

# Air Conditioner System

By Team 4

# 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
    - 3.2.3 : ADC API
      - 3.2.3.1 : Flowchart
      - 3.2.3.2 : Type definitions
      - 3.2.3.3 : 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 : LCD API
      - 3.3.2.1 : Flowchart
      - 3.3.2.2 : Type definitions
      - 3.3.2.3 : Services
    - 3.3.3 : Buzzer API
      - 3.3.3.1 : Flowchart
      - 3.3.3.2 : Type definitions
      - 3.3.3.3 : Services
    - 3.3.4 : Keypad API
      - 3.3.4.1 : Flowchart
      - 3.3.4.2 : Type definitions
      - 3.3.4.3 : Services
    - 3.3.5 : Temperature sensor 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

1- When system start LCD prompt welcome message for 1 second, then display the default temp is 20 ,the message appear for 1 second

2- ask to set initial temperature for 0.5 second and disappear

3- display range of temperature min=18,max=35

4- button\_1 and button\_2 used for increment and decrement respectively

5- each button press the temperature on the screen is update

Min=18

Temp

Max=35

|||||

6- Once button\_3 is pressed the temperature is set and LCD display current temp=.....

And display buzzer shape if temperature > set temperature & buzzer ON

7- once button\_4 is press back to step\_2 (readjust mode),stop buzzer if it was working add timeout

8- if button\_5 is press mean reset temperature to its default and display

Temp value is resettled to 20 degree

9- after set mode all buttons are not allowed except button\_4 and button\_5 and display error message for 0.5 second (the operation is not allowed)-add timeout

