIER**
 - **vuint32_t MTMRx_SR**
 - **vuint32_t MTMRx_EGR**
 - **vuint32_t MTMRx_CCMR1**
 - **vuint32_t MTMRx_CCMR2**
 - **vuint32_t MTMRx_CCER**
 - **vuint32_t MTMRx_CNT**
 - **vuint32_t MTMRx_PSC**
 - **vuint32_t MTMRx_ARR**
 - **vuint32_t MTMRx_RESERVED_1**
 - **vuint32_t MTMRx_CCR1**
 - **vuint32_t MTMRx_CCR2**
 - **vuint32_t MTMRx_CCR3**
 - **vuint32_t MTMRx_CCR4**
 - **vuint32_t MTMRx_RESERVED_2**
 - **vuint32_t MTMRx_DCR**
 - **vuint32_t MTMRx_DMAR**
 - **vuint32_t MTMRx_OR**
-

Field Documentation

vuint32_t MTMRx_ARR

vuint32_t MTMRx_CCER

vuint32_t MTMRx_CCMR1

vuint32_t MTMRx_CCMR2

vuint32_t MTMRx_CCR1

vuint32_t MTMRx_CCR2

vuint32_t MTMRx_CCR3

vuint32_t MTMRx_CCR4

vuint32_t MTMRx_CNT

vuint32_t MTMRx_CR1

vuint32_t MTMRx_CR2

vuint32_t MTMRx_DCR

vuint32_t MTMRx_DIER

vuint32_t MTMRx_DMAR

vuint32_t MTMRx_EGR

vuint32_t MTMRx_OR

vuint32_t MTMRx_PSC

vuint32_t MTMRx_RESERVED_1

vuint32_t MTMRx_RESERVED_2

vuint32_t MTMRx_SMCR

vuint32_t MTMRx_SR

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

- D:/Programming/Embedded System Diploma/ITI/grad
doc/Adaptive_Cruise_Control/Inc/MCAL/tmr/**tmr_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/Adaptive_Cruise_Control/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/Adaptive_Cruise_Control/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/Adaptive_Cruise_Control/Inc/MCAL/usart/usart_config.h`

File Documentation

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Adaptive_Cru ise_Control/acc_config.h File Reference

Configuration file for Adaptive Cruise Control (ACC) in the application.

Macros

- `#define RED_RANGE 0`
Red range definition for ACC.
- `#define BLUE_RANGE 20`
Blue range definition for ACC.
- `#define GREEN_RANGE 50`
Green range definition for ACC.

Detailed Description

Configuration file for Adaptive Cruise Control (ACC) in the application.

acc_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : acc_config.h  
5 // Date        : Oct 23, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
13 #ifndef APPLICATION_ADAPTIVE_CRUISE_CONTROL_ACC_CONFIG_H_  
14 #define APPLICATION_ADAPTIVE_CRUISE_CONTROL_ACC_CONFIG_H_  
15  
26 #define RED_RANGE    0  
27  
33 #define BLUE_RANGE   20  
34  
40 #define GREEN_RANGE  50  
41  
// End of ACC Configuration group43  
44 #endif /* APPLICATION_ADAPTIVE_CRUISE_CONTROL_ACC_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Adaptive_Cruise_Control/acc_interface.h File Reference

Interfacing file for Adaptive Cruise Control (ACC) in the application.

```
#include "../HAL/dcm/dcm_interface.h"
#include "../HAL/hc05/hc05_interface.h"
#include "acc_config.h"
```

Functions

- void **AACC_vSetSpeedLimit** (ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm, uint8_t copy_u8SpeedLimit)
Set speed limit for the AACC.
- void **AACC_vControllingCar** (ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm, uint32_t copy_u32CurrentDistance)
Control the car using AACC based on the current distance.
- void **AACC_vStopAcc** (ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm)
Stop the acceleration in the AACC.
- uint8_t **AACC_vChangeAccSpeedLimit** (uint8_t copy_u8SpeedAction)
Change the AACC acceleration speed limit.

Detailed Description

Interfacing file for Adaptive Cruise Control (ACC) in the application.

acc_interface.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : acc_interface.h  
5 // Date        : Oct 23, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
13 #ifndef ACC_INTERFACE_H_  
14 #define ACC_INTERFACE_H_  
15  
16 #include "../HAL/dcm/dcm_interface.h"  
17 #include "../HAL/hc05/hc05_interface.h"  
18  
19 #include "acc_config.h"  
20  
33 void AACC_vSetSpeedLimit(ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm, uint8_t  
copy_u8SpeedLimit);  
34  
42 void AACC_vControllingCar(ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm, uint32_t  
copy_u32CurrentDistance);  
43  
50 void AACC_vStopAcc(ST_DCM_cfg_t *rightdcm, ST_DCM_cfg_t *leftdcm);  
51  
58 uint8_t AACC_vChangeAccSpeedLimit(uint8_t copy_u8SpeedAction);  
59  
// End of ACC_INTERFACE_H_ group61  
62 #endif
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Automatic_Emergency_Brake/aeb_config.h File Reference

Configuration file for Automatic Emergency Brake (AEB) application.

Macros

- `#define STOP_SPEED 1`
Speed value to indicate stopping in the AEB system.
- `#define DANGEROUS_ZONE 10`
Distance threshold for a dangerous zone in the AEB system.

Enumerations

- `enum EN_AAEB_zones_t { AAEB_SAFE_ZONE = 0, AAEB_DANGEROUS_ZONE }`
Enumeration representing different zones for AEB action.

Detailed Description

Configuration file for Automatic Emergency Brake (AEB) application.

aeb_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : aeb_config.h  
5 // Date        : Oct 26, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
13 #ifndef APPLICATION_AUTOMATIC_EMERGENCY_BRAKE_AEB_CONFIG_H_  
14 #define APPLICATION_AUTOMATIC_EMERGENCY_BRAKE_AEB_CONFIG_H_  
15  
25 #define STOP_SPEED 1  
26  
31 #define DANGEROUS_ZONE 10  
32  
36 typedef enum  
37 {  
38     AAEB_SAFE_ZONE = 0,  
39     AAEB_DANGEROUS_ZONE  
40 } EN_AAEB_zones_t;  
41  
42 // End of AEB Configuration group43  
44 #endif /* APPLICATION_AUTOMATIC_EMERGENCY_BRAKE_AEB_CONFIG_H_ */
```


D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Automatic_Emergency_Brake/aeb_interface.h File Reference

Interface for Automatic Emergency Brake (AEB) application.

```
#include "../..//HAL/dcm/dcm_interface.h"
#include "../..//HAL/hc05/hc05_interface.h"
#include "aeb_config.h"
```

Functions

- **void AAEB_vIsReady** (void)
Check if the AEB system is ready.
- **EN_AAEB_zones_t AAEB_uddtCheckForObstacles** (ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm, uint32_t copy_u32CurrentDistance)
Check for obstacles and determine the AEB action.

Detailed Description

Interface for Automatic Emergency Brake (AEB) application.

aeb_interface.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : aeb_interface.h  
5 // Date        : Oct_26, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
13 #ifndef APPLICATION_AUTOMATIC_EMERGENCY_BRAKE_AEB_INTERFACE_H_  
14 #define APPLICATION_AUTOMATIC_EMERGENCY_BRAKE_AEB_INTERFACE_H_  
15  
16 #include "../HAL/dcm/dcm_interface.h"  
17 #include "../HAL/hc05/hc05_interface.h"  
18 #include "aeb_config.h"  
19  
20 void AAEB_vIsReady(void);  
21  
22 EN_AAEB_zones_t AAEB_uddtCheckForObstacles(ST_DCM_cfg_t *PS_uddtRightDcm,  
23                                             ST_DCM_cfg_t *PS_uddtLeftDcm,  
24                                             uint32_t copy_u32CurrentDistance);  
25  
26 // End of AEB Interface group47  
27 #endif /* APPLICATION_AUTOMATIC_EMERGENCY_BRAKE_AEB_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Body_Control I_Module/bcm_config.h File Reference

Configuration file for Body Control Module (BCM) in the application.

Enumerations

- enum EN_ABCM_carStates_t { ABCM_CAR_STANDBY = 0, ABCM_CAR_ON, ABCM_CAR_NCC_ACTIVE, ABCM_CAR_ACC_SET, ABCM_CAR_ACC_ACTIVE, ABCM_CAR_GET_FAULT, ABCM_CAR_NCC_OFF, ABCM_CAR_ACC_OFF, ABCM_CAR_IDLE, ABCM_UPDATE_FIRMWARE, ABCM_CHANGE_SPEED_LIMIT }
Enumeration representing different states of the car in BCM.
 - enum EN_ABCM_faultCodes_t { ABCM_FAULT_CAR_IS_ALREADY_ON = 1, ABCM_FAULT_NCC_IS_ALREADY_ACTIVE, ABCM_FAULT_ACC_IS_ALREADY_ACTIVE, ABCM_FAULT_CAR_IS_ALREADY_OFF, ABCM_FAULT_ACC_IS_ALREADY_OFF, ABCM_FAULT_NCC_IS_ALREADY_OFF, ABCM_FAULT_SPEED_RANGE_INVALID, ABCM_FAULT_ACC_NOR_NCC_IS_WORKING, ABCM_NO_FIRMWARE }
Enumeration representing different fault codes in BCM.
-

Detailed Description

Configuration file for Body Control Module (BCM) in the application.

bcm_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Body_Control_Module  
4 // File        : bcm_config.h  
5 // Date        : Oct 18, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
13 #ifndef APPLICATION_BODY_CONTROL_MODULE_BCM_CONFIG_H_  
14 #define APPLICATION_BODY_CONTROL_MODULE_BCM_CONFIG_H_  
15  
24 typedef enum  
25 {  
26     ABCM_CAR_STANDBY = 0,  
27     ABCM_CAR_ON,  
28     ABCM_CAR_NCC_ACTIVE,  
29     ABCM_CAR_ACC_SET,  
30     ABCM_CAR_ACC_ACTIVE,  
31     ABCM_CAR_GET_FAULT,  
32     ABCM_CAR_NCC_OFF,  
33     ABCM_CAR_ACC_OFF,  
34     ABCM_CAR_IDLE,  
35     ABCM_UPDATE_FIRMWARE,  
36     ABCM_CHANGE_SPEED_LIMIT  
37 } EN_ABCM_carStates_t;  
38  
42 typedef enum  
43 {  
44     ABCM_FAULT_CAR_IS_ALREADY_ON = 1,  
45     ABCM_FAULT_NCC_IS_ALREADY_ACTIVE,  
46     ABCM_FAULT_ACC_IS_ALREADY_ACTIVE,  
47     ABCM_FAULT_CAR_IS_ALREADY_OFF,  
48     ABCM_FAULT_ACC_IS_ALREADY_OFF,  
49     ABCM_FAULT_NCC_IS_ALREADY_OFF,  
50     ABCM_FAULT_SPEED_RANGE_INVALID,  
51     ABCM_FAULT_ACC_NOR_NCC_IS_WORKING,  
52     ABCM_NO_FIRMWARE  
53 } EN_ABCM_faultCodes_t;  
54  
55 // End of BCM Configuration group56  
57 #endif /* APPLICATION_BODY_CONTROL_MODULE_BCM_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Body_Control I_Module/bcm_interface.h File Reference

Interface for Body Control Module (BCM) in the application.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "bcm_config.h"
#include "../..//COMMON/std_types.h"
#include "../..//MCAL/rcc/rcc_interface.h"
#include "../..//MCAL/gpio/gpio_interface.h"
#include "../..//MCAL/tmr/tmr_interface.h"
#include "../..//MCAL/nvic/nvic_interface.h"
#include "../..//HAL/ir/IR_interface.h"
#include "../..//HAL/ultrasonic/ult_interface.h"
#include "../..//MCAL/exti/exti_interface.h"
#include "../User_Dashboard_Interface/udi_interface.h"
#include "../Normal_Cruise_Control/ncc_interface.h"
#include "../Adaptive_Cruise_Control/acc_interface.h"
#include "../Automatic_Emergency_Brake/aeb_interface.h"
#include "../Lane_Keep_Assistant/LKA_interface.h"
```

Functions

- **void ABCM_vSysInit (void)**
Initialize the Body Control Module (BCM) system.
- **void ABCM_vSysMangment (void)**
Manage the Body Control Module (BCM) system.
- **void ABCM_vThreadMode (void)**
Execute the Body Control Module (BCM) in thread mode.
- **EN_ABCM_carStates_t ABCM_uddtDetermineCarState (uint8_t copy_u8Action)**
Determine the car state based on the given action.
- **EN_ABCM_carStates_t ABCM_uddtFaultDetection (EN_ABCM_faultCodes_t copy_uddtFaultCode)**
Detect faults in the Body Control Module (BCM) system.

Detailed Description

Interface for Body Control Module (BCM) in the application.

bcm_interface.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Body_Control_Module  
4 // File        : bcm_interface.h  
5 // Date        : Oct 17, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
13 #ifndef APPLICATION_BODY_CONTROL_MODULE_BCM_INTERFACE_H_  
14 #define APPLICATION_BODY_CONTROL_MODULE_BCM_INTERFACE_H_  
15  
16 #include <stdio.h>  
17 #include <stdlib.h>  
18 #include <string.h>  
19 #include "bcm_config.h"  
20 #include "../COMMON/std_types.h"  
21 #include "../MCAL/rcc/rcc_interface.h"  
22 #include "../MCAL/gpio/gpio_interface.h"  
23 #include "../MCAL/tmr/tmr_interface.h"  
24 #include "../MCAL/nvic/nvic_interface.h"  
25 #include "../HAL/ir/IR_interface.h"  
26 #include "../HAL/ultrasonic/ult_interface.h"  
27 #include "../MCAL/exti/exti_interface.h"  
28 #include "../User_Dashboard_Interface/udi_interface.h"  
29 #include "../Normal_Cruise_Control/ncc_interface.h"  
30 #include "../Adaptive_Cruise_Control/acc_interface.h"  
31 #include "../Automatic_Emergency_Brake/aeb_interface.h"  
32 #include "../Lane_Keep_Assistant/LKA_interface.h"  
33  
43 void ABCM_vSysInit(void);  
44  
50 void ABCM_vSysMangment(void);  
51  
57 void ABCM_vThreadMode(void);  
58  
66 EN_ABCM_carStates_t ABCM_uddtDetermineCarState(uint8_t copy_u8Action);  
67  
75 EN_ABCM_carStates_t ABCM_uddtFaultDetection(EN_ABCM_faultCodes_t  
copy_uddtFaultCode);  
76  
// End of BCM Interface group78  
79 #endif /* APPLICATION_BODY_CONTROL_MODULE_BCM_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Lane_Keep_ Assistant/LKA_config.h File Reference

Configuration file for Lane Keep Assistant (LKA) application.

Enumerations

- enum **EN_ALKA_systeamState_t** { **ALKA_IN_LANE** = 0, **ALKA_OUT_LEFT_LANE**, **ALKA_OUT_RIGHT_LANE**, **ALKA_OUT_BOTH_LANE**, **ALKA_PTR_NULL**, **ALKA_POS_SET** }

Enumeration representing different system states for Lane Keep Assistant (LKA).

Detailed Description

Configuration file for Lane Keep Assistant (LKA) application.

LKA_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : LKA_config.h  
5 // Date        : Nov 7, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
13 #ifndef APPLICATION_LANE_KEEP_ASSISTANT_LKA_CONFIG_H_  
14 #define APPLICATION_LANE_KEEP_ASSISTANT_LKA_CONFIG_H_  
15  
24 typedef enum  
25 {  
26     ALKA_IN_LANE = 0,  
27     ALKA_OUT_LEFT_LANE,  
28     ALKA_OUT_RIGHT_LANE,  
29     ALKA_OUT_BOTH_LANE,  
30     ALKA_PTR_NULL,  
31     ALKA_POS_SET  
32 } EN_ALKA_systemState_t;  
33  
34 // End of LKA Configuration group35  
36 #endif /* APPLICATION_LANE_KEEP_ASSISTANT_LKA_CONFIG_H_ */
```


D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Lane_Keep_ Assistant/LKA_interface.h File Reference

Interface for Lane Keep Assistant (LKA) in the application.

