

Obstacle avoidance car V1.0 Design

By : Mahmoud Sarhan

Table Of Content:

1. Detailed Requirements
2. Layered architecture
3. System module
 1. Module architecture
 2. MCAL APIs
 - 3.2.1 : DIO API
 - 3.2.1.1 : Flowchart
 - 3.2.1.2 : Type definitions
 - 3.2.1.3 : Services
 - 3.2.2 : Timer API
 - 3.2.2.1 : Flowchart
 - 3.2.2.2 : Type definitions
 - 3.2.2.3 : Services
 3. HAL APIs
 - 3.3.1 : Timer Manager API
 - 3.3.1.1 : Flowchart
 - 3.3.1.2 : Type definitions
 - 3.3.1.3 : Services
 - 3.3.2 : Motor API
 - 3.3.2.1 : Flowchart
 - 3.3.2.2 : Type definitions
 - 3.3.2.3 : Services
 - 3.3.3 : keypad API
 - 3.3.3.1 : Flowchart
 - 3.3.3.2 : Type definitions
 - 3.3.3.3 : Services
 - 3.3.4 : Car control API
 - 3.3.4.1 : Flowchart
 - 3.3.4.2 : Type definitions
 - 3.3.4.3 : Services
 - 3.3.5 : Button API
 - 3.3.5.1 : Flowchart
 - 3.3.5.2 : Type definitions
 - 3.3.5.3 : Services
4. APP APIs
 - 3.4.1 : APP API
 - 3.4.1.1 : Flowchart
 - 3.4.1.2 : Type definitions
 - 3.4.1.3 : Services

1 : Detailed Requirements

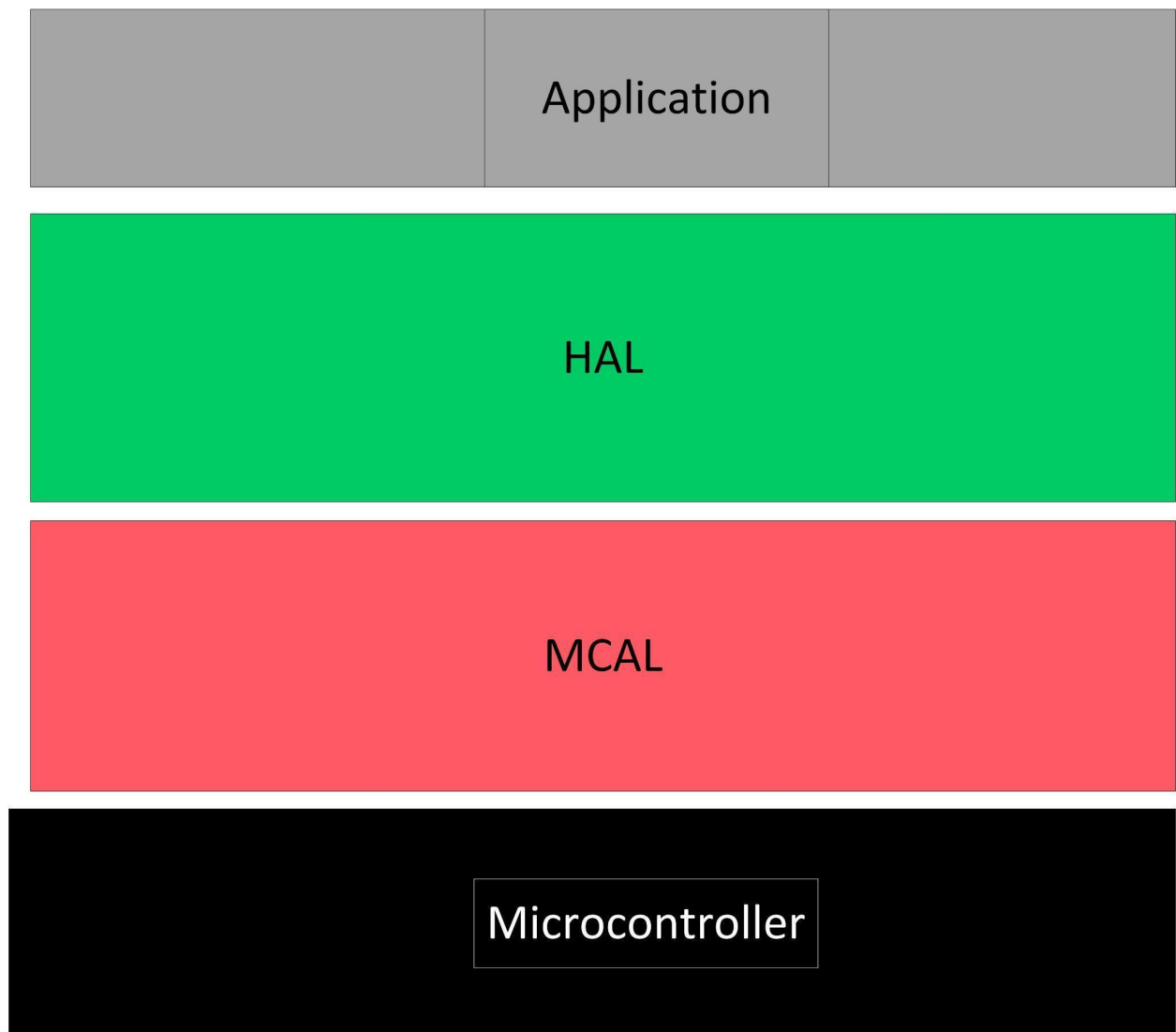
System Requirements:

1. The car starts initially from 0 speed
2. The default rotation direction is to the right
3. Press PB2 to start or stop the robot
4. After Pressing Start:
 1. The LCD will display a centered message in line 1 "Set Def. Rot."
 2. The LCD will display the selected option in line 2 "Right"
 3. The robot will wait for 5 seconds to choose between Right and Left
 1. When PB1 is pressed once, the default rotation will be Left and the LCD line 2 will be updated
 2. When PB1 is pressed again, the default rotation will be Right and the LCD line 2 will be updated
 3. For each press the default rotation will changed and the LCD line 2 is updated
 4. After the 5 seconds the default value of rotation is set
 4. The robot will move after 2 seconds from setting the default direction of rotation.
 5. For No obstacles or object is far than 70 centimeters:
 1. The robot will move forward with 30% speed for 5 seconds
 2. After 5 seconds it will move with 50% speed as long as there was no object or objects are located at more than 70 centimeters distance
 3. The LCD will display the speed and moving direction in line 1: "Speed:00% Dir: F/B/R/S", F: forward, B: Backwards, R: Rotating, and S: Stopped
 4. The LCD will display Object distance in line 2 "Dist.: 000 Cm"
 6. For Obstacles located between 30 and 70 centimeters
 1. The robot will decrease its speed to 30%
 2. LCD data is updated
 7. For Obstacles located between 20 and 30 centimeters
 1. The robot will stop and rotates 90 degrees to right/left according to the chosen configuration
 2. The LCD data is updated
 8. For Obstacles located less than 20 centimeters
 1. The robot will stop, move backwards with 30% speed until distance is greater than 20 and less than 30
 2. The LCD data is updated
 3. Then preform point 8