Button\_1 : Increment

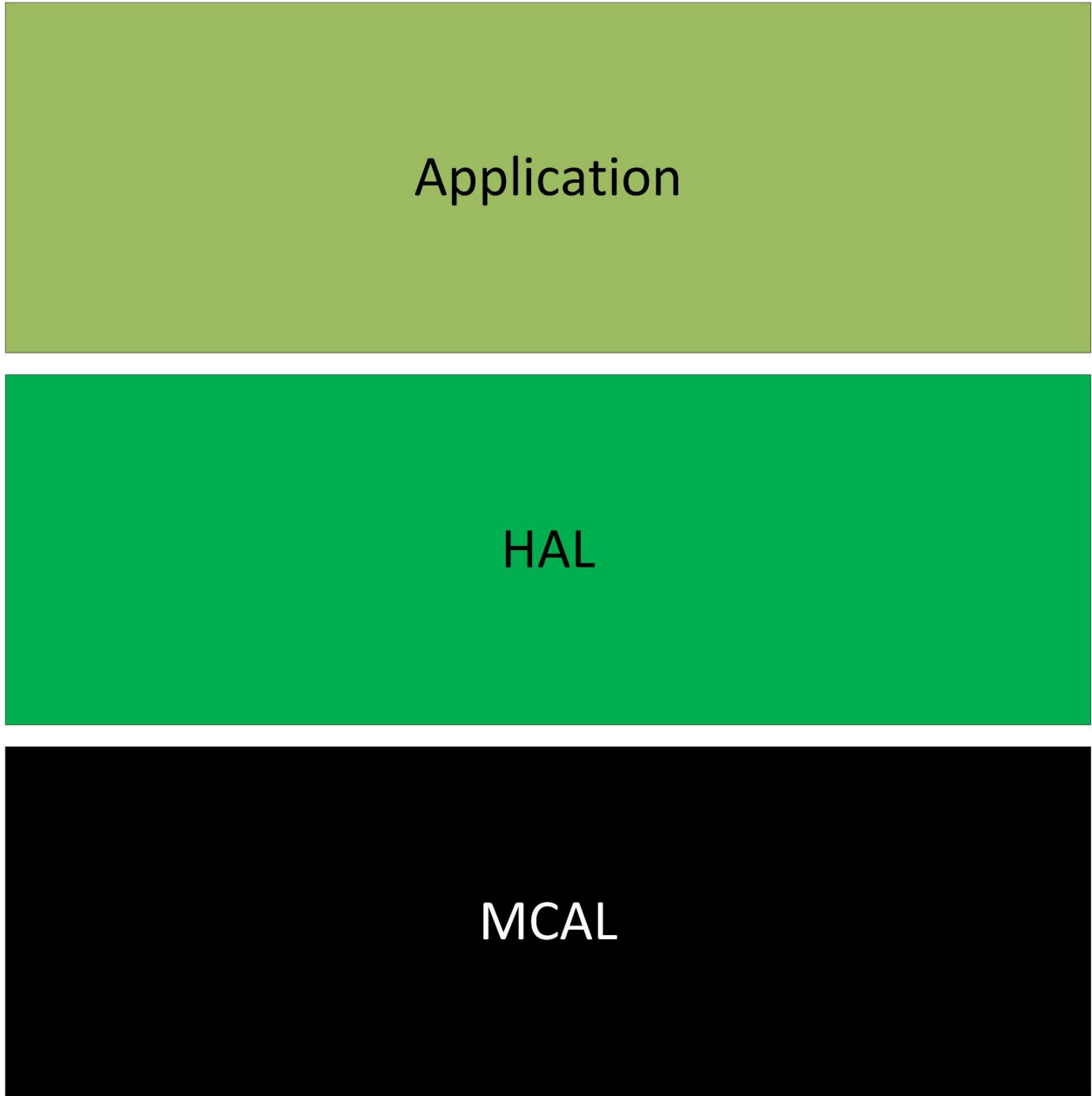
Button\_2 : Decrement

Button\_3 : Set temperature

Button\_4 : Adjust temperature

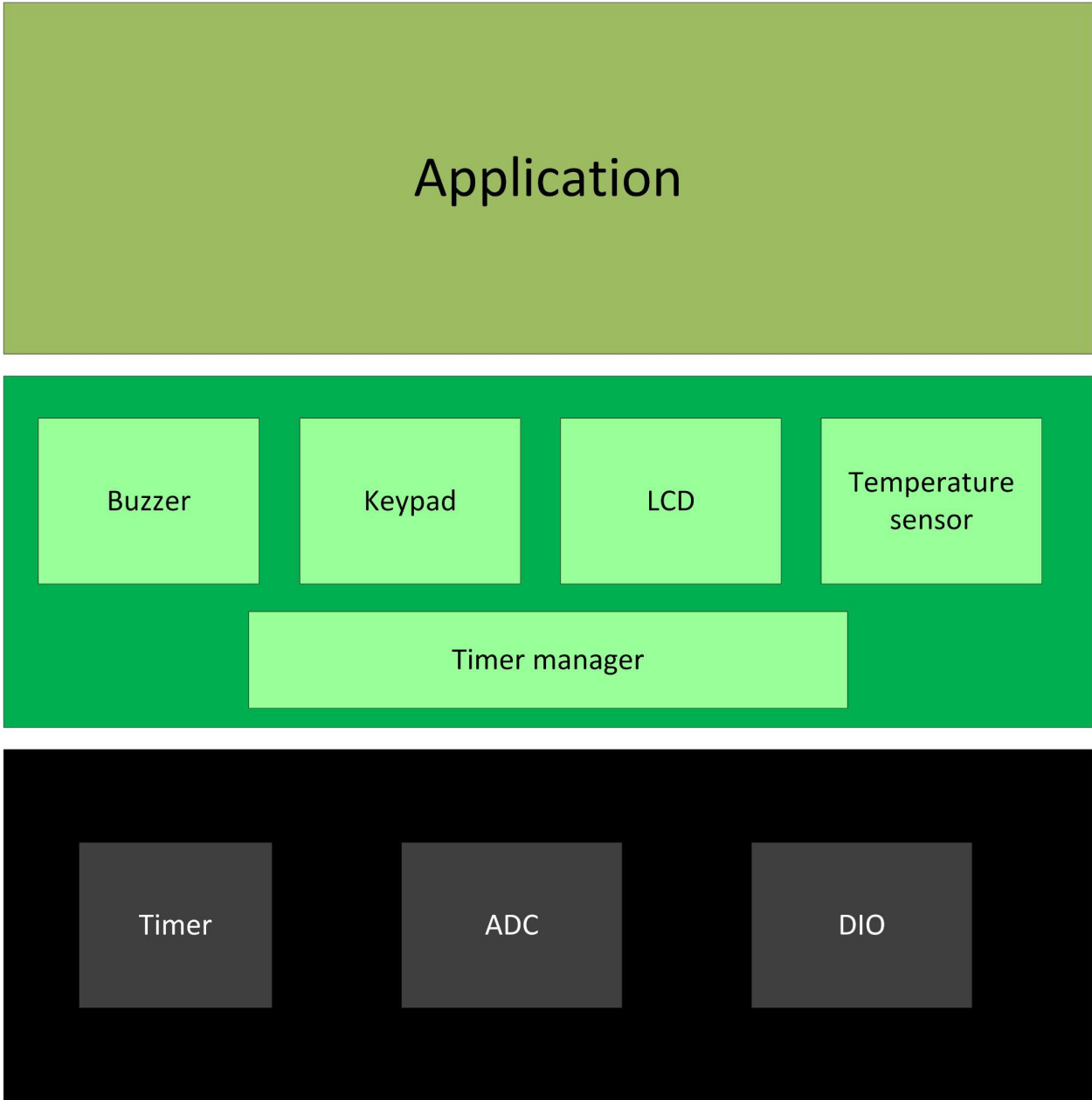
Button\_5 : Reset to default

## 2 : Layered architecture



# 3 : System modules

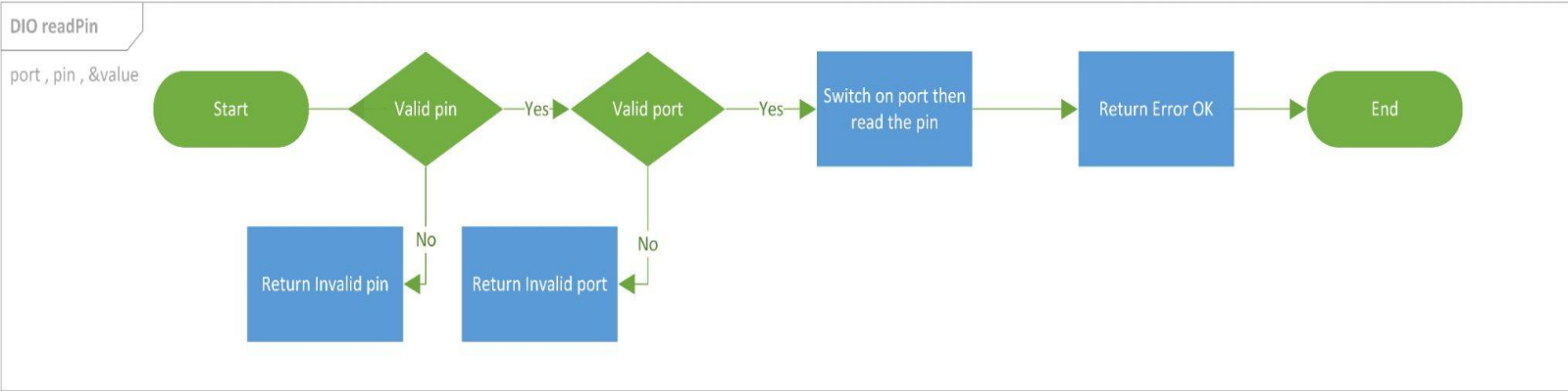
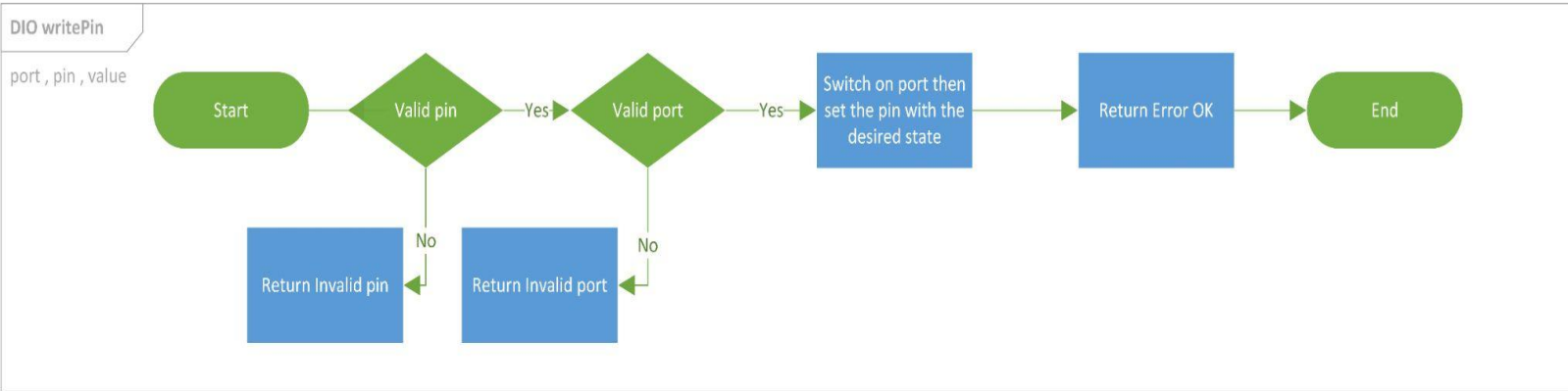
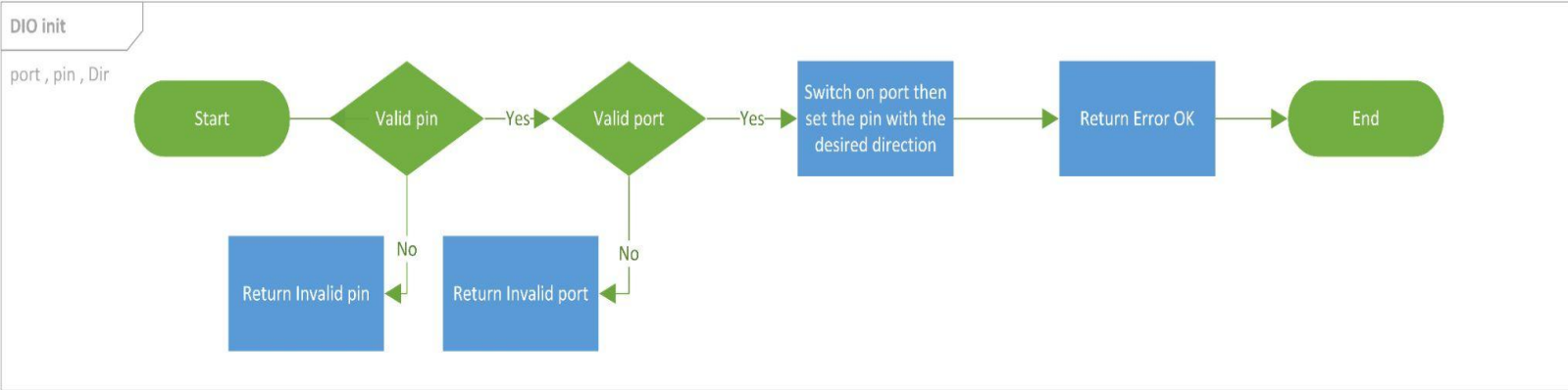
## 3.1: Module architecture



# 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	u8_en_dioErrors DIO_readPIN ( en_dioPortsType port, en_dioPinsType pin, uint8_t* value

	);		
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.