```
#include "../.../Inc/COMMON/std_types.h"
#include "../.../Inc/HAL/ir/IR_interface.h"
#include "../.../Inc/HAL/dcm/dcm_interface.h"
#include "LKA_config.h"
```

Functions

- **EN_ALKA_systeamState_t ALKA_uddtGetLanePosition (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr)**
Get the lane position using IR sensors.
- **EN_ALKA_systeamState_t ALKA_uddtSetCarInLanes (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr, ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm)**
Set the car in lanes based on IR sensor readings and DC motor configurations.
- **EN_ALKA_systeamState_t ALKA_forward (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr, ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm)**
Move the car forward based on IR sensor readings and DC motor configurations.
- **EN_ALKA_systeamState_t ALKA_right (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr, ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm)**
Move the car to the right based on IR sensor readings and DC motor configurations.
- **EN_ALKA_systeamState_t ALKA_left (ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t *PS_uddtLeftIr, ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t *PS_uddtLeftDcm)**
Move the car to the left based on IR sensor readings and DC motor configurations.

Detailed Description

Interface for Lane Keep Assistant (LKA) in the application.

LKA_interface.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : LKA_interface.h  
5 // Date        : Nov 7, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
13 #ifndef APPLICATION_LANE_KEEP_ASSISTANT_LKA_INTERFACE_H_  
14 #define APPLICATION_LANE_KEEP_ASSISTANT_LKA_INTERFACE_H_  
15  
16 #include "../Inc/COMMON/std_types.h"  
17 #include "../Inc/HAL/ir/IR_interface.h"  
18 #include "../Inc/HAL/dcm/dcm_interface.h"  
19 #include "LKA_config.h"  
20  
34 EN_ALKA_systemState_t ALKA_uddtGetLanePosition(ST_HIR_cfg_t *PS_uddtRightIr,  
ST_HIR_cfg_t *PS_uddtLeftIr);  
35  
46 EN_ALKA_systemState_t ALKA_uddtSetCarInLanes(ST_HIR_cfg_t *PS_uddtRightIr,  
ST_HIR_cfg_t *PS_uddtLeftIr,  
47                                     ST_DCM_cfg_t *PS_uddtRightDcm,  
ST_DCM_cfg_t *PS_uddtLeftDcm);  
48  
59 EN_ALKA_systemState_t ALKA_forward(ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t  
*PS_uddtLeftIr,  
60                                     ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t  
*PS_uddtLeftDcm);  
61  
72 EN_ALKA_systemState_t ALKA_right(ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t  
*PS_uddtLeftIr,  
73                                     ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t  
*PS_uddtLeftDcm);  
74  
85 EN_ALKA_systemState_t ALKA_left(ST_HIR_cfg_t *PS_uddtRightIr, ST_HIR_cfg_t  
*PS_uddtLeftIr,  
86                                     ST_DCM_cfg_t *PS_uddtRightDcm, ST_DCM_cfg_t  
*PS_uddtLeftDcm);  
87  
88 // End of LKA Interface group  
90 #endif /* APPLICATION_LANE_KEEP_ASSISTANT_LKA_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Normal_Cruise_Control/ncc_config.h File Reference

Configuration file for Normal Cruise Control (NCC) in the Adaptive Cruise Control project.

Macros

- `#define NCC_INCREMENT_SPEED +`
Macro to represent the speed increment operation in NCC.
- `#define NCC_DECREMENT_SPEED -`
Macro to represent the speed decrement operation in NCC.

Detailed Description

Configuration file for Normal Cruise Control (NCC) in the Adaptive Cruise Control project.

Author

Sherif Ashraf Khadr

Date

Oct 18, 2023

See also

<https://github.com/sherifkhadr>

ncc_config.h

Go to the documentation of this file.

```
1
10 #ifndef APPLICATION_NORMAL_CRUISE_CONTROL_NCC_CONFIG_H_
11 #define APPLICATION_NORMAL_CRUISE_CONTROL_NCC_CONFIG_H_
12
22 #define NCC_INCREMENT_SPEED +
23
28 #define NCC_DECREMENT_SPEED -
29
// End of NCC_Configuration group31
32 #endif /* APPLICATION_NORMAL_CRUISE_CONTROL_NCC_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/Normal_Cruise_Control/ncc_interface.h File Reference

Interface for Normal Cruise Control (NCC) in the Adaptive Cruise Control project.

```
#include "../HAL/dcm/dcm_interface.h"
#include "../HAL/hc05/hc05_interface.h"
#include "ncc_config.h"
```

Macros

- **#define STOP_SPEED 1**
Speed value to indicate stopping in the NCC system.

Functions

- **void ANCC_vStartNcc (ST_DCM_cfg_t *leftdcm, ST_DCM_cfg_t *rightdcm, uint8_t copy_u8Speed)**
Start the Normal Cruise Control (NCC) system.
- **void ANCC_vStopNcc (ST_DCM_cfg_t *leftdcm, ST_DCM_cfg_t *rightdcm)**
Stop the Normal Cruise Control (NCC) system.
- **uint8_t ANCC_vChangeNccSpeedLimit (uint8_t copy_u8SpeedAction)**
Change the Normal Cruise Control (NCC) speed limit.

Detailed Description

Interface for Normal Cruise Control (NCC) in the Adaptive Cruise Control project.

Author

Sherif Ashraf Khadr

Date

Oct 18, 2023

See also

<https://github.com/sherifkhadr>

ncc_interface.h

Go to the documentation of this file.

```
1
10 #ifndef APPLICATION_NORMAL_CRUISE_CONTROL_NCC_INTERFACE_H_
11 #define APPLICATION_NORMAL_CRUISE_CONTROL_NCC_INTERFACE_H_
12
13 #include "../../HAL/dcm/dcm_interface.h"
14 #include "../../HAL/hc05/hc05_interface.h"
15 #include "ncc_config.h"
16
26 #define STOP_SPEED 1
27
35 void ANCC_vStartNcc(ST_DCM_cfg_t *leftdcm, ST_DCM_cfg_t *rightdcm, uint8_t
copy_u8Speed);
36
43 void ANCC_vStopNcc(ST_DCM_cfg_t *leftdcm, ST_DCM_cfg_t *rightdcm);
44
51 uint8_t ANCC_vChangeNccSpeedLimit(uint8_t copy_u8SpeedAction);
52
// End of NCC_Interface group54
55 #endif /* APPLICATION_NORMAL_CRUISE_CONTROL_NCC_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/APPLICATION/User_Dashbo ard_Interface/udi_interface.h File Reference

Interface for User Dashboard in the Adaptive Cruise Control project.

```
#include "../HAL/hc05/hc05_interface.h"
```

Functions

- void **AUDI_vInitInterface** (ST_MUART_RegistersMap_t *PS_USARTx, ST_MUSART_cfg_t const *PS_uddtUartCfg, void(*ptr)(void))
Initialize the User Dashboard Interface (UDI).
- void **AUDI_vStandByDashboard** (ST_MUART_RegistersMap_t *PS_USARTx)
Put the User Dashboard Interface (UDI) in standby mode.
- void **AUDI_vOnDashboard** (ST_MUART_RegistersMap_t *PS_USARTx)
Turn on the User Dashboard Interface (UDI).

Detailed Description

Interface for User Dashboard in the Adaptive Cruise Control project.

Author

Sherif Ashraf Khadr

Date

Oct 18, 2023

See also

<https://github.com/sherifkhadr>

udi_interface.h

Go to the documentation of this file.

```
1
10 #ifndef APPLICATION_USER_DASHBOARD_INTERFACE_UDI_INTERFACE_H_
11 #define APPLICATION_USER_DASHBOARD_INTERFACE_UDI_INTERFACE_H_
12
13 #include "../../HAL/hc05/hc05_interface.h"
14
27 void AUDI_vInitInterface(ST_MUART_RegistersMap_t *PS_USARTx, ST_MUSART_cfg_t const
*PS_uddtUartCfg, void (*ptr)(void));
28
34 void AUDI_vStandByDashboard(ST_MUART_RegistersMap_t *PS_USARTx);
35
41 void AUDI_vOnDashboard(ST_MUART_RegistersMap_t *PS_USARTx);
42
// End of UDI_Interface group44
45 #endif /* APPLICATION_USER_DASHBOARD_INTERFACE_UDI_INTERFACE_H_ */
```


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

Macros

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

Typedefs

- `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 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 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          uint8_t      ;  
13 typedef unsigned short int     uint16_t     ;  
14 typedef unsigned long int      uint32_t     ;  
15 typedef signed char            sint8_t      ;  
16 typedef signed short int       sint16_t     ;  
17 typedef signed long int        sint32_t     ;  
18 typedef float                  float32_t    ;  
19 typedef double                  float64_t    ;  
20 typedef long double             float96_t    ;  
21  
22  
23 typedef volatile unsigned char  vuint8_t    ;  
24 typedef volatile unsigned short int vuint16_t ;  
25 typedef volatile unsigned long int vuint32_t ;  
26 typedef volatile signed char    vsint8_t    ;  
27 typedef volatile signed short int vsint16_t  ;  
28 typedef volatile signed long int vsint32_t  ;  
29 typedef volatile float          vfloat32_t   ;  
30 typedef volatile double          vfloat64_t   ;  
31 typedef volatile long double     vfloat96_t   ;  
32  
33  
34 #ifndef TRUE  
35 #define TRUE    1  
36 #endif  
37  
38  
39 #ifndef FALSE  
40 #define FALSE    0  
41 #endif  
42  
43  
44 #ifndef STR_NULL  
45 #define STR_NULL    '\0'  
46 #endif  
47  
48 #ifndef PTR_NULL  
49 #define PTR_NULL    (void*)0  
50 #endif  
51  
52  
53 #endif /* COMMON_STD_TYPES_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/dcm/dcm_config.h File Reference

Data Structures

struct ST_DCM_cfg_tEnumerations

- enum **EN_DCM_systemState_t** { **DCM_OK** = 0, **DCM_NOK** }
Enumeration for the system states of the DC motor.
 - enum **EN_DCM_states_t** { **DCM_TURN_OFF** = 0, **DCM_TURN_ON** }
Enumeration for the states of the DC motor (turn on or off).
 - enum **EN_DCM_direction_t** { **DCM_DIR_CLOCKWISE** = 0, **DCM_DIR_ANTI_CLOCKWISE** }
Enumeration for the direction of the DC motor.
 - enum **EN_DCM_rotation_t** { **DCM_ROTATE_LEFT** = 0, **DCM_ROTATE_RIGHT** }
Enumeration for the rotation of the DC motor.
-

Enumeration Type Documentation

enum EN_DCM_direction_t

Enumeration for the direction of the DC motor.

Enumerator:

DCM_DIR_CLOC KWISE	DC motor rotates in the clockwise direction.
DCM_DIR_ANTI _CLOCKWISE	DC motor rotates in the anti-clockwise direction.

enum EN_DCM_rotation_t

Enumeration for the rotation of the DC motor.

Enumerator:

DCM_ROTATE_ LEFT	DC motor rotates to the left.
DCM_ROTATE_ RIGHT	DC motor rotates to the right.

enum EN_DCM_states_t

Enumeration for the states of the DC motor (turn on or off).

Enumerator:

DCM_TURN_OFF	DC motor turned off.
DCM_TURN_ON	DC motor turned on.

enum EN_DCM_systemState_t

Enumeration for the system states of the DC motor.

Enumerator:

DCM_OK	DC motor operation successful.
DCM_NOK	DC motor operation failed.

dcm_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : dcm_config.h  
5 // Date        : Oct 17, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef HAL_DCM_DCM_CONFIG_H_  
9 #define HAL_DCM_DCM_CONFIG_H_  
10  
11  
12  
13  
14  
15 typedef enum  
16 {  
17     DCM_OK = 0,  
18     DCM_NOK  
19 } EN_DCM_systemState_t;  
20  
21  
22  
23  
24 typedef enum  
25 {  
26     DCM_TURN_OFF = 0,  
27     DCM_TURN_ON  
28 } EN_DCM_states_t;  
29  
30  
31  
32  
33 typedef enum  
34 {  
35     DCM_DIR_CLOCKWISE = 0,  
36     DCM_DIR_ANTI_CLOCKWISE  
37 } EN_DCM_direction_t;  
38  
39  
40  
41  
42 typedef enum  
43 {  
44     DCM_ROTATE_LEFT = 0,  
45     DCM_ROTATE_RIGHT  
46 } EN_DCM_rotation_t;  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59 #endif /* HAL_DCM_DCM_CONFIG_H_ */
```


D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/dcm/dcm_interface.h File Reference

Header file for the Direct Current Motor (DCM) module.

```
#include "../..//COMMON/std_types.h"
#include "../..//COMMON/bit_math.h"
#include "../..//MCAL/gpio/gpio_interface.h"
#include "../..//MCAL/tmr/tmr_interface.h"
#include "dcm_config.h"
```

Functions

- **EN_DCM_systemState_t HDCM_init (ST_DCM_cfg_t *dcmCfg)**
Initialize the DC motor.
- **EN_DCM_systemState_t HDCM_turnOff (ST_DCM_cfg_t *dcmCfg)**
Turn off the DC motor.
- **EN_DCM_systemState_t HDCM_turnOn (ST_DCM_cfg_t *dcmCfg)**
Turn on the DC motor.
- **EN_DCM_systemState_t HDCM_changeDirection (ST_DCM_cfg_t *dcmCfg, EN_DCM_direction_t dcmDirection)**
Change the direction of the DC motor.
- **EN_DCM_systemState_t HDCM_controlSpeed (uint8_t copy_u8Speed)**
Control the speed of the DC motor.

Detailed Description

Header file for the Direct Current Motor (DCM) module.

Function Documentation

EN_DCM_systemState_t HDCM_changeDirection (ST_DCM_cfg_t * *dcmCfg*, EN_DCM_direction_t *dcmDirection*)

Change the direction of the DC motor.

This function changes the direction of the DC motor.

Parameters

<i>dcmCfg</i>	Pointer to the configuration structure of the DC motor.
<i>dcmDirection</i>	The new direction for the DC motor.

Returns

The system state after changing the direction of the DC motor.

- **DCM_OK**: DC motor direction changed successfully.
- **DCM_NOK**: Failed to change the DC motor direction.

EN_DCM_systemState_t HDCM_controlSpeed (uint8_t *copy_u8Speed*)

Control the speed of the DC motor.

This function controls the speed of the DC motor.

Parameters

<i>copy_u8Speed</i>	The desired speed value for the DC motor.
---------------------	-------------------------------------------

Returns

The system state after controlling the speed of the DC motor.

- **DCM_OK**: DC motor speed controlled successfully.
- **DCM_NOK**: Failed to control the DC motor speed.

EN_DCM_systemState_t HDCM_init (ST_DCM_cfg_t * *dcmCfg*)

Initialize the DC motor.

This function initializes the DC motor based on the provided configuration.

Parameters

<i>dcmCfg</i>	Pointer to the configuration structure of the DC motor.
---------------	---------------------------------------------------------

Returns

The system state after initializing the DC motor.

- **DCM_OK**: DC motor initialization successful.
- **DCM_NOK**: DC motor initialization failed.

EN_DCM_systemState_t HDCM_turnOff (ST_DCM_cfg_t * *dcmCfg*)

Turn off the DC motor.

This function turns off the DC motor.

Parameters

<i>dcmCfg</i>	Pointer to the configuration structure of the DC motor.
---------------	---------------------------------------------------------

Returns

The system state after turning off the DC motor.

- **DCM_OK**: DC motor turned off successfully.
- **DCM_NOK**: Failed to turn off the DC motor.

EN_DCM_systemState_t HDCM_turnOn (ST_DCM_cfg_t * *dcmCfg*)

Turn on the DC motor.

This function turns on the DC motor.

Parameters

<i>dcmCfg</i>	Pointer to the configuration structure of the DC motor.
---------------	---------------------------------------------------------

Returns

The system state after turning on the DC motor.

- **DCM_OK**: DC motor turned on successfully.
- **DCM_NOK**: Failed to turn on the DC motor.

dcm_interface.h

Go to the documentation of this file.

```
1
7 #ifndef HAL_DCM_DCM_INTERFACE_H
8 #define HAL_DCM_DCM_INTERFACE_H
9
10 #include "../COMMON/std_types.h"
11 #include "../COMMON/bit_math.h"
12 #include "../MCAL/gpio/gpio_interface.h"
13 #include "../MCAL/tmr/tmr_interface.h"
14 #include "dcm_config.h"
15
27 EN_DCM_systemState_t HDCM_init(ST_DCM_cfg_t *dcmCfg);
28
40 EN_DCM_systemState_t HDCM_turnOff(ST_DCM_cfg_t *dcmCfg);
41
53 EN_DCM_systemState_t HDCM_turnOn(ST_DCM_cfg_t *dcmCfg);
54
67 EN_DCM_systemState_t HDCM_changeDirection(ST_DCM_cfg_t *dcmCfg, EN_DCM_direction_t
dcmDirection);
68
80 EN_DCM_systemState_t HDCM_controlSpeed(uint8_t copy_u8Speed);
81 #endif /* HAL_DCM_DCM_INTERFACE_H */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/hc05/hc05_config.h File Reference

Enumerations

- enum **EN_HHC05_systeamState_t** { **HHC05_NOK** = 0, **HHC05_OK**, **HHC05_PTR_NULL** }
Enumeration for the system states of the HHC05 module.

Enumeration Type Documentation

enum **EN_HHC05_systeamState_t**

Enumeration for the system states of the HHC05 module.

Enumerator:

HHC05_NOK	HHC05 module operation unsuccessful.
HHC05_OK	HHC05 module operation successful.
HHC05_PTR_NU LL	Null pointer encountered during the operation.

hc05_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : hc05_config.h  
5 // Date        : Oct 18, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef HAL_HC05_HC05_CONFIG_H_  
9 #define HAL_HC05_HC05_CONFIG_H_  
10  
11  
12  
13  
14 typedef enum  
15 {  
16     HHC05_NOK = 0,  
17     HHC05_OK,  
18     HHC05_PTR_NULL  
19 } EN_HHC05_systemState_t;  
20  
21 #endif /* HAL_HC05_HC05_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/hc05/hc05_interface.h File Reference

Header file for the Bluetooth Module (HHC05) module.

