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

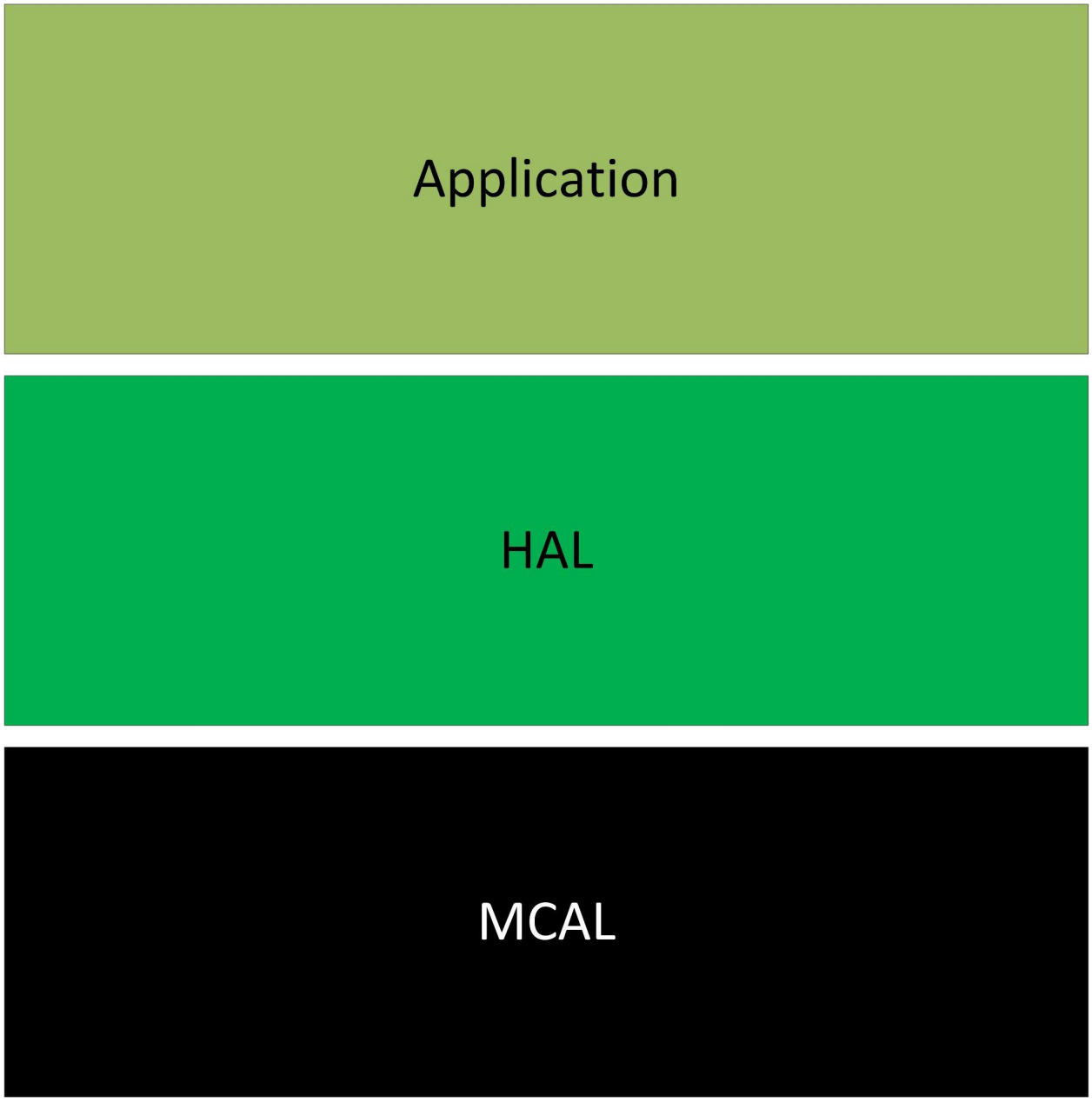
3.4.1.2: Type definitions
3.4.1.3: Services

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

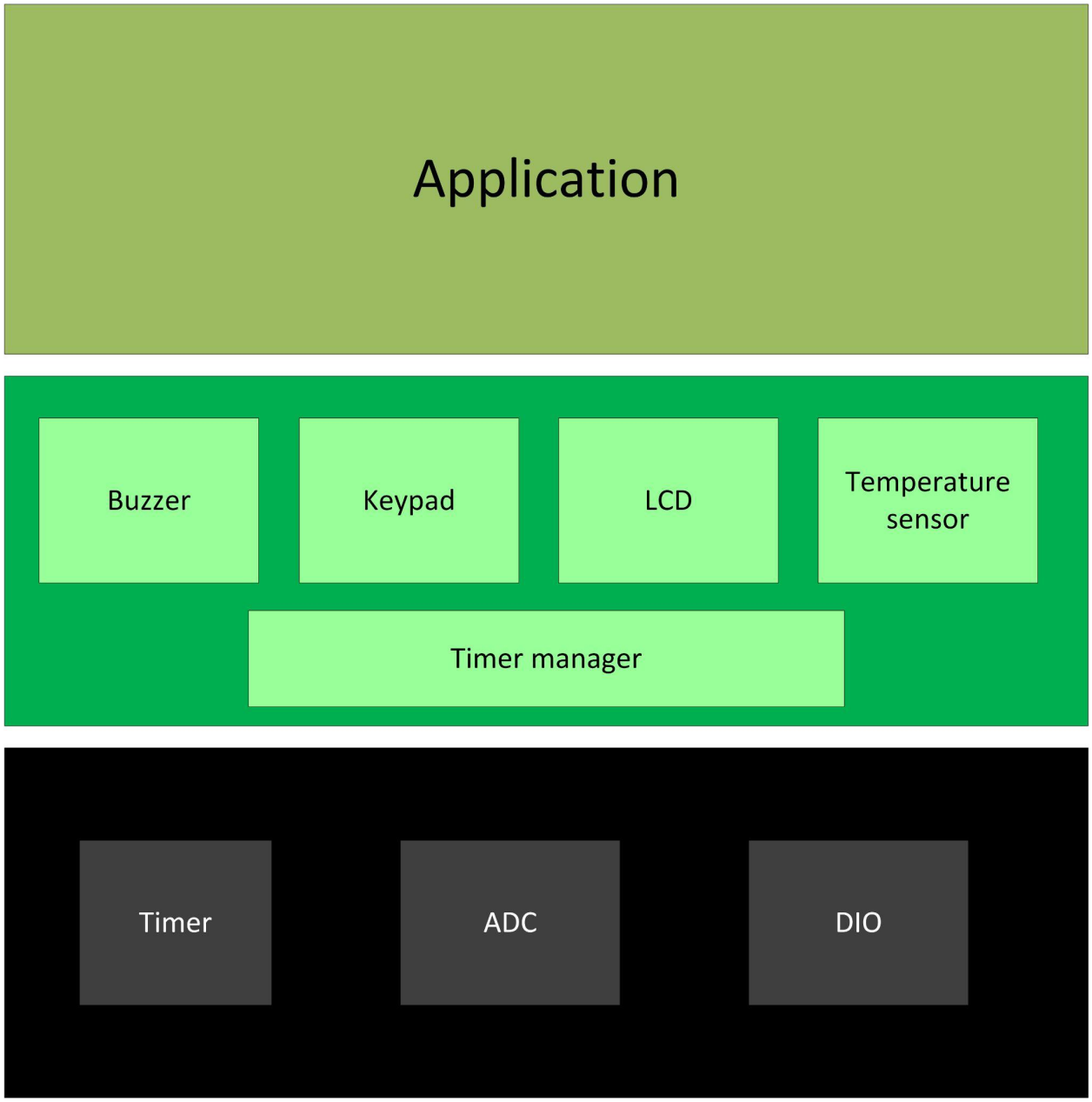
Button_1 : Increment
Button_2 : Decrement
Button_3 : Set temperature
Button_4 : Adjust temperature
Button_5 : Reset to default