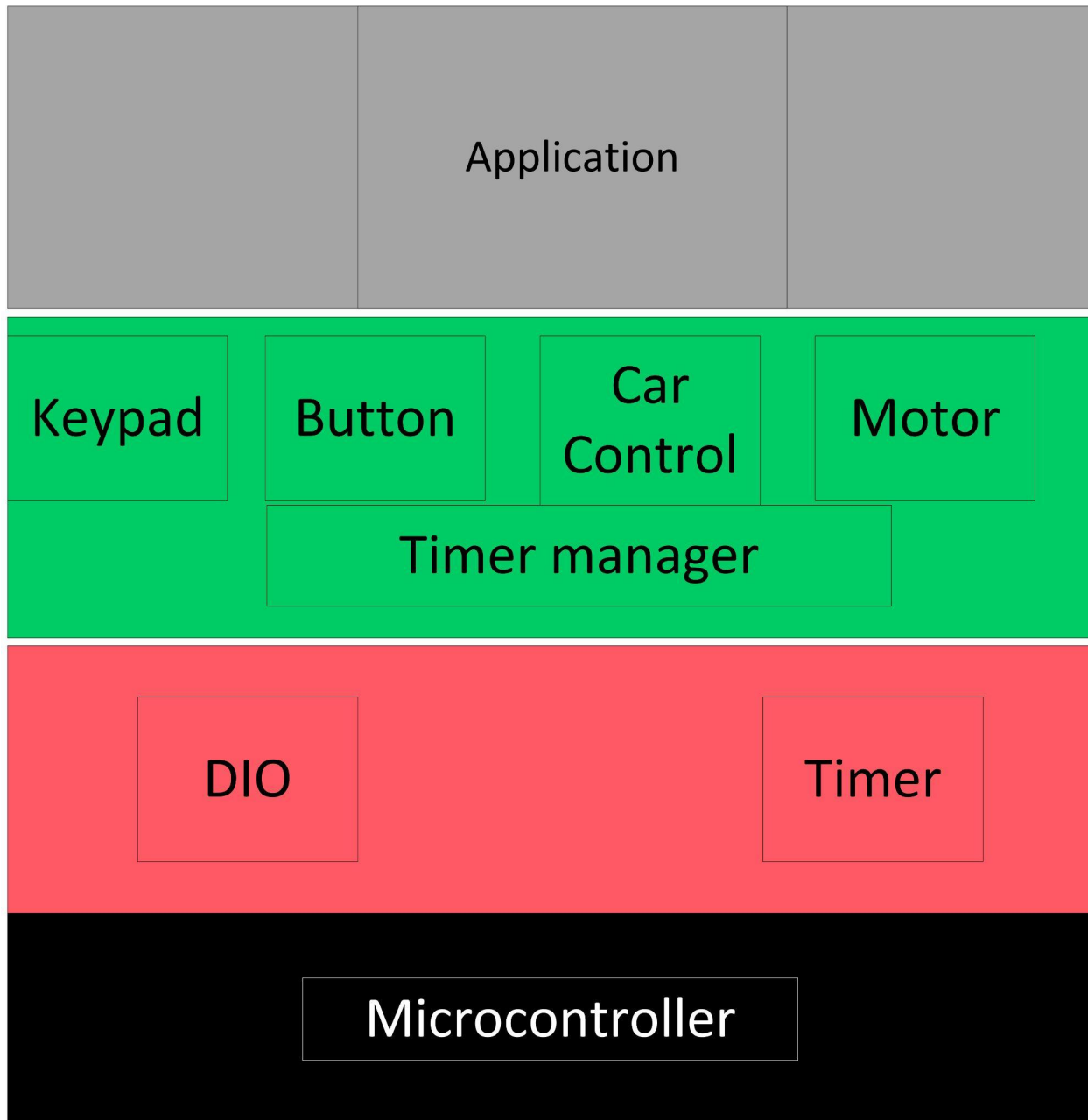
9. Obstacles surrounding the robot (Bonus)

1. If the robot rotated for 360 degrees without finding any distance greater than 20 it will stop
2. LCD data will be updated.
3. The robot will frequently (each 3 seconds) check if any of the obstacles was removed or not and move in the direction of the furthest object

2 : Layered architecture



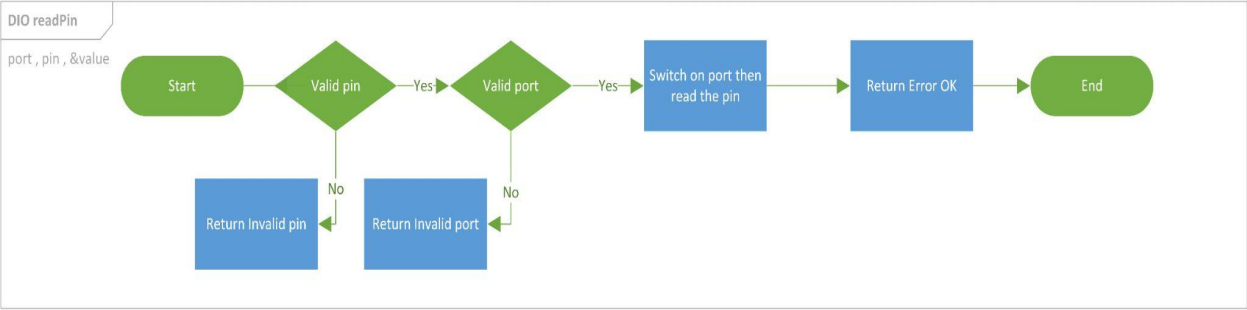
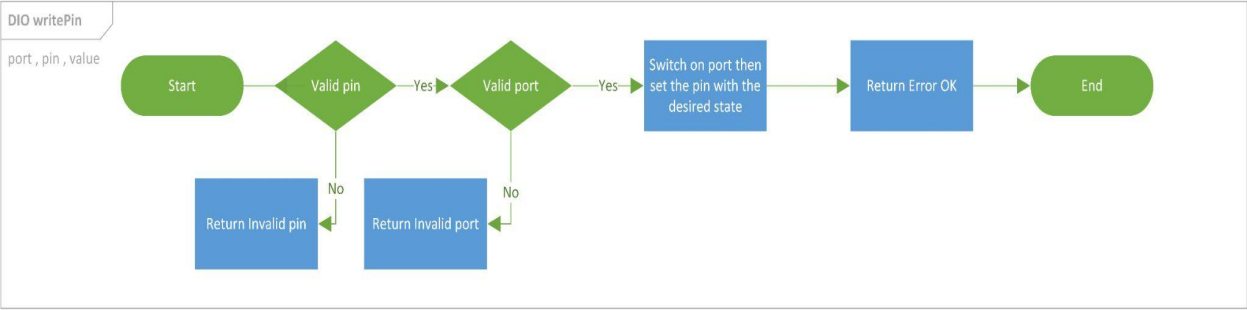
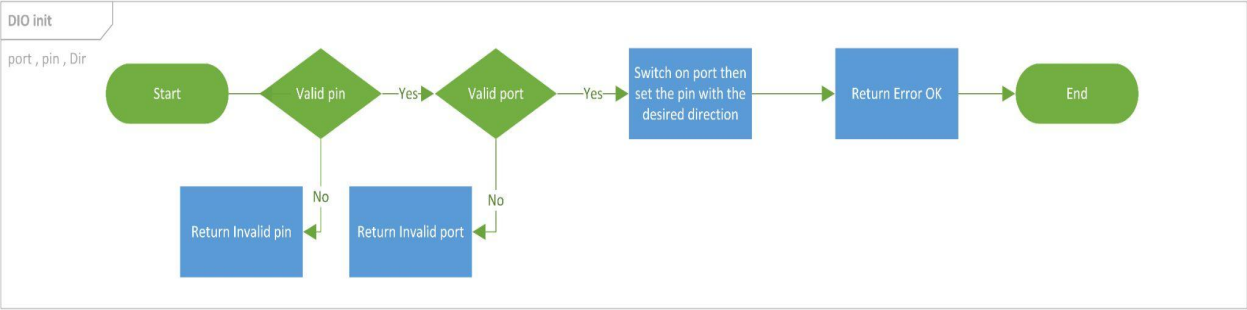
3 : System modules