```
#include "../..//MCAL/usart/usart_interface.h"  
#include "hc05_config.h"
```

Functions

- **EN_HHC05_systeamState_t HHC05_uddtInit (ST_MUART_RegistersMap_t *PS_USARTx, ST_MUSART_cfg_t const *PS_uddtUartCfg)**
Initialize the HHC05 module using the specified UART configuration.
- **EN_HHC05_systeamState_t HHC05_uddtEnable (ST_MUART_RegistersMap_t *PS_USARTx)**
Enable the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_uddtDisable (ST_MUART_RegistersMap_t *PS_USARTx)**
Disable the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_uddtTransmitByte (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t copy_u8ByteToSend)**
Transmit a byte through the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_uddtTransmitString (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8StringToSend)**
Transmit a string through the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_uddtReadDataRegister (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8ByteToReceive)**
Read a byte from the HHC05 module data register.
- **EN_HHC05_systeamState_t HHC05_uddtReceiveByteSynchNonBlocking (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8ByteToReceive)**
Receive a byte synchronously (non-blocking) from the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_uddtReceiveStringSynchNonBlocking (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8ByteToReceive)**
Receive a string synchronously (non-blocking) from the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_uddtReceiveStringAsynchBlocking (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8ByteToReceive)**
Receive a string asynchronously (blocking) from the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_RxIntSetStatus (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t copy_u8Status)**
Set the status of the receive interrupt for the HHC05 module.

- **EN_HHC05_systeamState_t HHC05_uddtSetCallBackUart1** (void(*ptr)(void))
Set the callback function for UART1 communication with the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_uddtSetCallBackUart2** (void(*ptr)(void))
Set the callback function for UART2 communication with the HHC05 module.
- **EN_HHC05_systeamState_t HHC05_uddtSetCallBackUart6** (void(*ptr)(void))
Set the callback function for UART6 communication with the HHC05 module.

Detailed Description

Header file for the Bluetooth Module (HHC05) module.

Function Documentation

EN_HHC05_systeamState_t HHC05_RxIntSetStatus (ST_MUART_RegistersMap_t * PS_USARTx, uint8_t copy_u8Status)

Set the status of the receive interrupt for the HHC05 module.

This function sets the status of the receive interrupt for the HHC05 module.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
<i>copy_u8Status</i>	The status to set for the receive interrupt.

Returns

The system state after setting the interrupt status.

- **HHC05_OK**: Interrupt status setting successful.
- **HHC05_NOK**: Interrupt status setting failed.

EN_HHC05_systeamState_t HHC05_uddtDisable (ST_MUART_RegistersMap_t * PS_USARTx)

Disable the HHC05 module.

This function disables the HHC05 module.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
------------------	------------------------------------

Returns

The system state after disabling the HHC05 module.

- **HHC05_OK**: HHC05 module disabling successful.
- **HHC05_NOK**: HHC05 module disabling failed.

EN_HHC05_systeamState_t HHC05_uddtEnable (ST_MUART_RegistersMap_t * PS_USARTx)

Enable the HHC05 module.

This function enables the HHC05 module.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
------------------	------------------------------------

Returns

The system state after enabling the HHC05 module.

- **HHC05_OK**: HHC05 module enabling successful.
- **HHC05_NOK**: HHC05 module enabling failed.

**EN_HHC05_systemState_t HHC05_uddtInit (ST_MUART_RegistersMap_t *
PS_USARTx, ST_MUSART_cfg_t const * PS_uddtUartCfg)**

Initialize the HHC05 module using the specified UART configuration.

This function initializes the HHC05 module using the provided UART configuration.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
<i>PS_uddtUartCfg</i>	Pointer to the UART configuration structure.

Returns

The system state after initializing the HHC05 module.

- **HHC05_OK**: HHC05 module initialization successful.
- **HHC05_NOK**: HHC05 module initialization failed.
- **HHC05_PTR_NULL**: Null pointer encountered during the operation.

**EN_HHC05_systemState_t HHC05_uddtReadDataRegister
(ST_MUART_RegistersMap_t * PS_USARTx, uint8_t * copy_u8ByteToReceive)**

Read a byte from the HHC05 module data register.

This function reads a byte from the HHC05 module data register.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
<i>copy_u8ByteToReceive</i>	Pointer to store the received byte.

Returns

The system state after reading the byte.

- **HHC05_OK**: Byte reading successful.
- **HHC05_NOK**: Byte reading failed.

**EN_HHC05_systemState_t HHC05_uddtReceiveByteSynchNonBlocking
(ST_MUART_RegistersMap_t * PS_USARTx, uint8_t * copy_u8ByteToReceive)**

Receive a byte synchronously (non-blocking) from the HHC05 module.

This function receives a byte synchronously (non-blocking) from the HHC05 module.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
<i>copy_u8ByteToReceive</i>	Pointer to store the received byte.

Returns

The system state after receiving the byte.

- **HHC05_OK**: Byte reception successful.

- **HHC05_NOK**: Byte reception failed.

EN_HHC05_systemState_t HHC05_uddtReceiveStringAsynchBlocking
(ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t * *copy_u8ByteToReceive*)

Receive a string asynchronously (blocking) from the HHC05 module.

This function receives a string asynchronously (blocking) from the HHC05 module.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
<i>copy_u8ByteToReceive</i>	Pointer to store the received string.

Returns

The system state after receiving the string.

- **HHC05_OK**: String reception successful.
- **HHC05_NOK**: String reception failed.

EN_HHC05_systemState_t HHC05_uddtReceiveStringSynchNonBlocking
(ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t * *copy_u8ByteToReceive*)

Receive a string synchronously (non-blocking) from the HHC05 module.

This function receives a string synchronously (non-blocking) from the HHC05 module.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
<i>copy_u8ByteToReceive</i>	Pointer to store the received string.

Returns

The system state after receiving the string.

- **HHC05_OK**: String reception successful.
- **HHC05_NOK**: String reception failed.

EN_HHC05_systemState_t HHC05_uddtSetCallBackUart1 (void*)(void) ptr

Set the callback function for UART1 communication with the HHC05 module.

This function sets the callback function for UART1 communication with the HHC05 module.

Parameters

<i>ptr</i>	Pointer to the callback function.
------------	-----------------------------------

Returns

The system state after setting the callback function.

- **HHC05_OK**: Callback function setting successful.
- **HHC05_NOK**: Callback function setting failed.

EN_HHC05_systemState_t HHC05_uddtSetCallBackUart2 (void*)(void) ptr

Set the callback function for UART2 communication with the HHC05 module.

This function sets the callback function for UART2 communication with the HHC05 module.

Parameters

<i>ptr</i>	Pointer to the callback function.
------------	-----------------------------------

Returns

The system state after setting the callback function.

- **HHC05_OK**: Callback function setting successful.
- **HHC05_NOK**: Callback function setting failed.

EN_HHC05_systemState_t HHC05_uddtSetCallBackUart6 (void*)(void) *ptr*

Set the callback function for UART6 communication with the HHC05 module.

This function sets the callback function for UART6 communication with the HHC05 module.

Parameters

<i>ptr</i>	Pointer to the callback function.
------------	-----------------------------------

Returns

The system state after setting the callback function.

- **HHC05_OK**: Callback function setting successful.
- **HHC05_NOK**: Callback function setting failed.

EN_HHC05_systemState_t HHC05_uddtTransmitByte (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t *copy_u8ByteToSend*)

Transmit a byte through the HHC05 module.

This function transmits a byte through the HHC05 module.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
<i>copy_u8ByteToSend</i>	The byte to transmit.

Returns

The system state after transmitting the byte.

- **HHC05_OK**: Byte transmission successful.
- **HHC05_NOK**: Byte transmission failed.

EN_HHC05_systemState_t HHC05_uddtTransmitString (ST_MUART_RegistersMap_t * *PS_USARTx*, uint8_t * *copy_u8StringToSend*)

Transmit a string through the HHC05 module.

This function transmits a string through the HHC05 module.

Parameters

<i>PS_USARTx</i>	Pointer to the UART registers map.
<i>copy_u8StringToSend</i>	The string to transmit.

Returns

The system state after transmitting the string.

- **HHC05_OK**: String transmission successful.
- **HHC05_NOK**: String transmission failed.

hc05_interface.h

Go to the documentation of this file.

```
1
2
3
4
5
6 #ifndef HAL_HC05_HC05_INTERFACE_H_
7 #define HAL_HC05_HC05_INTERFACE_H_
8
9 #include "../MCAL/usart/usart_interface.h"
10 #include "hc05_config.h"
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26 EN_HHC05_systemState_t HHC05_uddtInit(ST_MUART_RegistersMap_t *PS_USARTx,
27 ST_MUSART_cfg_t const *PS_uddtUartCfg);
28
29
30
31
32
33
34
35
36
37
38
39 EN_HHC05_systemState_t HHC05_uddtEnable(ST_MUART_RegistersMap_t *PS_USARTx);
40
41
42
43
44
45
46
47
48
49
50
51
52 EN_HHC05_systemState_t HHC05_uddtDisable(ST_MUART_RegistersMap_t *PS_USARTx);
53
54
55
56
57
58
59
60
61
62
63
64
65
66 EN_HHC05_systemState_t HHC05_uddtTransmitByte(ST_MUART_RegistersMap_t *PS_USARTx,
67 uint8_t copy_u8ByteToSend);
68
69
70
71
72
73
74
75
76
77
78
79
80 EN_HHC05_systemState_t HHC05_uddtTransmitString(ST_MUART_RegistersMap_t *PS_USARTx,
81 uint8_t *copy_u8StringToSend);
82
83
84
85
86
87
88
89
90
91
92
93
94 EN_HHC05_systemState_t HHC05_uddtReadDataRegister(ST_MUART_RegistersMap_t
95 *PS_USARTx, uint8_t *copy_u8ByteToReceive);
96
97
98
99
100
101
102
103
104
105
106
107
108 EN_HHC05_systemState_t
109 HHC05_uddtReceiveByteSynchNonBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
110 *copy_u8ByteToReceive);
111
112
113
114
115
116
117
118
119
120
121
122 EN_HHC05_systemState_t
123 HHC05_uddtReceiveStringSynchNonBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
124 *copy_u8ByteToReceive);
125
126
127
128
129
130
131
132
133
134
135
136 EN_HHC05_systemState_t
137 HHC05_uddtReceiveStringAsynchBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
138 *copy_u8ByteToReceive);
139
140
141
142
143
144
145
146
147
148
149
150 EN_HHC05_systemState_t HHC05_RxIntSetStatus(ST_MUART_RegistersMap_t *PS_USARTx,
151 uint8_t copy_u8Status);
152
153
154
155
156
157
158
159
160
161
162
163 EN_HHC05_systemState_t HHC05_uddtSetCallBackUart1(void (*ptr)(void));
164
165
166
167
168
169
170
171
172
173
174
175
176 EN_HHC05_systemState_t HHC05_uddtSetCallBackUart2(void (*ptr)(void));
177
178
179
180
181
182
183
184
185
186
187
188
189 EN_HHC05_systemState_t HHC05_uddtSetCallBackUart6(void (*ptr)(void));
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193 #endif /* HAL_HC05_HC05_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ir/IR_config.h File Reference

Data Structures

struct **ST_HIR_cfg_t** *Configuration structure for the Human Interface Receiver (HIR) module.*

Enumerations

- enum **EN_HIR_systemState_t** { **HIR_NOK** = 0, **HIR_OK**, **HIR_PTR_NULL** }
Enumeration for the system state of the Human Interface Receiver (HIR) module.
 - enum **EN_HIR_surfaceState_t** { **HIR_ON_WHITE** = 0, **HIR_ON_BLACK** }
Enumeration for the surface state of the Human Interface Receiver (HIR) module.
-

Enumeration Type Documentation

enum **EN_HIR_surfaceState_t**

Enumeration for the surface state of the Human Interface Receiver (HIR) module.

This enumeration defines possible surface states for the HIR module.

Enumerator:

HIR_ON_WHITE	HIR module is on a white surface.
HIR_ON_BLACK	HIR module is on a black surface.

enum **EN_HIR_systemState_t**

Enumeration for the system state of the Human Interface Receiver (HIR) module.

This enumeration defines possible system states for the HIR module.

Enumerator:

HIR_NOK	HIR module encountered an error.
HIR_OK	HIR module operation successful.
HIR_PTR_NULL	Null pointer encountered during the operation.

IR_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : IR_config.h  
5 // Date        : Nov 7, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef HAL_IR_IR_CONFIG_H_  
9 #define HAL_IR_IR_CONFIG_H_  
10  
11  
12  
13  
14  
15  
16 typedef enum  
17 {  
18     HIR_NOK = 0,  
19     HIR_OK,  
20     HIR_PTR_NULL  
21 } EN_HIR_systemState_t;  
22  
23  
24  
25  
26  
27  
28 typedef enum  
29 {  
30     HIR_ON_WHITE = 0,  
31     HIR_ON_BLACK  
32 } EN_HIR_surfaceState_t;  
33  
34  
35  
36  
37  
38  
39 typedef struct  
40 {  
41     ST_MGPIox_RegistersMap_t *HIR_port;  
42     EN_MGPIIO_pinOptions_t    HIR_pin;  
43 } ST_HIR_cfg_t;  
44  
45 #endif /* HAL_IR_IR_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ir/IR_interface.h File Reference

Header file for the Infrared Sensor (HIR) module.

```
#include "../.../Inc/COMMON/std_types.h"
#include "../.../Inc/MCAL/gpio/gpio_interface.h"
#include "IR_config.h"
```

Functions

- **EN_HIR_systemState_t HIR_uddtInit (ST_HIR_cfg_t *PS_uddtIrInstance)**
Initialize the Human Interface Receiver (HIR) module.
- **EN_HIR_systemState_t HIR_uddtReadData (ST_HIR_cfg_t *PS_uddtIrInstance, EN_HIR_surfaceState_t *copy_uddtRetOfIrRead)**
Read data from the Human Interface Receiver (HIR) module.

Detailed Description

Header file for the Infrared Sensor (HIR) module.

Function Documentation

EN_HIR_systemState_t HIR_uddtInit (ST_HIR_cfg_t * PS_uddtIrInstance)

Initialize the Human Interface Receiver (HIR) module.

This function initializes the HIR module based on the provided configuration.

Parameters

<i>PS_uddtIrInstance</i>	Pointer to the configuration structure for the HIR module.
--------------------------	------------------------------------------------------------

Returns

The system state after initializing the HIR module.

- **HIR_OK**: HIR module initialization successful.
- **HIR_NOK**: HIR module initialization failed.
- **HIR_PTR_NULL**: Null pointer encountered during the operation.

EN_HIR_systemState_t HIR_uddtReadData (ST_HIR_cfg_t * PS_uddtIrInstance, EN_HIR_surfaceState_t * copy_uddtRetOfIrRead)

Read data from the Human Interface Receiver (HIR) module.

This function reads data from the HIR module and provides the surface state.

Parameters

<i>PS_uddtIrInstance</i>	Pointer to the configuration structure for the HIR module.
<i>copy_uddtRetOfIrRead</i>	Pointer to store the retrieved surface state.

Returns

The system state after reading data from the HIR module.

- **HIR_OK**: Data reading successful.
- **HIR_NOK**: Data reading failed.
- **HIR_PTR_NULL**: Null pointer encountered during the operation.

IR_interface.h

Go to the documentation of this file.

```
1
6 #ifndef HAL_IR_IR_INTERFACE_H_
7 #define HAL_IR_IR_INTERFACE_H_
8
9 #include "../Inc/COMMON/std_types.h"
10 #include "../Inc/MCAL/gpio/gpio_interface.h"
11 #include "IR_config.h"
12
25 EN_HIR_systemState_t HIR_uddtInit(ST_HIR_cfg_t *PS_uddtIrInstance);
26
40 EN_HIR_systemState_t HIR_uddtReadData(ST_HIR_cfg_t *PS_uddtIrInstance,
EN_HIR_surfaceState_t *copy_uddtRetOfIrRead);
41 #endif /* HAL_IR_IR_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ultrasonic/Delay.h File Reference

```
#include "../..//COMMON/std_types.h"
```

Functions

- void **delay_us** (uint32_t microseconds)

Function Documentation

void **delay_us** (uint32_t *microseconds*)

Delay.h

Go to the documentation of this file.

```
1 /*
2  * Delay.h
3  *
4  * Created on: Nov 7, 2023
5  * Author: Omar Abouzaid
6  */
7
8 #ifndef DELAY_H_
9 #define DELAY_H_
10
11 #include "../COMMON/std_types.h"
12
13 void delay_us(uint32_t microseconds);
14
15 #endif /* DELAY_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ultrasonic/ult_config.h File Reference

Macros

- `#define TRIG_PORT MGPIOA_PERIPHERAL`
 - `#define TRIG_PIN MGPIO_PIN9`
 - `#define ECHO_PORT MGPIOA_PERIPHERAL`
 - `#define ECHO_PIN MGPIO_PIN10`
-

Macro Definition Documentation

`#define ECHO_PIN MGPIO_PIN10`

`#define ECHO_PORT MGPIOA_PERIPHERAL`

`#define TRIG_PIN MGPIO_PIN9`

`#define TRIG_PORT MGPIOA_PERIPHERAL`

ult_config.h

Go to the documentation of this file.

```
1 /*
2  * ULTRASONIC_Config.h
3  *
4  * Created on: Nov 7, 2023
5  * Author: Omar Abouzaid
6  */
7
8 #ifndef SERVO_CONFIG_H_
9 #define SERVO_CONFIG_H_
10
11
12 /*TRIG PIN CONFIG*/
13 #define TRIG_PORT    MGPIOA_PERIPHERAL
14 #define TRIG_PIN     MGPIO_PIN9
15
16
17 /*ECHO PIN CONFIG*/
18 #define ECHO_PORT    MGPIOA_PERIPHERAL
19 #define ECHO_PIN     MGPIO_PIN10
20
21
22 #endif
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/HAL/ultrasonic/ult_interface .h File Reference

Functions

- void **UltraSonic_Init** (void)
• void **UltraSonic_Send_Pulse** (void)
Send a pulse on the Trig pin.
- **uint32_t UltraSonic_Measure_Distance** (void)
Measure the distance using the Ultrasonic sensor.

Function Documentation

void UltraSonic_Init (void)

uint32_t UltraSonic_Measure_Distance (void)

Measure the distance using the Ultrasonic sensor.

This function measures the distance using the Ultrasonic sensor. It returns the calculated distance value in centimeters.

Returns

The measured distance in centimeters.

void UltraSonic_Send_Pulse (void)

Send a pulse on the Trig pin.

This function triggers the Ultrasonic sensor to send a pulse on its Trig pin. It is used to initiate the distance measurement process.

Returns

No return.

ult_interface.h

Go to the documentation of this file.

