Modules for ECU1

- Typedefs:

GPT_ValueType

Name	GPT_ValueType
Type	uint
Range	The range of this type is µC dependent (width of the timer register) and has to be described by the supplier.
Description	Type for reading and setting the timer values (in number of ticks).

GPT_ModeType

Name	GPT_ModeType
Type	Enum
Range	TIM_MOD_NORMAL: Normal operation mode of the Timer TIM_MOD_SLEEP: Operation for reduced power operation mode. In sleep mode only wakeup capable channels are available.
Description	Select different power mode

GPT_ChannelType

Name	GPT_ChannelType
Type	uint
Range	The range of this type is µC and application dependent
Description	Numeric ID for channel timer

GPIO_LevelType

Name	GPIO_LevelType
Type	uint8
Range	STD_LOW 0V STD_HIGH 5V or 3.3V
Description	These are the possible levels a DIO channel can have (input or output)

GPIO_PortType

Name	GPIO_PortType
Type	uint8
Range	Shall cover all available DIO Ports.
Description	Numeric ID of a DIO port.

GPIO_ChannelType

Name	GPIO_ChannelType
Type	uint
Range	Shall cover all available DIO channels
Description	Numeric ID of a DIO channel.

Can_HwHandleType

Name	Can_HwHandleType
Type	uint8, uint16
Range	Shall cover all available DIO channels
Description	Numeric ID of a DIO channel.

- MCAL Layer API's

1- Timer

GPT_Init

Name	GPT_Init
Syntax	void GPT_Init(const GPT_ConfigType* ConfigPtr)
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	ConfigPtr: pointer to configuration set
Parameters (out)	None
Return Value	None
Description	Initializes the hardware timer module.

GPT_GetTime

Name	GPT_GetTime
Syntax	GPT_ValueType GPT_GetTime (GPT_ChannelType Channel)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Channel: Numeric id for timer channel
Parameters (out)	None
Return Value	GPT_ValueType : Elapsed timer value (in number of ticks)
Description	Returns the time already elapsed.

GPT_GetMod

Name	GPT_GetMod
Syntax	GPT_ModeType GPT_GetMod (GPT_ChannelType Channel)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Channel: Numeric id for timer channel
Parameters (out)	None
Return Value	GPT_ModeType: mode of timer channel
Description	Returns the timer mode

2- GPIO

GPIO_ReadChannel

Name	GPIO_ReadChannel
Syntax	GPIO_LevelType GPIO_ReadChannel(GPIO_ChannelType ChannelId)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	ChannelId
Parameters (out)	None
Return Value	Dio_LevelType
Description	Return the value of the specified channel

GPIO_WriteChannel

Name	GPIO_WriteChannel
Syntax	GPIO_LevelType GPIO_WriteChannel(GPIO_ChannelType ChannelId, GPIO_LevelType level)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	ChannelId
Parameters (out)	level
Return Value	None
Description	None

3- ADC

Adc_Init

Name	ADC_Init
Syntax	void Adc_Init(const Adc_ConfigType* ConfigPtr)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	ConfigPtr : pointer to configuration set
Parameters (out)	None
Return Value	None
Description	Initializes the ADC hardware units and driver.

Adc_StartGroupConversation

Name	Adc_StartGroupConversion
Syntax	void Adc_StartGroupConversion(Adc_GroupType Group)
Sync/Async	Asynchronous
Reentrancy	Reentrant
Parameters (in)	Group: ADC Channel group
Parameters (out)	None
Return Value	None
Description	Starts the conversion of all channels of the requested ADC Channel group.