3.2: MCAL APIs

3.2.1: DIO API:

3.2.1.1 :Flowcharts:



3.2.1.2 : Type definitions:

- en_dioPinsType

Name	en_dioPinsType
------	----------------

Type	Enumeration
Range	Shall contain all pins ID
Description	en_dioPinsType
Available via	dio.h

- en_dioPortsType

Name	en_dioPortsType
Type	Enumeration
Range	Shall contain all ports ID
Description	en_dioPortsType
Available via	dio.h

- u8_en_dioErrors

Name	u8_en_dioErrorsType		
Type	Enumeration		
Range	DIO_E_OK	0x00	DIO error OK
	DIO_InvalidPin	0x01	DIO error, invalid pin number.
	DIO_InvalidPort	0x02	DIO error, invalid port number.
Description	u8_en_dioErrors		
Available via	dio.h		

- u8_en_dioLevelType

Name	u8_en_dioLevelType
------	--------------------

Type	Enumeration		
Range	STD_LOW	0x00	Physical state 0V
	STD_HIGH	0x01	Physical state 5V or 3.3V.
Description	u8_en_dioLevelType		
Available via	dio.h		

- u8_en_dioDirType

Name	u8_en_dioDirType		
Type	Enumeration		
Range	STD_INPUT	0x00	Set pin as input pin
	STD_OUTPUT	0x01	Set pin as output pin
Description	u8_en_dioDirType		
Available via	dio.h		

3.2.1.3 : Services affecting the hardware unit:

- DIO_readPIN

Service name	DIO_readPIN		
Syntax	<pre>u8_en_dioErrors DIO_readPIN (en_dioPortsType port, en_dioPinsType pin, uint8_t* value);</pre>		
Parameters (in)	Port, pin	Channel ID	
	value	Pointer to store the level	STD_HIGH
			STD_LOW

Return	u8_en_dioErrors	DIO_E_OK
		DIO_InvalidPin
		DIO_InvalidPort
Description	This Function gets the level of the pin	

- This function shall return DIO_InvalidPin if pin number is invalid.
- This function shall return DIO_InvalidPort if port number is invalid.

- DIO_writePIN

Service name	DIO_writePIN		
Syntax	u8_en_dioErrors DIO_writePIN (en_dioPortsType port, en_dioPinsType pin, u8_en_dioLevelType state);		
Parameters (in)	Port, pin	Channel ID	
	state	Value to be set	STD_HIGH
			STD_LOW
Return	u8_en_dioErrors	DIO_E_OK	
		DIO_InvalidPin	
		DIO_InvalidPort	
	Description	This Function sets the level of the pin	

- This function shall return DIO_InvalidPin if pin number is invalid.
- This function shall return DIO_InvalidPort if port number is invalid.

- DIO_init

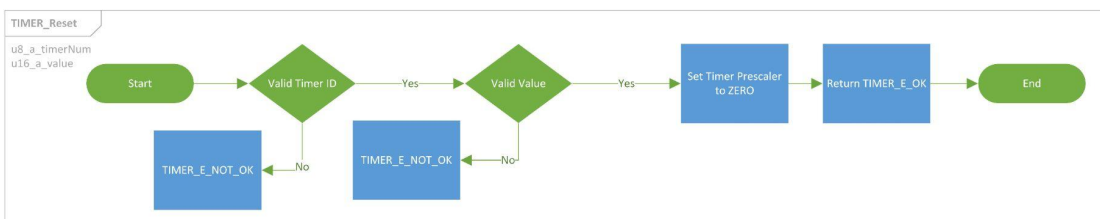
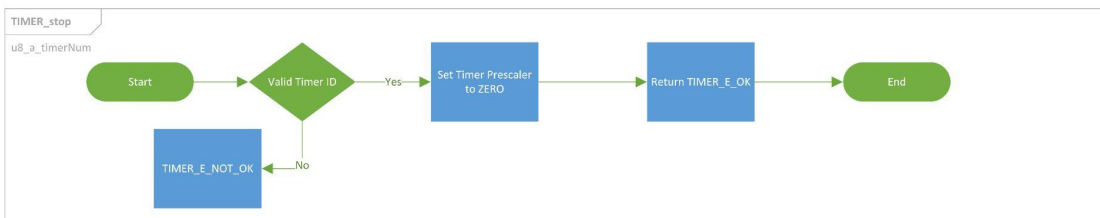
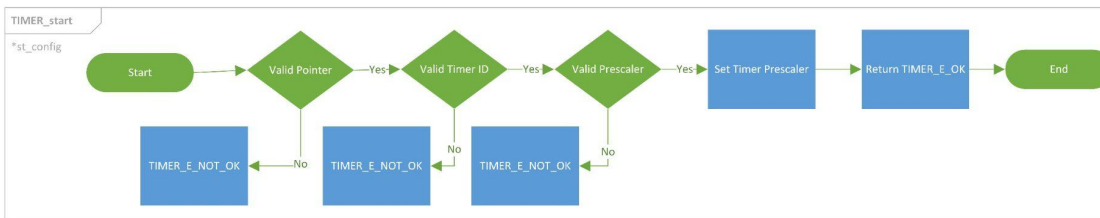
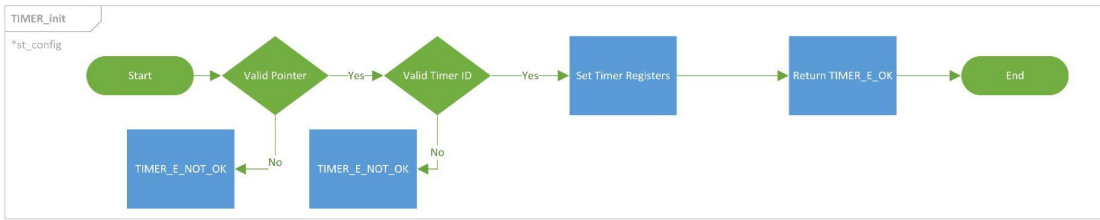
Service name	DIO_init		
Syntax	u8_en_dioErrors DIO_init (en_dioPortsType port, en_dioPinsType pin, u8_en_dioDirType direction);		

);						
Parameters (in)	Port, pin	Channel ID					
	direction	Value to be set	STD_INPUT				
			STD_OUTPUT				
Return	<table><tr><td rowspan="3">DIO_Errors</td><td>DIO_E_OK</td></tr><tr><td>DIO_InvalidPin</td></tr><tr><td>DIO_InvalidPort</td></tr></table>			DIO_Errors	DIO_E_OK	DIO_InvalidPin	DIO_InvalidPort
					DIO_Errors	DIO_E_OK	
						DIO_InvalidPin	
				DIO_InvalidPort			
Description	This Function sets the Direction of the pin						

- This function shall return DIO_InvalidPin if pin number is invalid
- This function shall return DIO_InvalidPort if port number is invalid.