```
1 /*
2  * ULTRASONIC_Interface.h
3  *
4  * Created on: Nov 7, 2023
5  * Author: Omar Abouzaid
6  */
7
8 #ifndef ULTRASONIC_INTERFACE_H_
9 #define ULTRASONIC_INTERFACE_H_
10
11 void UltraSonic_Init(void);
12
21 void UltraSonic_Send_Pulse(void);
22
31 uint32_t UltraSonic_Measure_Distance(void);
32
33 #endif /* ULTRASONIC_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/exti/exti_config.h File Reference

Enumerations

- enum EN_MEXTI_systemState_t { MEXTI_OK = 0, MEXTI_NOK, MEXTI_INVALID_PARAMTER }
Enumeration for the system state of EXTI functions.
- enum EN_MEXTI_triggerOptions_t { MEXTI_RISING_EDGE = 0, MEXTI_FALLING_EDGE, MEXTI_ON_CHANGE }
- enum EN_MEXTI_lines_t { MEXTI_LINE0 = 0, MEXTI_LINE1, MEXTI_LINE2, MEXTI_LINE3, MEXTI_LINE4, MEXTI_LINE5, MEXTI_LINE6, MEXTI_LINE7, MEXTI_LINE8, MEXTI_LINE9, MEXTI_LINE10, MEXTI_LINE11, MEXTI_LINE12, MEXTI_LINE13, MEXTI_LINE14, MEXTI_LINE15, MEXTI_LINE16, MEXTI_LINE17, MEXTI_LINE18, MEXTI_LINE19, MEXTI_LINE20, MEXTI_LINE21, MEXTI_LINE22 }
- enum EN_MEXTI_port_t { MEXTI_PORTA = 0, MEXTI_PORTB, MEXTI_PORTC, MEXTI_PORTD, MEXTI_PORTE, MEXTI_PORTH }

Enumeration Type Documentation

enum EN_MEXTI_lines_t

Enumerator:

MEXTI_LINE0	
MEXTI_LINE1	
MEXTI_LINE2	
MEXTI_LINE3	
MEXTI_LINE4	
MEXTI_LINE5	
MEXTI_LINE6	
MEXTI_LINE7	
MEXTI_LINE8	
MEXTI_LINE9	
MEXTI_LINE10	
MEXTI_LINE11	
MEXTI_LINE12	
MEXTI_LINE13	
MEXTI_LINE14	
MEXTI_LINE15	
MEXTI_LINE16	
MEXTI_LINE17	
MEXTI_LINE18	
MEXTI_LINE19	
MEXTI_LINE20	
MEXTI_LINE21	
MEXTI_LINE22	

enum EN_MEXTI_port_t

Enumerator:

MEXTI_PORTA	
-------------	--

MEXTI_PORTB	
MEXTI_PORTC	
MEXTI_PORTD	
MEXTI_PORTE	
MEXTI_PORTH	

enum EN_MEXTI_systemState_t

Enumeration for the system state of EXTI functions.

Enumerator:

MEXTI_OK	Operation successful.
MEXTI_NOK	Operation failed.
MEXTI_INVALID_PARAMETER	Invalid parameter detected.

enum EN_MEXTI_triggerOptions_t

Enumerator:

MEXTI_RISING_EDGE	
MEXTI_FALLING_EDGE	
MEXTI_ON_CHANGE	

exti_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : STM32F401xC  
4 // File        : exti_config.h  
5 // Date        : Sep 11, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef MCAL_EXTI_EXTI_CONFIG_H_  
9 #define MCAL_EXTI_EXTI_CONFIG_H_  
10  
11  
12  
13  
14  
15  
16 typedef enum  
17 {  
18     MEXTI_OK = 0,  
19     MEXTI_NOK,  
20     MEXTI_INVALID_PARAMTER  
21 } EN_MEXTI_systemState_t;  
22  
23  
24 typedef enum  
25 {  
26     MEXTI_RISING_EDGE = 0,  
27     MEXTI_FALLING_EDGE,  
28     MEXTI_ON_CHANGE  
29 } EN_MEXTI_triggerOptions_t;  
30  
31  
32 typedef enum  
33 {  
34     MEXTI_LINE0 = 0,  
35     MEXTI_LINE1,  
36     MEXTI_LINE2,  
37     MEXTI_LINE3,  
38     MEXTI_LINE4,  
39     MEXTI_LINE5,  
40     MEXTI_LINE6,  
41     MEXTI_LINE7,  
42     MEXTI_LINE8,  
43     MEXTI_LINE9,  
44     MEXTI_LINE10,  
45     MEXTI_LINE11,  
46     MEXTI_LINE12,  
47     MEXTI_LINE13,  
48     MEXTI_LINE14,  
49     MEXTI_LINE15,  
50     MEXTI_LINE16,  
51     MEXTI_LINE17,  
52     MEXTI_LINE18,  
53     MEXTI_LINE19,  
54     MEXTI_LINE20,  
55     MEXTI_LINE21,  
56     MEXTI_LINE22  
57 } EN_MEXTI_lines_t;  
58  
59  
60 typedef enum  
61 {  
62     MEXTI_PORTA = 0,  
63     MEXTI_PORTB,  
64     MEXTI_PORTC,  
65     MEXTI_PORTD,  
66     MEXTI PORTE,  
67     MEXTI_PORTH  
68 } EN_MEXTI_port_t;  
69  
70 #endif /* MCAL_EXTI_EXTI_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/exti/exti_interface.h File Reference

Header file for the EXTI (External Interrupt) module interface.

```
#include "../..../COMMON/bit_math.h"
#include "../..../COMMON/std_types.h"
#include "exti_private.h"
#include "exti_config.h"
```

Functions

- **EN_MEXTI_systemState_t MEXTI_enableEXTI (EN_MEXTI_lines_t copy_uddtLineNum)**
Enable the EXTI line for a specific GPIO pin.
- **EN_MEXTI_systemState_t MEXTI_disableEXTI (EN_MEXTI_lines_t copy_uddtLineNum)**
Disable the EXTI line for a specific GPIO pin.
- **EN_MEXTI_systemState_t MEXTI_softwareInterrupt (EN_MEXTI_lines_t copy_uddtLineNum)**
Generate a software interrupt on the specified EXTI line.
- **EN_MEXTI_systemState_t MEXTI_setTriggerSource (EN_MEXTI_lines_t copy_uddtLineNum, EN_MEXTI_triggerOptions_t copy_uddtTriggerOption)**
Set the trigger source for the specified EXTI line.
- **EN_MEXTI_systemState_t MEXTI_setExtiConfig (EN_MEXTI_lines_t copy_uddtLineNum, EN_MEXTI_port_t copy_uddtPortNum)**
Set the EXTI configuration for the specified GPIO pin.
- **EN_MEXTI_systemState_t MEXTI_setCallBack (void(*ptr)(void), EN_MEXTI_lines_t copy_uddtLineNum)**
Set the callback function for the specified EXTI line.

Detailed Description

Header file for the EXTI (External Interrupt) module interface.

Function Documentation

EN_MEXTI_systemState_t MEXTI_disableEXTI (EN_MEXTI_lines_t copy_uddtLineNum)

Disable the EXTI line for a specific GPIO pin.

This function disables the EXTI line for a specified GPIO pin.

Parameters

<i>copy_uddtLineNumber</i>	The EXTI line number to disable. Possible values are: <ul style="list-style-type: none">• #EN_MEXTI_LINE0• #EN_MEXTI_LINE1• ...• #EN_MEXTI_LINE15
----------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

The system state after disabling the EXTI line.

- #EN_MEXTI_OK: EXTI line disabling successful.
- #EN_MEXTI_NOK: EXTI line disabling failed.
- #EN_MEXTI_INVALID_PARAMTER: Invalid parameter detected during the operation.

EN_MEXTI_systemState_t MEXTI_enableEXTI (EN_MEXTI_lines_t copy_uddtLineNumber)

Enable the EXTI line for a specific GPIO pin.

This function enables the EXTI line for a specified GPIO pin.

Parameters

<i>copy_uddtLineNumber</i>	The EXTI line number to enable. Possible values are: <ul style="list-style-type: none">• #EN_MEXTI_LINE0• #EN_MEXTI_LINE1• ...• #EN_MEXTI_LINE15
----------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

The system state after enabling the EXTI line.

- #EN_MEXTI_OK: EXTI line enabling successful.
- #EN_MEXTI_NOK: EXTI line enabling failed.
- #EN_MEXTI_INVALID_PARAMTER: Invalid parameter detected during the operation.

EN_MEXTI_systemState_t MEXTI_setCallBack (void(*) (void) ptr, EN_MEXTI_lines_t copy_uddtLineNumber)

Set the callback function for the specified EXTI line.

This function sets the callback function for the specified EXTI line.

Parameters

<i>ptr</i>	Pointer to the callback function.
<i>copy_uddtLineNumber</i>	The EXTI line number to set the callback for. Possible values are: <ul style="list-style-type: none">• #EN_MEXTI_LINE0• #EN_MEXTI_LINE1• ...• #EN_MEXTI_LINE15

Returns

The system state after setting the callback function.

- #EN_MEXTI_OK: Callback function setting successful.
- #EN_MEXTI_NOK: Callback function setting failed.
- #EN_MEXTI_INVALID_PARAMTER: Invalid parameter detected during the operation.
- #EN_MEXTI_PTR_NULL: Null pointer encountered during the operation.

**EN_MEXTI_systemState_t MEXTI_setExtiConfig (EN_MEXTI_lines_t
copy_uddtLineNum, EN_MEXTI_port_t copy_uddtPortNum)**

Set the EXTI configuration for the specified GPIO pin.

This function sets the EXTI configuration for the specified GPIO pin.

Parameters

<i>copy_uddtLineNum</i>	The EXTI line number to configure. Possible values are: <ul style="list-style-type: none"> • #EN_MEXTI_LINE0 • #EN_MEXTI_LINE1 • ... • #EN_MEXTI_LINE15
<i>copy_uddtPortNum</i>	The GPIO port number. Possible values are: <ul style="list-style-type: none"> • #EN_MEXTI_PORTA • #EN_MEXTI_PORTB • #EN_MEXTI_PORTC • ... • #EN_MEXTI_PORTH

Returns

The system state after setting the EXTI configuration.

- #EN_MEXTI_OK: EXTI configuration setting successful.
- #EN_MEXTI_NOK: EXTI configuration setting failed.
- #EN_MEXTI_INVALID_PARAMTER: Invalid parameter detected during the operation.

**EN_MEXTI_systemState_t MEXTI_setTriggerSource (EN_MEXTI_lines_t
copy_uddtLineNum, EN_MEXTI_triggerOptions_t copy_uddtTriggerOption)**

Set the trigger source for the specified EXTI line.

This function sets the trigger source for the specified EXTI line.

Parameters

<i>copy_uddtLineNum</i>	The EXTI line number to configure. Possible values are: <ul style="list-style-type: none"> • #EN_MEXTI_LINE0 • #EN_MEXTI_LINE1 • ... • #EN_MEXTI_LINE15
<i>copy_uddtTriggerOption</i>	The trigger source option. Possible values are: <ul style="list-style-type: none"> • #EN_MEXTI_TRIGGER_RISING_EDGE • #EN_MEXTI_TRIGGER_FALLING_EDGE • #EN_MEXTI_TRIGGER_BOTH_EDGES

Returns

The system state after setting the trigger source.

- #EN_MEXTI_OK: Trigger source setting successful.
- #EN_MEXTI_NOK: Trigger source setting failed.
- #EN_MEXTI_INVALID_PARAMTER: Invalid parameter detected during the operation.

**EN_MEXTI_systemState_t MEXTI_softwareInterrupt (EN_MEXTI_lines_t
copy_uddtLineNum)**

Generate a software interrupt on the specified EXTI line.

This function generates a software interrupt on the specified EXTI line.

Parameters

<i>copy_uddtLineNum</i>	The EXTI line number to trigger. Possible values are: <ul style="list-style-type: none">• #EN_MEXTI_LINE0• #EN_MEXTI_LINE1• ...• #EN_MEXTI_LINE15
-------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

The system state after triggering the software interrupt.

- #EN_MEXTI_OK: Software interrupt triggering successful.
- #EN_MEXTI_NOK: Software interrupt triggering failed.
- #EN_MEXTI_INVALID_PARAMTER: Invalid parameter detected during the operation.