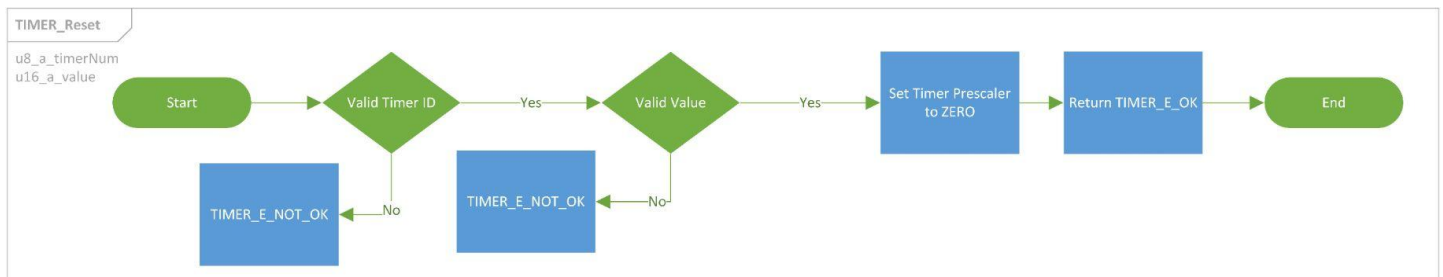
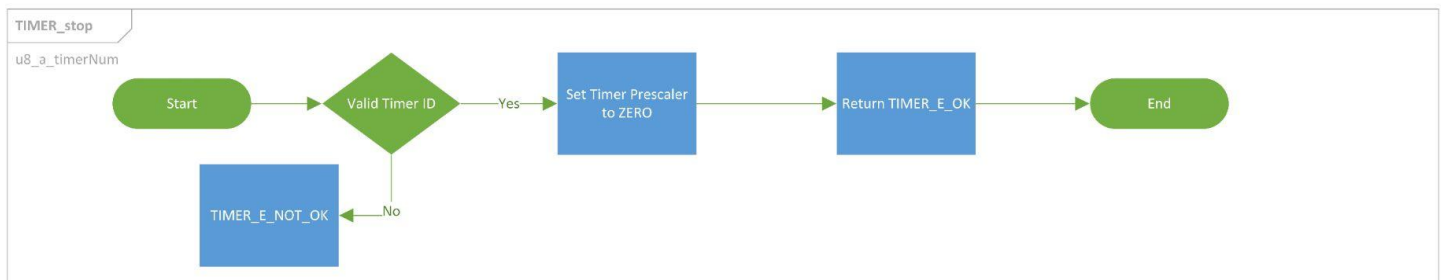
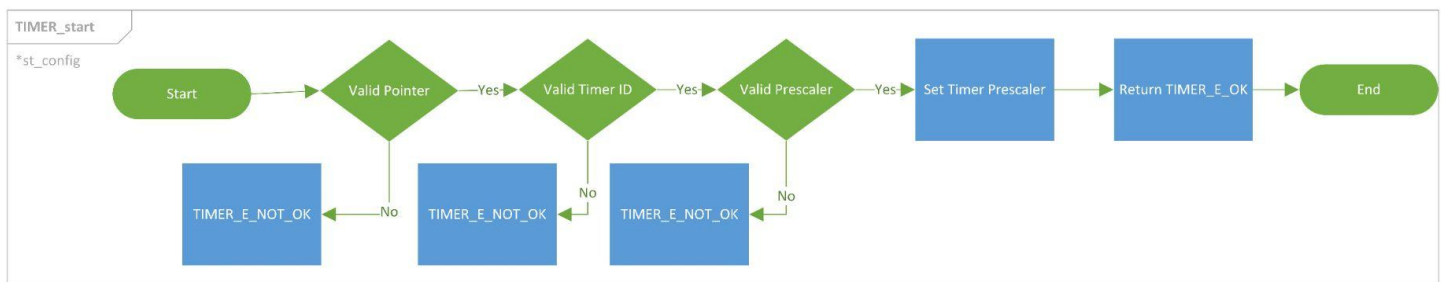
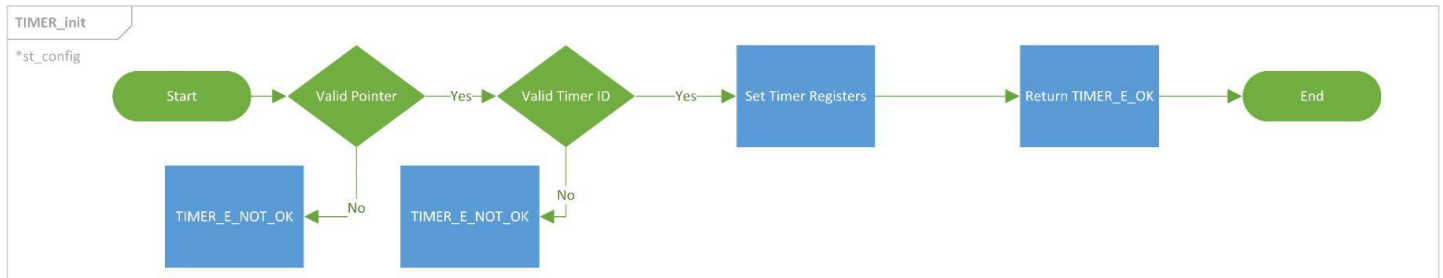
- DIO\_init

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	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	Timer ID
Return	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**

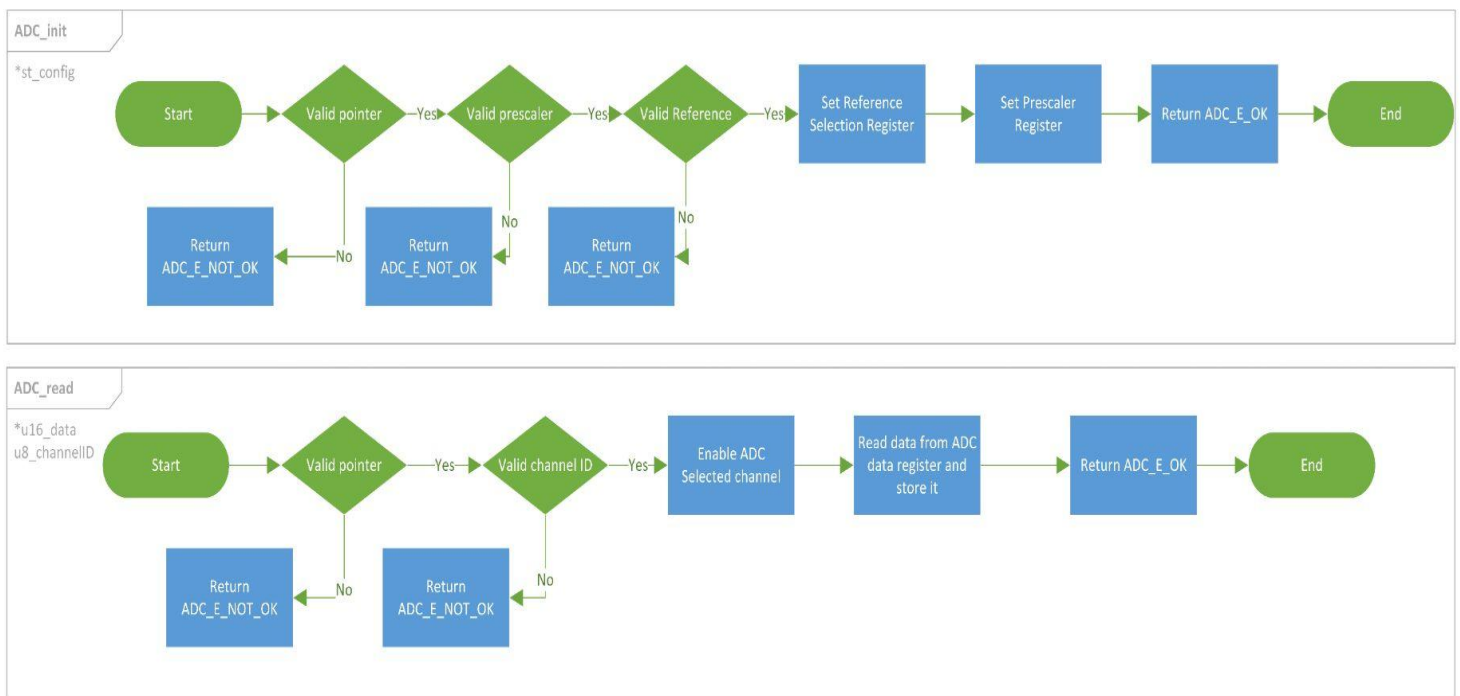
Service name	TIMER_reset	
Syntax	u8_en_timerErrorsType TIMER_reset ( u8_en_timerNumberType u8_en_timerNum, uint16_t u16_value );	
Parameters (in)	u8_en_timerNum	Timer ID
	u16_value	Value to be set to the timer
Return		

	u8_en_timerErrorsType	TIMER_E_OK
		TIMER_E_NOT_OK
Description	This Function reset the TIMER	

