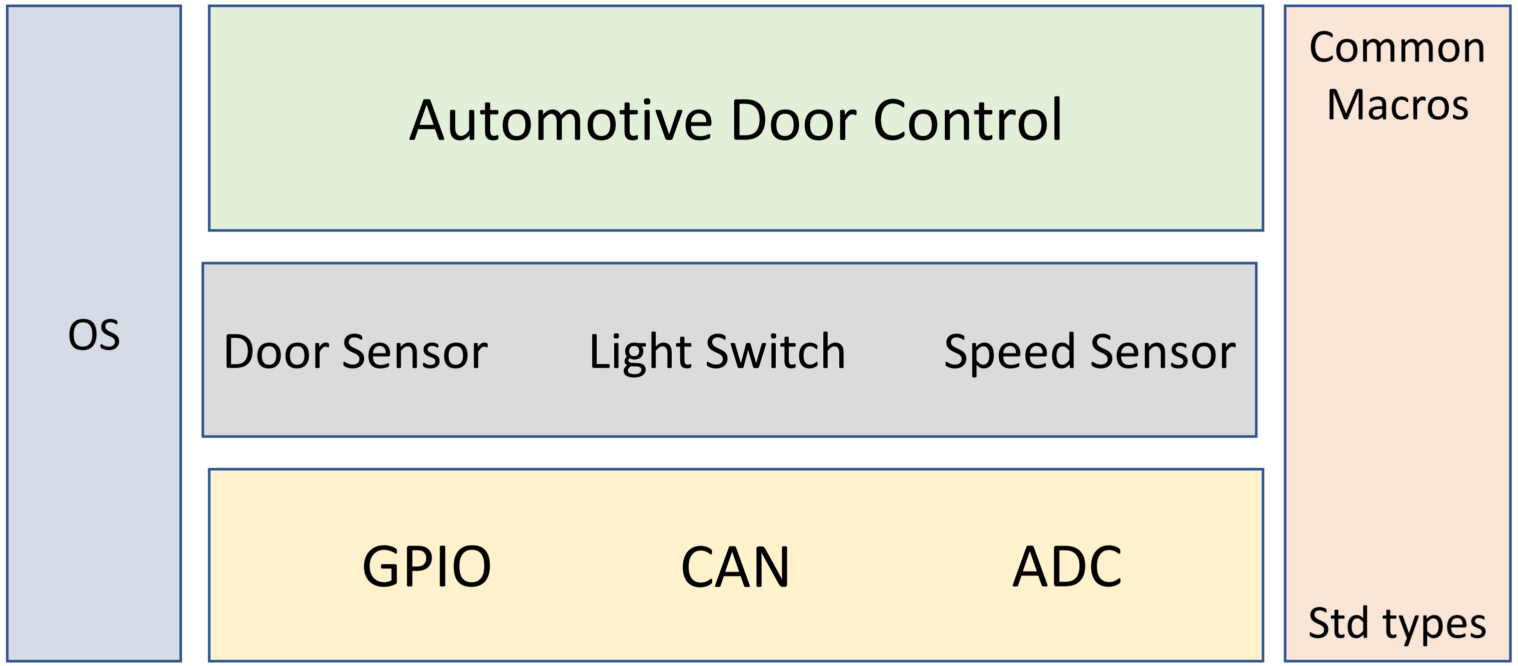
ECU 1


OS
Automotive Door Control
Door Sensor
Light Switch
Speed Sensor
GPIO
CAN
ADC
Common Macros
Std types

GPIO

API – Types

Gpio_ChannelType

Gpio_PortIDType

Gpio_LevelType

Gpio_PortLevelType

API - Functions

void GPIO_Init(const Gpio_ConfigType * ConfigPtr)

Gpio_LevelType GPIO_ReadChannel(Gpio_ChannelType ChannelID)

void GPIO_WriteChannel(Gpio_ChannelType ChannelID, Gpio_LevelType Level)

Gpio_LevelType GPIO_FlipChannel(Gpio_ChannelType ChannelID);