exti_interface.h

Go to the documentation of this file.

```
1
7 #ifndef MCAL_EXTI_EXTI_INTERFACE_H_
8 #define MCAL_EXTI_EXTI_INTERFACE_H_
9
10 #include "../COMMON/bit_math.h"
11 #include "../COMMON/std_types.h"
12 #include "exti_private.h"
13 #include "exti_config.h"
14
32 EN_MEXTI_systemState_t MEXTI_enableEXTI(EN_MEXTI_lines_t copy_uddtLineNum);
33
51 EN_MEXTI_systemState_t MEXTI_disableEXTI(EN_MEXTI_lines_t copy_uddtLineNum);
52
70 EN_MEXTI_systemState_t MEXTI_softwareInterrupt(EN_MEXTI_lines_t copy_uddtLineNum);
71
94 EN_MEXTI_systemState_t MEXTI_setTriggerSource(EN_MEXTI_lines_t copy_uddtLineNum,
EN_MEXTI_triggerOptions_t copy_uddtTriggerOption);
95
120 EN_MEXTI_systemState_t MEXTI_setExtiConfig(EN_MEXTI_lines_t copy_uddtLineNum,
EN_MEXTI_port_t copy_uddtPortNum);
121
141 EN_MEXTI_systemState_t MEXTI_setCallBack(void (*ptr)(void), EN_MEXTI_lines_t
copy_uddtLineNum);
142
143
144 #endif /* MCAL_EXTI_EXTI_INTERFACE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/exti/exti_private.h File Reference

Data Structures

- struct ST_MEXTI_RegistersMap_t struct ST_MSYS_CFG_RegistersMap_t

Macros

- #define MEXTI_PERIPHERAL_BASE_ADDR (0x40013C00)
 - #define MSYS_CFG_PERIPHERAL_BASE_ADDR (0x40013800)
 - #define EXTI_CFG_MASK1 0xFFFFFFFF0UL
 - #define EXTI_CFG_MASK2 0xFFFFFFFF0FUL
 - #define EXTI_CFG_MASK3 0xFFFFF0FFUL
 - #define EXTI_CFG_MASK4 0xFFFF0FFFUL
 - #define MEXTI_PERIPHERAL ((volatile ST_MEXTI_RegistersMap_t *)MEXTI_PERIPHERAL_BASE_ADDR)
 - #define MSYS_CFG_PERIPHERAL ((volatile ST_MSYS_CFG_RegistersMap_t *)MSYS_CFG_PERIPHERAL_BASE_ADDR)
-

Macro Definition Documentation

#define EXTI_CFG_MASK1 0xFFFFFFFF0UL

#define EXTI_CFG_MASK2 0xFFFFFFFF0FUL

#define EXTI_CFG_MASK3 0xFFFFF0FFUL

#define EXTI_CFG_MASK4 0xFFFF0FFFUL

#define MEXTI_PERIPHERAL ((volatile ST_MEXTI_RegistersMap_t *)MEXTI_PERIPHERAL_BASE_ADDR)

#define MEXTI_PERIPHERAL_BASE_ADDR (0x40013C00)

#define MSYS_CFG_PERIPHERAL ((volatile ST_MSYS_CFG_RegistersMap_t *)MSYS_CFG_PERIPHERAL_BASE_ADDR)

#define MSYS_CFG_PERIPHERAL_BASE_ADDR (0x40013800)

exti_private.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : STM32F401xC  
4 // File        : exti_private.h  
5 // Date        : Sep 11, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef MCAL_EXTI_PRIVATE_H_  
9 #define MCAL_EXTI_PRIVATE_H_  
10  
11  
12  
13 #define MEXTI_PERIPHERAL_BASE_ADDR      (0x40013C00)  
14 #define MSYSCFG_PERIPHERAL_BASE_ADDR    (0x40013800)  
15  
16  
17 #define EXTI_CONFIG_MASK1               0xFFFFFFF0UL  
18 #define EXTI_CONFIG_MASK2               0xFFFFFFF0FUL  
19 #define EXTI_CONFIG_MASK3               0xFFFFFFF0FUL  
20 #define EXTI_CONFIG_MASK4               0xFFFFFFF0FUL  
21  
22 typedef struct  
23 {  
24  
25     vuint32_t MEXTI_IMR;  
26     vuint32_t MEXTI_EMR;  
27     vuint32_t MEXTI_RTSM;  
28     vuint32_t MEXTI_FTSR;  
29     vuint32_t MEXTI_SWIER;  
30     vuint32_t MEXTI_PR;  
31  
32 }ST_MEXTI_RegistersMap_t;  
33  
34  
35 typedef struct  
36 {  
37     vuint32_t MSYSCFG_MEMRMP;  
38     vuint32_t MSYSCFG_PMC;  
39     vuint32_t MSYSCFG_EXTICR1;  
40     vuint32_t MSYSCFG_EXTICR2;  
41     vuint32_t MSYSCFG_EXTICR3;  
42     vuint32_t MSYSCFG_EXTICR4;  
43     vuint32_t MSYSCFG_CMPCR;  
44 }ST_MSYSCFG_RegistersMap_t;  
45  
46  
47 #define MEXTI_PERIPHERAL      ((volatile ST_MEXTI_RegistersMap_t  
48 *)MEXTI_PERIPHERAL_BASE_ADDR)  
49 #define MSYSCFG_PERIPHERAL    ((volatile ST_MSYSCFG_RegistersMap_t  
50 *)MSYSCFG_PERIPHERAL_BASE_ADDR)  
51 #endif /* MCAL_EXTI_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad
doc/Adaptive_Cruise_Control/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_PUS H_PULL	
MGPIO_OUTPUT _RESISTOR_OPE N_DRAIN	

enum EN_MGPIO_outputSpeedOptions_t

Enumerator:

MGPIO_OUTPUT _SPEED_LOW	
MGPIO_OUTPUT _SPEED_MEDIU M	
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_I NPOT	
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_F LOADING	
MGPIO_PULL_P ULL_UP	
MGPIO_PULL_P ULL_DOWN	

enum EN_MGPIO_setResetOptions_t

Enumerator:

MGPIO_PIN_RES ET	
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/Adaptive_Cruise_Control/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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
<i>copy_uddtPinNum</i>	The pin number to configure. Possible values are: <ul style="list-style-type: none"> • MGPIO_PIN0 • MGPIO_PIN1 • ... • MGPIO_PIN15
<i>copy_uddtSetReset State</i>	The set/reset option. Possible values are: <ul style="list-style-type: none"> • 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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
<i>copy_uddtPinNum</i>	The pin number to read. Possible values are: <ul style="list-style-type: none">• MGPIO_PIN0• MGPIO_PIN1• ...• MGPIO_PIN15
<i>copy_uddtPtrRetOfPinLogic</i>	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

<i>PS_altPinInstance</i>	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

<i>PS_pinInstance</i>	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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
<i>copy_uddtPinNum</i>	The pin number to configure. Possible values are: <ul style="list-style-type: none"> • MGPIO_PIN0 • MGPIO_PIN1 • ... • MGPIO_PIN15
<i>Copy_uddtAltFun</i>	The alternate function option. Possible values are: <ul style="list-style-type: none"> • 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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
<i>copy_uddtPinNum</i>	The pin number to configure. Possible values are: <ul style="list-style-type: none"> • MGPIO_PIN0 • MGPIO_PIN1 • ... • MGPIO_PIN15
<i>copy_uddtOutputResistor</i>	The output resistor option. Possible values are: <ul style="list-style-type: none"> • 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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
<i>copy_uddtPinNum</i>	The pin number to configure. Possible values are: <ul style="list-style-type: none"> • MGPIO_PIN0 • MGPIO_PIN1 • ... • MGPIO_PIN15
<i>copy_uddtOutputSpeed</i>	The output speed option. Possible values are: <ul style="list-style-type: none"> • 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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
<i>copy_uddtPinNum</i>	The pin number to configure. Possible values are: <ul style="list-style-type: none"> • MGPIO_PIN0 • MGPIO_PIN1 • ... • MGPIO_PIN15
<i>copy_uddtPinMode</i>	The mode to set for the pin. Possible values are: <ul style="list-style-type: none"> • 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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
<i>copy_uddtPinNum</i>	The pin number to configure. Possible values are: <ul style="list-style-type: none"> • MGPIO_PIN0 • MGPIO_PIN1 • ... • MGPIO_PIN15
<i>copy_uddtPinLogic</i>	The logic level to set for the pin. Possible values are: <ul style="list-style-type: none"> • 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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
<i>copy_u16OutputVal</i>	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

<i>PS_GPIOx</i>	Pointer to the GPIOx registers map.
-----------------	-------------------------------------

<i>copy_uddtPinNum</i>	The pin number to configure. Possible values are: <ul style="list-style-type: none"> • MGPIO_PIN0 • MGPIO_PIN1 • ... • MGPIO_PIN15
<i>copy_uddtPullState</i>	The pull state option. Possible values are: <ul style="list-style-type: none"> • 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
2
3
4
5
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
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29 EN_MGPIO_systemState_t MGPIO_uddtSetPinMode(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
30 EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_pinModeOptions_t copy_uddtPinMode);
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64 EN_MGPIO_systemState_t MGPIO_uddtSetOutputMode(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
65 EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_outputResistorOptions_t
66 copy_uddtOutputResistor);
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91 EN_MGPIO_systemState_t MGPIO_uddtSetOutputSpeed(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
92 EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_outputSpeedOptions_t
93 copy_uddtOutputSpeed);
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109 EN_MGPIO_systemState_t MGPIO_uddtSetPullState(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
110 EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_pushPullOptions_t copy_uddtPullState);
111
112
113
114
115
116
117
118
119 EN_MGPIO_systemState_t MGPIO_uddtGetPinVal(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
120 EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_pinLogicOptions_t
121 *copy_uddtPtrRetOfPinLogic);
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139 EN_MGPIO_systemState_t MGPIO_uddtSetPinVal(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
140 EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_pinLogicOptions_t copy_uddtPinLogic);
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159 EN_MGPIO_systemState_t MGPIO_uddtDirectSetReset(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
160 EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_setResetOptions_t
161 copy_uddtSetResetState);
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180 EN_MGPIO_systemState_t MGPIO_uddtSetPortVal(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
181 uint16_t copy_ul16OutputVal);
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200 EN_MGPIO_systemState_t MGPIO_uddtSetAltFun(ST_MGPIOx_RegistersMap_t *PS_GPIOx,
201 EN_MGPIO_pinOptions_t copy_uddtPinNum, EN_MGPIO_altfnOptions_t Copy_uddtAltFun);
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219 EN_MGPIO_systemState_t MGPIO_uddtInitPin(ST_MGPIO_pinCfg_t *PS_pinInstance);
220
221
222
223
224
225
226
227
228
229 EN_MGPIO_systemState_t MGPIO_uddtInitAltPin(ST_MGPIO_altPinCfg_t
230 *PS_altPinInstance);
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259 #endif /* MCAL_GPIO_GPIO_INTERFACE_H_ */
```

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

Data Structures

struct ST_MGPIOx_RegistersMap_tMacros

- **#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    MGPIOn_MODER;  
24     vuint32_t    MGPIOn_OTYPER;  
25     vuint32_t    MGPIOn_OSPEEDR;  
26     vuint32_t    MGPIOn_PUPDR;  
27     vuint32_t    MGPIOn_IDR;  
28     vuint32_t    MGPIOn_ODR;  
29     vuint32_t    MGPIOn_BSRR;  
30     vuint32_t    MGPIOn_LCKR;  
31     vuint32_t    MGPIOn_AFR1;  
32     vuint32_t    MGPIOn_AFRH;  
33  
34  
35 }ST_MGPIOn_RegistersMap_t;  
36  
37  
38  
39  
40 #define MGPIOA_PERIPHERAL (( ST_MGPIOn_RegistersMap_t *)MGPIOA_PERIPHERAL_BASE_ADDR)  
41 #define MGPIOB_PERIPHERAL (( ST_MGPIOn_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/Adaptive_Cruise_Control/Inc/MCAL/nvic/nvic_config.h File Reference

Macros

- `#define POS_OF_FIRST_INT 0`
- `#define POS_OF_LAST_INT 84`
- `#define getIntRegNumber(IntNumber) (IntNumber / 32)`
- `#define getIntBitNumber(IntNumber) (IntNumber % 32)`

Enumerations

- `enum EN_MNVIC_systemState_t { MNVIC_OK = 0, MNVIC_NOK, MNVIC_INVALID_PARAMTER }`
Enumeration for the system state of NVIC functions.
- `enum EN_MNVIC_priorityGrouping_t { GP_16_SP_00 = 0, GP_08_SP_02, GP_04_SP_04, GP_02_SP_08, GP_00_SP_16 }`
Enumeration for the priority grouping options in the NVIC.

Macro Definition Documentation

`#define getIntBitNumber(IntNumber) (IntNumber % 32)`

`#define getIntRegNumber(IntNumber) (IntNumber / 32)`

`#define POS_OF_FIRST_INT 0`

`#define POS_OF_LAST_INT 84`

Enumeration Type Documentation

`enum EN_MNVIC_priorityGrouping_t`

Enumeration for the priority grouping options in the NVIC.

Enumerator:

GP_16_SP_00	16 priority levels, 0 subpriority levels.
GP_08_SP_02	8 priority levels, 2 subpriority levels.
GP_04_SP_04	4 priority levels, 4 subpriority levels.
GP_02_SP_08	2 priority levels, 8 subpriority levels.
GP_00_SP_16	0 priority levels, 16 subpriority levels.

enum EN_MNVIC_systemState_t

Enumeration for the system state of NVIC functions.

Enumerator:

MNVIC_OK	Operation successful.
MNVIC_NOK	Operation failed.
MNVIC_INVALID_PARAMETER	Invalid parameter detected.

nvic_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : STM32F401xC  
4 // File        : nvic_config.h  
5 // Date        : Sep 10, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef MCAL_NVIC_NVIC_CONFIG_H_  
9 #define MCAL_NVIC_NVIC_CONFIG_H_  
10  
11 #define POS_OF_FIRST_INT      0  
12 #define POS_OF_LAST_INT      84  
13  
14 #define getIntRegNumber(IntNumber) (IntNumber / 32)  
15 #define getIntBitNumber(IntNumber) (IntNumber % 32)  
16  
17  
21 typedef enum  
22 {  
23     MNVIC_OK = 0,  
24     MNVIC_NOK,  
25     MNVIC_INVALID_PARAMTER  
26 } EN_MNVIC_systemState_t;  
27  
31 typedef enum  
32 {  
33     GP_16_SP_00 = 0,  
34     GP_08_SP_02,  
35     GP_04_SP_04,  
36     GP_02_SP_08,  
37     GP_00_SP_16  
38 } EN_MNVIC_priorityGrouping_t;  
39  
40  
41 #endif /* MCAL_NVIC_NVIC_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/nvic/nvic_interface.h File Reference

Header file for the NVIC (Nested Vectored Interrupt Controller) module interface.

```
#include "../..../COMMON/bit_math.h"
#include "../..../COMMON/std_types.h"
#include "nvic_private.h"
#include "nvic_config.h"
```

Functions

- **EN_MNVIC_systemState_t MNVIC_enableInterrupt (uint8_t copy_u8IntPos)**
Enable an interrupt in the NVIC.
- **EN_MNVIC_systemState_t MNVIC_disableInterrupt (uint8_t copy_u8IntPos)**
Disable an interrupt in the NVIC.
- **EN_MNVIC_systemState_t MNVIC_enableInterruptPending (uint8_t copy_u8IntPos)**
Enable pending status for an interrupt in the NVIC.
- **EN_MNVIC_systemState_t MNVIC_disableInterruptPending (uint8_t copy_u8IntPos)**
Disable pending status for an interrupt in the NVIC.
- **EN_MNVIC_systemState_t MNVIC_IsInterruptActive (uint8_t copy_u8IntPos, uint8_t *ptrOfRetReading)**
Check if an interrupt is active in the NVIC.
- **EN_MNVIC_systemState_t MNVIC_SetInterruptPriority (uint8_t copy_u8IntPos, EN_MNVIC_priorityGrouping_t copy_uddtGroupOption, uint8_t copy_u8GroupPriority, uint8_t copy_u8SubPriority)**
Set the priority of an interrupt in the NVIC.
- **EN_MNVIC_systemState_t MNVIC_SetInterruptGroup (EN_MNVIC_priorityGrouping_t copy_uddtGroupOption)**
Set the priority grouping for the NVIC.

Detailed Description

Header file for the NVIC (Nested Vectored Interrupt Controller) module interface.

Function Documentation

EN_MNVIC_systemState_t MNVIC_disableInterrupt (uint8_t copy_u8IntPos)

Disable an interrupt in the NVIC.

This function disables the specified interrupt in the NVIC.

Parameters

<i>copy_u8IntPos</i>	The position of the interrupt to disable.
----------------------	-------------------------------------------

Returns

The system state after disabling the interrupt.

- MNVIC_OK: Operation successful.
- MNVIC_NOK: Operation failed.
- MNVIC_INVALID_PARAMTER: Invalid parameter detected.

EN_MNVIC_systemState_t MNVIC_disableInterruptPending (uint8_t *copy_u8IntPos*)

Disable pending status for an interrupt in the NVIC.

This function disables the pending status for the specified interrupt in the NVIC.

Parameters

<i>copy_u8IntPos</i>	The position of the interrupt to disable pending status for.
----------------------	--------------------------------------------------------------

Returns

The system state after disabling pending status for the interrupt.

- MNVIC_OK: Operation successful.
- MNVIC_NOK: Operation failed.
- MNVIC_INVALID_PARAMTER: Invalid parameter detected.

EN_MNVIC_systemState_t MNVIC_enableInterrupt (uint8_t *copy_u8IntPos*)

Enable an interrupt in the NVIC.

This function enables the specified interrupt in the NVIC.

Parameters

<i>copy_u8IntPos</i>	The position of the interrupt to enable.
----------------------	------------------------------------------

Returns

The system state after enabling the interrupt.

- MNVIC_OK: Operation successful.
- MNVIC_NOK: Operation failed.
- MNVIC_INVALID_PARAMTER: Invalid parameter detected.

EN_MNVIC_systemState_t MNVIC_enableInterruptPending (uint8_t *copy_u8IntPos*)

Enable pending status for an interrupt in the NVIC.

This function enables the pending status for the specified interrupt in the NVIC.

Parameters

<i>copy_u8IntPos</i>	The position of the interrupt to enable pending status for.
----------------------	-------------------------------------------------------------

Returns

The system state after enabling pending status for the interrupt.

- MNVIC_OK: Operation successful.
- MNVIC_NOK: Operation failed.
- MNVIC_INVALID_PARAMTER: Invalid parameter detected.

EN_MNVIC_systemState_t MNVIC_IsInterruptActive (uint8_t *copy_u8IntPos*, uint8_t **ptrOfRetReading*)

Check if an interrupt is active in the NVIC.

This function checks if the specified interrupt is active in the NVIC.

Parameters

<i>copy_u8IntPos</i>	The position of the interrupt to check.
<i>ptrOfRetReading</i>	Pointer to store the result of the interrupt's active status.

Returns

The system state after checking the interrupt's active status.

- MNVIC_OK: Operation successful.
- MNVIC_NOK: Operation failed.
- MNVIC_INVALID_PARAMTER: Invalid parameter detected.

EN_MNVIC_systemState_t MNVIC_SetInterruptGroup (EN_MNVIC_priorityGrouping_t *copy_uddtGroupOption*)

Set the priority grouping for the NVIC.

This function sets the priority grouping for the NVIC.

Parameters

<i>copy_uddtGroupOption</i>	The priority grouping option (GP_16_SP_00, GP_08_SP_02, GP_04_SP_04, GP_02_SP_08, GP_00_SP_16).
-----------------------------	-------------------------------------------------------------------------------------------------

Returns

The system state after setting the priority grouping.

- MNVIC_OK: Operation successful.
- MNVIC_NOK: Operation failed.
- MNVIC_INVALID_PARAMTER: Invalid parameter detected.

EN_MNVIC_systemState_t MNVIC_SetInterruptPriority (uint8_t *copy_u8IntPos*, EN_MNVIC_priorityGrouping_t *copy_uddtGroupOption*, uint8_t *copy_u8GroupPriority*, uint8_t *copy_u8SubPriority*)

Set the priority of an interrupt in the NVIC.

This function sets the priority of the specified interrupt in the NVIC.

Parameters

<i>copy_u8IntPos</i>	The position of the interrupt to set the priority for.
<i>copy_uddtGroupOption</i>	The priority grouping option (GP_16_SP_00, GP_08_SP_02, GP_04_SP_04, GP_02_SP_08, GP_00_SP_16).
<i>copy_u8GroupPriority</i>	The group priority value (0 to 15).
<i>copy_u8SubPriority</i>	The subpriority value (0 to 15).

Returns

The system state after setting the interrupt priority.