- This function shall return TIMER\_E\_NOK if u8\_en\_timerNum is invalid
- This function shall return TIMER\_E\_NOK if u16\_value is invalid

### 3.2.3 : ADC API :

#### 3.2.3.1 :Flowcharts:



#### 3.2.3.2 : Type definitions:

- st\_adcConfigType

Name	st_adcConfigType
Type	Structure



Range	Shall contain required ADC configuration
Description	st_adcConfigType
Available via	adc.h

- u8\_en\_adcErrorsType

Name	u8_en_adcErrorsType		
Type	Enumeration		
Range	ADC_E_OK	0x00	ADC error OK
	ADC_E_NOT_OK	0x04	ADC error
Description	u8_en_adcErrorsType		
Available via	adc.h		

- u8\_en\_adcChannelId

Name	u8_en_adcChannelId		
Type	Enumeration		
Range	Shall contain all ADC channel ID		
Description	u8_en_adcChannelId		
Available via	adc.h		

- u8\_en\_adcRefType

Name	u8_en_adcRefType		
Type	Enumeration		
Range	Shall contain all reference selection modes		
Description	u8_en_adcRefType		

Available via	adc.h
---------------	-------

- u8\_en\_adcPrescalerType

Name	u8_en_adcPrescalerType
Type	Enumeration
Range	Shall contain all Prescaler selection modes
Description	u8_en_adcPrescalerType
Available via	adc.h

### 3.2.3.3 : Services affecting the hardware unit

- ADC\_init

Service name	ADC_init					
Syntax	u8_en_adcErrorsType ADC_init ( st_adcConfigType* st_config );					
Parameters (in)	st_config	Pointer to the configuration structure				
Return	<table><tr><td rowspan="2">u8_en_adcErrorsType</td><td>ADC_E_OK</td></tr><tr><td>ADC_E_NOT_OK</td></tr></table>			u8_en_adcErrorsType	ADC_E_OK	ADC_E_NOT_OK
u8_en_adcErrorsType	ADC_E_OK					
	ADC_E_NOT_OK					
Description	This Function Initialize ADC module					

- This function shall return ADC\_E\_NOK if st\_config is NULL
- This function shall return ADC\_E\_NOK if any of the configuration elements is invalid.

- ADC\_read

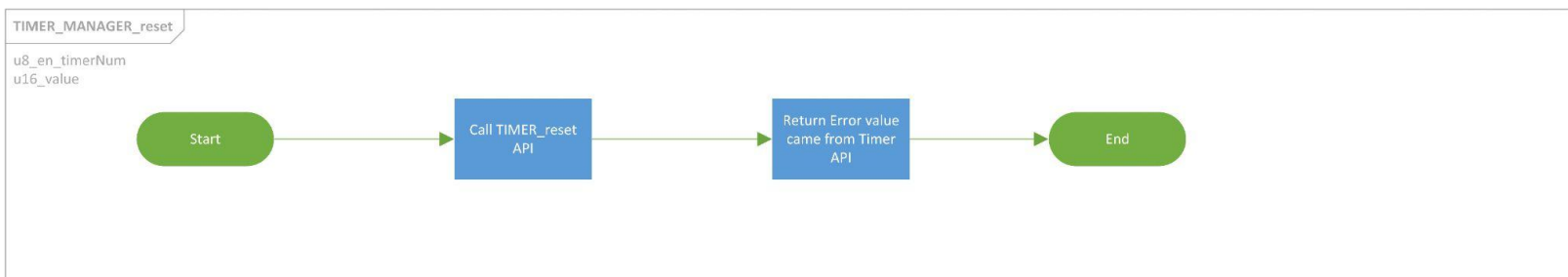
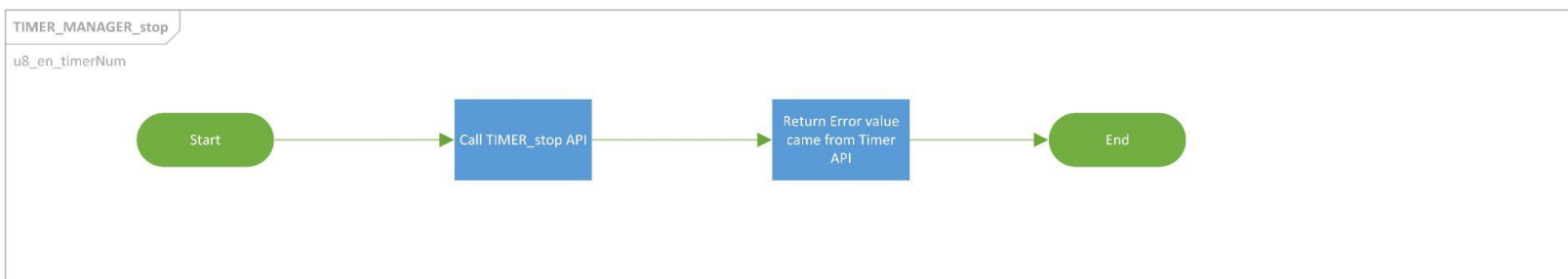
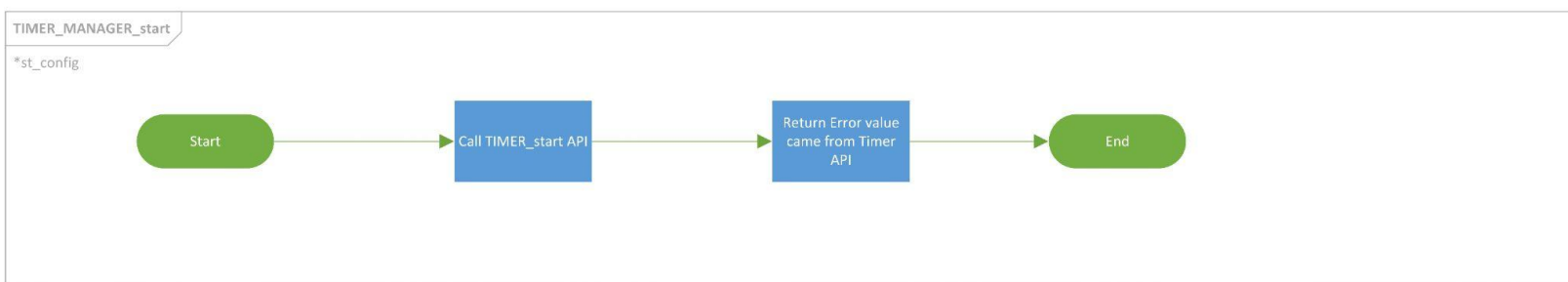
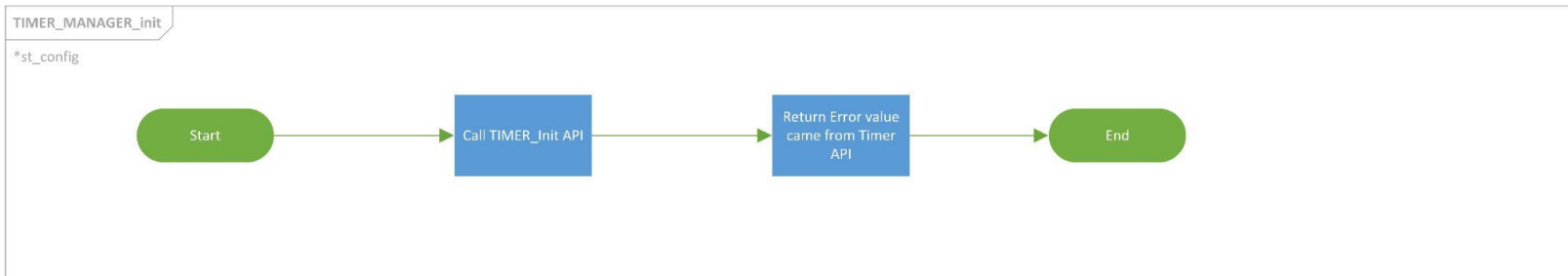
Service name	ADC_read
Syntax	u8_en_adcErrorsType ADC_read (