Gpio_PortLevelType GPIO_ReadPort(Gpio_PortIDType PortID)

void GPIO_WritePort(Gpio_PortType PortID, Gpio_PortLevelType Level)

Configurations

- PortPinMode
- PortPinLevelValue
- PortPinDirection
- PortPinInternalAttach

Name	Channel ID
Type	uint8
Range	
Description	Numeric ID of Dio Pins

Name	Gpio_LevelType	
Type	uint8	
Range	0	Physical state 0V
	1	Physical state 5V or 3.3V
Description	These are the possible levels a DIO channel can have (input or output)	

Name	Gpio_ConfigType
Type	Structure
Range	uint8
Description	This structure contains all post-build configurable parameters of the DIO driver. A pointer to this structure is passed to the DIO driver initialization function for configuration.

Name	Gpio_PortIDType
Type	uint8
Range	
Description	Numeric ID of Port Numbers

Name	Gpio_PortLevelType	
Type	uint8	
Range	0	Physical state 0V
	1	Physical state 5V or 3.3V
Description	These are the possible levels a Port can have (input or output)	

Function Name:	GPIO_Init		
Arguments	INPUTS	* ConfigPtr	Gpio_ConfigType
		Pointer to post-build configuration data	
		-	-
		-	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Initializes the GPIO module.		

Function Name:	GPIO_ReadChannel		
Arguments	INPUTS	ChannelID	Gpio_ChannelType
		ID of DIO PIN	
		-	-
		-	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Returns the value of the specified DIO Pin.		

Function Name:	GPIO_WriteChannel		
Arguments	INPUTS	ChannelID	Gpio_ChannelType
		ID of DIO PIN	
		Level	Gpio_LevelType
		Value to be written	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Set a level of a DIO_Pin		

Function Name:	GPIO_ReadPort		
Arguments	INPUTS	PortID	Gpio_PortIDType
		ID of the Port	
		-	-
		-	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Read a level of a Port		

Function Name:	GPIO_WritePort		
Arguments	INPUTS	PortID	Gpio_PortIDType
		ID of the Port	
		Level	Gpio_LevelType
		Value to be written	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Set a level of a Port		

CAN

API – Types

Can_ConfigType “Structure”

API – Functions

void Can_Init(const Can_ConfigType* Config)

Std_ReturnType Can_SetBaudrate(uint8 Controller, uint16 BaudRateConfigID)

void Can_MainFunction_Write(uint32 Data)

void Can_MainFunction_Read(void)

Function Name:	CAN_Init		
Arguments	INPUTS	* ConfigPtr	CAN_ConfigType
		Pointer to a selected configuration structure	
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Initializes the hardware CAN module.		

Function Name:	Can_SetBaudrate		
Arguments	INPUTS	Controller	uint8
		CAN Controller, whose baudrate shall be changed	
		Baudrate	uint16
		Requested baudrate in kbps	
	Output	None	None
	Input/Output	None	None
Return	E_OK	Service request accepted, baudrate change started	
	E_NOK	Service request not accepted	
Description	Set the baudrate of the CAN controller.		

Function Name:	Can_MainFunction_Write		
Arguments	INPUTS	Data	uint32
		Data required to be send	
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Send Data from the CAN controller.		

Function Name:	Can_MainFunction_Read		
Arguments	INPUTS	void	None
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Receive Data from the CAN controller.		

ADC

API – Types

ADC_ConfigType “Structure”

ADC_Prescalar

ADC_RefVolatge

API – Functions

void ADC_Init(const ADC_ConfigType * Config_Ptr);

uint32 ADC_readChannel(uint8 CH_num);

Function Name:	ADC_Init		
Arguments	INPUTS	* ConfigPtr	ADC_ConfigType
		Pointer to a selected configuration structure	
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Initializes the ADC module.		

Function Name:	ADC_readChannel		
Arguments	INPUTS	Ch_num	uint8
		ID of ADC Channel	
		-	-
		-	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Returns the value of the specified ADC Channel		

GPT

API Types	
GPT_ConfigType	Structure Implemenation
GPT_ValueType	uint8

API Functions

void Timer_Init(const GPT_ConfigType * Config_Ptr)

void Timer_Start(GPT_ValueType Value)

void Timer_Stop(GPT_ValueType Value)

Name	Timer_ValueType
Type	uint8
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).