2 : Layered architecture



3 : System modules

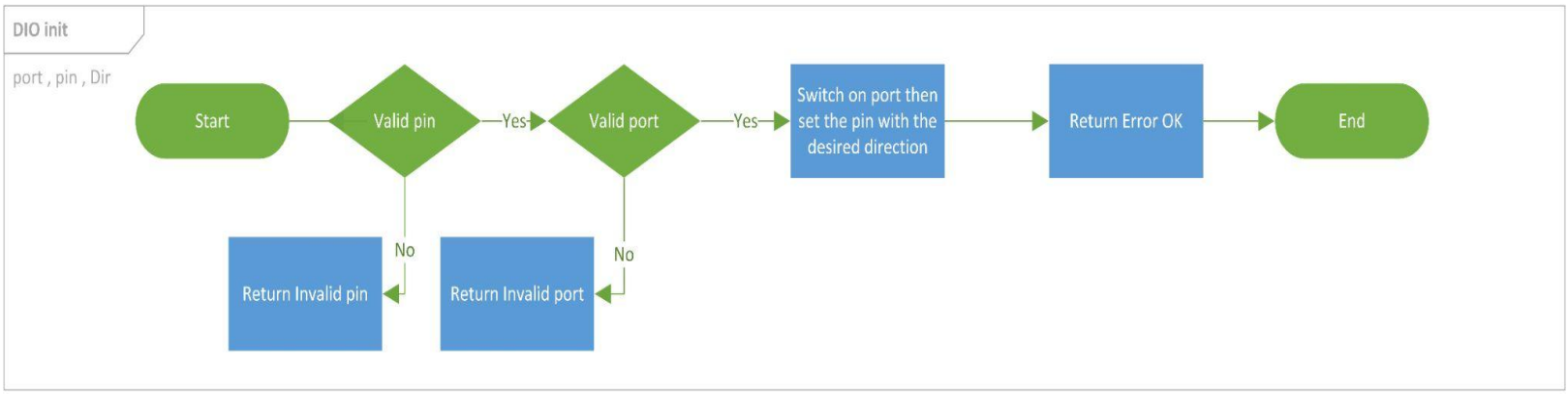
3.1: Module architecture

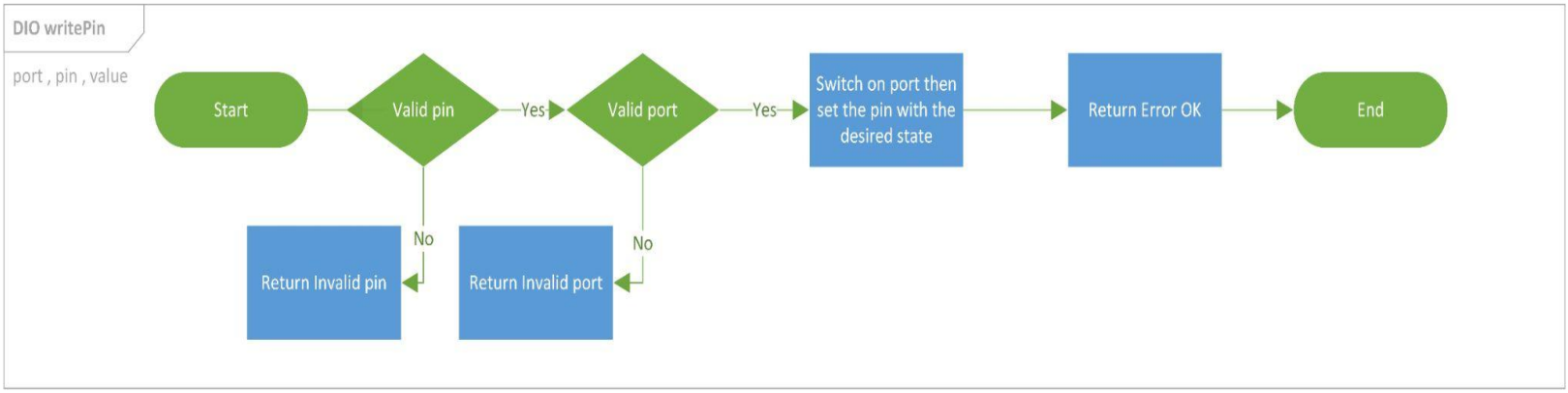


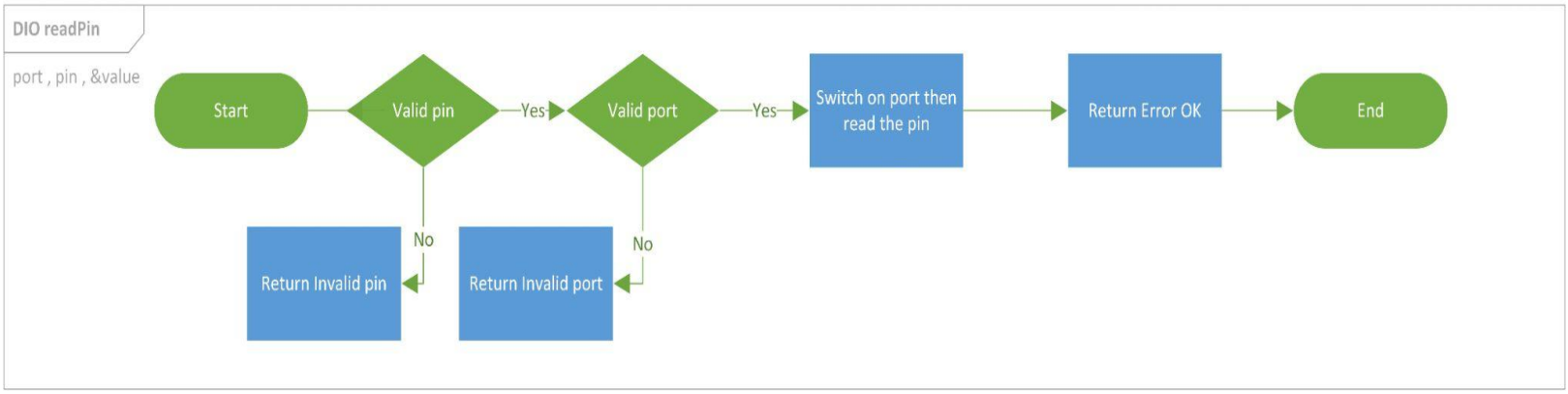
3.2: MCAL APIs

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

<table>
<tr><td></td><td colspan="3">);</td></tr>
<tr><td rowspan="3">Parameters (in)</td><td>Port, pin</td><td colspan="2">Channel ID</td></tr>
<tr><td rowspan="2">value</td><td rowspan="2">Pointer to store the level</td><td>STD_HIGH</td></tr>
<tr><td>STD_LOW</td></tr>
<tr><td>Return</td><td colspan="3"><table><tr><td rowspan="3">u8_en_dioErrors</td><td>DIO_E_OK</td></tr><tr><td>DIO_InvalidPin</td></tr><tr><td>DIO_InvalidPort</td></tr></table></td></tr>
<tr><td>Description</td><td colspan="3">This Function gets the level of the pin</td></tr>
</table>