	u8_en_adcChannelId u8_channelID, uint16_t * u16_data );	
Parameters (in)	u16_data	Pointer to variable where to store the ADC value
	u8_channel ID	ADC Channel ID
Return	u8_en_adcErrorsType	ADC_E_OK
		ADC_E_NOT_OK
Description	This Function read ADC	

### 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 ( 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\_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	<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 stop TIMER				

- This function shall return TIMER\_E\_NOK if u8\_en\_timerNum is invalid

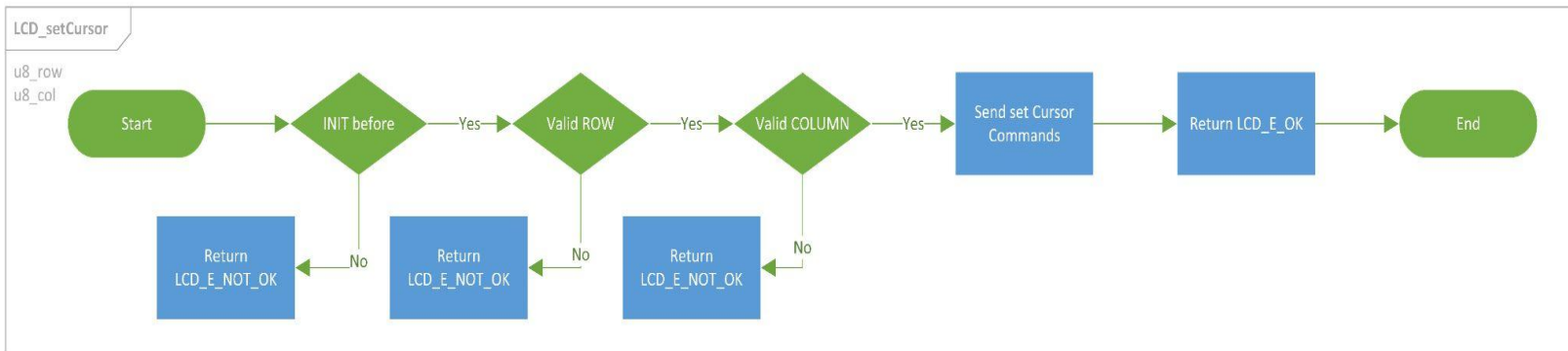
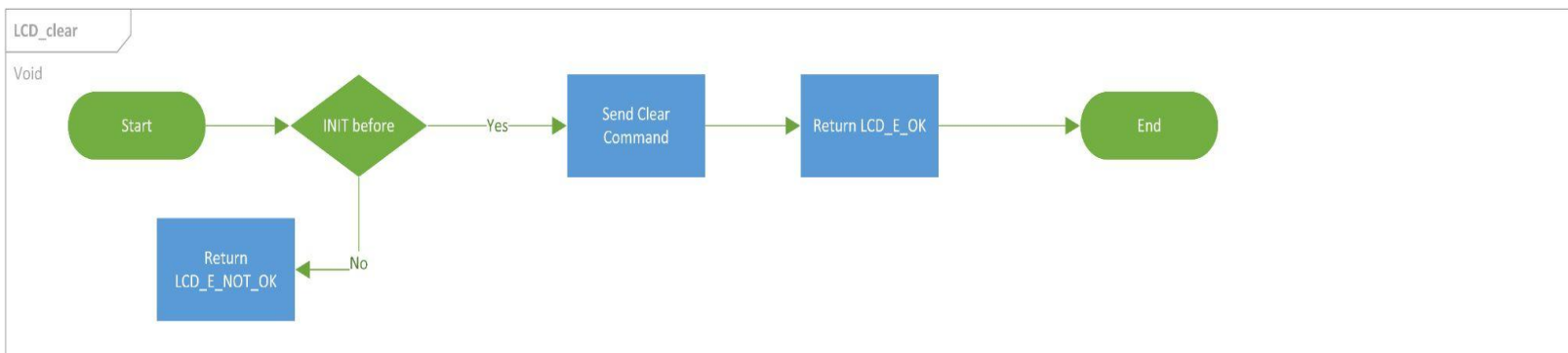
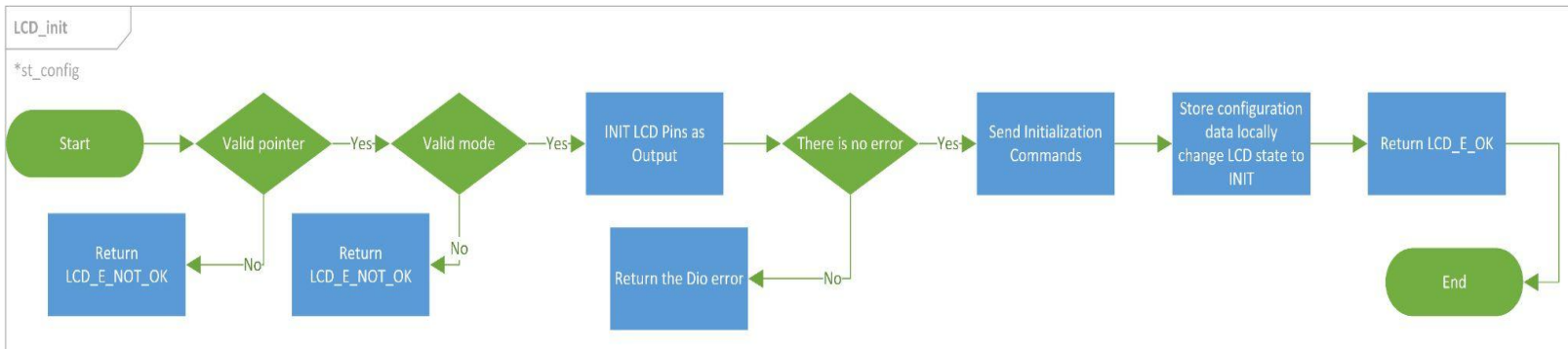
- **TIMER\_Manager\_reset**