Name	Timer_ConfigType
Type	Structure
Range	
Description	This is the type of the data structure including the configuration set required for initializing the timer unit.

Function Name:	Timer_Init		
Arguments	INPUTS	* ConfigPtr	Timer_ConfigType
		Pointer to a selected configuration structure	
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Initializes the hardware timer module.		

Function Name:	Timer_Start		
Arguments	INPUTS	-	-
		-	
		Value	GPT_ValueType
		Target time in number of ticks.	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Starts a Timer Channel		

Function Name:	Timer_Stop		
Arguments	INPUTS	-	-
		-	
		Value	GPT_ValueType
		Target time in number of ticks.	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Stops a Timer Channel		

Door Sensor

Must include “GPIO Driver”

API – Functions

- void D_Init(void)
- uint8 D_ReadLevel(Gpio_ChannelType ChannelID)

Function Name	D_Init()	
API Type	Init	
Parameters (INPUTS)	None	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Initializes the door sensor module	

Function Name	D_ReadValue()	
API Type	Getter	
Parameters (INPUTS)	ChannelID	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Get the state of door sensor module	

Light Switch

Must include “GPIO Driver”

API – Functions

void L_Init(void)

uint8 L_GetState(Gpio_ChannelType ChannelID)

Light Switch APIs:

Function Name	L_Init()	
API Type	Init	
Parameters (INPUTS)	None	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Initializes the Light Switch module	

Function Name	L_GetState()	
API Type	Getter	
Parameters (INPUTS)	ChannelID	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Get the state of Light Switch module	

Speed Sensor

Must include “ADC Driver”

API – Functions

void S_Init(void)

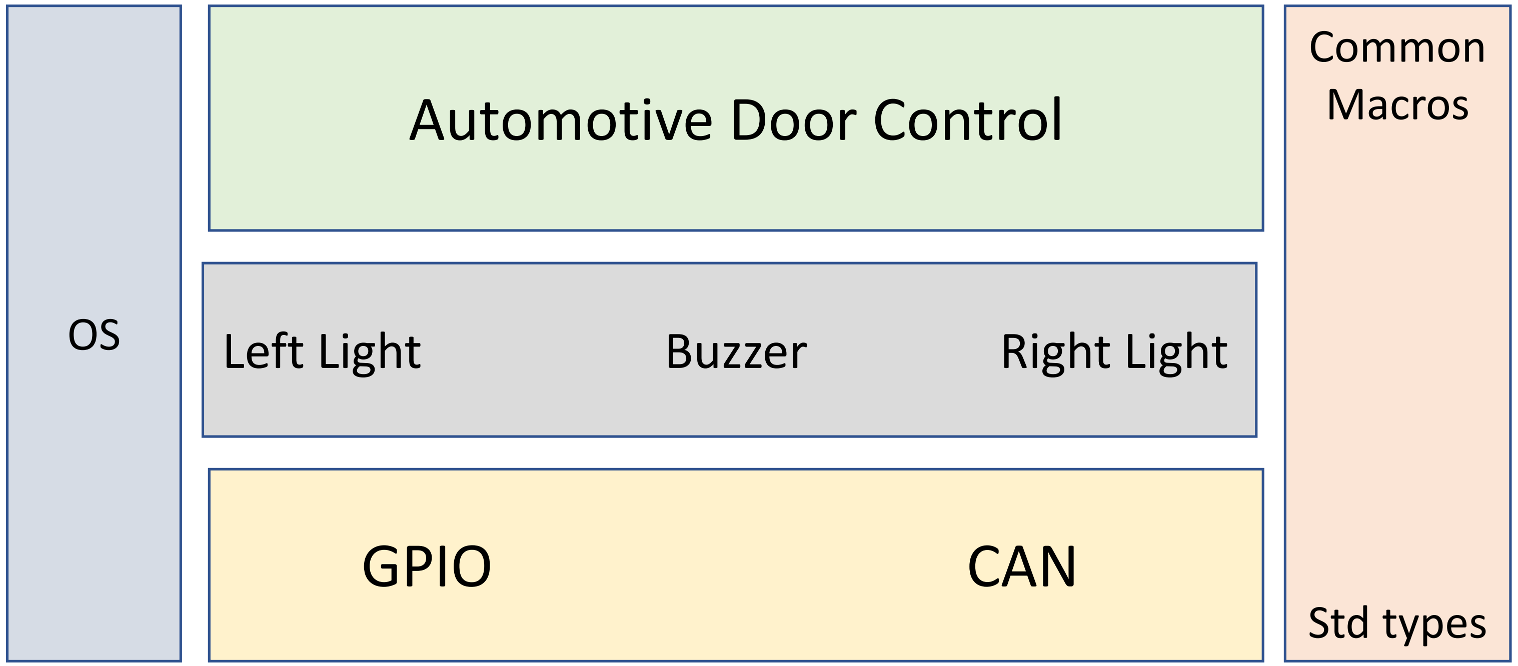
uint8 S_ReadValue(uint8 ADC_Channel)