3.2.2: Timer API:

3.2.2.1 :Flowcharts:



3.2.2.2 : Type definitions:

- st_timerConfigType

Name	st_timerConfigType
Type	Structure

Range	Shall contain required timer configuration
Description	st_timerConfigType
Available via	timer_types.h

- u8_en_timerErrorsType

Name	u8_en_timerErrorsType		
Type	Enumeration		
Range	TIMER_E_OK	0x00	Timer error OK
	TIMER_E_NOT_OK	0x03	Timer error
Description	u8_en_timerErrorsType		
Available via	timer_types.h		

- u8_en_timerPrescalerType

Name	u8_en_timerPrescalerType		
Type	Enumeration		
Range	Shall Contain all Prescaler values		
Description	u8_en_timerPrescalerType		
Available via	timer_types.h		

- u8_en_timerNumberType

Name	u8_en_timerNumberType		
Type	Enumeration		

Range	Shall Contain all Timers IDs
Description	u8_en_timerNumberType
Available via	timer_types.h

3.2.2.3 : Services affecting the hardware unit

- **TIMER_init**

Service name	TIMER_init					
Syntax	u8_en_timerErrorsType TIMER_init (st_timerConfigType* st_config);					
Parameters (in)	st_config	Pointer to the configuration structure				
Return	<table><tr><td rowspan="2">u8_en_timerErrorsType</td><td>TIMER_E_OK</td></tr><tr><td>TIMER_E_NOT_OK</td></tr></table>			u8_en_timerErrorsType	TIMER_E_OK	TIMER_E_NOT_OK
u8_en_timerErrorsType	TIMER_E_OK					
	TIMER_E_NOT_OK					
Description	This Function Initialize TIMER module					

- This function shall return TIMER_E_NOK if st_config is NULL
- This function shall return TIMER_E_NOK if any of the configuration elements is invalid.

- **TIMER_start**

Service name	TIMER_start				
Syntax	u8_en_timerErrorsType TIMER_start (st_timerConfigType* st_config);				
Parameters (in)	st_config	Pointer to the configuration structure			
Return	u8_en_timerErrorsType	<table><tr><td>TIMER_E_OK</td></tr><tr><td>TIMER_E_NOT_OK</td></tr></table>		TIMER_E_OK	TIMER_E_NOT_OK
TIMER_E_OK					
TIMER_E_NOT_OK					

Description	This Function start TIMER
-------------	---------------------------

- This function shall return TIMER_E_NOK if st_config is NULL
- This function shall return TIMER_E_NOK if any of the configuration elements is invalid.

- **TIMER_stop**

Service name	TIMER_stop					
Syntax	u8_en_timerErrorsType TIMER_stop (u8_en_timerNumberType u8_a_timerNum);					
Parameters (in)	u8_a_timerNum	Pointer to the configuration structure				
Return	<table><tr><td rowspan="2">u8_en_timerErrorsType</td><td>TIMER_E_OK</td></tr><tr><td>TIMER_E_NOT_OK</td></tr></table>			u8_en_timerErrorsType	TIMER_E_OK	TIMER_E_NOT_OK
u8_en_timerErrorsType	TIMER_E_OK					
	TIMER_E_NOT_OK					
Description	This Function stop TIMER					

- This function shall return TIMER_E_NOK if u8_a_timerNum is invalid

- **TIMER_reset**

Service name	TIMER_reset					
Syntax	u8_en_timerErrorsType TIMER_reset (st_timerConfigType* st_config);					
Parameters (in)	st_config	Timer ID				
Return	<table><tr><td rowspan="2">u8_en_timerErrorsType</td><td>TIMER_E_OK</td></tr><tr><td>TIMER_E_NOT_OK</td></tr></table>			u8_en_timerErrorsType	TIMER_E_OK	TIMER_E_NOT_OK
u8_en_timerErrorsType	TIMER_E_OK					
	TIMER_E_NOT_OK					
Description	This Function reset the TIMER					

- This function shall return TIMER_E_NOK if st_config is NULL

- This function shall return `TIMER_E_NOK` if any of the configuration elements is invalid.
- `TIMER_setCallback`

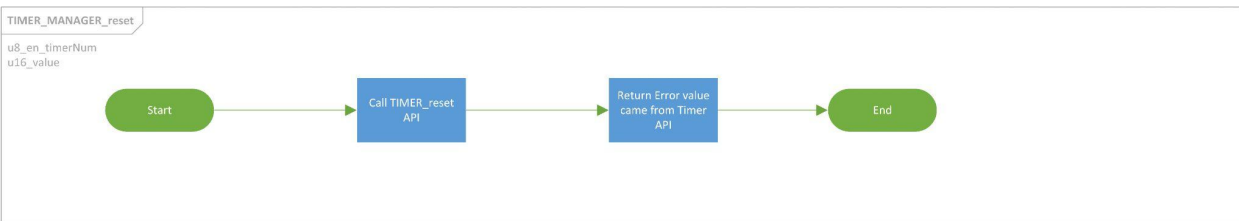
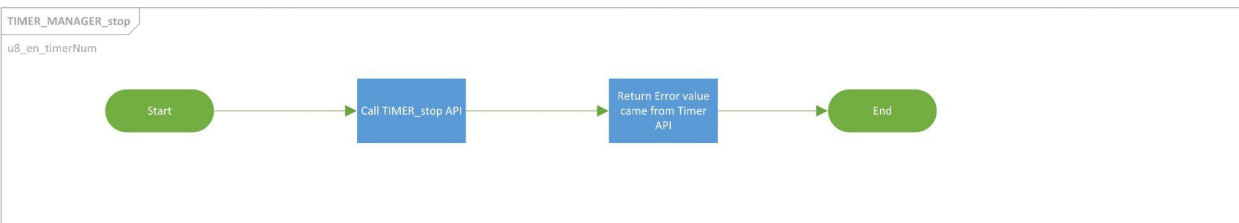
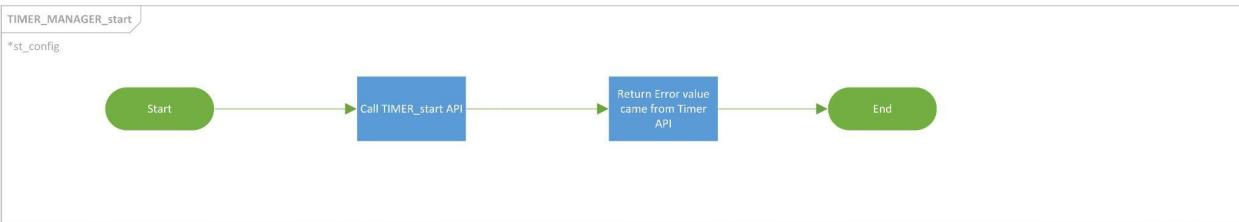
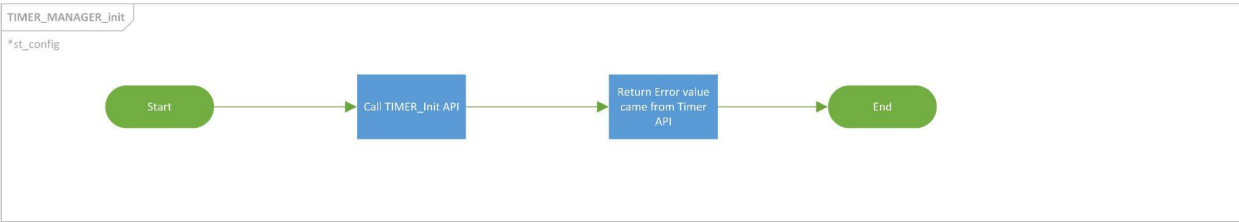
Service name	TIMER_setCallBack		
Syntax	u8_en_timerErrorsType TIMER_setCallBack (void(*a_timerCallBack)(void), u8_en_timerNumberType u8_a_timerNum);		
Parameters (in)	*a_timerCallBack	Pointer to the Callback function	
	u8_a_timerNum	Timer ID	
Return	u8_en_timerErrorsType	TIMER_E_OK	
		TIMER_E_NOT_OK	
Description	This Function reset the TIMER		