Adc_ReadGroup

Name	Adc_ReadGroup
Syntax	Std_ReturnType Adc_ReadGroup(Adc_GroupType Group, Adc_ValueGroupType* DataBufferPtr)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Group: ADC Channel group
Parameters (out)	DataBufferPtr: ADC results of all channels of the selected group are stored in the data buffer addressed with the pointer.
Return Value	Std_ReturnType: E_OK: results are available and written to the data buffer E_NOT_OK: no results are available or development error occured
Description	Reads the group conversion result of the last completed conversion

4- CAN

CAN_Init

Name	Can_Init
Syntax	void Can_Init(const Can_ConfigType* Config)
Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	Config: Pointer to driver configuration.
Parameters (out)	DataBufferPtr: ADC results of all channels of the selected group are stored in the data buffer addressed with the pointer.
Return Value	Std_ReturnType: E_OK: results are available and written to the data buffer E_NOT_OK: no results are available or development error occured
Description	Initializes the module

CAN_SetBaudrate

Name	Can_SetBaudrate
Syntax	Std_ReturnType Can_SetBaudrate(uint8 Controller, uint16 BaudRateConfig)
Sync/Async	Synchronous
Reentrancy	Reentrant for different Controllers. Non reentrant for the same Controller.
Parameters (in)	Controller: CAN controller, whose baud rate shall be set
Parameters (out)	BaudRateConfig
Return Value	None
Description	Std_ReturnType: E_OK: Service request accepted, setting of (new) baud rate started E_NOT_OK: Service request not accepted

CAN_EnableControllerInterrupts

Name	Can_EnableControllerInterrupts
Syntax	void Can_EnableControllerInterrupts(uint8 Controller)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Controller: CAN controller for which interrupts shall be re-enabled
Parameters (out)	None.
Return Value	None
Description	Enables all allowed interrupts

CAN_Write

Name	Can_Write
Syntax	Std_ReturnType Can_Write(Can_HwHandleType Hth, const Can_PduType* PduInfo)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Hth: information which HW-transmit handle shall be used for transmit. Implicitly this is also the information about the controller to use because the Hth numbers are unique inside one hardware unit.
Parameters (out)	PduInfo: Pointer to SDU user memory, Data Length and Identifier.
Return Value	None
Description	Std_ReturnType: E_OK: Write command has been accepted E_NOT_OK: development error occurred

- HAL Layer

LSW_Init

Name	LSW_Init
Syntax	void LSW_Init (const Port_ConfigType* ConfigPtr)
Sync/Async:	Synchronous
Reentrancy	Non Reentrant
Parameters(in):	ConfigPtr : pointer to configuration set
Parameters(out):	None
Return value:	None
Description:	Initializes the port at which the switch will be connected

LSW_GetState

Name	LSW_GetState
Syntax	Dio_LevelType LSW_GetState (Dio_ChannelType SwId)
Sync/Async:	Synchronous
Reentrancy	Reentrant
Parameters(in):	SwId
Parameters(out):	None
Return value:	Dio_LevelType
Description:	Return the value of the specified Switch

LSW_Update

Name	LSW_Update
Syntax	void LSW_Update (Dio_ChannelType SwId)
Sync/Async:	Synchronous
Reentrancy	Reentrant
Parameters(in):	SwId
Parameters(out):	None
Return value:	None
Description:	update the specified Switch state

SpSens_Init

Name	SpSens_Init
Syntax	void SpSens_Init(Dio_PortType Port, Dio_PinType Pin)
Sync/Async:	Synchronous
Reentrancy	Reentrant
Parameters(in):	Port number Pin Number
Parameters(out):	None
Return value:	None
Description:	Initializes the speed sensor.

SpSens_GetSpeed

Name	SpSens_GetSpeed
Syntax	Uint16 SpSens_GetSpeed (Dio_PortType Port, Dio_PinType Pin)
Sync/Async:	Synchronous
Reentrancy	Reentrant
Parameters(in):	ConfigPtr : pointer to configuration set
Parameters(out):	None
Return value:	The speed value
Description:	Initializes the speed sensor.

DSens_Init

Name	DSens_Init
Syntax	void DSens_Init (const Port_ConfigType* ConfigPtr)
Sync/Async:	Synchronous
Reentrancy	Non Reentrant
Parameters(in):	ConfigPtr : pointer to configuration set
Parameters(out):	None
Return value:	None
Description:	Initializes the port at which the Door sensor will be connected

DSens_GetState

Name	DSens_GetState
Syntax	Dio_LevelType DSens_GetState (Dio_ChannelType DoorId)
Sync/Async:	Synchronous
Reentrancy	Reentrant
Parameters(in):	DoorId
Parameters(out):	None
Return value:	Dio_LevelType
Description:	Return the state of the Door sensor