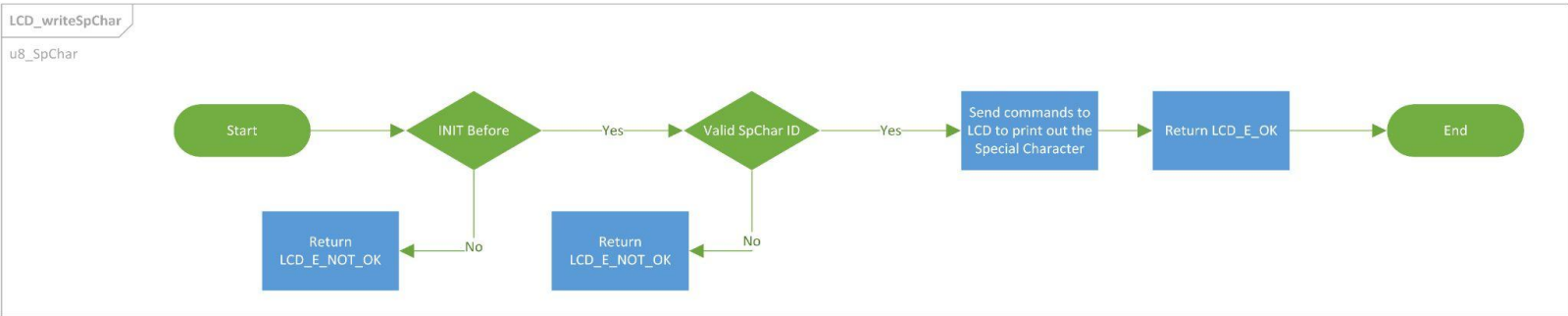
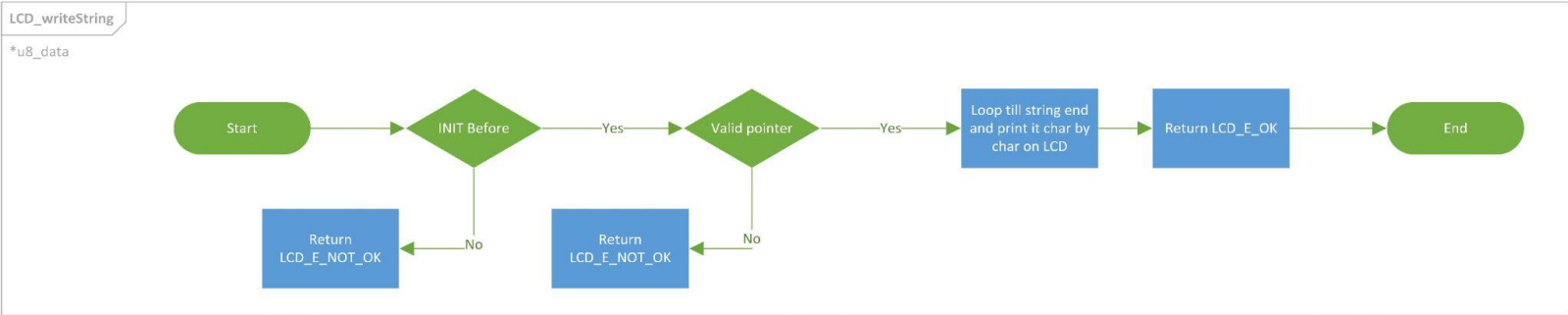
Service name	TIMER_Manager_reset				
Syntax	u8_en_timerErrorsType TIMER_Manager_reset ( u8_en_timerNumberType u8_en_timerNum, uint16_t u16_value );				
Parameters (in)	u8_en_timerNum	Timer ID			
	u16_value	Value to be set to the timer			
Return	u8_en_timerErrorsType	<table><tr><td>TIMER_E_OK</td></tr><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 stop TIMER				

- This function shall return TIMER\_E\_NOK if u8\_en\_timerNum is invalid
- This function shall return TIMER\_E\_NOK if u16\_value 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	<table><tr><td>LCD_E_OK</td><td>0x00</td><td>LCD error OK</td></tr></table>			LCD_E_OK	0x00	LCD error OK
LCD_E_OK	0x00	LCD error OK				



	<table><tr><td>LCD_E_NOT_OK</td><td>0x05</td><td>LCD error</td></tr></table>	LCD_E_NOT_OK	0x05	LCD error
LCD_E_NOT_OK	0x05	LCD error		
Description	u8_en_lcdErrorsType			
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	u8_en_lcdErrorsType LCD_setCursor ( uint8_t u8_row, uint8_t u8_col );	
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	u8_en_lcdErrorsType LCD_writeString ( uint8_t* u8_data );	

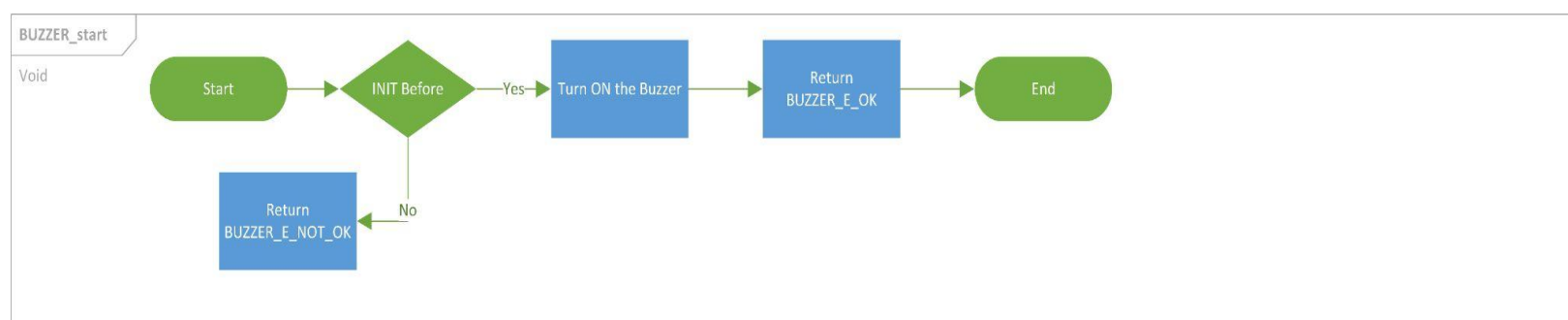
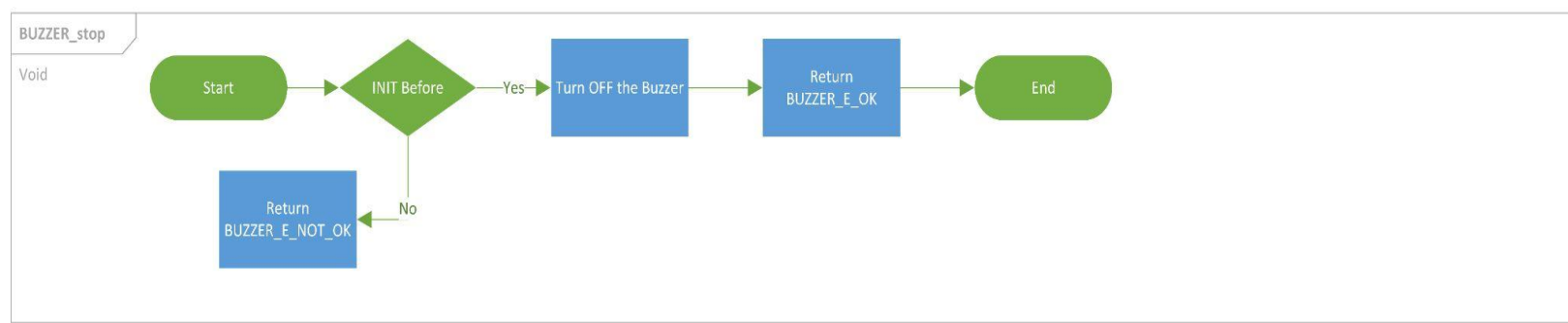
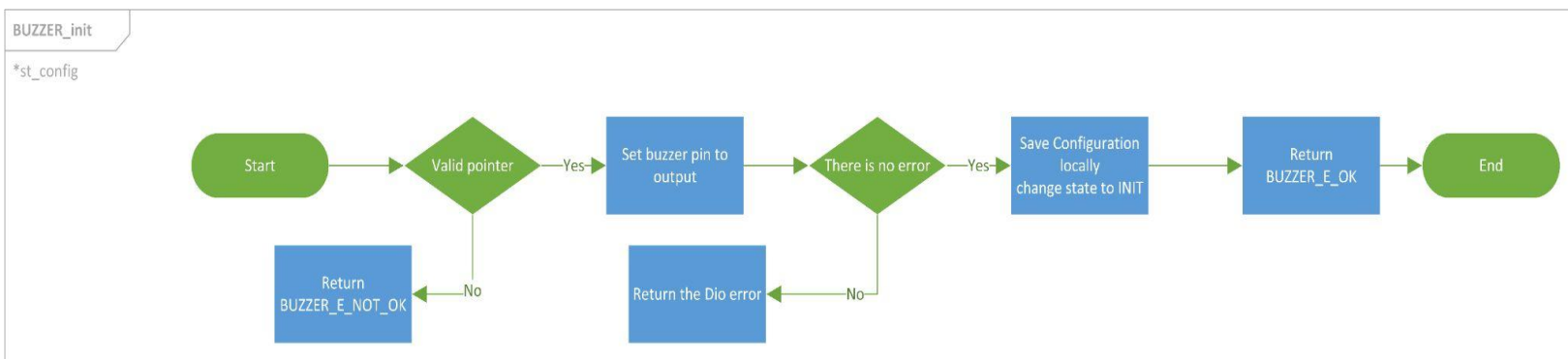
Parameters (in)	u8_data	Pointer to string to it print on screen
Return	u8_en_lcdErrorsType	LCD_E_OK
		LCD_E_NOT_OK
Description	This Function write a string on LCD	

