

Air Conditioner System Design

By : Team 4

Contents

Flowchart Table	3
Detailed Requirements.....	4
Layered architecture	5
System modules	5
MCAL	6
DIO API :.....	6
<i>ADC API</i> :.....	7
TIMER API :.....	11
HAL	16
BUZZER API:.....	16
LCD API:.....	18
LM35 API :	21
Timer Manager API :.....	23
KEYPAD API:	25
App	27
App API:.....	27

Flowchart Table

Figure 1 Desgin_arch	5
Figure 2 Air_Conditioner_Desgin_Arch	5
Figure 3 DIO_APIS Flowchar	6
Figure 4 ADC_init()	7
Figure 5 ADC_channel_init()	8
Figure 6 ADC_channel_read()	8
Figure 7 ADC_channel_read_INT()	9
Figure 8 ADC_channel_read_ISR()	10
Figure 9 TIMERx_init	11
Figure 10 TIMERx_reset()	12
Figure 12 TIMERx_start()	13
Figure 11 TIMERx_stop()	13
Figure 13 TIMERx_setCallBack()	14
Figure 15 TIMERxSetValue	14
Figure 14 TIMERx_setValue()	14
Figure 16 TIMERx_CTC_SetCompare()	15
Figure 17 BUZZER_init()	16
Figure 18 BUZZER_start()	17
Figure 19 BUZZER_stop()	17
Figure 20 LCD_clear()	18
Figure 21 LCD_init ()	18
Figure 22 LCD_writeString()	19
Figure 23 LCD_setCursor()	19
Figure 24 LCD_writeSpChar()	20
Figure 25 LM35_INIT()	21
Figure 26 LM35_read_temp()	22
Figure 27 LM35_read_temp_INT()	22
Figure 28 TIMER_MANGER_init()	23
Figure 29 TIMER_MANGER_start()	24
Figure 30 TIMER_MANGER_setValue()	24
Figure 31 TIMER_MANGER_stop()	24
Figure 32 KEYPAD_init()	25
Figure 33 KEYPAD_read()	26
Figure 34 APP_init()	27
Figure 35 APP_start()	28
Figure 36 APP_set()	29
Figure 37 APP_welcome()	30
Figure 38 APP_working()	31
Figure 39 intToString()	32

Detailed Requirements

System Requirements:

- 1- When system start LCD prompt welcome message for 1 second, then display the default temp is 20, the message appears 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
 1. Button_1 : Increment
 2. Button_2 : Decrement
 3. Button_3 : Set temperature
 4. Button_4 : Adjust temperature
 5. Button_5 : Reset to default