- This function shall return `TIMER_E_NOK` if `a_timerCallback` is `NULL`
- This function shall return `TIMER_E_NOK` if `u8_a_timerNum` is invalid.

3.3 : HAL APIs

3.3.1: Timer Manager API:

3.3.1.1 :Flowcharts:



3.3.1.2 : Type definitions:

Imported from Timer Module

3.3.1.3 : Services affecting the hardware unit

- **TIMER_Manager_init**

Service name	TIMER_Manager_init
--------------	--------------------

Syntax	u8_en_timerErrorsType TIMER_Manager_init (st_timerConfigType* st_config);					
Parameters (in)	st_config	Pointer to the configuration structure				
Return	<table><tr><td rowspan="2">u8_en_timerErrorsType</td><td>TIMER_E_OK</td></tr><tr><td>TIMER_E_NOT_OK</td></tr></table>			u8_en_timerErrorsType	TIMER_E_OK	TIMER_E_NOT_OK
u8_en_timerErrorsType	TIMER_E_OK					
	TIMER_E_NOT_OK					
Description	This Function Initialize TIMER module					

- This function shall return TIMER_E_NOK if st_config is NULL
- This function shall return TIMER_E_NOK if any of the configuration elements is invalid.

- TIMER_Manager_start

Service name	TIMER_Manager_start		
Syntax	u8_en_timerErrorsType TIMER_Manager_start (

- This function shall return TIMER_E_NOK if st_config is NULL
- This function shall return TIMER_E_NOK if any of the configuration elements is invalid.

- TIMER_Manager_stop

Service name	TIMER_Manager_stop	
Syntax	u8_en_timerErrorsType TIMER_Manager_stop (u8_en_timerNumberType u8_en_timerNum);	

Parameters (in)	u8_en_timerNum	Timer ID
Return	u8_en_timerErrorsType	<div>TIMER_E_OK</div> <div>TIMER_E_NOT_OK</div>
Description	This Function stop TIMER	

- This function shall return TIMER_E_NOK if u8_en_timerNum is invalid

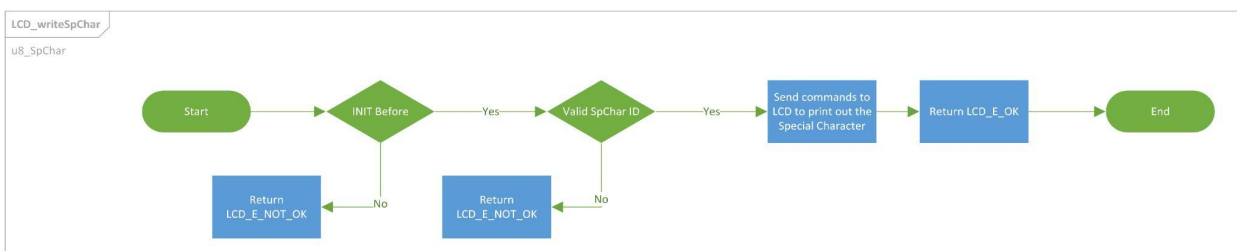
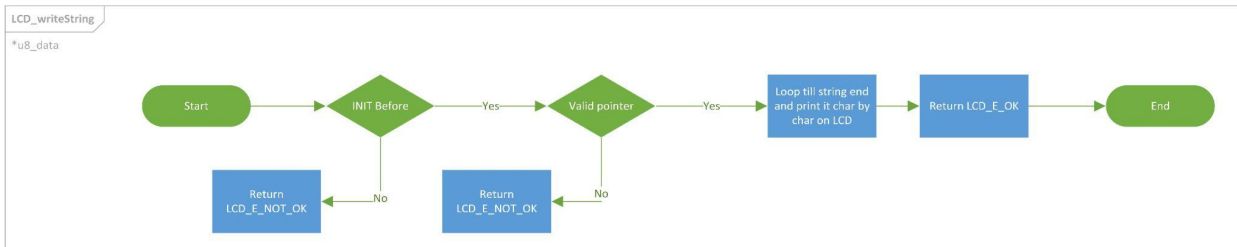
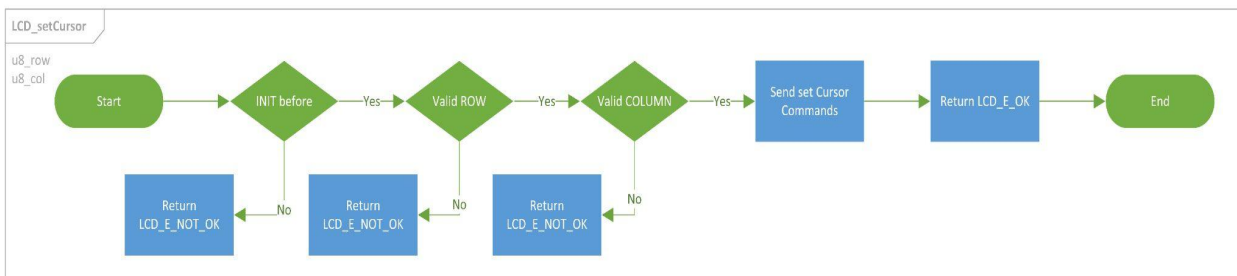
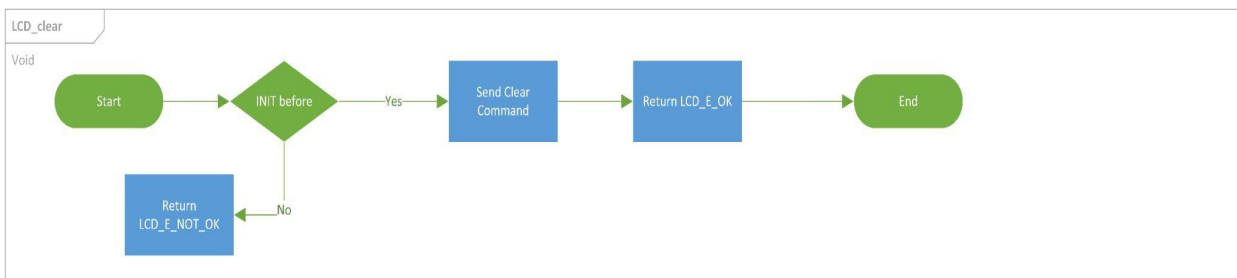
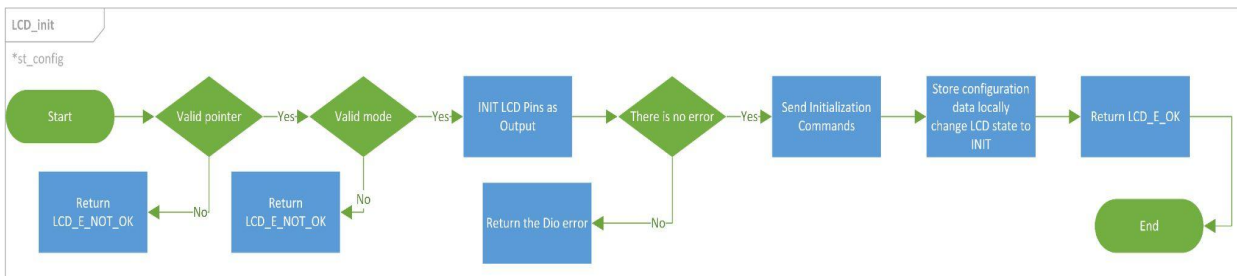
- **TIMER_Manager_reset**