- 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: Buzzer API:

#### 3.3.3.1 :Flowcharts:



### 3.3.3.2 : Type definitions:

- st\_buzzerConfigType

Name	st_buzzerConfigType
Type	Structure
Range	Shall contain required Buzzer configuration
Description	st_buzzerConfigType
Available via	buzzer.h

- u8\_en\_buzzerErrorsType

Name	u8_en_buzzerErrorsType		
Type	Enumeration		
Range	BUZZER_E_OK	0x00	Buzzer error OK
	BUZZER_E_NOT_OK	0x06	Buzzer error
Description	u8_en_buzzerErrorsType		
Available via	buzzer.h		

### 3.3.3.3 : Services affecting the hardware unit

- BUZZER\_init

Service name	BUZZER_init					
Syntax	u8_en_buzzerErrorsType BUZZER_init ( st_buzzerConfigType* st_config );					
Parameters (in)	st_config	Pointer to the configuration structure				
Return	<table><tr><td rowspan="2">u8_en_buzzerErrorsType</td><td>BUZZER_E_OK</td></tr><tr><td>BUZZER_E_NOT_OK</td></tr></table>			u8_en_buzzerErrorsType	BUZZER_E_OK	BUZZER_E_NOT_OK
u8_en_buzzerErrorsType	BUZZER_E_OK					
	BUZZER_E_NOT_OK					
Description	This Function Initialize Buzzer module					

- This function shall return BUZZER\_E\_NOK if st\_config is NULL
- This function shall return BUZZER\_E\_NOK if any of the configuration elements is invalid.

- BUZZER\_start

Service name	BUZZER_start				
Syntax	u8_en_buzzerErrorsType BUZZER_start( void );				
Parameters (in)	None				
Return	u8_en_buzzerErrorsType	<table><tr><td>BUZZER_E_OK</td></tr><tr><td>BUZZER_E_NOT_OK</td></tr></table>		BUZZER_E_OK	BUZZER_E_NOT_OK
BUZZER_E_OK					
BUZZER_E_NOT_OK					
Description	This Function starts Buzzer				

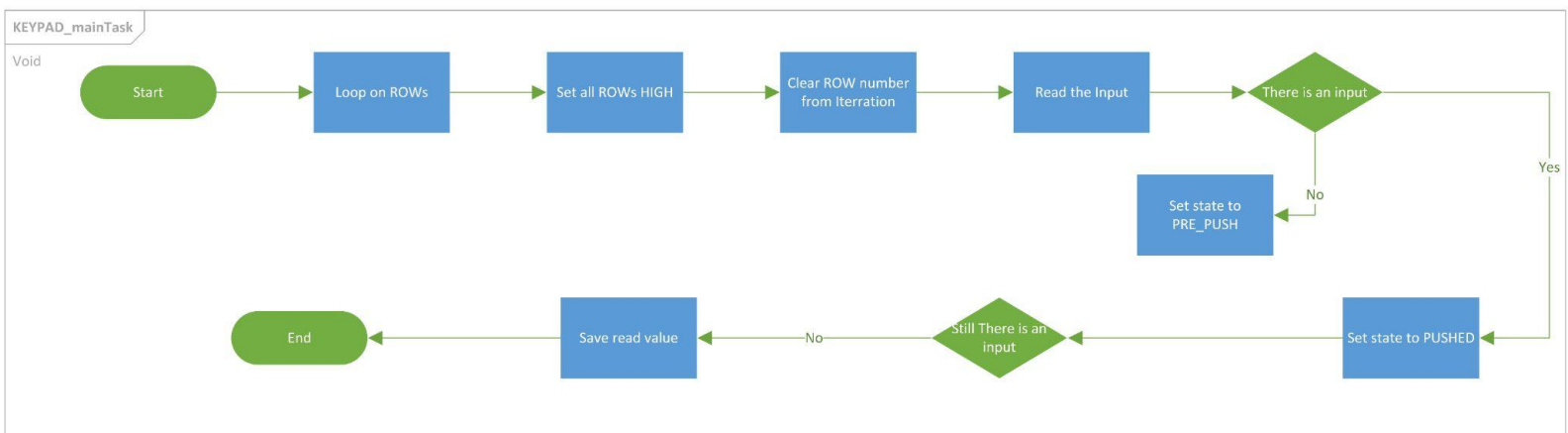
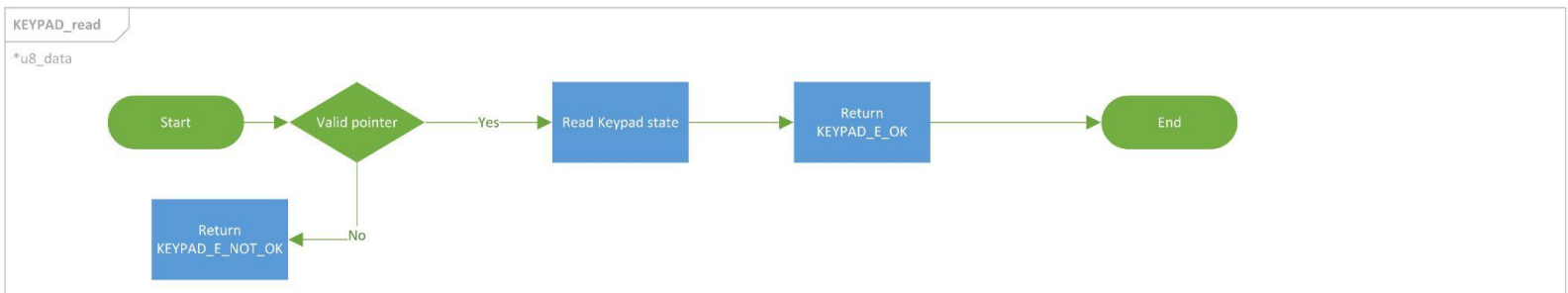
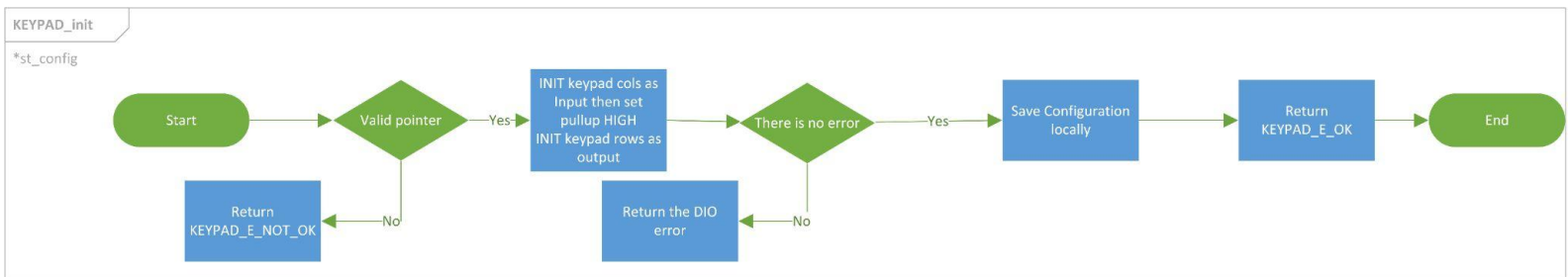
- BUZZER\_stop