Speed Sensor APIs:

Function Name	S_Init()	
API Type	Init	
Parameters (INPUTS)	None	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Initializes the Speed sensor module	

Function Name	S_ReadValue()	
API Type	Getter	
Parameters (INPUTS)	ADC_Channel	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Get the state of Speed Sensor module	

ECU 2


OS
Automotive Door Control
Left Light
Buzzer
Right Light
GPIO
CAN
Common Macros
Std types

GPIO

API – Types

Gpio_ChannelType

Gpio_PortType

Gpio_LevelType

Gpio_PortLevelType

API - Functions

void GPIO_Init(const Gpio_ConfigType * ConfigPtr)

Gpio_LevelType GPIO_ReadChannel(Gpio_ChannelType ChannelID)

void GPIO_WriteChannel(Gpio_ChannelType ChannelID, Gpio_LevelType Level)

Gpio_LevelType GPIO_FlipChannel(Gpio_ChannelType ChannelID);

Gpio_PortLevelType GPIO_ReadPort(Gpio_PortType PortID)

void GPIO_WritePort(Gpio_PortType PortID, Gpio_PortLevelType Level)

Configurations

- PortPinMode
- PortPinLevelValue
- PortPinDirection
- PortPinInternalAttach

Name	Channel ID
Type	uint8
Range	
Description	Numeric ID of Dio Pins

Name	Gpio_LevelType	
Type	uint8	
Range	0	Physical state 0V
	1	Physical state 5V or 3.3V
Description	These are the possible levels a DIO channel can have (input or output)	

Name	Gpio_ConfigType
Type	Structure
Range	uint8
Description	This structure contains all post-build configurable parameters of the DIO driver. A pointer to this structure is passed to the DIO driver initialization function for configuration.

Name	Gpio_PortIDType	
Type	uint8	
Range		
Description	Numeric ID of Port Numbers	

Name	Gpio_PortLevelType	
Type	uint8	
Range	0	Physical state 0V
	1	Physical state 5V or 3.3V
Description	These are the possible levels a Port can have (input or output)	

Function Name:	GPIO_Init		
Arguments	INPUTS	* ConfigPtr	Gpio_ConfigType
		Pointer to post-build configuration data	
		-	-
		-	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Initializes the GPIO module.		

Function Name:	GPIO_ReadChannel		
Arguments	INPUTS	ChannelID	Gpio_ChannelType
		ID of DIO PIN	
		-	-
		-	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Returns the value of the specified DIO Pin.		