Service name	TIMER_Manager_reset	
Syntax	u8_en_timerErrorsType TIMER_Manager_reset (st_timerConfigType* st_config);	
Parameters (in)	st_config	Pointer to the configuration structure
Return	u8_en_timerErrorsType	<div>TIMER_E_OK</div> <div>TIMER_E_NOT_OK</div>
Description	This Function reset the TIMER with the initial value	

- This function shall return TIMER_E_NOK if st_config is NULL
- This function shall return TIMER_E_NOK if any of the configuration elements is invalid.

3.3.2: LCD API:

3.3.2.1 :Flowcharts:



3.3.2.2 : Type definitions:

- st_lcdConfigType

Name	st_lcdConfigType
Type	Structure
Range	Shall contain required LCD configuration
Description	st_lcdConfigType
Available via	lcd.h

- u8_en_lcdErrorsType

Name	u8_en_lcdErrorsType		
Type	Enumeration		
Range	LCD_E_OK	0x00	LCD error OK
	LCD_E_NOT_OK	0x05	LCD error
Description	u8_en_lcdErrorsType		
Available via	lcd.h		

- u8_en_lcdModeType

Name	u8_en_lcdModeType		
Type	Enumeration		
Range	LCD_4_BIT_MODE	0x00	LCD 4-bit mode
	LCD_8_BIT_MODE	0x01	LCD 8-bit mode
	LCD_INVALID_MODE	0X02	LCD invalid mode
Description	u8_en_lcdModeType		
Available via	lcd.h		

- u8_en_lcdSpCharType

Name	u8_en_lcdSpCharType
Type	Enumeration
Range	Shall contain all special characters IDs
Description	u8_en_lcdSpCharType
Available via	lcd.h

3.3.2.3 : Services affecting the hardware unit

- LCD_init

Service name	LCD_init					
Syntax	u8_en_lcdErrorsType LCD_init (st_lcdConfigType* st_config);					
Parameters (in)	st_config	Pointer to the configuration structure				
Return	<table><tr><td rowspan="2">u8_en_lcdErrorsType</td><td>LCD_E_OK</td></tr><tr><td>LCD_E_NOT_OK</td></tr></table>			u8_en_lcdErrorsType	LCD_E_OK	LCD_E_NOT_OK
u8_en_lcdErrorsType	LCD_E_OK					
	LCD_E_NOT_OK					
Description	This Function Initialize LCD module					

- This function shall return LCD_E_NOK if st_config is NULL
- This function shall return LCD_E_NOK if any of the configuration elements is invalid.

- LCD_clear

Service name	LCD_clear
Syntax	u8_en_lcdErrorsType LCD_clear (void

);	
Parameters (in)	None	
Return	u8_en_lcdErrorsType	LCD_E_OK
		LCD_E_NOT_OK
Description	This Function Clear LCD	

- LCD_setCursor

Service name	LCD_setCursor	
Syntax	<pre>u8_en_lcdErrorsType LCD_setCursor (uint8_t u8_row, uint8_t u8_col);</pre>	
Parameters (in)	u8_row	The desired row to set cursor
	u8_col	The desired column to set cursor
Return	u8_en_lcdErrorsType	LCD_E_OK
		LCD_E_NOT_OK
Description	This Function sets the cursor location on LCD	

- LCD_writeString

Service name	LCD_writeString	
Syntax	<pre>u8_en_lcdErrorsType LCD_writeString (uint8_t* u8_data);</pre>	
Parameters (in)	u8_data	Pointer to string to it print on screen
Return	u8_en_lcdErrorsType	LCD_E_OK