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

- DIO_writePIN

<table>
<tr><td>Service name</td><td colspan="3">DIO_writePIN</td></tr>
<tr><td>Syntax</td><td colspan="3">u8_en_dioErrors DIO_writePIN (
en_dioPortsType port, en_dioPinsType pin,
u8_en_dioLevelType state
);</td></tr>
<tr><td rowspan="3">Parameters (in)</td><td>Port, pin</td><td colspan="2">Channel ID</td></tr>
<tr><td rowspan="2">state</td><td rowspan="2">Value to be set</td><td>STD_HIGH</td></tr>
<tr><td>STD_LOW</td></tr>
<tr><td>Return</td><td colspan="3"><table><tr><td rowspan="3">u8_en_dioErrors</td><td>DIO_E_OK</td></tr><tr><td>DIO_InvalidPin</td></tr><tr><td>DIO_InvalidPort</td></tr></table></td></tr>
<tr><td>Description</td><td colspan="3">This Function sets the level of the pin</td></tr>
</table>

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

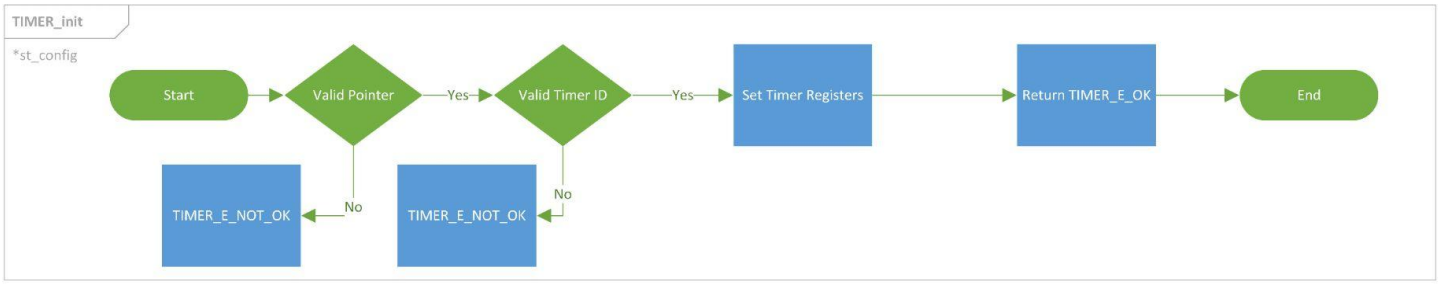
- DIO_init

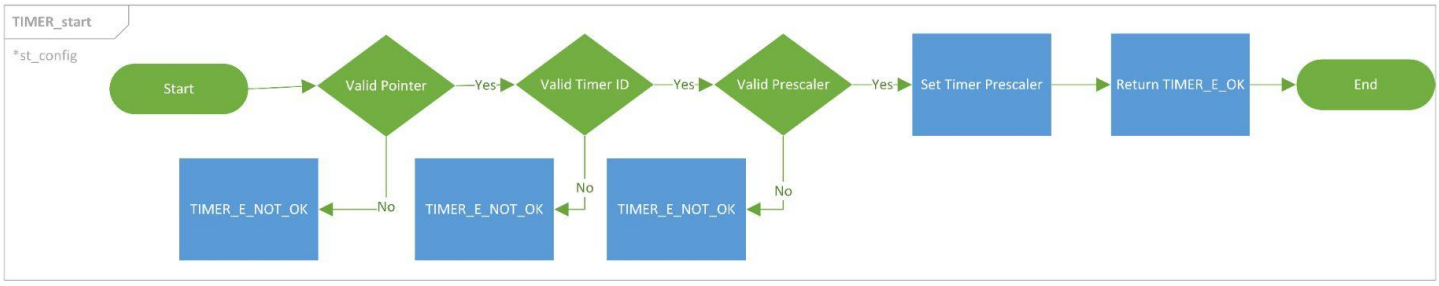
<table>
<tr><td>Service name</td><td colspan="3">DIO_init</td></tr>
<tr><td>Syntax</td><td colspan="3">u8_en_dioErrors DIO_init (
en_dioPortsType port,
en_dioPinsType pin,
u8_en_dioDirType direction
);</td></tr>
<tr><td rowspan="3">Parameters (in)</td><td>Port, pin</td><td colspan="2">Channel ID</td></tr>
<tr><td rowspan="2">direction</td><td rowspan="2">Value to be set</td><td>STD_INPUT</td></tr>
<tr><td>STD_OUTPUT</td></tr>
<tr><td>Return</td><td colspan="3"><table><tr><td>DIO_Errors</td><td>DIO_E_OK</td></tr><tr><td></td><td>DIO_InvalidPin</td></tr><tr><td></td><td>DIO_InvalidPort</td></tr></table></td></tr>
<tr><td>Description</td><td colspan="3">This Function sets the Direction of the pin</td></tr>
</table>

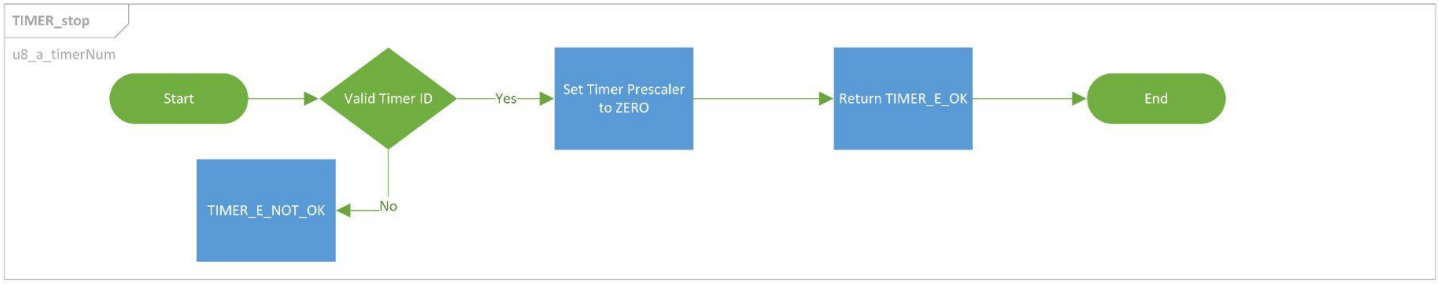
- This function shall return DIO_InvalidPin if pin number is invalid
- This function shall return DIO_InvalidPort if port number is invalid.

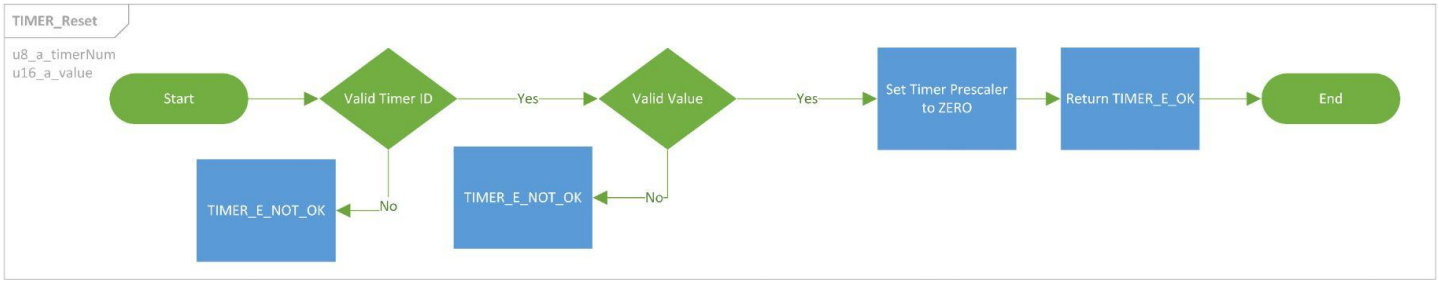
3.2.2: Timer API:

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	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	Pointer to the configuration structure
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 (st_timerConfigType* st_config);	
Parameters (in)	st_config	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 st_config is NULL
- This function shall return TIMER_E_NOK if any of the configuration elements is invalid.