- MNVIC_OK: Operation successful.
- MNVIC_NOK: Operation failed.
- MNVIC_INVALID_PARAMTER: Invalid parameter detected.

nvic_interface.h

Go to the documentation of this file.

```
1
7 #ifndef MCAL_NVIC_NVIC_INTERFACE_H_
8 #define MCAL_NVIC_NVIC_INTERFACE_H_
9
10 #include "../COMMON/bit_math.h"
11 #include "../COMMON/std_types.h"
12 #include "nvic_private.h"
13 #include "nvic_config.h"
14
27 EN_MNVIC_systemState_t MNVIC_enableInterrupt(uint8_t copy_u8IntPos);
28
41 EN_MNVIC_systemState_t MNVIC_disableInterrupt(uint8_t copy_u8IntPos);
42
55 EN_MNVIC_systemState_t MNVIC_enableInterruptPending(uint8_t copy_u8IntPos);
56
69 EN_MNVIC_systemState_t MNVIC_disableInterruptPending(uint8_t copy_u8IntPos);
70
84 EN_MNVIC_systemState_t MNVIC_IsInterruptActive(uint8_t copy_u8IntPos, uint8_t
*ptrOfRetReading);
85
101 EN_MNVIC_systemState_t MNVIC_SetInterruptPriority(uint8_t copy_u8IntPos,
EN_MNVIC_priorityGrouping_t copy_uddtGroupOption, uint8_t copy_u8GroupPriority, uint8_t
copy_u8SubPriority);
102
115 EN_MNVIC_systemState_t MNVIC_SetInterruptGroup(EN_MNVIC_priorityGrouping_t
copy_uddtGroupOption);
116
117 #endif /* MCAL_NVIC_NVIC_INTERFACE_H_ */
```


D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/nvic/nvic_private.h File Reference

Data Structures

struct ST_MNVIC_RegistersMap_t **Macros**

- `#define MNVIC_PERIPHERAL_BASE_ADDR (0xE000E100)`
 - `#define MNVIC_PERIPHERAL ((volatile ST_MNVIC_RegistersMap_t *)MNVIC_PERIPHERAL_BASE_ADDR)`
 - `#define SCB_AIRCR *((volatile uint32_t *) (0xE000ED0C))`
 - `#define MNVIC_VECTKEY (0x05FA0000) /* Key to write to AIRCR register */`
-

Macro Definition Documentation

`#define MNVIC_PERIPHERAL ((volatile ST_MNVIC_RegistersMap_t *)MNVIC_PERIPHERAL_BASE_ADDR)`

`#define MNVIC_PERIPHERAL_BASE_ADDR (0xE000E100)`

`#define MNVIC_VECTKEY (0x05FA0000) /* Key to write to AIRCR register */`

`#define SCB_AIRCR *((volatile uint32_t *) (0xE000ED0C))`

nvic_private.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : STM32F401xC  
4 // File        : nvic_private.h  
5 // Date        : Sep 10, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef MCAL_NVIC_NVIC_PRIVATE_H_  
9 #define MCAL_NVIC_NVIC_PRIVATE_H_  
10  
11  
12 #define MNVIC_PERIPHERAL_BASE_ADDR    (0xE000E100)  
13  
14  
15 typedef struct  
16 {  
17  
18     vuint32_t MNVIC_ISERx[8];  
19     vuint32_t MNVIC_RESERVED0[24];  
20     vuint32_t MNVIC_ICERx[8];  
21     vuint32_t MNVIC_RESERVED1[24];  
22     vuint32_t MNVIC_ISPRx[8];  
23     vuint32_t MNVIC_RESERVED2[24];  
24     vuint32_t MNVIC_ICPRx[8];  
25     vuint32_t MNVIC_RESERVED3[24];  
26     vuint32_t MNVIC_IABRx[8];  
27     vuint32_t MNVIC_RESERVED4[56];  
28     vuint8_t  MNVIC_IPRx[240];  
29     vuint32_t MNVIC_RESERVED5[580];  
30     vuint32_t MNVIC_STIR;  
31  
32 }ST_MNVIC_RegistersMap_t;  
33  
34  
35 #define MNVIC_PERIPHERAL ((volatile ST_MNVIC_RegistersMap_t  
36 *)MNVIC_PERIPHERAL_BASE_ADDR)  
37 #define SCB_AIRCR        *((volatile uint32_t *) (0xE000ED0C))  
38  
39 #define MNVIC_VECTKEY     (0x05FA0000) /* Key to write to AIRCR register */  
40  
41  
42 #endif /* MCAL_NVIC_NVIC_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/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 pllDivisionFactor 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_PR ESCALER_128	AHB prescaler: 128.
MRCC_AHB_PR ESCALER_256	AHB prescaler: 256.
MRCC_AHB_PR ESCALER_512	AHB prescaler: 512.

enum EN_MRCC_apbPrescalerSpeed_t

Enumeration for APB (Advanced Peripheral Bus) prescaler speed.

Enumerator:

MRCC_APB_PRE SCALER_SPEED _NOT_DIVIDED	APB prescaler speed: Not divided.
MRCC_APB_PRE SCALER_SPEED _2	APB prescaler speed: 2.
MRCC_APB_PRE SCALER_SPEED _4	APB prescaler speed: 4.
MRCC_APB_PRE SCALER_SPEED _8	APB prescaler speed: 8.
MRCC_APB_PRE SCALER_SPEED _16	APB prescaler speed: 16.

enum EN_MRCC_busOptions_t

Enumeration for different buses in MRCC.

Enumerator:

MRCC_AHP1_B US	AHP1 bus.
MRCC_AHP2_B US	AHP2 bus.
MRCC_APB1_BU S	APB1 bus.
MRCC_APB2_BU S	APB2 bus.

enum EN_MRCC_peripheralOptions_t

Enumeration for different peripheral options in MRCC.

Enumerator:

MRCC_GPIOA_P ERIPHERAL	
MRCC_GPIOB_P	

ERIPHERAL	
MRCC_CRC_PE RIPHERAL	
MRCC_DMA1_P ERIPHERAL	
MRCC_DMA2_P ERIPHERAL	
MRCC_OTGFS_P ERIPHERAL	
MRCC_TIM2_PE RIPHERAL	
MRCC_TIM3_PE RIPHERAL	
MRCC_TIM4_PE RIPHERAL	
MRCC_TIM5_PE RIPHERAL	
MRCC_WWDG_ PERIPHERAL	
MRCC_SPI2_PER IPHERAL	
MRCC_SPI3_PER IPHERAL	
MRCC_USART2_ PERIPHERAL	
MRCC_I2C1_PER IPHERAL	
MRCC_I2C2_PER IPHERAL	
MRCC_I2C3_PER IPHERAL	
MRCC_PWR_PE RIPHERAL	
MRCC_TIM1_PE RIPHERAL	
MRCC_USART1_ PERIPHERAL	
MRCC_USART6_ PERIPHERAL	
MRCC_ADC1_PE RIPHERAL	
MRCC_SDIO_PE RIPHERAL	
MRCC_SPI1_PER IPHERAL	
MRCC_SPI4_PER IPHERAL	
MRCC_SYSCFG_ PERIPHERAL	
MRCC_TIM9_PE RIPHERAL	
MRCC_TIM10_P ERIPHERAL	
MRCC_TIM11_P ERIPHERAL	

enum EN_MRCC_pllClkSourceEntry_t

Enumeration for the PLL entry clock source.

Enumerator:

MRCC_PLL_ENT RY_CLK_HSI	PLL entry clock source: HSI.
MRCC_PLL_ENT RY_CLK_HSE	PLL entry clock source: HSE.

enum EN_MRCC_pllDivisionFactor_t

Enumeration for the PLL division factor.

Enumerator:

MRCC_PLL_DIV ISION_FACTOR_ 2	PLL division factor: 2.
MRCC_PLL_DIV ISION_FACTOR_ 4	PLL division factor: 4.
MRCC_PLL_DIV ISION_FACTOR_ 6	PLL division factor: 6.
MRCC_PLL_DIV ISION_FACTOR_ 8	PLL division factor: 8.

enum EN_MRCC_pllStatus_t

Enumeration for PLL status.

Enumerator:

MRCC_PLL_DIS ABLE	PLL is disabled.
MRCC_PLL_ENA BLE	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_BYP ASS	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_PARAMETER	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 pllDivisionFactor     MRCC_PLL_DIVISION_FACTOR_2  
15 #define pllDivisionFactor     0  
16 #define pllMulFactor          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  
29 typedef enum  
30 {  
31     MRCC_SYS_CLK_HSI = 0,  
32     MRCC_SYS_CLK_HSE_BYPASS,  
33     MRCC_SYS_CLK_HSE_NOT_BYPASS = 1,  
34     MRCC_SYS_CLK_PLL,  
35     MRCC_SYS_CLK_NOT_ALLOWED  
36 } EN_MRCC_sysClkSelect_t;  
37  
38 typedef enum  
39 {  
40     MRCC_PLL_ENTRY_CLK_HSI = 0,  
41     MRCC_PLL_ENTRY_CLK_HSE  
42 } EN_MRCC_pllClkSourceEntry_t;  
43  
44 typedef enum  
45 {  
46     MRCC_PLL_DIVISION_FACTOR_2 = 0,  
47     MRCC_PLL_DIVISION_FACTOR_4,  
48     MRCC_PLL_DIVISION_FACTOR_6,  
49     MRCC_PLL_DIVISION_FACTOR_8  
50 } EN_MRCC_pllDivisionFactor_t;  
51  
52 typedef enum  
53 {  
54     MRCC_APB_PRESCALER_SPEED_NOT_DIVIDED = 0,  
55     MRCC_APB_PRESCALER_SPEED_2 = 4,  
56     MRCC_APB_PRESCALER_SPEED_4,  
57     MRCC_APB_PRESCALER_SPEED_8,  
58     MRCC_APB_PRESCALER_SPEED_16  
59 } EN_MRCC_apbPrescalerSpeed_t;  
60  
61 typedef enum  
62 {  
63     MRCC_AHB_PRESCALER_NOT_DIVIDED = 0,  
64     MRCC_AHB_PRESCALER_2 = 8,  
65     MRCC_AHB_PRESCALER_4,  
66     MRCC_AHB_PRESCALER_8,  
67     MRCC_AHB_PRESCALER_16,  
68     MRCC_AHB_PRESCALER_64,  
69     MRCC_AHB_PRESCALER_128,  
70     MRCC_AHB_PRESCALER_256,  
71     MRCC_AHB_PRESCALER_512  
72 } 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 }EN_MRCC_peripheralOptions_t;
144
145
146
147 #endif /* MCAL_RCC_RCC_CONFIG_H_ */

```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/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

<i>busSelection</i>	The bus on which the peripheral is located. Possible values are: <ul style="list-style-type: none">• #EN_MRCC_AHP1_BUS• #EN_MRCC_AHP2_BUS• #EN_MRCC_APB1_BUS• #EN_MRCC_APB2_BUS
<i>PeripheralNumber</i>	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

<i>busSelection</i>	The bus on which the peripheral is located. Possible values are: <ul style="list-style-type: none">• #EN_MRCC_AHP1_BUS• #EN_MRCC_AHP2_BUS• #EN_MRCC_APB1_BUS• #EN_MRCC_APB2_BUS
<i>PeripheralNumber</i>	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
2
3
4
5
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
15
16
17
18
19
20
21
22
23
24
25
26
27
28 EN_MRCC_systemState_t MRCC_Init(void);
29
30
31 EN_MRCC_systemState_t MRCC_enablePeripheral(EN_MRCC_busOptions_t busSelection,
32 EN_MRCC_peripheralOptions_t PeripheralNumber);
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51 EN_MRCC_systemState_t MRCC_disablePeripheral(EN_MRCC_busOptions_t busSelection,
52 EN_MRCC_peripheralOptions_t PeripheralNumber);
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76 void HAL_DeInit(void);
77
78 #endif /* MCAL_RCC_RCC_INTERFACE_H_ */
```

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

Data Structures

struct ST_MRCC_RegistersMap_tMacros

- #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
= 0xFFFFFFFFFU)

#define
__HAL_RCC_AHB1_RELEASE_RESET() (MRCC_PERIPHERAL->RCC_AHB1RSTR_RE
G = 0x00U)

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

#define
__HAL_RCC_APB1_RELEASE_RESET() (MRCC_PERIPHERAL->RCC_APB1RSTR_RE
G = 0x00U)

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

#define
__HAL_RCC_APB2_RELEASE_RESET() (MRCC_PERIPHERAL->RCC_APB2RSTR_RE
G = 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/Adaptive_Cruise_Control/Inc/MCAL/systick/systick_conf g.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:

<code>MSTK_CLK_AH B_8</code>	SysTick clock source is AHB/8.
<code>MSTK_CLK_PRO CESSOR_AHB</code>	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_PARAMETER	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  
19 typedef enum  
20 {  
21     MSTK_OK = 0,  
22     MSTK_NOK,  
23     MSTK_INVALID_PARAMTER  
24 } EN_MSTK_systemState_t;  
25  
26 typedef enum  
27 {  
28     MSTK_CLK_AHB_8 = 0,  
29     MSTK_CLK_PROCESSOR_AHB  
30 } EN_MSTK_clkSourceOptions_t;  
31  
32 typedef enum  
33 {  
34     MSTK_INTERRUPT_ENABLED = 0,  
35     MSTK_INTERRUPT_DISABLED  
36 } EN_MSTK_interruptStates_t;  
37  
38 #endif /* MCAL_SYSTICK_SYSTICK_CONFIG_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/systick/systick_interf ace.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

<i>copy_u32PtrRetOfElapsedTicks</i>	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

<i>copy_uddtClkSource</i>	The clock source option (MSTK_CLK_AHB_8, MSTK_CLK_PROCESSOR_AHB).
<i>copy_uddtIntStates</i>	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

<i>copy_u32NumberOfTicks</i>	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

<i>copy_u32NumberOfTicks</i>	The number of ticks for the periodic interval.
<i>Pf</i>	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

<i>copy_u32NumberOfTicks</i>	The number of ticks for the single-shot interval.
<i>Pf</i>	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/Adaptive_Cruise_Control/Inc/MCAL/systick/systick_privat e.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  
30 *)MSTK_PERIPHERAL_BASE_ADDR)  
31  
32 #endif /* MCAL_SYSTICK_SYSTICK_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/tmr/tmr_config.h File Reference

Enumerations

- enum **EN_MTMR_number_t** { MTMR2 = 0, MTMR3, MTMR4, MTMR5 }
Enumeration for Multi-Function Timer numbers.
 - enum **EN_MTMR_channel_t** { MTMR_CH1 = 1, MTMR_CH2, MTMR_CH3, MTMR_CH4 }
Enumeration for Multi-Function Timer channels.
 - enum **EN_MTMR_selectedMode_t** { MTMR_MODE_FROZEN = 0, MTMR_MODE_ACTIVE, MTMR_MODE_INACTIVE, MTMR_MODE_TOGGLE, MTMR_MODE_INACTIVE_FORCE, MTMR_MODE_ACTIVE_FORCE, MTMR_MODE_PWM_MODE1, MTMR_MODE_PWM_MODE2 }
Enumeration for selected modes of Multi-Function Timer.
 - enum **CH_MODE_t** { OUTPUT, IC_T2, IC_T1, IC_TRC }
Enumeration for Multi-Function Timer channel modes.
 - enum **EDGE_t** { RISIN, FALLIN, RESERVED, BOTH }
Enumeration for Multi-Function Timer edge options.
-

Enumeration Type Documentation

enum CH_MODE_t

Enumeration for Multi-Function Timer channel modes.

Enumerator:

OUTPUT	Output mode.
IC_T2	Input capture mode T2.
IC_T1	Input capture mode T1.
IC_TRC	Input capture mode TRC.

enum EDGE_t

Enumeration for Multi-Function Timer edge options.

Enumerator:

RISIN	Rising edge.
FALLIN	Falling edge.

RESERVED	Reserved.
BOTH	Both edges.

enum EN_MTMR_channel_t

Enumeration for Multi-Function Timer channels.

Enumerator:

MTMR_CH1	Multi-Function Timer Channel 1.
MTMR_CH2	Multi-Function Timer Channel 2.
MTMR_CH3	Multi-Function Timer Channel 3.
MTMR_CH4	Multi-Function Timer Channel 4.

enum EN_MTMR_number_t

Enumeration for Multi-Function Timer numbers.

Enumerator:

MTMR2	Multi-Function Timer 2.
MTMR3	Multi-Function Timer 3.
MTMR4	Multi-Function Timer 4.
MTMR5	Multi-Function Timer 5.

enum EN_MTMR_selectedMode_t

Enumeration for selected modes of Multi-Function Timer.

Enumerator:

MTMR_MODE_FROZEN	Frozen mode.
MTMR_MODE_ACTIVE	Active mode.
MTMR_MODE_INACTIVE	Inactive mode.
MTMR_MODE_TOGGLE	Toggle mode.

MTMR_MODE_I NACTIVE_FORC E	Inactive force mode.
MTMR_MODE_A CTIVE_FORCE	Active force mode.
MTMR_MODE_P WM_MODE1	PWM mode 1.
MTMR_MODE_P WM_MODE2	PWM mode 2.

tmr_config.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : tmr_config.h  
5 // Date        : Oct 14, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef MCAL_TMR_TMR_CONFIG_H_  
9 #define MCAL_TMR_TMR_CONFIG_H_  
10  
11  
12  
13  
14  
15 typedef enum  
16 {  
17     MTMR2 = 0,  
18     MTMR3,  
19     MTMR4,  
20     MTMR5  
21 } EN_MTMR_number_t;  
22  
23  
24  
25  
26 typedef enum  
27 {  
28     MTMR_CH1 = 1,  
29     MTMR_CH2,  
30     MTMR_CH3,  
31     MTMR_CH4  
32 } EN_MTMR_channel_t;  
33  
34  
35  
36  
37 typedef enum  
38 {  
39     MTMR_MODE_FROZEN = 0,  
40     MTMR_MODE_ACTIVE,  
41     MTMR_MODE_INACTIVE,  
42     MTMR_MODE_TOGGLE,  
43     MTMR_MODE_INACTIVE_FORCE,  
44     MTMR_MODE_ACTIVE_FORCE,  
45     MTMR_MODE_PWM_MODE1,  
46     MTMR_MODE_PWM_MODE2  
47 } EN_MTMR_selectedMode_t;  
48  
49  
50  
51  
52 typedef enum  
53 {  
54     OUTPUT,  
55     IC_T2,  
56     IC_T1,  
57     IC_TRC  
58 } CH_MODE_t;  
59  
60  
61  
62  
63 typedef enum  
64 {  
65     RISIN,  
66     FALLIN,  
67     RESERVED,  
68     BOTH  
69 } EDGE_t;  
70 #endif /* MCAL_TMR_TMR_CONFIG_H_ */
```


D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/tmr/tmr_interface.h File Reference

Header file for the Multi-Mode Timer (MTMR) module.