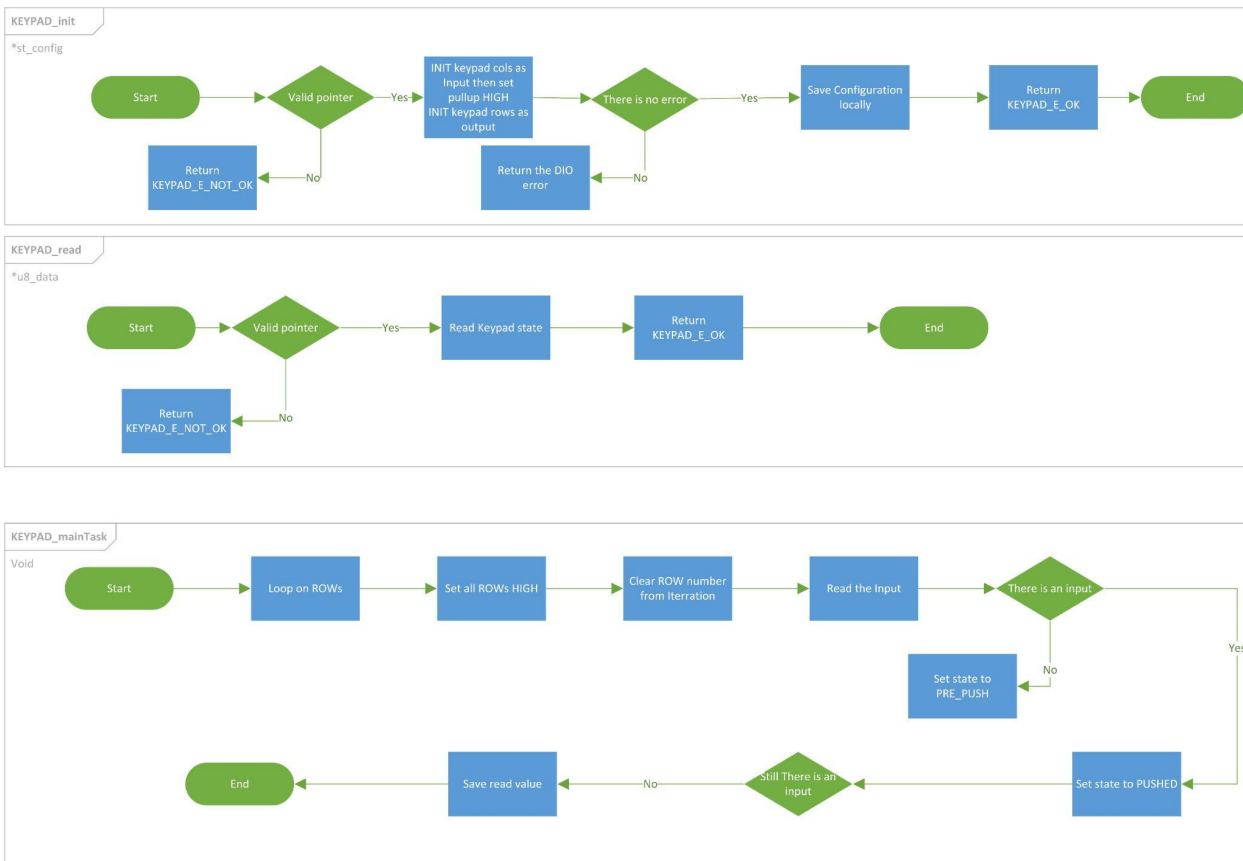
	<table> <tr> <td></td><td>LCD_E_NOT_OK</td></tr> </table>		LCD_E_NOT_OK
	LCD_E_NOT_OK		
Description	This Function write a string on LCD		

- LCD_writeSpChar

Service name	LCD_writeSpChar	
Syntax	u8_en_lcdErrorsType LCD_writeSpChar (u8_en_lcdSpCharType u8_SpChar);	
Parameters (in)	u8_SpChar	Special character ID to it print on screen
Return	u8_en_lcdErrorsType	LCD_E_OK
		LCD_E_NOT_OK
Description	This Function write a special character on LCD	

3.3.3 : Keypad API :

3.3.3.1 :Flowcharts:



3.3.3.2 : Type definitions:

- st_keypadConfigType

Name	st_keypadConfigType
Type	Structure
Range	Shall contain required Keypad configuration
Description	st_keypadConfigType
Available via	keypad.h

- u8_en_keypadErrorsType

Name	u8_en_keypadErrorsType		
Type	Enumeration		
Range	KEYPAD_E_OK	0x00	Keypad error OK
	KEYPAD_E_NOT_OK	0x07	Keypad error
Description	u8_en_keypadErrorsType		
Available via	keypad.h		

3.3.3.3 : Services affecting the hardware unit

- KEYPAD_init

Service name	KEYPAD_init		
Syntax	u8_en_keypadErrorsType KEYPAD_init (st_keypadConfigType* st_config);		
Parameters (in)	st_config	Pointer to the configuration structure	
Return	u8_en_keypadErrorsType	KEYPAD_E_OK	
		KEYPAD_E_NOT_OK	
Description	This Function Initialize Keypad module		

- This function shall return KEYPAD_E_NOK if st_config is NULL
- This function shall return KEYPAD_E_NOK if any of the configuration elements is invalid.
- This function shall return DIO_E_NOT_OK if failed to initialize the pin direction to be OUTPUT or INPUT

- KEYPAD_read

Service name	KEYPAD_read		
Syntax	u8_en_keypadErrorsType KEYPAD_read (uint8_t * u8_data);		

Parameters (in)	u8_data	Pointer to variable where to store value read from keypad	
Return	u8_en_keypadErrorsType	KEYPAD_E_OK	
		KEYPAD_E_NOT_OK	
Description	This Function read Keypad		

3.3.4 : Car Control API :

3.3.4.1: Flowcharts:

3.3.4.2 : Type definitions:

- st_carControlConfigType

Name	st_carControlConfigType
Type	Structure
Range	Shall contain required car motors configuration
Description	st_carControlConfigType
Available via	car_control.h

- u8_en_carControlErrorsType

Name	u8_en_carControlErrorsType		
Type	Enumeration		
Range	CAR_E_OK	0x00	CAR error OK
	CAR_E_NOT_OK	0x07	CAR error
Description	u8_en_carControlErrorsType		
Available via	car_control.h		

3.3.4.2 : Services affecting the hardware unit

- CAR_init

Service name	CAR_init					
Syntax	u8_en_carControlErrorsType CAR_init (st_carControlConfigType* st_config_R, st_carControlConfigType* st_config_L);					
Parameters (in)	st_config	Pointer to the configuration structure				
Return	<table><tr><td rowspan="2">u8_en_carControlErrorsType</td><td>CAR_E_OK</td></tr><tr><td>CAR_E_NOT_OK</td></tr></table>			u8_en_carControlErrorsType	CAR_E_OK	CAR_E_NOT_OK
u8_en_carControlErrorsType	CAR_E_OK					
	CAR_E_NOT_OK					
Description	This Function Initialize car module					

- CAR_moveForward

Service name	● CAR_moveForward					
Syntax	u8_en_carControlErrorsType CAR_moveForward(st_carControlConfigType* st_config_R, st_carControlConfigType* st_config_L);					
Parameters (in)	st_config	Pointer to the configuration structure				
Return	<table><tr><td rowspan="2">u8_en_carControlErrorsType</td><td>CAR_E_OK</td></tr><tr><td>CAR_E_NOT_OK</td></tr></table>			u8_en_carControlErrorsType	CAR_E_OK	CAR_E_NOT_OK
u8_en_carControlErrorsType	CAR_E_OK					
	CAR_E_NOT_OK					
Description	This Function moving the car forward					

- CAR_turnRight

Service name	CAR_turnRight		
Syntax	u8_en_carControlErrorsType CAR_turnRight(st_carControlConfigType* st_config_R, st_carControlConfigType* st_config_L);		
Parameters (in)	st_config	Pointer to the configuration structure	