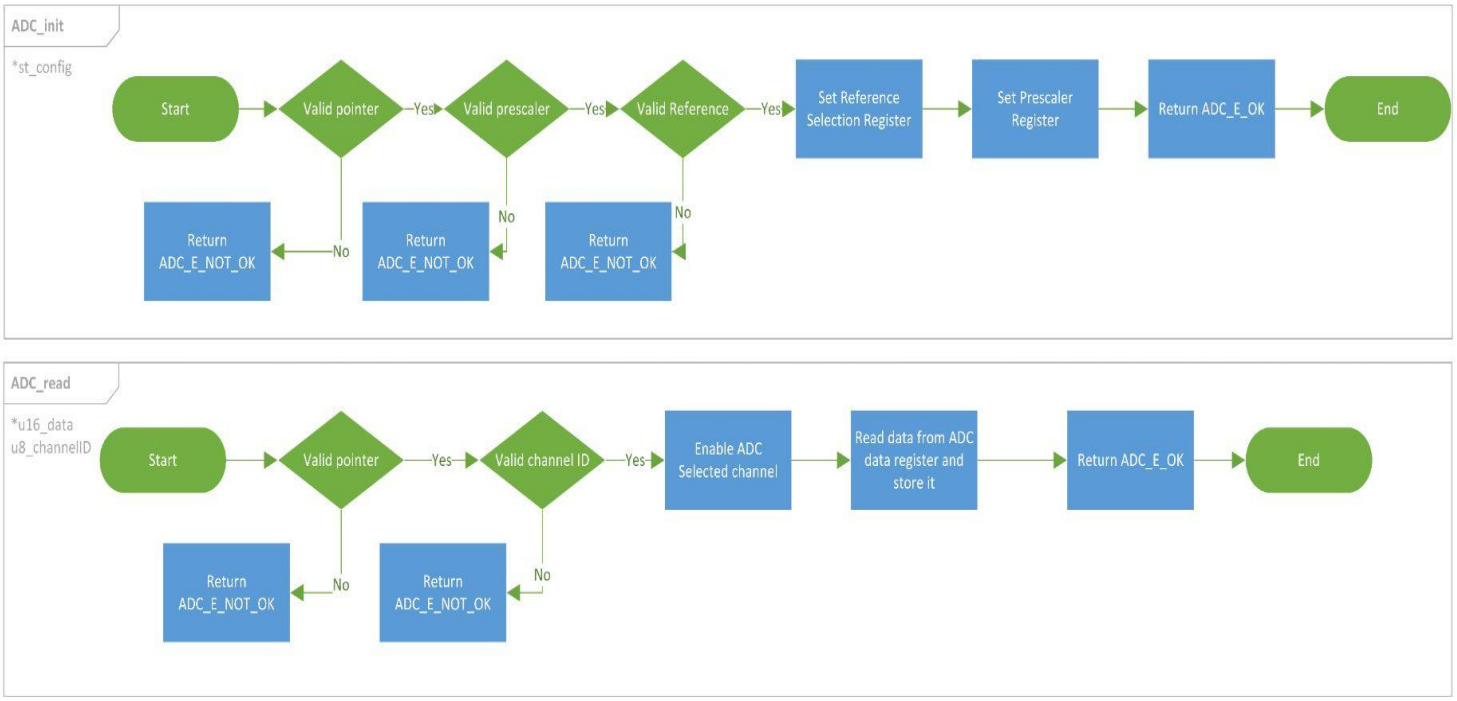
- TIMER_setCallBack

<table>
<tr><td>Service name</td><td colspan="2">TIMER_setCallBack</td></tr>
<tr><td>Syntax</td><td colspan="2">u8_en_timerErrorsType TIMER_setCallBack (
void(*a_timerCallBack)(void),
u8_en_timerNumberType u8_a_timerNum
);</td></tr>
<tr><td rowspan="2">Parameters (in)</td><td>*a_timerCallBack</td><td>Pointer to the Callback function</td></tr>
<tr><td>u8_a_timerNum</td><td>Timer ID</td></tr>
<tr><td>Return</td><td colspan="2"><table><tr><td rowspan="2">u8_en_timerErrorsType</td><td>TIMER_E_OK</td></tr><tr><td>TIMER_E_NOT_OK</td></tr></table></td></tr>
<tr><td>Description</td><td colspan="2">This Function reset the TIMER</td></tr>
</table>

- 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.

3.2.3 : 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_types.h

- u8_en_adcErrorsType

Name	u8_en_adcErrorsType
Type	Enumeration

Range	<table><tr><td>ADC_E_OK</td><td>0x00</td><td>ADC error OK</td></tr><tr><td>ADC_E_NOT_OK</td><td>0x04</td><td>ADC error</td></tr></table>
Description	u8_en_adcErrorsType
Available via	adc_types.h

- u8_en_adcChannelId

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

- u8_en_adcRefType

Name	u8_en_adcRefType
Type	Enumeration
Range	Shall contain all reference selection modes
Description	u8_en_adcRefType
Available via	adc_types.h

- u8_en_adcPrescalerType

Name	u8_en_adcPrescalerType
Type	Enumeration
Range	Shall contain all Prescaler selection modes

Description	u8_en_adcPrescalerType
Available via	adc_types.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	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

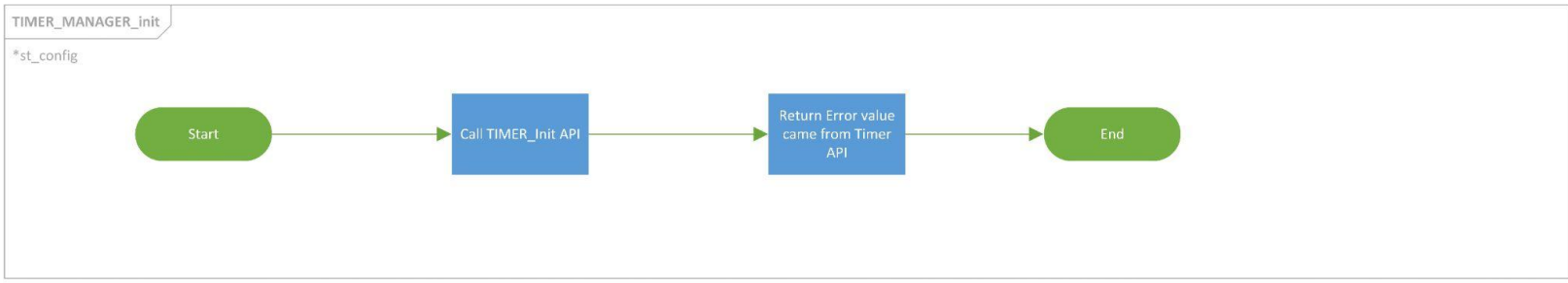
Service name	ADC_read	
Syntax	u8_en_adcErrorsType ADC_read (u8_en_adcChannelId u8_channelID, uint16_t * u16_data);	
Parameters (in)	u16_data	Pointer to variable where to store the ADC value
	u8_channel ID	ADC Channel ID
Return	u8_en_adcErrorsType	ADC_E_OK