```
#include "../..../COMMON/bit_math.h"
#include "../..../COMMON/std_types.h"
#include "tmr_private.h"
#include "tmr_config.h"
```

Functions

- void **MTMR_vStartTimer** (EN_MTMR_number_t copy_uddtTMRNumber)
Start the specified Multi-Function Timer.
- void **MTMR_vStopTimer** (EN_MTMR_number_t copy_uddtTMRNumber)
Stop the specified Multi-Function Timer.
- void **MTMR_vSetTimerPrescaler** (EN_MTMR_number_t copy_uddtTMRNumber, uint16_t copy_u16Value)
Set the prescaler value for the specified Multi-Function Timer.
- void **MTMR_vEnableTimerOPM** (EN_MTMR_number_t copy_uddtTMRNumber)
Enable One-Pulse Mode for the specified Multi-Function Timer.
- void **MTMR_vTimerCountRst** (EN_MTMR_number_t copy_uddtTMRNumber)
Reset the count of the specified Multi-Function Timer.
- void **MTMR_vSetTimerChannelOutput** (EN_MTMR_number_t copy_uddtTMRNumber, EN_MTMR_selectedMode_t copy_uddtTimerMode, EN_MTMR_channel_t copy_uddtChannelNumber)
Set the output mode for a specific channel of the Multi-Function Timer.
- void **MTMR_vSetTimerChannelInput** (EN_MTMR_number_t copy_uddtTMRNumber, EN_MTMR_channel_t copy_uddtChannelNumber)
Set the input mode for a specific channel of the Multi-Function Timer.
- void **MTMR_vSetTimerARR** (EN_MTMR_number_t copy_uddtTMRNumber, uint32_t copy_u32Value)
Set the Auto-Reload Register value for the specified Multi-Function Timer.
- void **MTMR_vSetTimerStop** (EN_MTMR_number_t copy_uddtTMRNumber)
Stop the specified Multi-Function Timer.
- void **MTMR_vClearTimerCount** (EN_MTMR_number_t copy_uddtTMRNumber)
Clear the count of the specified Multi-Function Timer.
- void **MTMR_vEnableTimerICUInt** (EN_MTMR_number_t copy_uddtTMRNumber)
Enable the interrupt for the specified Multi-Function Timer input capture.

- **void MTMR_vSetTimerCMPVal** (EN_MTMR_number_t copy_uddtTMRNumber, EN_MTMR_channel_t copy_uddtChannelNumber, uint32_t copy_u32CmpValue)
Set the compare value for the specified channel of the Multi-Function Timer.
- **uint32_t MTMR_vReadCaptureVal** (EN_MTMR_number_t copy_uddtTMRNumber, EN_MTMR_channel_t copy_uddtChannelNumber)
Read the capture value for the specified channel of the Multi-Function Timer.
- **void MTMR3_vCaptureCompareInit** (void)
Initialize the capture compare functionality for Multi-Function Timer 3.

Detailed Description

Header file for the Multi-Mode Timer (MTMR) module.

Function Documentation

void MTMR3_vCaptureCompareInit (void)

Initialize the capture compare functionality for Multi-Function Timer 3.

This function initializes the capture compare functionality for Multi-Function Timer 3.

Returns

No return.

void MTMR_vClearTimerCount (EN_MTMR_number_t copy_uddtTMRNumber)

Clear the count of the specified Multi-Function Timer.

This function clears the count of the specified Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	<p>The Multi-Function Timer to clear. Possible values are:</p> <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
---------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

No return.

void MTMR_vEnableTimerICUInt (EN_MTMR_number_t copy_uddtTMRNumber)

Enable the interrupt for the specified Multi-Function Timer input capture.

This function enables the interrupt for the specified Multi-Function Timer input capture.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to configure. Possible values are: <ul style="list-style-type: none">• MTMR2• MTMR3• MTMR4• MTMR5
---------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

No return.

void MTMR_vEnableTimerOPM (EN_MTMR_number_t *copy_uddtTMRNumber*)

Enable One-Pulse Mode for the specified Multi-Function Timer.

This function enables One-Pulse Mode for the specified Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to configure. Possible values are: <ul style="list-style-type: none">• MTMR2• MTMR3• MTMR4• MTMR5
---------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

No return.

uint32_t MTMR_vReadCaptureVal (EN_MTMR_number_t *copy_uddtTMRNumber*, EN_MTMR_channel_t *copy_uddtChannelNumber*)

Read the capture value for the specified channel of the Multi-Function Timer.

This function reads the capture value for the specified channel of the Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to read from. Possible values are: <ul style="list-style-type: none">• MTMR2• MTMR3• MTMR4• MTMR5
<i>copy_uddtChannelNumber</i>	The channel number to read from. Possible values are: <ul style="list-style-type: none">• MTMR_CH1• MTMR_CH2• MTMR_CH3• MTMR_CH4

Returns

The captured value from the specified channel.

void MTMR_vSetTimerARR (EN_MTMR_number_t *copy_uddtTMRNumber*, uint32_t *copy_u32Value*)

Set the Auto-Reload Register value for the specified Multi-Function Timer.

This function sets the Auto-Reload Register value for the specified Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to configure. Possible values are: <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
<i>copy_u32Value</i>	The value to set for the Auto-Reload Register.

Returns

No return.

void MTMR_vSetTimerChannelInput (EN_MTMR_number_t *copy_uddtTMRNumber*, EN_MTMR_channel_t *copy_uddtChannelNumber*)

Set the input mode for a specific channel of the Multi-Function Timer.

This function sets the input mode for a specific channel of the Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to configure. Possible values are: <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
<i>copy_uddtChannelNumber</i>	The channel number to configure. Possible values are: <ul style="list-style-type: none"> • MTMR_CH1 • MTMR_CH2 • MTMR_CH3 • MTMR_CH4

Returns

No return.

void MTMR_vSetTimerChannelOutput (EN_MTMR_number_t *copy_uddtTMRNumber*, EN_MTMR_selectedMode_t *copy_uddtTimerMode*, EN_MTMR_channel_t *copy_uddtChannelNumber*)

Set the output mode for a specific channel of the Multi-Function Timer.

This function sets the output mode for a specific channel of the Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to configure. Possible values are: <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
<i>copy_uddtTimerMode</i>	The mode to set for the timer channel. Possible values are: <ul style="list-style-type: none"> • MTMR_MODE_FROZEN • MTMR_MODE_ACTIVE • MTMR_MODE_INACTIVE • MTMR_MODE_TOGGLE • MTMR_MODE_INACTIVE_FORCE

	<ul style="list-style-type: none"> • MTMR_MODE_ACTIVE_FORCE • MTMR_MODE_PWM_MODE1 • MTMR_MODE_PWM_MODE2
<i>copy_uddtChannelNumber</i>	The channel number to configure. Possible values are: <ul style="list-style-type: none"> • MTMR_CH1 • MTMR_CH2 • MTMR_CH3 • MTMR_CH4

Returns

No return.

void MTMR_vSetTimerCMPVal (EN_MTMR_number_t *copy_uddtTMRNumber*, EN_MTMR_channel_t *copy_uddtChannelNumber*, uint32_t *copy_u32CmpValue*)

Set the compare value for the specified channel of the Multi-Function Timer.

This function sets the compare value for the specified channel of the Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to configure. Possible values are: <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
<i>copy_uddtChannelNumber</i>	The channel number to configure. Possible values are: <ul style="list-style-type: none"> • MTMR_CH1 • MTMR_CH2 • MTMR_CH3 • MTMR_CH4
<i>copy_u32CmpValue</i>	The compare value to set.

Returns

No return.

void MTMR_vSetTimerPrescaler (EN_MTMR_number_t *copy_uddtTMRNumber*, uint16_t *copy_u16Value*)

Set the prescaler value for the specified Multi-Function Timer.

This function sets the prescaler value for the specified Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to configure. Possible values are: <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
<i>copy_u16Value</i>	The prescaler value to set.

Returns

No return.

void MTMR_vSetTimerStop (EN_MTMR_number_t *copy_uddtTMRNumber*)

Stop the specified Multi-Function Timer.

This function stops the specified Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to stop. Possible values are: <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
---------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

No return.

void MTMR_vStartTimer (EN_MTMR_number_t *copy_uddtTMRNumber*)

Start the specified Multi-Function Timer.

This function starts the specified Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to start. Possible values are: <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
---------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

No return.

void MTMR_vStopTimer (EN_MTMR_number_t *copy_uddtTMRNumber*)

Stop the specified Multi-Function Timer.

This function stops the specified Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to stop. Possible values are: <ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
---------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

No return.

void MTMR_vTimerCountRst (EN_MTMR_number_t *copy_uddtTMRNumber*)

Reset the count of the specified Multi-Function Timer.

This function resets the count of the specified Multi-Function Timer.

Parameters

<i>copy_uddtTMRNumber</i>	The Multi-Function Timer to reset. Possible values are:
---------------------------	---------------------------------------------------------

<i>mber</i>	<ul style="list-style-type: none"> • MTMR2 • MTMR3 • MTMR4 • MTMR5
-------------	------------------------------------------------------------------------------------------------------------------------------------------

Returns

No return.

tmr_interface.h

Go to the documentation of this file.

```
1
2
3
4
5
6 #ifndef MCAL_TMR_TMR_INTERFACE_H_
7 #define MCAL_TMR_TMR_INTERFACE_H_
8
9 #include "../COMMON/bit_math.h"
10 #include "../COMMON/std_types.h"
11 #include "tmr_private.h"
12 #include "tmr_config.h"
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 void MTMR_vStartTimer(EN_MTMR_number_t copy_uddtTMRNumber);
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42 void MTMR_vStopTimer(EN_MTMR_number_t copy_uddtTMRNumber);
43
44
45
46
47
48
49
50
51
52
53
54
55
56 void MTMR_vSetTimerPrescaler(EN_MTMR_number_t copy_uddtTMRNumber, uint16_t
copy_ul6Value);
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73 void MTMR_vEnableTimerOPM(EN_MTMR_number_t copy_uddtTMRNumber);
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88 void MTMR_vTimerCountRst(EN_MTMR_number_t copy_uddtTMRNumber);
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109 void MTMR_vSetTimerChannelOutput(EN_MTMR_number_t copy_uddtTMRNumber,
EN_MTMR_selectedMode_t copy_uddtTimerMode, EN_MTMR_channel_t copy_uddtChannelNumber);
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140 void MTMR_vSetTimerChannelInput(EN_MTMR_number_t copy_uddtTMRNumber,
EN_MTMR_channel_t copy_uddtChannelNumber);
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156 void MTMR_vSetTimerARR(EN_MTMR_number_t copy_uddtTMRNumber, uint32_t copy_u32Value);
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171 void MTMR_vSetTimerStop(EN_MTMR_number_t copy_uddtTMRNumber);
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186 void MTMR_vClearTimerCount(EN_MTMR_number_t copy_uddtTMRNumber);
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201 void MTMR_vEnableTimerICUInt(EN_MTMR_number_t copy_uddtTMRNumber);
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223 void MTMR_vSetTimerCMPVal(EN_MTMR_number_t copy_uddtTMRNumber, EN_MTMR_channel_t
copy_uddtChannelNumber, uint32_t copy_u32CmpValue);
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244 uint32_t MTMR_vReadCaptureVal(EN_MTMR_number_t copy_uddtTMRNumber, EN_MTMR_channel_t
copy_uddtChannelNumber);
245
246
247
248
249
250
251
252
253 void MTMR3_vCaptureCompareInit(void);
254
255 #endif /* MCAL_TMR_TMR_INTERFACE_H_ */
```


D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/Inc/MCAL/tmr/tmr_private.h File Reference

Data Structures

struct ST_MTMRx_RegistersMap_tMacros

- #define MTMR2_PERIPHERAL_BASE_ADDR (0x40000000)
 - #define MTMR3_PERIPHERAL_BASE_ADDR (0x40000400)
 - #define MTMR4_PERIPHERAL_BASE_ADDR (0x40000800)
 - #define MTMR5_PERIPHERAL_BASE_ADDR (0x40000C00)
 - #define CEN_BIT 0
 - #define OPM_BIT 3
 - #define CC1S_SHIFT 0
 - #define OC1M_SHIFT 4
 - #define CC2S_SHIFT 8
 - #define OC2M_SHIFT 12
 - #define CC3S_SHIFT 0
 - #define OC3M_SHIFT 4
 - #define CC4S_SHIFT 8
 - #define OC4M_SHIFT 12
 - #define CC1IE_BIT 1
 - #define CC1P_BIT 1
 - #define CC1NP_BIT 3
 - #define CC3P_BIT 9
 - #define CC3NP_BIT 11
 - #define CC3EN_BIT 8
 - #define CC1EN_BIT 0
 - #define MTMR2_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t *)MTMR2_PERIPHERAL_BASE_ADDR)
 - #define MTMR3_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t *)MTMR3_PERIPHERAL_BASE_ADDR)
 - #define MTMR4_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t *)MTMR4_PERIPHERAL_BASE_ADDR)
 - #define MTMR5_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t *)MTMR5_PERIPHERAL_BASE_ADDR)
-

Macro Definition Documentation

#define CC1EN_BIT 0

#define CC1IE_BIT 1

#define CC1NP_BIT 3

#define CC1P_BIT 1

#define CC1S_SHIFT 0

#define CC2S_SHIFT 8

#define CC3EN_BIT 8

#define CC3NP_BIT 11

#define CC3P_BIT 9

#define CC3S_SHIFT 0

#define CC4S_SHIFT 8

#define CEN_BIT 0

#define MTMR2_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t *)MTMR2_PERIPHERAL_BASE_ADDR)

#define MTMR2_PERIPHERAL_BASE_ADDR (0x40000000)

#define MTMR3_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t *)MTMR3_PERIPHERAL_BASE_ADDR)

#define MTMR3_PERIPHERAL_BASE_ADDR (0x40000400)

#define MTMR4_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t *)MTMR4_PERIPHERAL_BASE_ADDR)

#define MTMR4_PERIPHERAL_BASE_ADDR (0x40000800)

#define MTMR5_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t *)MTMR5_PERIPHERAL_BASE_ADDR)

#define MTMR5_PERIPHERAL_BASE_ADDR (0x40000C00)

#define OC1M_SHIFT 4

#define OC2M_SHIFT 12

#define OC3M_SHIFT 4

#define OC4M_SHIFT 12

```
#define OPM_BIT 3
```

tmr_private.h

Go to the documentation of this file.

```
1 /*****  
2 // Author      : Sherif Ashraf Khadr  
3 // Project     : Adaptive_Cruise_Control  
4 // File        : tmr_private.h  
5 // Date        : Oct 14, 2023  
6 // GitHub      : https://github.com/sherifkhadr  
7 *****/  
8 #ifndef MCAL_TMR_TMR_PRIVATE_H_  
9 #define MCAL_TMR_TMR_PRIVATE_H_  
10  
11  
12 #define MTMR2_PERIPHERAL_BASE_ADDR (0x40000000)  
13 #define MTMR3_PERIPHERAL_BASE_ADDR (0x40000400)  
14 #define MTMR4_PERIPHERAL_BASE_ADDR (0x40000800)  
15 #define MTMR5_PERIPHERAL_BASE_ADDR (0x40000C00)  
16  
17  
18  
19 typedef struct  
20 {  
21  
22     vuint32_t MTMRx_CR1;  
23     vuint32_t MTMRx_CR2;  
24     vuint32_t MTMRx_SMCR;  
25     vuint32_t MTMRx_DIER;  
26     vuint32_t MTMRx_SR;  
27     vuint32_t MTMRx_EGR;  
28     vuint32_t MTMRx_CCMR1;  
29     vuint32_t MTMRx_CCMR2;  
30     vuint32_t MTMRx_CCER;  
31     vuint32_t MTMRx_CNT;  
32     vuint32_t MTMRx_PSC;  
33     vuint32_t MTMRx_ARR;  
34     vuint32_t MTMRx_RESERVED_1;  
35     vuint32_t MTMRx_CCR1;  
36     vuint32_t MTMRx_CCR2;  
37     vuint32_t MTMRx_CCR3;  
38     vuint32_t MTMRx_CCR4;  
39     vuint32_t MTMRx_RESERVED_2;  
40     vuint32_t MTMRx_DCR;  
41     vuint32_t MTMRx_DMAR;  
42     vuint32_t MTMRx_OR;  
43  
44 }ST_MTMRx_RegistersMap_t;  
45  
46  
47 #define CEN_BIT 0  
48 #define OPM_BIT 3  
49 #define CC1S_SHIFT 0  
50 #define OC1M_SHIFT 4  
51 #define CC2S_SHIFT 8  
52 #define OC2M_SHIFT 12  
53 #define CC3S_SHIFT 0  
54 #define OC3M_SHIFT 4  
55 #define CC4S_SHIFT 8  
56 #define OC4M_SHIFT 12  
57 #define CC1IE_BIT 1  
58 #define CC1P_BIT 1  
59 #define CC1NP_BIT 3  
60 #define CC3P_BIT 9  
61 #define CC3NP_BIT 11  
62 #define CC3EN_BIT 8  
63 #define CC1EN_BIT 0  
64  
65  
66  
67 #define MTMR2_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t  
68 *)MTMR2_PERIPHERAL_BASE_ADDR)  
69 #define MTMR3_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t  
70 *)MTMR3_PERIPHERAL_BASE_ADDR)  
71 #define MTMR4_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t  
72 *)MTMR4_PERIPHERAL_BASE_ADDR)
```