Layered architecture

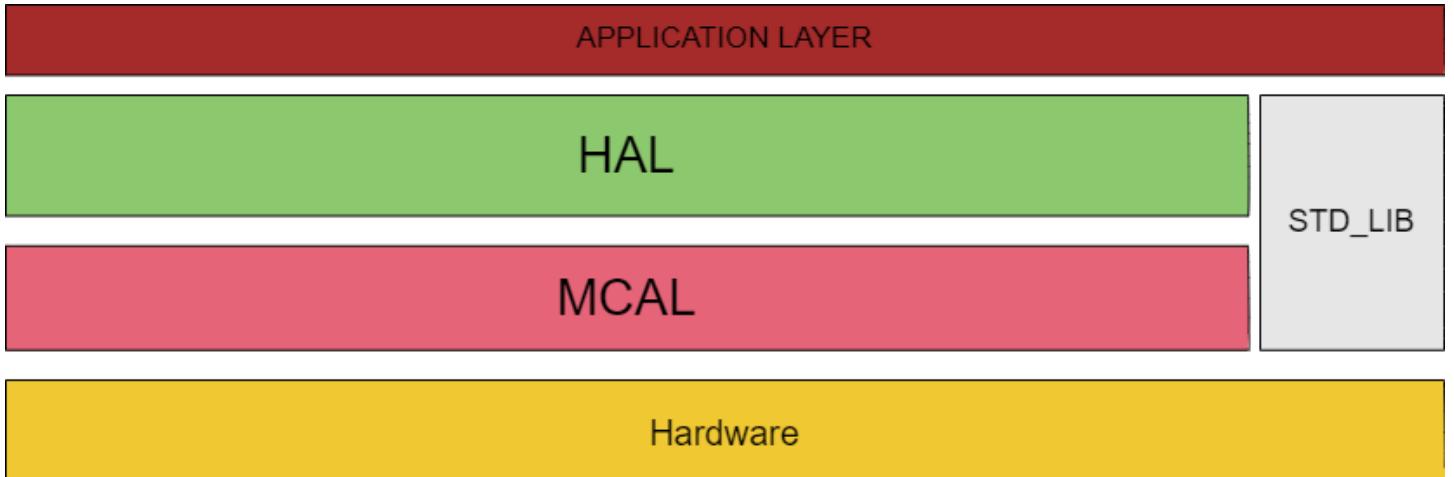


Figure 1 Desgin_arch

System modules

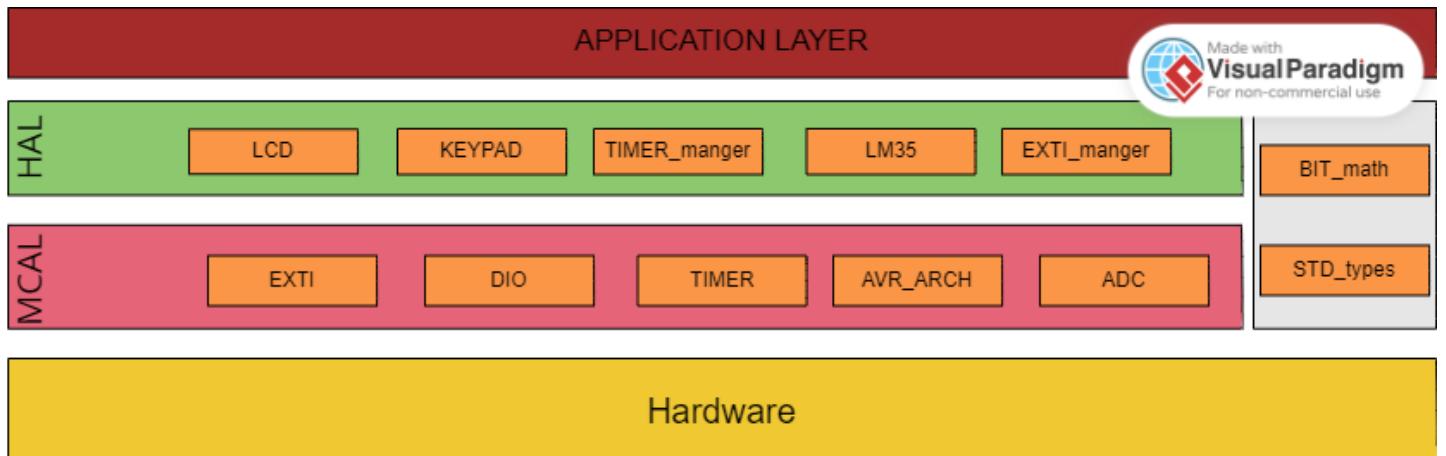


Figure 2 Air_Conditioner_Desgin_Arch

MCAL

DIO API :

Flowcharts:

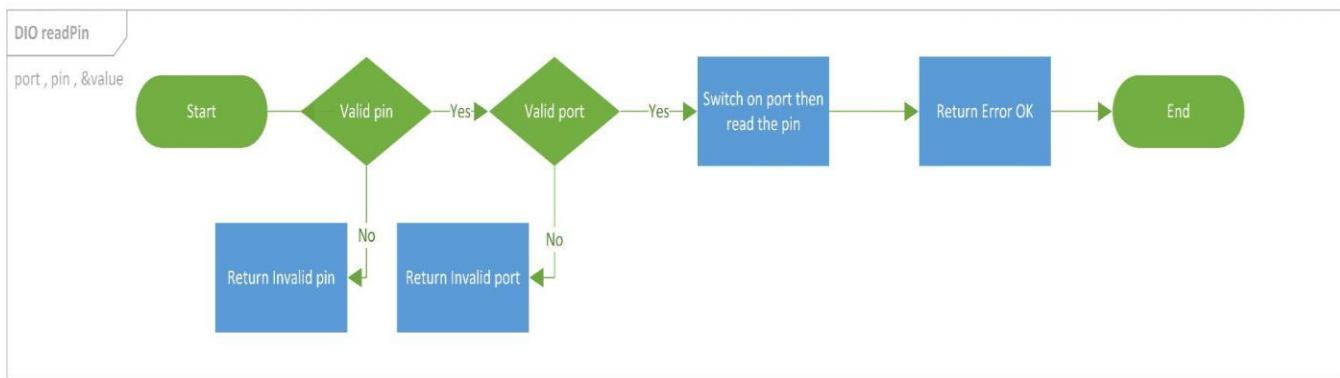
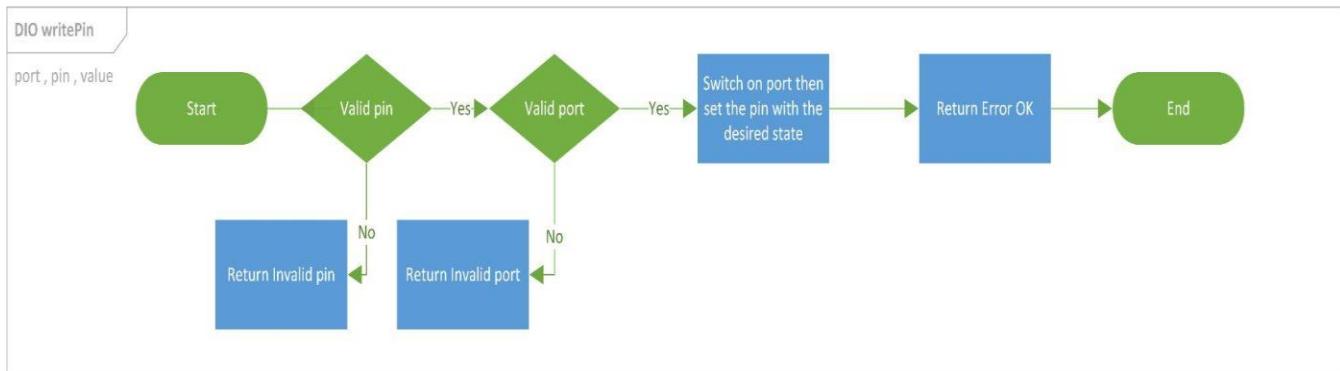
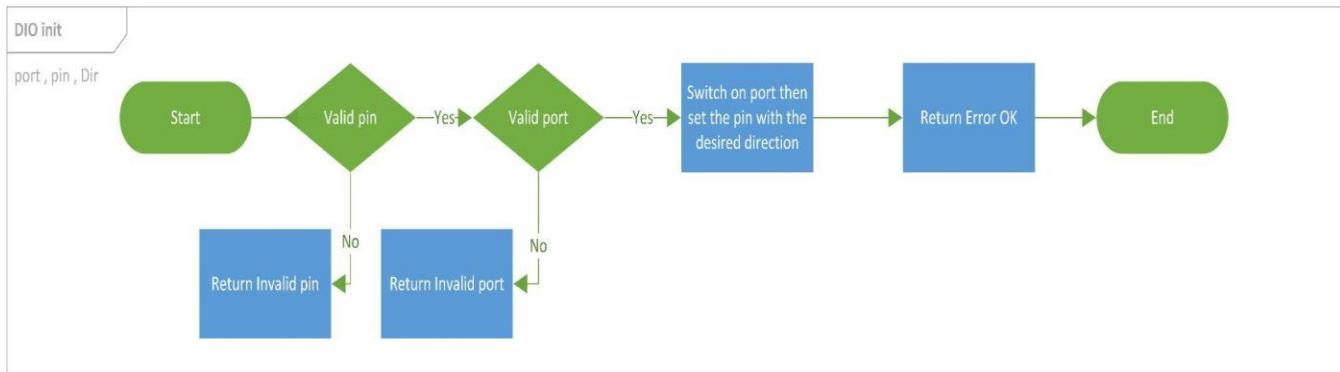


Figure 3 DIO_APIS Flowchar

ADC API :

Flowcharts:

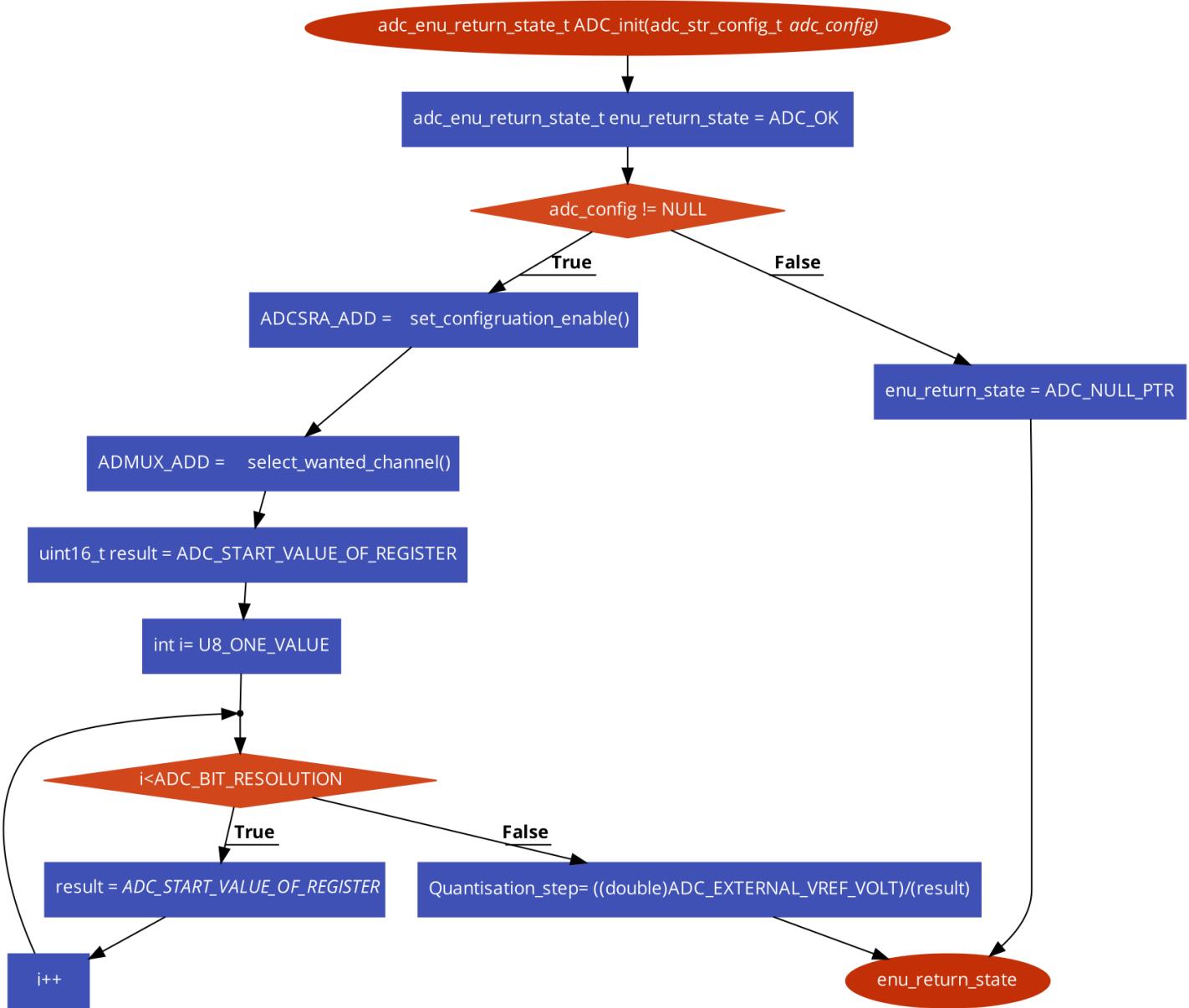


Figure 4 ADC_init()