Function Name:	GPIO_WriteChannel		
Arguments	INPUTS	ChannelID	Gpio_ChannelType
		ID of DIO PIN	
		Level	Gpio_LevelType
		Value to be written	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Set a level of a DIO_Pin		

Function Name:	GPIO_ReadPort		
Arguments	INPUTS	PortID	Gpio_PortIDType
		ID of the Port	
		-	-
		-	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Read a level of a Port		

Function Name:	GPIO_WritePort		
Arguments	INPUTS	PortID	Gpio_PortIDType
		ID of the Port	
		Level	Gpio_LevelType
		Value to be written	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Set a level of a Port		

CAN

API – Types

Can_ConfigType “Structure”

API – Functions

void Can_Init(const Can_ConfigType* Config)

Std_ReturnType Can_SetBaudrate(uint8 Controller, uint16 BaudRateConfigID)

Std_ReturnType Can_SetControllerMode(uint8 Controller, Can_ControllerStateType Transition)

void Can_MainFunction_Write(void)

void Can_MainFunction_Read(void)

Function Name:	CAN_Init		
Arguments	INPUTS	* ConfigPtr	CAN_ConfigType
		Pointer to a selected configuration structure	
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Initializes the hardware CAN module.		

Function Name:	Can_SetBaudrate		
Arguments	INPUTS	Controller	uint8
		CAN Controller, whose baudrate shall be changed	
		Baudrate	uint16
		Requested baudrate in kbps	
	Output	None	None
	Input/Output	None	None
Return	E_OK	Service request accepted, baudrate change started	
	E_NOK	Service request not accepted	
Description	Set the baudrate of the CAN controller.		

Function Name:	Can_MainFunction_Write		
Arguments	INPUTS	Data	uint32
		Data required to be send	
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Send Data from the CAN controller.		

Function Name:	Can_MainFunction_Read		
Arguments	INPUTS	void	None
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Receive Data from the CAN controller.		

GPT

API Types	
GPT_ConfigType	Structure Implemenation
GPT_ValueType	uint8

API Functions

- void Timer_Init(const GPT_ConfigType * Config_Ptr)
- void Timer_Start(GPT_ValueType Value)
- void Timer_Stop(GPT_ValueType Value)

Name	Timer_ValueType
Type	uint8
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).

Name	Timer_ConfigType
Type	Structure
Range	
Description	This is the type of the data structure including the configuration set required for initializing the timer unit.

Function Name:	Timer_Init		
Arguments	INPUTS	* ConfigPtr	Timer_ConfigType
		Pointer to a selected configuration structure	
		None	None
		None	
	Output	None	None
	Input/Output	None	None
Return	E_OK	0	
	E_NOK	1	
Description	Initializes the hardware timer module.		

Function Name:	Timer_Start		
Arguments	INPUTS	-	-
		-	
		Value	GPT_ValueType
		Target time in number of ticks.	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Starts a Timer Channel		

Function Name:	Timer_Stop		
Arguments	INPUTS	-	-
		-	
		Value	GPT_ValueType
		Target time in number of ticks.	
	Output	None	None
	Input/Output	None	None
Return	E_OK	None	
	E_NOK	None	
Description	Stops a Timer Channel		

Left Light

Must include “GPIO Driver”

API – Functions

- void LL_ON(Gpio_ChannelType ChannelID)
- void LL_OFF(Gpio_ChannelType ChannelID)

Function Name	LL_ON()	
API Type	-	
Parameters (INPUTS)	ChannelID	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Make Left Light on	

Function Name	LL_OFF()	
API Type	-	
Parameters (INPUTS)	ChannelID	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Make Left Light off	

Buzzer

Must include “GPIO Driver”

API – Functions

void B_ON(Gpio_ChannelType ChannelID)

void B_OFF(Gpio_ChannelType ChannelID)

Function Name	Buzzer_ON()	
API Type	-	
Parameters (INPUTS)	ChannelID	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Turn on the buzzer	

Function Name	Buzzer_OFF()	
API Type	-	
Parameters (INPUTS)	ChannelID	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Turn off the buzzer	

Right Light

Must include "GPIO Driver"

API – Functions

void RL_ON(Gpio_ChannelType ChannelID)

void RL_OFF(Gpio_ChannelType ChannelID)

Function Name	LR_ON()	
API Type	-	
Parameters (INPUTS)	ChannelID	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Make Right Light ON	

Function Name	LR_OFF()	
API Type	-	
Parameters (INPUTS)	ChannelID	
Parameters (OUTPUT)	None	
Return	E_OK	0
	E_NOK	1
Description	Make Right Light OFF	