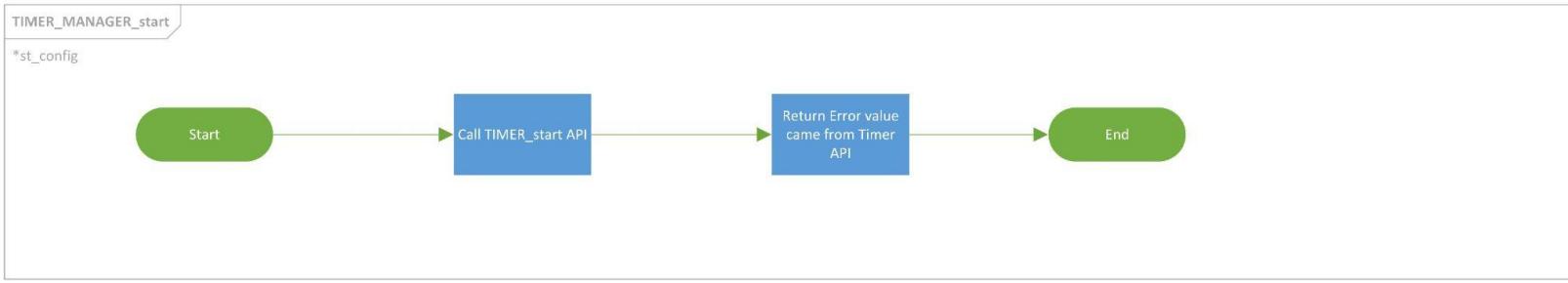
	ADC_E_NOT_OK
Description	This Function read ADC

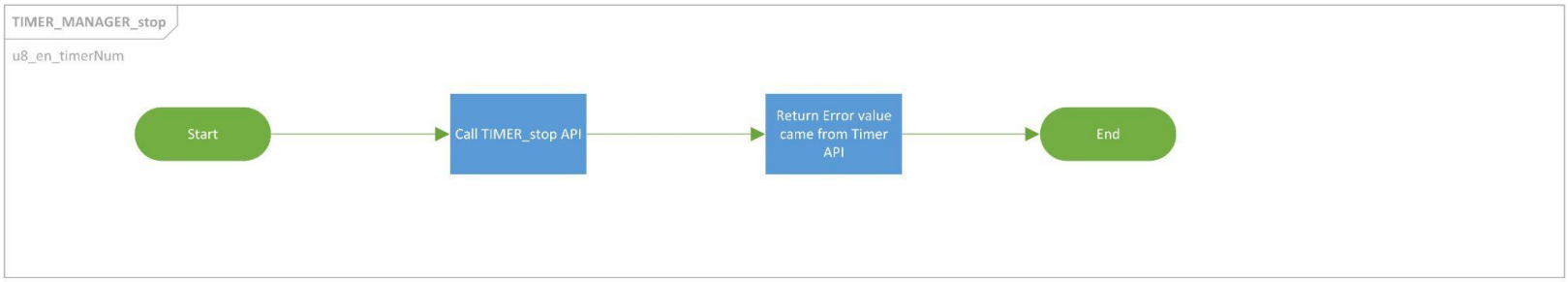
3.3 : HAL APIs

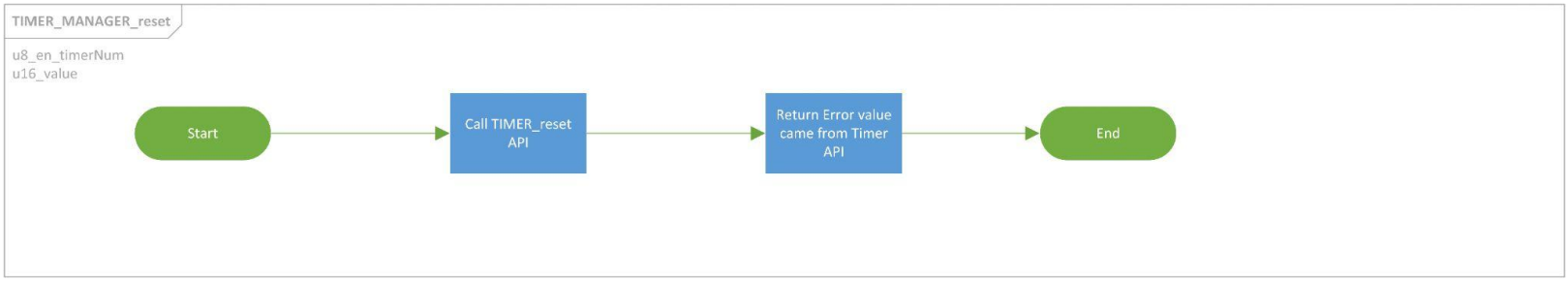
3.3.1: Timer Manager API:

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	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	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	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

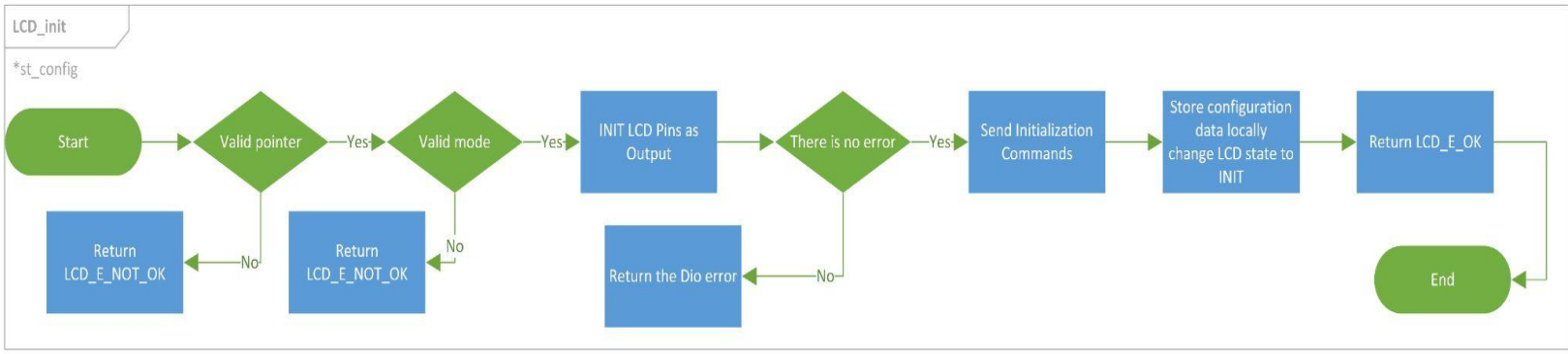
- TIMER_Manager_reset

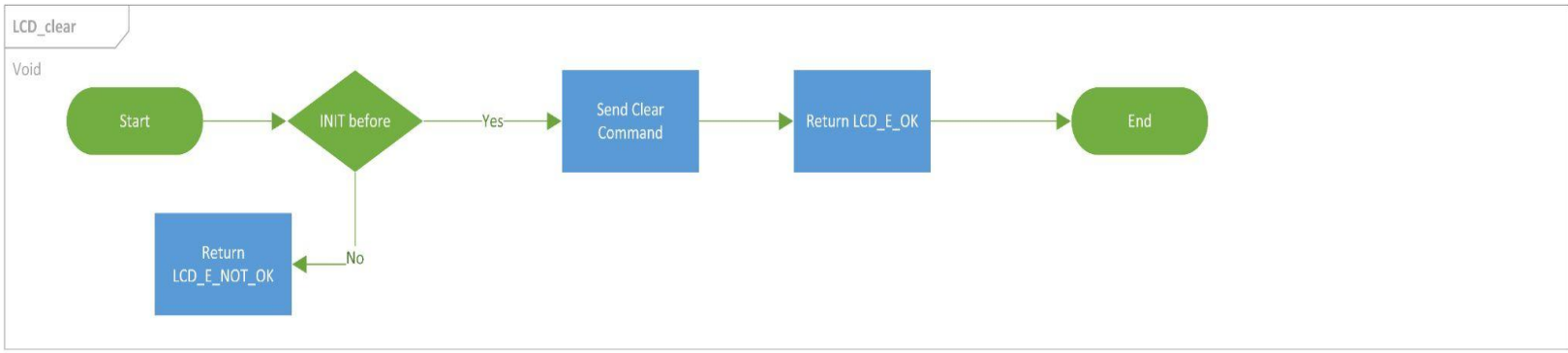
Service name	TIMER_Manager_reset	
Syntax	u8_en_timerErrorsType TIMER_Manager_reset (st_timerConfigType* st_config);	
Parameters (in)	st_config	Pointer to the configuration structure
Return	u8_en_timerErrorsType	TIMER_E_OK TIMER_E_NOT_OK
Description	This Function stop TIMER	