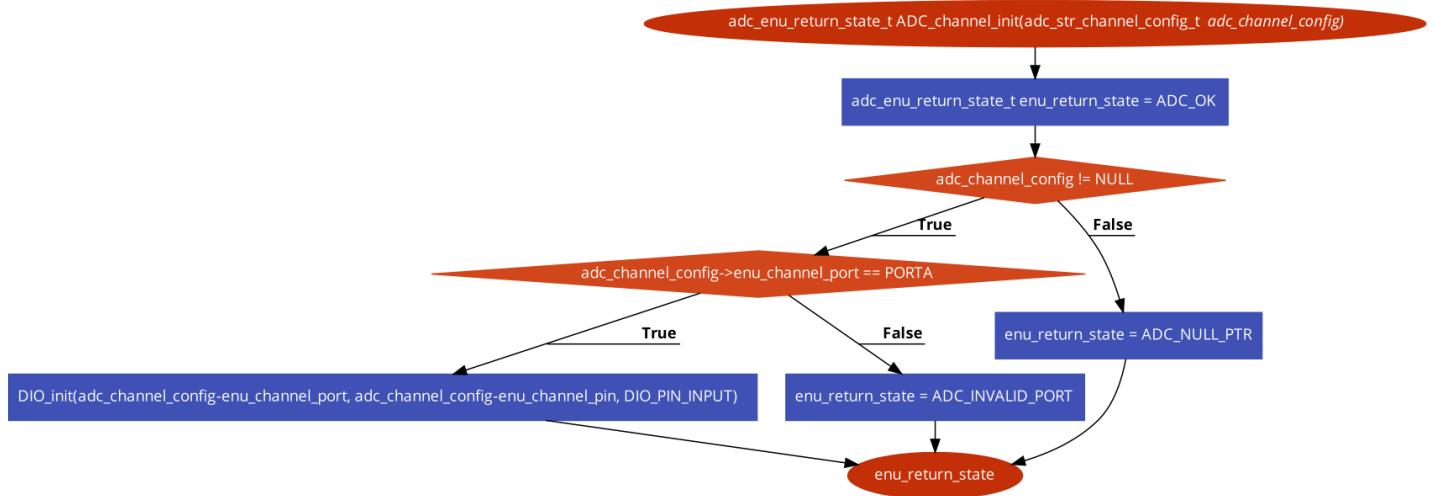


Figure 5 ADC_channel_init()