Service name	BUZZER_stop	
Syntax	u8_en_buzzerErrorsType BUZZER_stop( void );	
Parameters (in)	None	
Return		

	<table> <tr> <td>u8_en_buzzerErrorsType</td><td>BUZZER_E_OK</td></tr> <tr> <td></td><td>BUZZER_E_NOT_OK</td></tr> </table>	u8_en_buzzerErrorsType	BUZZER_E_OK		BUZZER_E_NOT_OK
u8_en_buzzerErrorsType	BUZZER_E_OK				
	BUZZER_E_NOT_OK				
Description	This Function stops Buzzer				

### 3.3.4 : Keypad API :

#### 3.3.4.1 :Flowcharts:



#### 3.3.4.2 : 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		

### 3.3.4.3 : Services affecting the hardware unit

- KEYPAD\_init

Service name	KEYPAD_init		
Syntax	u8_en_keypadErrorsType KEYPAD_init ( 		



- 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.

- KEYPAD\_read

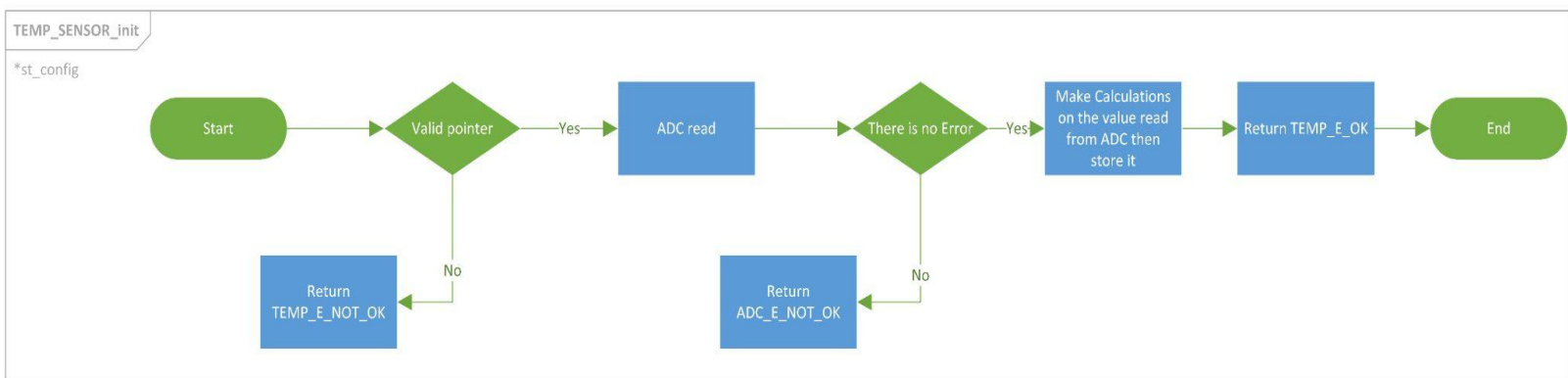
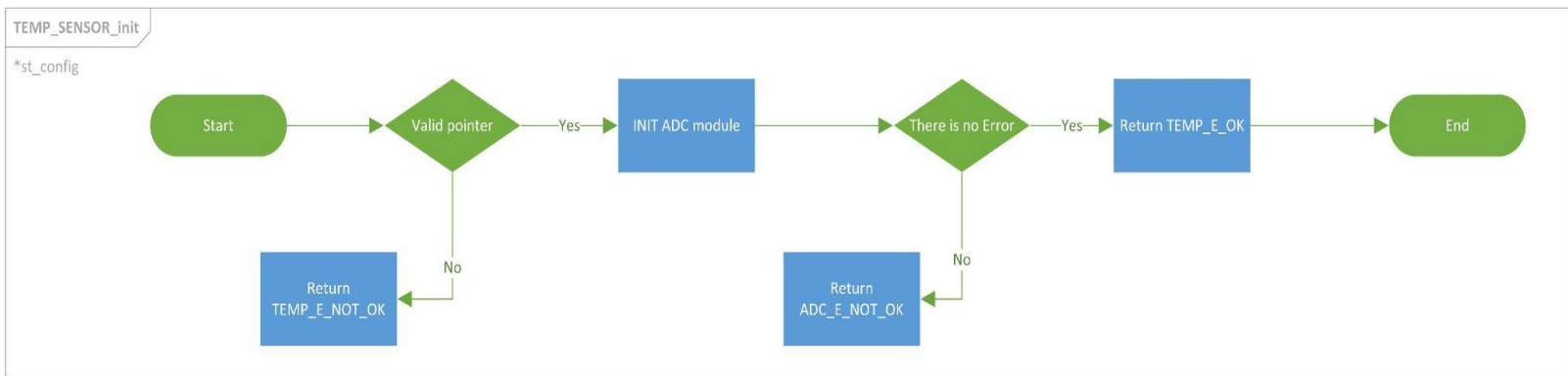
Service name	KEYPAD_read					
Syntax	u8_en_keypadErrorsType KEYPAD_read ( uint8_t * u8_data );					
Parameters (in)	u8_data	Pointer to variable where to store value read from keypad				
Return	<table><tr><td rowspan="2">u8_en_keypadErrorsType</td><td>KEYPAD_E_OK</td></tr><tr><td>KEYPAD_E_NOT_OK</td></tr></table>			u8_en_keypadErrorsType	KEYPAD_E_OK	KEYPAD_E_NOT_OK
u8_en_keypadErrorsType	KEYPAD_E_OK					
	KEYPAD_E_NOT_OK					
Description	This Function read Keypad					

- KEYPAD\_mainTask

Service name	KEYPAD_mainTask
Syntax	void KEYPAD_mainTask ( void );
Parameters (in)	None
Return	None
Description	This Function update the value read from the Keypad

### 3.3.5 : Temperature sensor API :

#### 3.3.5.1 :Flowcharts:



#### 3.3.5.2 : Type definitions:

- st\_tempSensorConfigType

Name	st_tempSensorConfigType
Type	Structure
Range	Shall contain required Temperature Sensor configuration
Description	st_tempSensorConfigType
Available via	temp_sensor.h

- u8\_en\_tempSensorErrorsType

Name	u8_en_tempSensorErrorsType
------	----------------------------

Type	Enumeration		
Range	TEMP_E_OK	0x00	Temp Sensor error OK
	TEMP_E_NOT_OK	0x08	Temp Sensor error
Description	u8_en_tempSensorErrorsType		
Available via	temp_sensor.h		

### 3.3.5.3 : Services affecting the hardware unit

- TEMP\_SENSOR\_init

Service name	TEMP_SENSOR_init		
Syntax	u8_en_tempSensorErrorsType TEMP_SENSOR_init ( 		

- This function shall return TEMP\_E\_NOK if st\_config is NULL
- This function shall return TEMP\_E\_NOK if any of the configuration elements is invalid.

- TEMP\_SENSOR\_read

Service name	TEMP_SENSOR_read		
Syntax	u8_en_tempSensorErrorsType TEMP_SENSOR_read ( uint8_t * u8_data );		
Parameters (in)	u8_data	Pointer to variable where to store value read from keypad	

Return	u8_en_tempSensorErrorsType	TEMP_E_OK
		TEMP_E_NOT_OK
Description	This Function read the Temperature	

## 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