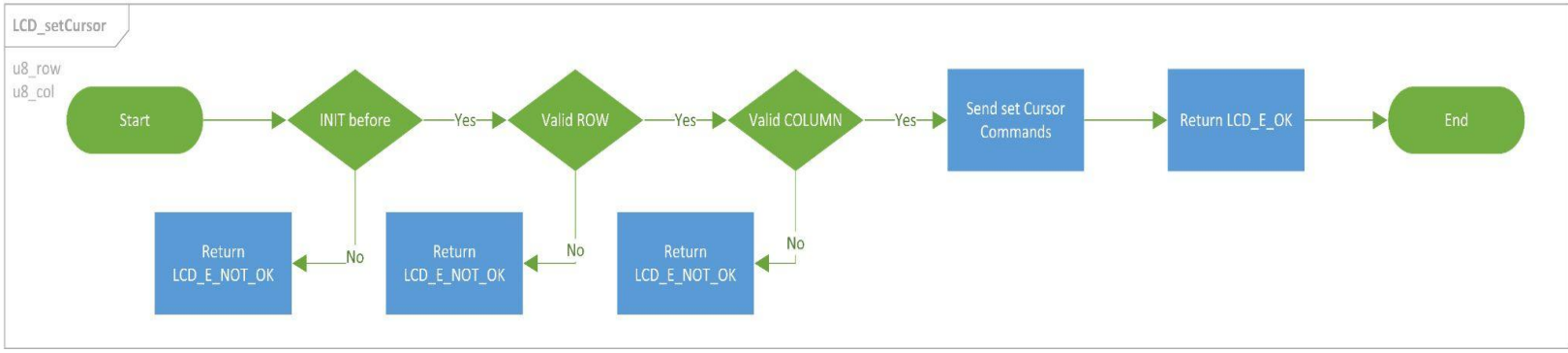
  - This function shall return TIMER_E_NOK if st_config is NULL
  - This function shall return TIMER_E_NOK if any of the configuration elements is invalid.

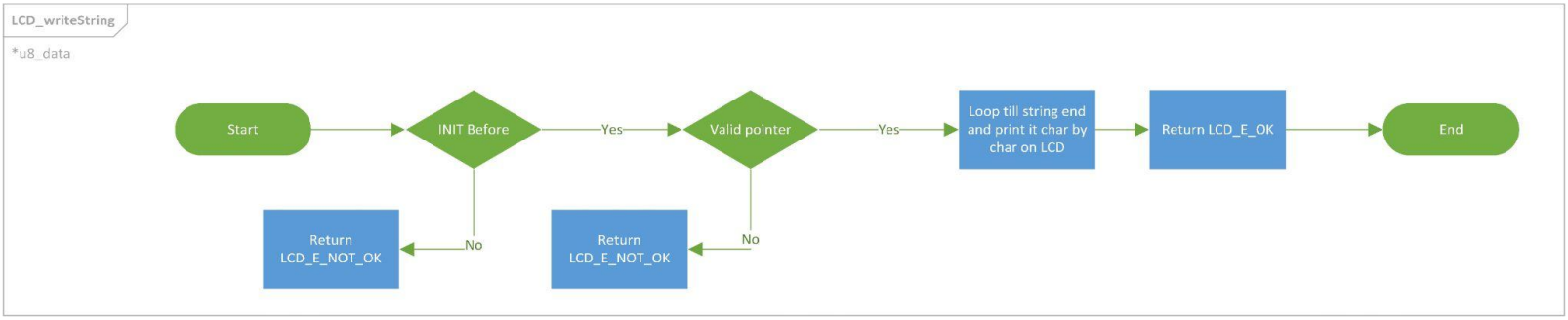
3.3.2: LCD API:

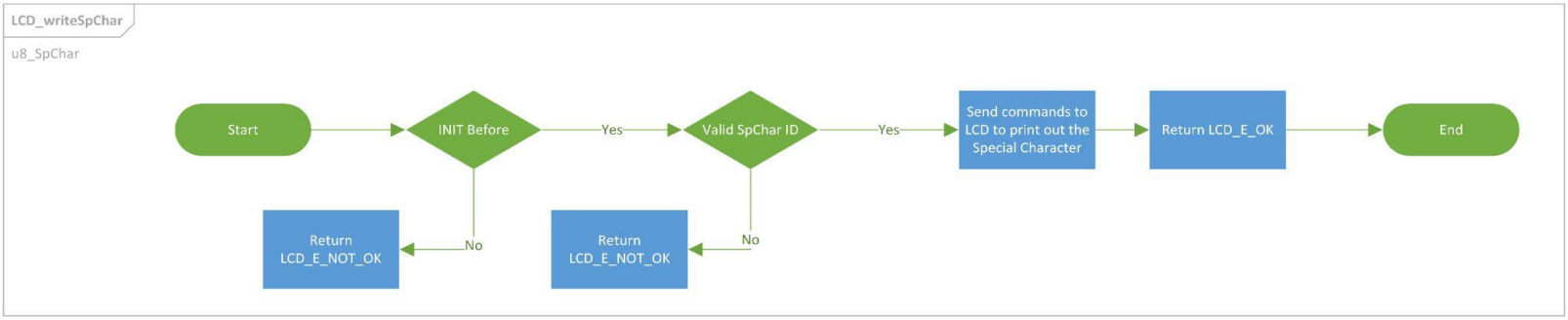
3.3.2.1 :Flowcharts:











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	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

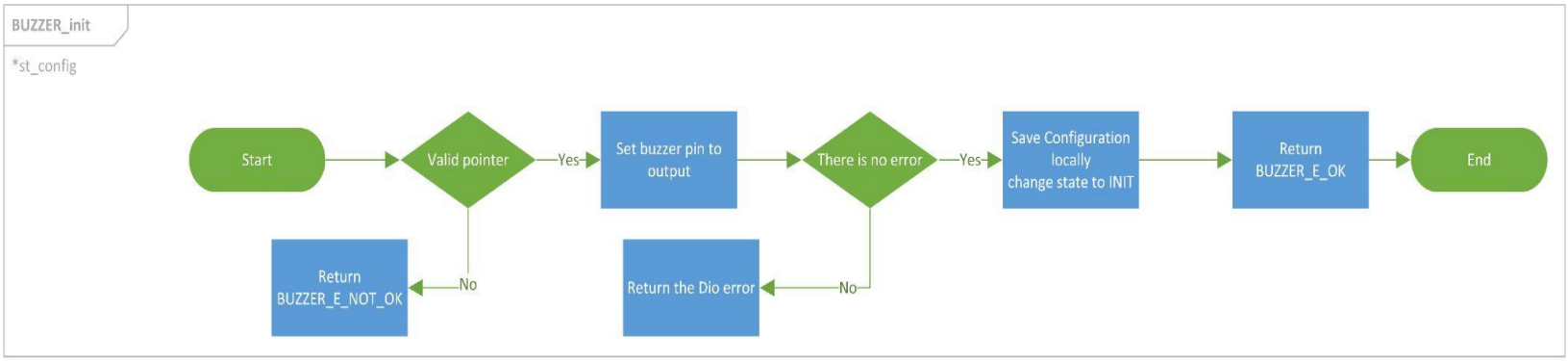
Service name	LCD_writeString	
Syntax	u8_en_lcdErrorsType LCD_writeString (uint8_t* u8_data);	
Parameters (in)	u8_data	Pointer to string to it print on screen
Return	u8_en_lcdErrorsType	LCD_E_OK LCD_E_NOT_OK
Description	This Function write a string on LCD	