BCM_Init

Name	BCM_Init
Syntax	void BCM_Init (const ComM_ConfigType* ConfigPtr)
Sync/Async:	Synchronous
Reentrancy	Non Reentrant
Parameters(in):	ConfigPtr : pointer to configuration set
Parameters(out):	None
Return value:	None
Description:	Initializes the communication manager

BCM_GetData

Name	BCM_GetData
Syntax	Uint8 BCM_GetData (uint8 BCMId, Uint8*pdata)
Sync/Async:	Synchronous
Reentrancy	Reentrant
Parameters(in):	BCMId : numeric id to determine which comm protocol is used pdata: pointer to data location
Parameters(out):	None
Return value:	Uint8 data length to be read
Description:	Return the received data length

BCM_SetData

Name	BCM_SetData
Syntax	void BCM_SetData (uint8 BCMId, uint8 *pdata, uint8 len)
Sync/Async:	Synchronous
Reentrancy	Reentrant
Parameters(in):	BCMId : numeric id to determine which comm protocol is used pdata: data to be sent len: length of data
Parameters(out):	None
Return value:	void
Description:	Send data over a dedicated communication bus

Modules for ECU2

- Typedefs:

GPT_ValueType

Name	TIM_ValueType
Type	uint
Range	The range of this type is µC dependent (width of the timer register) and has to be described by the supplier.
Description	Type for reading and setting the timer values (in number of ticks).

GPT_ModeType

Name	TIM_ModeType
Type	Enum
Range	TIM_MOD_NORMAL: Normal operation mode of the Timer TIM_MOD_SLEEP: Operation for reduced power operation mode. In sleep mode only wakeup capable channels are available.
Description	Select different power mode

GPT_ChannelType

Name	TIM_ChannelType
Type	uint
Range	The range of this type is µC and application dependent
Description	Numeric ID for channel timer

GPIO_LevelType

Name	Dio_LevelType
Type	uint8
Range	STD_LOW 0V STD_HIGH 5V or 3.3V
Description	These are the possible levels a DIO channel can have (input or output)

GPIO_PortType

Name	Dio_PortType
Type	uint8
Range	Shall cover all available DIO Ports.
Description	Numeric ID of a DIO port.

GPIO_ChannelType

Name	Dio_ChannelType
Type	uint
Range	Shall cover all available DIO channels
Description	Numeric ID of a DIO channel.

CAN_HwHandleType

Name	Can_HwHandleType
Type	uint8, uint16
Range	Shall cover all available DIO channels
Description	Numeric ID of a DIO channel

- MCAL Layer API's

1- Timer

GPT_Init

Name	GPT_Init
Syntax	void GPT_Init(const GPT_ConfigType* ConfigPtr)
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	ConfigPtr: pointer to configuration set
Parameters (out)	None
Return Value	None
Description	Initializes the hardware timer module.

GPT_GetTime

Name	GPT_GetTime
Syntax	GPT_ValueType GPT_GetTime (GPT_ChannelType Channel)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Channel: Numeric id for timer channel
Parameters (out)	None
Return Value	GPT_ValueType: Elapsed timer value (in number of ticks)
Description	Returns the time already elapsed.

GPT_GetMod

Name	GPT_GetMod
Syntax	GPT_ModeType GPT_GetMod (GPT_ChannelType Channel)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Channel : Numeric id for timer channel
Parameters (out)	None
Return Value	TIM_ModeType : mode of timer channel
Description	Returns the timer mode

2- GPIO

GPIO_ReadChannel

Name	Dio_ReadChannel
Syntax	Dio_LevelType Dio_ReadChannel(Dio_ChannelType ChannelId)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	ChannelId
Parameters (out)	None
Return Value	Dio_LevelType
Description	Return the value of the specified channel

GPIO_WriteChannel

Name	Dio_WriteChannel
Syntax	Dio_LevelType Dio_WriteChannel(Dio_ChannelType ChannelId, Dio_LevelType level)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	ChannelId
Parameters (out)	level
Return Value	None
Description	None