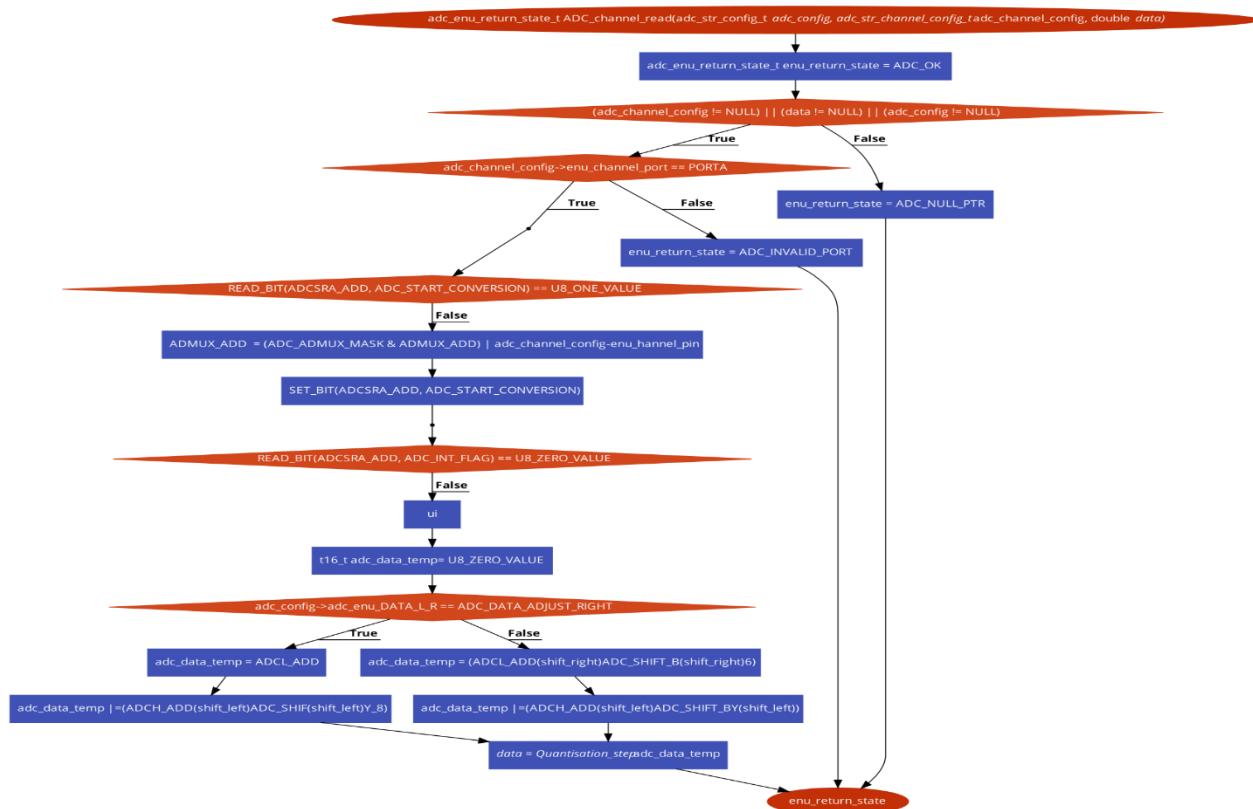


Figure 6 ADC_channel_read()

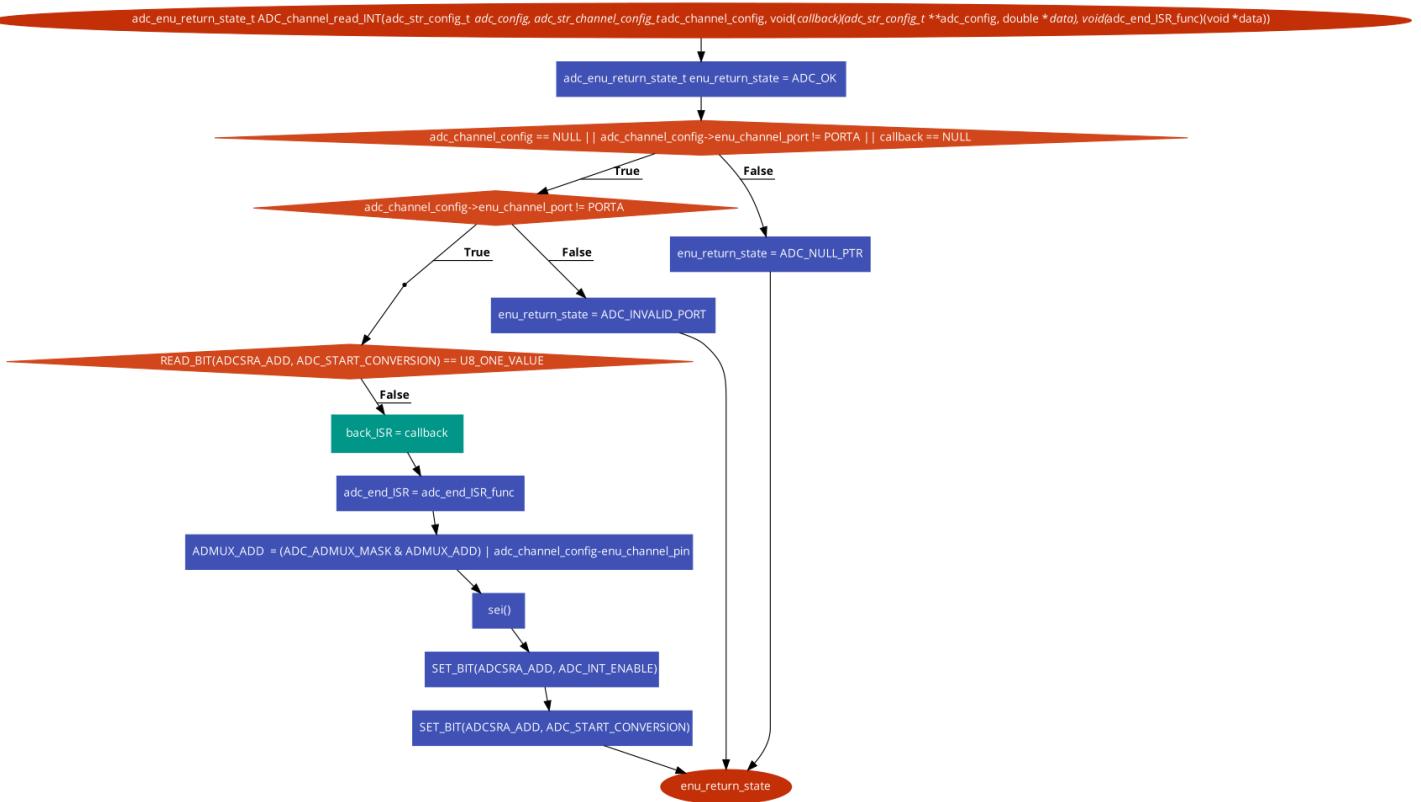


Figure 7 ADC_channel_read_INT()

```
static void ADC_channel_read_ISR(adc_str_config_t *adc_config, double data)
```