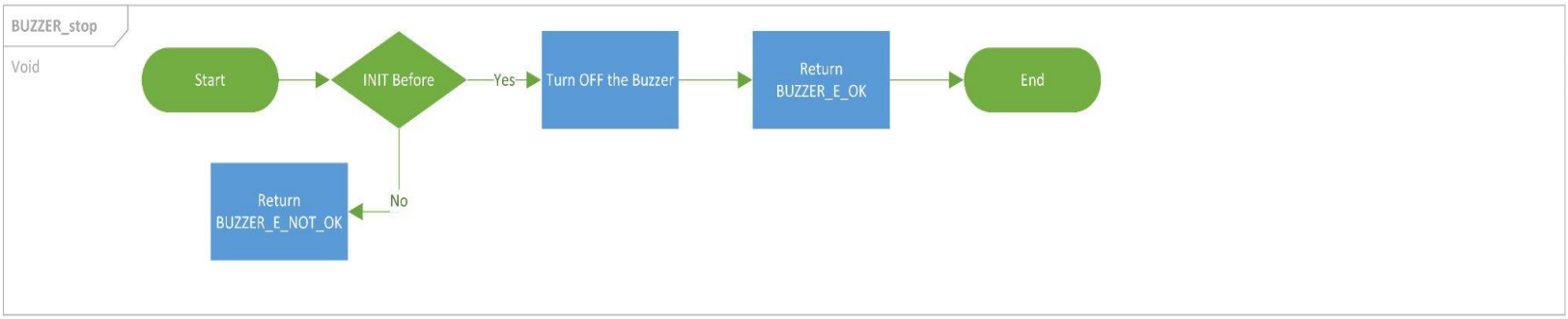
- LCD_writeSpChar

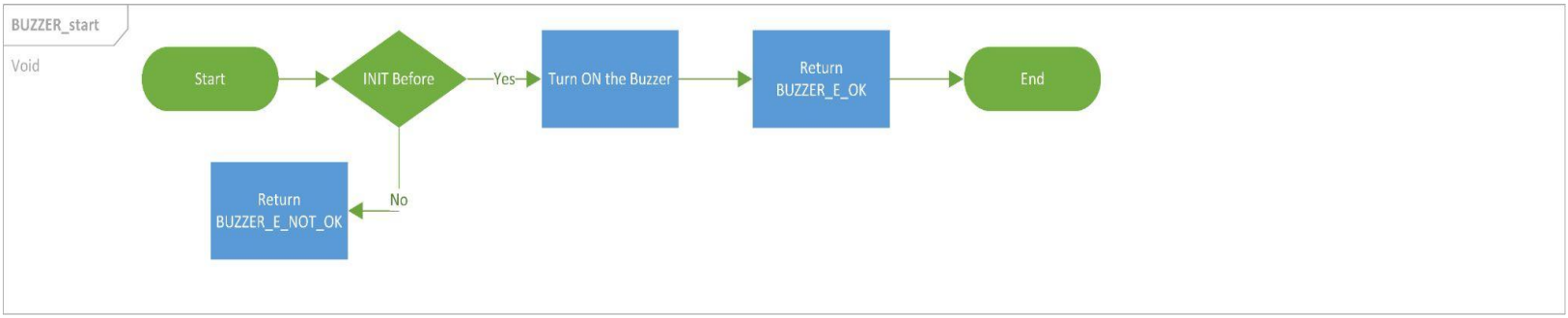
Service name	LCD_writeSpChar	
Syntax	u8_en_lcdErrorsType LCD_writeSpChar (u8_en_lcdSpCharType u8_SpChar);	
Parameters (in)	u8_SpChar	Special character ID to it print on screen
Return	u8_en_lcdErrorsType	LCD_E_OK LCD_E_NOT_OK
Description	This Function write a special character on LCD	

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	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	BUZZER_E_OK BUZZER_E_NOT_OK
Description	This Function starts Buzzer	

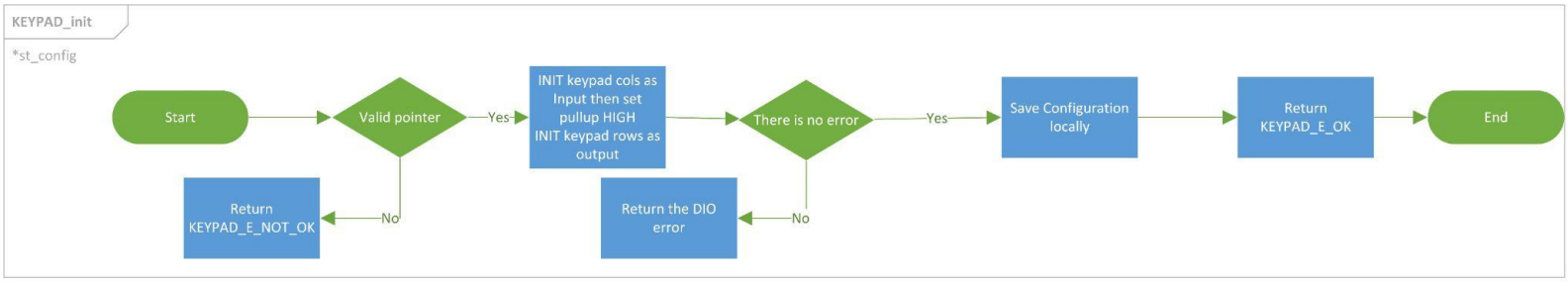
- BUZZER_stop

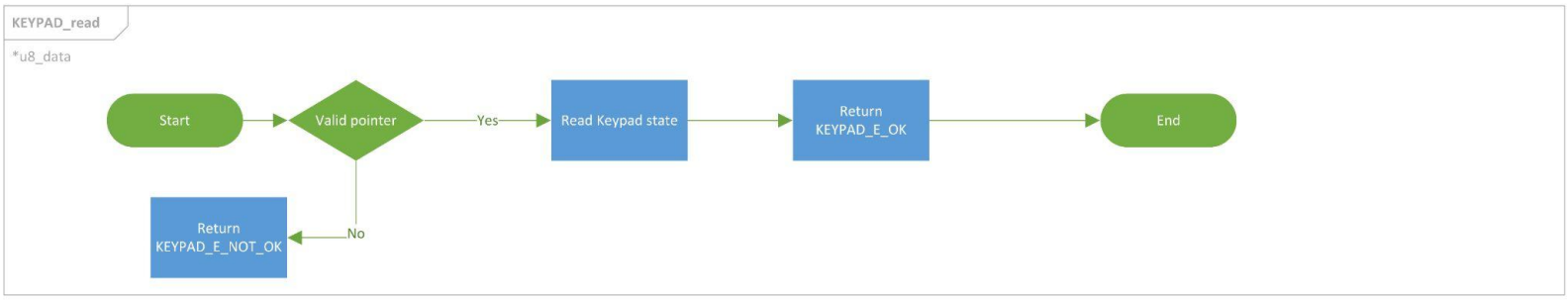
Service name	BUZZER_stop
Syntax	u8_en_buzzerErrorsType BUZZER_stop(void);
Parameters (in)	None
Return	