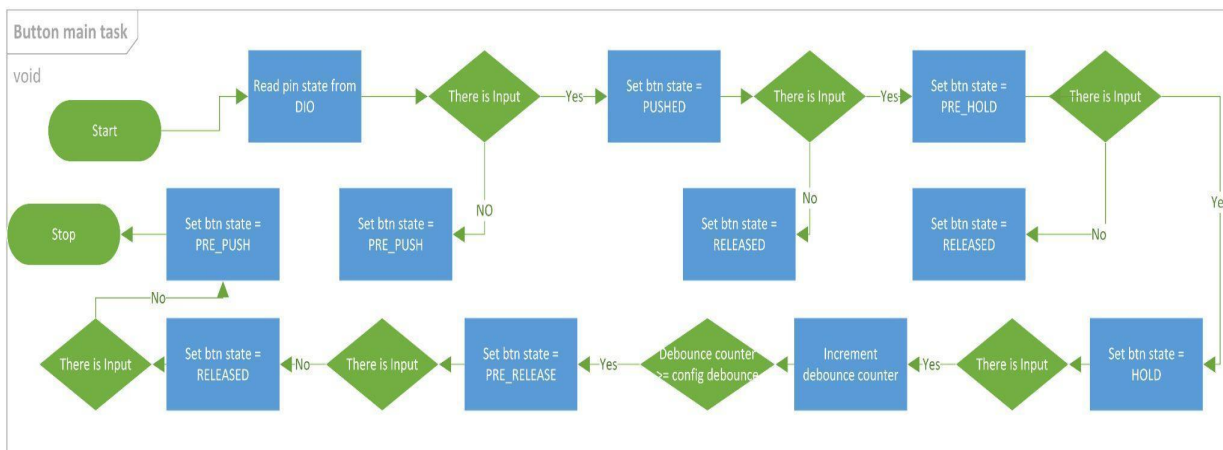
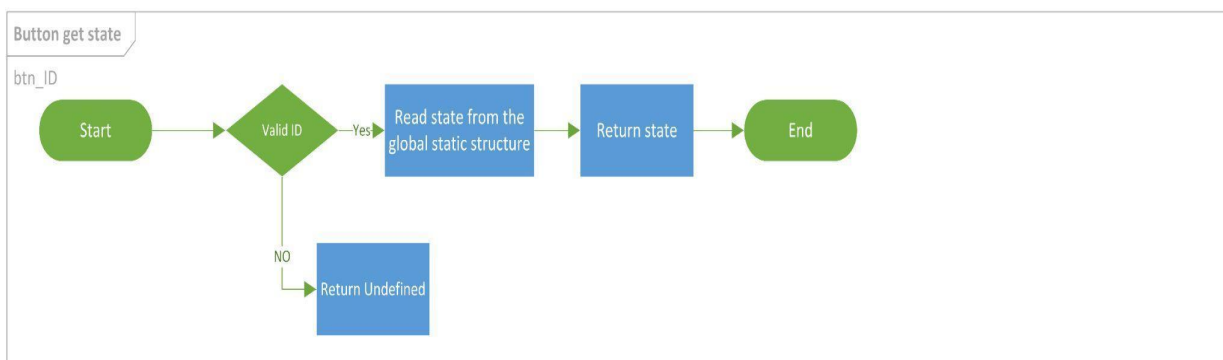
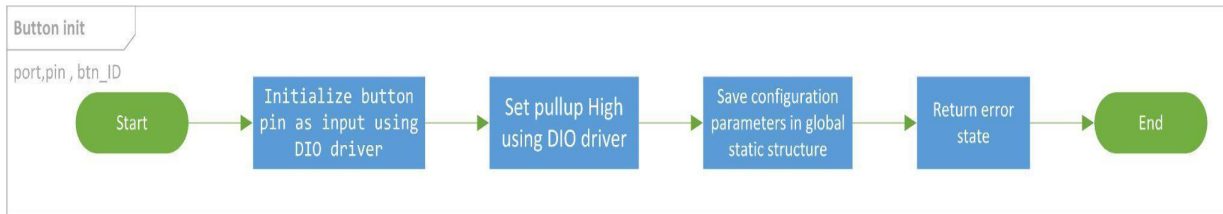
Return	u8_en_carControlErrorsType	CAR_E_OK
		CAR_E_NOT_OK
Description	This Function turn the car right	

- CAR_turnLeft

Service name	CAR_turnLeft	
Syntax	<pre>u8_en_carControlErrorsType CAR_turnLeft(st_carControlConfigType* st_config_R, st_carControlConfigType* st_config_L);</pre>	
Parameters (in)	st_config	Pointer to the configuration structure
Return	u8_en_carControlErrorsType	CAR_E_OK
		CAR_E_NOT_OK
Description	This Function turn the car left	

3.3.5: Button API:

3.3.5.1: Flowcharts:



3.3.5.2 : Type definitions:

- st_btnConfigType

Name	st_btnConfigType
Type	Structure

Description	This is the type of the external data structure containing the overall configuration data for the Button API
Available via	button_types.h

- u8_en_btnLevelType

Name	u8_en_btnLevelType		
Type	Enumeration		
Range	BT_PUSH_LEVEL	0x00	Push Level
	BT_RELEASE_LEVEL	0x01	Release Level
Description	Button Level Enum		
Available via	button_types.h		

- u8_en_btnStateType

Name	u8_en_btnStateType		
Type	Enumeration		
Range	BT_PRE_PUSH	0x00	Pre Push Level
	BT_PUSHED	0x01	Pushed Level
	BT_PRE_HOLD	0x02	Pre Hold Level
	BT_HOLD	0x03	Hold Level
	BT_PRE_RELEASE	0x04	Pre Release Level
	BT_RELEASED	0x05	Released Level
	BT_UNDEFINED	0x06	Undefined
Description	Button state Enum		
Available via	button_types.h		

- Button_IdType

Name	u8_en_btnIdType		
Type	Enumeration		
Range	Button_Start	0x00	Start Button
Description	Button ID Enum		
Available via	button_types.h		

3.3.5.2 : Services affecting the hardware unit

- BUTTON_getState

Service name	BUTTON_getState										
Syntax	u8_en_btnStateType BUTTON_getState(u8_en_btnIdType en_btnId);										
Parameters (in)	en_btnId	Start 0x00									
Return	<table><tr><td rowspan="7">Button_StateTyp</td><td>BT_PRE_PUSH</td></tr><tr><td>BT_PUSHED</td></tr><tr><td>BT_PRE_HOLD</td></tr><tr><td>BT_HOLD</td></tr><tr><td>BT_PRE_RELEASE</td></tr><tr><td>BT_RELEASED</td></tr><tr><td>BT_UNDEFINED</td></tr></table>			Button_StateTyp	BT_PRE_PUSH	BT_PUSHED	BT_PRE_HOLD	BT_HOLD	BT_PRE_RELEASE	BT_RELEASED	BT_UNDEFINED
Button_StateTyp	BT_PRE_PUSH										
	BT_PUSHED										
	BT_PRE_HOLD										
	BT_HOLD										
	BT_PRE_RELEASE										
	BT_RELEASED										
	BT_UNDEFINED										
Description	This Function gets the Button state.										

- button_Main_Task

Service name	button_Main_Task
--------------	------------------

Syntax	void button_Main_Task(void);
Parameters (in)	NONE
Return	NONE
Description	This Function update all button states Shall call periodic

- BUTTON_init

Service name	BUTTON_init		
Syntax	u8_en_btnStateType BUTTON_init(uint8_t u8_a_port, uint8_t u8_a_pin, u8_en_btnIdType en_btnId);		
Parameters (in)	Port, pin	Channel ID	
	en_btnId	Start 0x00	
Return	Button_StateTyp	BT_PRE_PUSH	
		BT_UNDEFINED	
Description	This Function sets the Direction of the button pin as input		

3.4 : App APIs

3.4.1 : APP API :

3.4.1.1 :Flowcharts:

3.4.1.2 : Type definitions:

3.4.1.3 : Services affecting the hardware unit

- APP_start

Service name	APP_start
Syntax	void APP_start(void);
Description	This Function Start the Application.
Available via	app.h