3- CAN

CAN_Init

Name	Can_Init
Syntax	void Can_Init(const Can_ConfigType* Config)
Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	Config: Pointer to driver configuration.
Parameters (out)	DataBufferPtr: ADC results of all channels of stored in the data buffer addressed with the
Return Value	Std_ReturnType: E_OK: results are available and written to the data buffer E_NOT_OK: no results are available or development error occured
Description	initializes the module

CAN_SetBaudrate

Name	Can_SetBaudrate
Syntax	Std_ReturnType Can_SetBaudrate(uint8 Controller, uint16 BaudRateConfig)
Sync/Async	Synchronous
Reentrancy	Reentrant for different Controllers. Non reentrant for the same Controller.
Parameters (in)	Controller: CAN controller, whose baud rate shall be set
Parameters (out)	BaudRateConfig
Return Value	None
Description	Std_ReturnType: E_OK: Service request accepted, setting of (new) baud rate started E_NOT_OK: Service request not accepted

CAN_EnableControllerInterrupts

Name	Can_EnableControllerInterrupts
Syntax	void Can_EnableControllerInterrupts(uint8 Controller)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Controller: CAN controller for which interrupts shall be re-enabled
Parameters (out)	None.
Return Value	None
Description	enables all allowed interrupts

CAN_Write

Name	Can_Write
Syntax	Std_ReturnType Can_Write(Can_HwHandleType Hth, const Can_PduType* PduInfo)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Hth: information which HW-transmit handle shall be used for transmit. Implicitly this is also the information about the controller to use because the Hth numbers are unique inside one hardware unit.
Parameters (out)	None
Return Value	Std_ReturnType: E_OK: Write command has been accepted E_NOT_OK: development error occurred
Description	pass a CAN message to CanDrv for transmission

- HAL Layer

Buzz_Init

Name	BUZ_Init
Syntax	void BUZ_Init (const Port_ConfigType* ConfigPtr)
Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	ConfigPtr : pointer to configuration set
Parameters (out)	None
Return Value	None
Description	Initializes the port at which the buzzer will be connected

Buzz_Update

Name	BUZ_Update
Syntax	void BUZ_Update (Dio_ChannelType BUZId, bool val)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	BUZId: buzzer port pin Val : On/Off
Parameters (out)	None
Return Value	None
Description	update the buzzer state(On/Off)

Buzz_GetState

Name	BUZ_GetState
Syntax	bool BUZ_GetState (Dio_ChannelType BUZId)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	BUZId : buzzer port pin
Parameters (out)	None
Return Value	The state of the buzzer On/Off
Description	Returns the buzzer state

Light_Init

Name	Light_Init
Syntax	void Light_Init(const Port_ConfigType* ConfigPtr)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	ConfigPtr : pointer to configuration set
Parameters (out)	None
Return Value	None
Description	Initializes Lights.

Light_Update

Name	Light_Update
Syntax	void Light_Update (Dio_ChannelType LId, bool val)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	LId: Light port pin Val : On/Off
Parameters (out)	None
Return Value	None
Description	update the Light state(On/Off)

Light_GetState

Name	Light_GetState
Syntax	bool Light_GetState (Dio_ChannelType LId)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	Lid : Light port pin
Parameters (out)	None
Return Value	The light state (On/Off)
Description	Returns the state of the light

BCM_Init

Name	BCM_Init
Syntax	void BCM_Init (const ComM_ConfigType* ConfigPtr)
Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	ConfigPtr : pointer to configuration set
Parameters (out)	None
Return Value	None
Description	Initializes the communication manager

BCM_GetData

Name	BCM_GetData
Syntax	Uint8 BCM_GetData (uint8 BCMId, Uint8*pdata)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	BCMId : numeric id to determine which comm protocol is used pdata: pointer to data location
Parameters (out)	None
Return Value	Uint8 data length to be read
Description	Return the received data length

BCM_SetData

Name	BCM_SetData
Syntax	void BCM_SetData (uint8 BCMId, uint8 *pdata, uint8 len)
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	BCMId : numeric id to determine which comm protocol is used pdata: data to be sent len: length of data
Parameters (out)	None
Return Value	void
Description	Send data over a dedicated communication bus