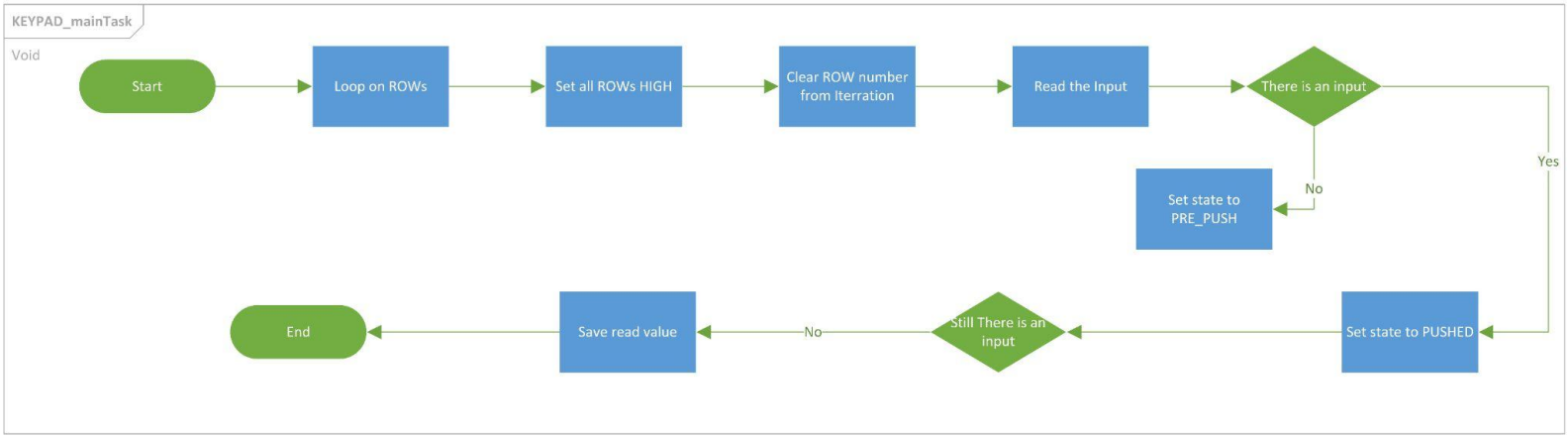
	u8_en_buzzerErrorsType	BUZZER_E_OK
		BUZZER_E_NOT_OK
Description	This Function stops Buzzer	

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 (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	

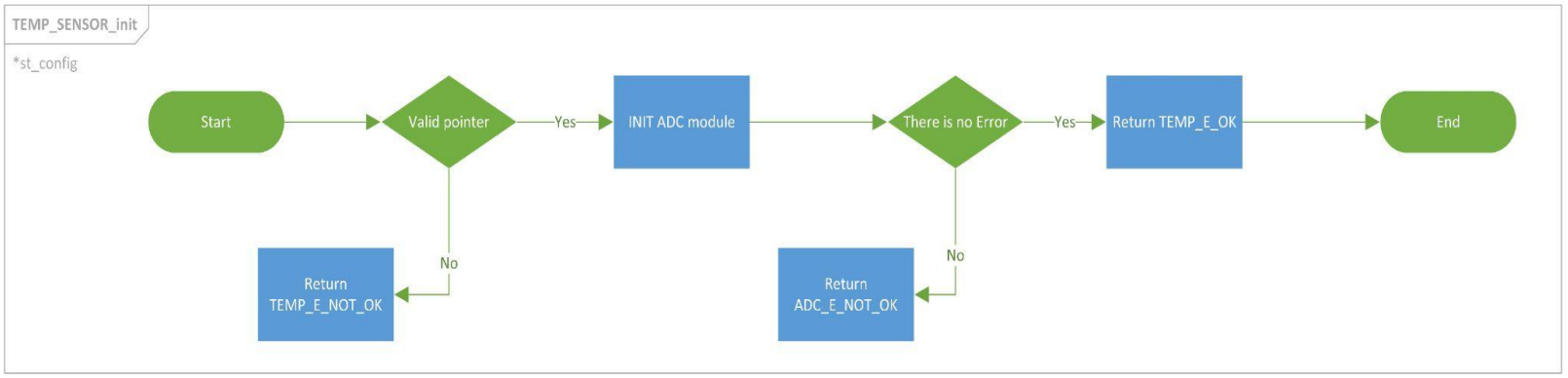
- This function shall return KEYPAD_E_NOK if st_config is NULL
- This function shall return KEYPAD_E_NOK if any of the configuration elements is invalid.

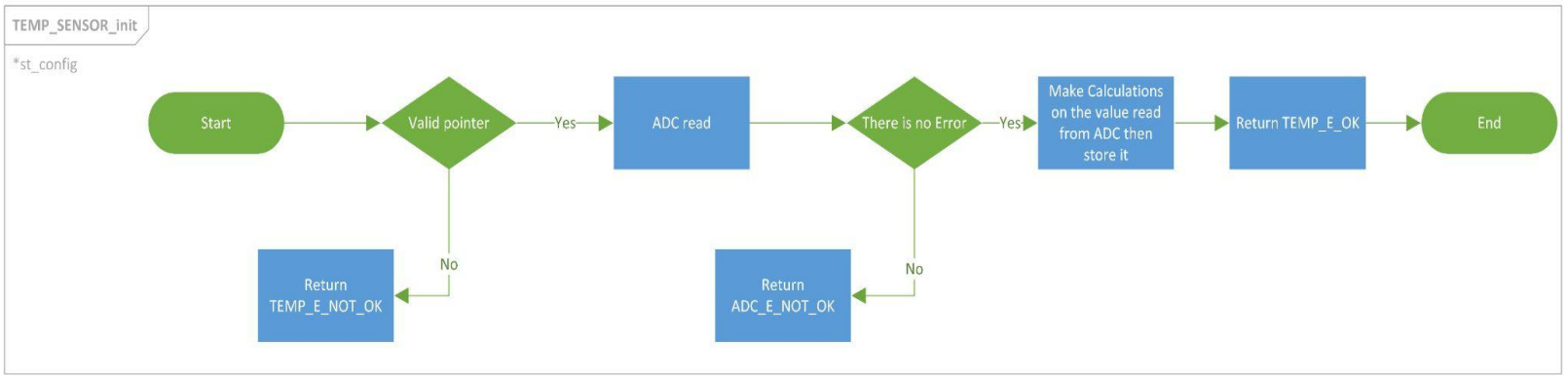
- KEYPAD_read

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

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 (st_tempSensorConfigType* st_config);	
Parameters (in)	st_config	Pointer to the configuration structure
Return	u8_en_tempSensorErrorsType	TEMP_E_OK TEMP_E_NOT_OK
Description	This Function Initialize Temperature sensor module	

  - 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 (st_tempSensorConfigType* st_config, 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:

- u8_programStateType

Name	u8_programStateType		
Type	Enumeration		
Range	APP_WELCOME	0x00	Welcome mode
	APP_SET_TEMP	0x01	Setting temp mode
	APP_WORKING	0x02	Working mode
Description	u8_programStateType		
Available via	app.h		

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