```
70 #define MTMR5_PERIPHERAL ((volatile ST_MTMRx_RegistersMap_t
*)MTMR5_PERIPHERAL_BASE_ADDR)
71
72
73
74
75 #endif /* MCAL_TMR_TMR_PRIVATE_H_ */
```

D:/Programing/Embedded System Diploma/ITI/grad doc/Adaptive_Cruise_Control/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_systemState_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	

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_SAMP LING_BY_16	USART sampling by 16.
MUSART_SAMP LING_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_systemState_t

Enumeration for USART system states.

Enumerator:

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

enum EN_MUSART_transferControl_t

Enumeration for USART transfer control options.

Enumerator:

MUSART_TX_O NLY	USART transmit only.
MUSART_RX_O NLY	USART receive only.
MUSART_TX_R X	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_systemState_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
92 typedef enum
93 {
94     USART_EVEN_PARITY = 0,
95     USART_ODD_PARITY
96 } EN_USART_paritySelectionOption_t;
97
98 typedef enum
99 {
100     USART_DATA_SIZE_8_BIT = 0,
101     USART_DATA_SIZE_9_BIT
102 } EN_USART_dataSizeOptions_t;
103
104 typedef struct
105 {
106     uint8_t clockOutput;
107     uint8_t clockPolarity;
108     uint8_t clockPhase;
109     uint8_t lastBitClockPulse;
110 } ST_USART_clockInit_t;
111
112 typedef struct
113 {
114     EN_USART_transferControl_t copy_uddtTransferDirection;
115     EN_USART_samplingModeOptions_t copy_uddtSamplingModeOption;
116     EN_USART_baudRateOptions_t copy_uddtBuadRateOption;
117     EN_USART_dataSizeOptions_t copy_uddtDataSizeOption;
118     EN_USART_parityControlOption_t copy_uddtParityControl;
119     EN_USART_paritySelectionOption_t copy_uddtParitySelection;
120     EN_USART_stopBitOption_t copy_uddtStopBitSelection;
121     uint8_t copy_HardwareFlowControl;
122     ST_USART_clockInit_t copy_uddtUartClockInit;
123 } ST_USART_cfg_t;
124
125 #endif /* MCAL_USART_USART_CONFIG_H */

```

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

Header file for the Universal Synchronous/Asynchronous Receiver Transmitter (USART) 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_USART_systemState_t USART_uddtInit (ST_MUART_RegistersMap_t *PS_USARTx, ST_USART_cfg_t const *PS_uddtUartCfg)**
Initialize the USART module with the given configuration.
- **EN_USART_systemState_t USART_uddtEnable (ST_MUART_RegistersMap_t *PS_USARTx)**
Enable the USART module.
- **EN_USART_systemState_t USART_uddtDisable (ST_MUART_RegistersMap_t *PS_USARTx)**
Disable the USART module.
- **EN_USART_systemState_t USART_uddtTransmitByte (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t copy_u8ByteToSend)**
Transmit a byte through the USART module.
- **EN_USART_systemState_t USART_uddtTransmitString (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8StringToSend)**
Transmit a string through the USART module.
- **EN_USART_systemState_t USART_uddtReadDataRegister (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8ByteToReceive)**
Read data from the USART data register.
- **EN_USART_systemState_t USART_uddtClearFlags (ST_MUART_RegistersMap_t *PS_USARTx)**
Clear the USART flags.
- **EN_USART_systemState_t USART_uddtReceiveByteSynchNonBlocking (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8ByteToReceive)**
Receive a byte asynchronously in a non-blocking manner.
- **EN_USART_systemState_t USART_uddtReceiveStringAsynchBlocking (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8ByteToReceive)**
Receive a string asynchronously in a blocking manner.

- **EN_MUSART_systemState_t MUSART_uddtReceiveStringSynchNonBlocking** (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t *copy_u8ByteToReceive)
Receive a string asynchronously in a non-blocking manner.
- **EN_MUSART_systemState_t MUSART_RxIntSetStatus** (ST_MUART_RegistersMap_t *PS_USARTx, uint8_t copy_u8Status)
Set the receive interrupt status for the USART module.
- **EN_MUSART_systemState_t MUSART1_uddtSetCallBack** (void(*ptr)(void))
Set the callback function for USART1.
- **EN_MUSART_systemState_t MUSART2_uddtSetCallBack** (void(*ptr)(void))
Set the callback function for USART2.
- **EN_MUSART_systemState_t MUSART6_uddtSetCallBack** (void(*ptr)(void))
Set the callback function for USART6.

Detailed Description

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

Function Documentation

EN_MUSART_systemState_t MUSART1_uddtSetCallBack (void(*) (void) ptr)

Set the callback function for USART1.

This function sets the callback function for USART1.

Parameters

<i>ptr</i>	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_systemState_t MUSART2_uddtSetCallBack (void(*) (void) ptr)

Set the callback function for USART2.

This function sets the callback function for USART2.

Parameters

<i>ptr</i>	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_systemState_t MUSART6_uddtSetCallBack (void*)(void) ptr)

Set the callback function for USART6.

This function sets the callback function for USART6.

Parameters

<i>ptr</i>	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_systemState_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

<i>PS_USARTx</i>	Pointer to the USARTx registers map.
<i>copy_u8Status</i>	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_systemState_t MUSART_uddtClearFlags (ST_MUART_RegistersMap_t * PS_USARTx)

Clear the USART flags.

This function clears the USART flags.

Parameters

<i>PS_USARTx</i>	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_systemState_t MUSART_uddtDisable (ST_MUART_RegistersMap_t * PS_USARTx)

Disable the USART module.

This function disables the USART module.

Parameters

<i>PS_USARTx</i>	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_systemState_t MUSART_uddtEnable (ST_MUART_RegistersMap_t * *PS_USARTx*)

Enable the USART module.

This function enables the USART module.

Parameters

<i>PS_USARTx</i>	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_systemState_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

<i>PS_USARTx</i>	Pointer to the USARTx registers map.
<i>PS_uddtUartCfg</i>	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_systemState_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

<i>PS_USARTx</i>	Pointer to the USARTx registers map.
<i>copy_u8ByteToReceive</i>	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_systemState_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

<i>PS_USARTx</i>	Pointer to the USARTx registers map.
<i>copy_u8ByteToReceive</i>	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_systemState_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

<i>PS_USARTx</i>	Pointer to the USARTx registers map.
<i>copy_u8ByteToReceive</i>	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_systemState_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

<i>PS_USARTx</i>	Pointer to the USARTx registers map.
<i>copy_u8ByteToReceive</i>	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_systemState_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

<i>PS_USARTx</i>	Pointer to the USARTx registers map.
<i>copy_u8ByteToSend</i>	The byte to transmit.

Returns

The system state after transmitting the byte.

- #EN_USART_OK: Byte transmission successful.
- #EN_USART_NOK: Byte transmission failed.
- #EN_USART_PTR_NULL: Null pointer encountered during the operation.

EN_USART_systemState_t USART_uddtTransmitString
(ST_USART_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

<i>PS_USARTx</i>	Pointer to the USARTx registers map.
<i>copy_u8StringToSend</i>	Pointer to the string to transmit.

Returns

The system state after transmitting the string.

- #EN_USART_OK: String transmission successful.
- #EN_USART_NOK: String transmission failed.
- #EN_USART_PTR_NULL: Null pointer encountered during the operation.

usart_interface.h

Go to the documentation of this file.

```
1
2
3
4
5
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
16
17
18
19
20
21
22
23
24
25
26
27
28
29 EN_MUSART_systemState_t MUSART_uddtInit(ST_MUART_RegistersMap_t *PS_USARTx,
30 ST_MUART_cfg_t const *PS_uddtUartCfg);
31
32
33
34
35
36
37
38
39
40
41
42
43 EN_MUSART_systemState_t MUSART_uddtEnable(ST_MUART_RegistersMap_t *PS_USARTx);
44
45
46
47
48
49
50
51
52
53
54
55
56
57 EN_MUSART_systemState_t MUSART_uddtDisable(ST_MUART_RegistersMap_t *PS_USARTx);
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72 EN_MUSART_systemState_t MUSART_uddtTransmitByte(ST_MUART_RegistersMap_t *PS_USARTx,
73 uint8_t copy_u8ByteToSend);
74
75
76
77
78
79
80
81
82
83
84
85
86
87 EN_MUSART_systemState_t MUSART_uddtTransmitString(ST_MUART_RegistersMap_t
88 *PS_USARTx, uint8_t *copy_u8StringToSend);
89
90
91
92
93
94
95
96
97
98
99
100
101
102 EN_MUSART_systemState_t MUSART_uddtReadDataRegister(ST_MUART_RegistersMap_t
103 *PS_USARTx, uint8_t *copy_u8ByteToReceive);
104
105
106
107
108
109
110
111
112
113
114
115
116 EN_MUSART_systemState_t MUSART_uddtClearFlags(ST_MUART_RegistersMap_t *PS_USARTx);
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131 EN_MUSART_systemState_t
132 MUSART_uddtReceiveByteSynchNonBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
133 *copy_u8ByteToReceive);
134
135
136
137
138
139
140
141
142
143
144
145
146 EN_MUSART_systemState_t
147 MUSART_uddtReceiveStringAsynchBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
148 *copy_u8ByteToReceive);
149
150
151
152
153
154
155
156
157
158
159
160
161 EN_MUSART_systemState_t
162 MUSART_uddtReceiveStringSynchNonBlocking(ST_MUART_RegistersMap_t *PS_USARTx, uint8_t
163 *copy_u8ByteToReceive);
164
165
166
167
168
169
170
171
172
173
174
175
176 EN_MUSART_systemState_t MUSART_RxIntSetStatus(ST_MUART_RegistersMap_t *PS_USARTx,
177 uint8_t copy_u8Status);
178
179
180
181
182
183
184
185
186
187
188
189
190 EN_MUSART_systemState_t MUSART1_uddtSetCallBack(void (*ptr)(void));
191
192
193
194
195
196
197
198
199
200
201
202
203
204 EN_MUSART_systemState_t MUSART2_uddtSetCallBack(void (*ptr)(void));
205
206
207
208
209
210
211
212
213
214
215
216
217
218 EN_MUSART_systemState_t MUSART6_uddtSetCallBack(void (*ptr)(void));
219
220
221
222
223
224
225
226
227
228
229
230
231 #endif /* MCAL_USART_USART_INTERFACE_H */
```

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

Data Structures

struct ST_MUART_RegistersMap_tMacros

- #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        : usart_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_ )  
38 ((uint32_t) (((float64_t) (PCLK_)) * 25U) / (4U * ((float64_t) (BAUD_))))  
39 #define UART_DIVMANT_SAMPLING16( PCLK_, BAUD_ )  
40 ((UART_DIV_SAMPLING16( PCLK_, BAUD_ )) / 100U)  
41 #define UART_DIVFRAQ_SAMPLING16( PCLK_, BAUD_ )  
42 (((UART_DIV_SAMPLING16( PCLK_, BAUD_ )) - (UART_DIVMANT_SAMPLING16( PCLK_, BAUD_ )) * 100U) * 16U) + 50U) / 100U  
43 /* UART BRR = mantissa + overflow + fraction  
44 = (UART_DIVMANT << 4) + (UART_DIVFRAQ & 0xF0) + (UART_DIVFRAQ & 0x0FU) */  
45 #define UART_BRR_SAMPLING16( PCLK_, BAUD_ )  
46 ((UART_DIVMANT_SAMPLING16( PCLK_, BAUD_ )) << 4U) + \  
47 (UART_DIVFRAQ_SAMPLING16( PCLK_, BAUD_ )) & 0xF0U) + \  
48 (UART_DIVFRAQ_SAMPLING16( PCLK_, BAUD_ )) & 0x0FU)  
49  
50 #define UART_DIV_SAMPLING8( PCLK_, BAUD_ )  
51 ((uint32_t) (((float64_t) (PCLK_)) * 25U) / (2U * ((float64_t) (BAUD_))))  
52 #define UART_DIVMANT_SAMPLING8( PCLK_, BAUD_ )  
53 ((UART_DIV_SAMPLING8( PCLK_, BAUD_ )) / 100U)  
54 #define UART_DIVFRAQ_SAMPLING8( PCLK_, BAUD_ )  
55 (((UART_DIV_SAMPLING8( PCLK_, BAUD_ )) - (UART_DIVMANT_SAMPLING8( PCLK_, BAUD_ )) * 100U) * 8U) + 50U) / 100U  
56 /* UART BRR = mantissa + overflow + fraction  
57 = (UART_DIVMANT << 4) + ((UART_DIVFRAQ & 0xF8) << 1) + (UART_DIVFRAQ & 0x07U) */  
58 #define UART_BRR_SAMPLING8( PCLK_, BAUD_ )  
59 ((UART_DIVMANT_SAMPLING8( PCLK_, BAUD_ )) << 4U) + \  
60 (UART_DIVFRAQ_SAMPLING8( PCLK_, BAUD_ )) & 0xF8U << 1U) + \  
61 (UART_DIVFRAQ_SAMPLING8( PCLK_, BAUD_ )) & 0x07U)  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  

```



```

58
59 /*****
60 /*          SR BITS Mapping          */
61 /*****
62 /* Parity error                        */
63 #define USART_SR_PE_BIT    0
64 /* Framing error                  */
65 #define USART_SR_FE_BIT    1
66 /* Noise error flag              */
67 #define USART_SR_NE_BIT    2
68 /* Overrun error                  */
69 #define USART_SR_ORE_BIT    3
70 /* IDLE line detected            */
71 #define USART_SR_IDLE_BIT   4
72 /* Read data register not empty */
73 #define USART_SR_RXNE_BIT   5
74 /* Transmission complete         */
75 #define USART_SR_TC_BIT     6
76 /* Transmit data register empty */
77 #define USART_SR_TXE_BIT     7
78 /* LIN break detection flag      */
79 #define USART_SR_LBD_BIT     8
80 /* CTS flag                       */
81 #define USART_SR_CTS_BIT     9
82
83
84
85 /*****
86 /*          CR1 BITS Mapping          */
87 /*****
88 /* Send break bit */
89 #define USART_CR1_SBK_BIT 0
90 /* Receiver Wakeup bit */
91 #define USART_CR1_RWU_BIT 1
92 /* Receiver Enable bit */
93 #define USART_CR1_RE_BIT 2
94 /* Transmitter Enable bit */
95 #define USART_CR1_TE_BIT 3
96 /* IDLE interrupt enable bit */
97 #define USART_CR1_IDLEIE_BIT 4
98 /* RXNEIE interrupt enable bit */
99 #define USART_CR1_RXNEIE_BIT 5
100 /* Transmission complete interrupt enable bit */
101 #define USART_CR1_TCIE_BIT 6
102 /* TXE interrupt enable bit */
103 #define USART_CR1_TXEIE_BIT 7
104 /* PE interrupt enable bit */
105 #define USART_CR1_PEIE_BIT 8
106 /* Parity selection bit */
107 #define USART_CR1_PS_BIT 9
108 /* Parity control enable bit */
109 #define USART_CR1_PCE_BIT 10
110 /* Wakeup method bit */
111 #define USART_CR1_WAKE_BIT 11
112 /* Word length bit */
113 #define USART_CR1_M_BIT 12
114 /* USART enable bit */
115 #define USART_CR1_UE_BIT 13
116 /* USART Oversampling bit */
117 #define USART_CR1_OVER8_BIT 15
118
119 /*****
120 /*          CR2 BITS Mapping          */
121 /*****
122 /* Address of the USART node bits */
123 #define USART_CR2_ADD0_BIT 0
124 #define USART_CR2_ADD1_BIT 1
125 #define USART_CR2_ADD2_BIT 2
126 #define USART_CR2_ADD3_BIT 3
127 /* lin break detection length bit */
128 #define USART_CR2_LBDL_BIT 5
129 /* LIN break detection interrupt enable bit */
130 #define USART_CR2_LBDIE_BIT 6
131 /* Last bit clock pulse bit */
132 #define USART_CR2_LBCL_BIT 8
133 /* Clock phase bit */
134 #define USART_CR2_CPHA_BIT 9

```

```

135 /* Clock polarity bit */
136 #define USART_CR2_CPOL_BIT 10
137 /* Clock enable bit */
138 #define USART_CR2_CLKEN_BIT 11
139 /* STOP bit start */
140 #define USART_CR2_STOP_BIT 12
141 /* STOP bits */
142 #define USART_CR2_STOP0_BIT 12
143 #define USART_CR2_STOP1_BIT 13
144 /* LIN mode enable bit */
145 #define USART_CR2_LINEN_BIT 14
146
147
148 /*****
149 /* CR3 BITS Mapping */
150 *****/
151 /* CTS interrupt enable bit */
152 #define USART_CR3_CTSIE_BIT 10
153 /* CTS enable bit */
154 #define USART_CR3_CTSE_BIT 9
155 /* RTS enable bit */
156 #define USART_CR3_RTSE_BIT 8
157 /* DMA enable transmitter bit */
158 #define USART_CR3_DMAT_BIT 7
159 /* DMA enable receiver bit */
160 #define USART_CR3_DMAR_BIT 6
161 /* Smartcard mode enable bit */
162 #define USART_CR3_SCEN_BIT 5
163 /* Smartcard NACK enable bit */
164 #define USART_CR3_NACK_BIT 4
165 /* Half-duplex selection bit */
166 #define USART_CR3_HDSEL_BIT 3
167 /* IrDA low-power bit */
168 #define USART_CR3_IRLP_BIT 2
169 /* IrDA mode enable bit */
170 #define USART_CR3_IREN_BIT 1
171 /* Error interrupt enable bit */
172 #define USART_CR3_EIE_BIT 0
173
174 #endif /* MCAL_USART_USART_PRIVATE_H_ */

```

Index

INDEX