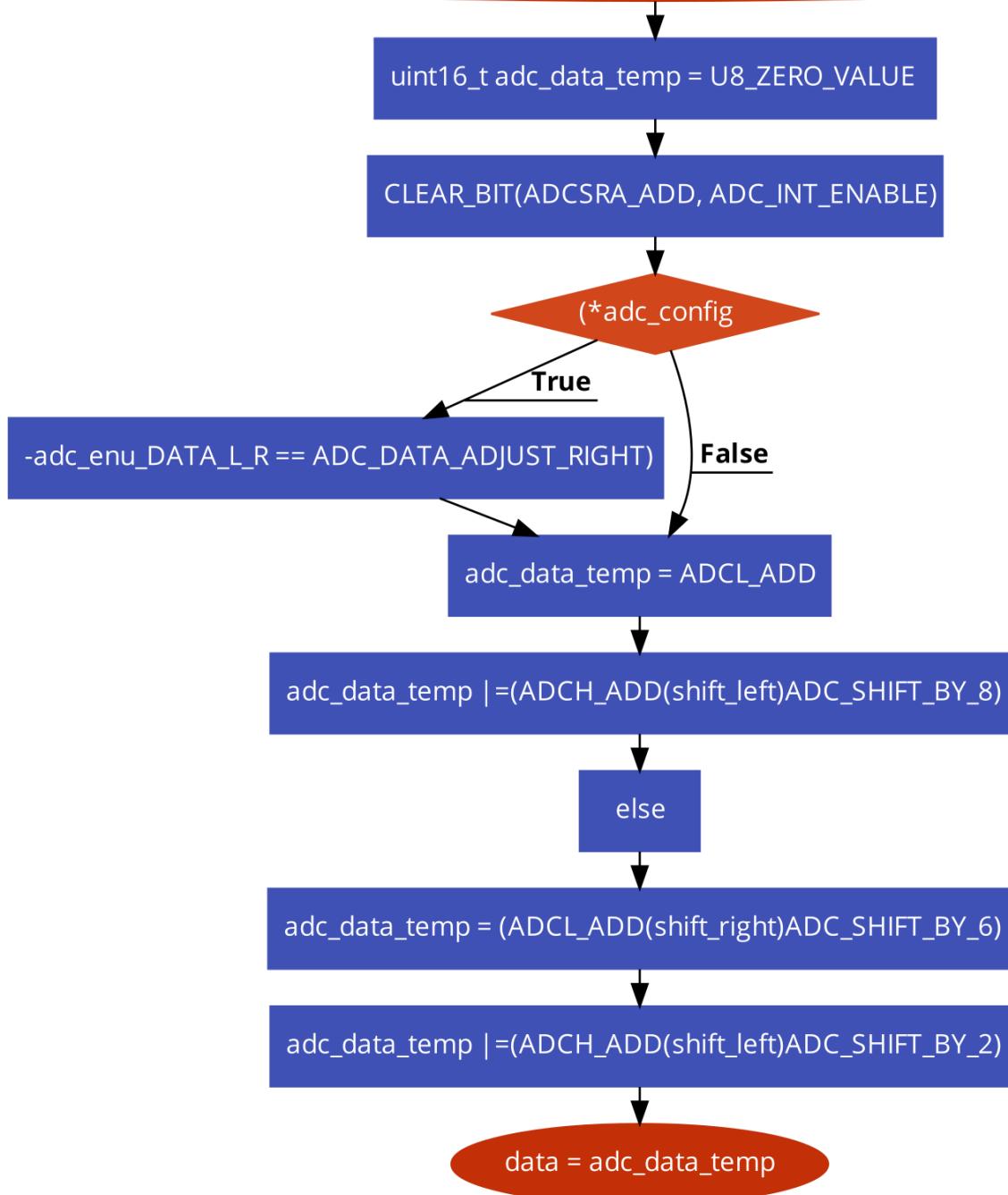


Figure 8 ADC_channel_read_ISR()

TIMER API :

Flowcharts:

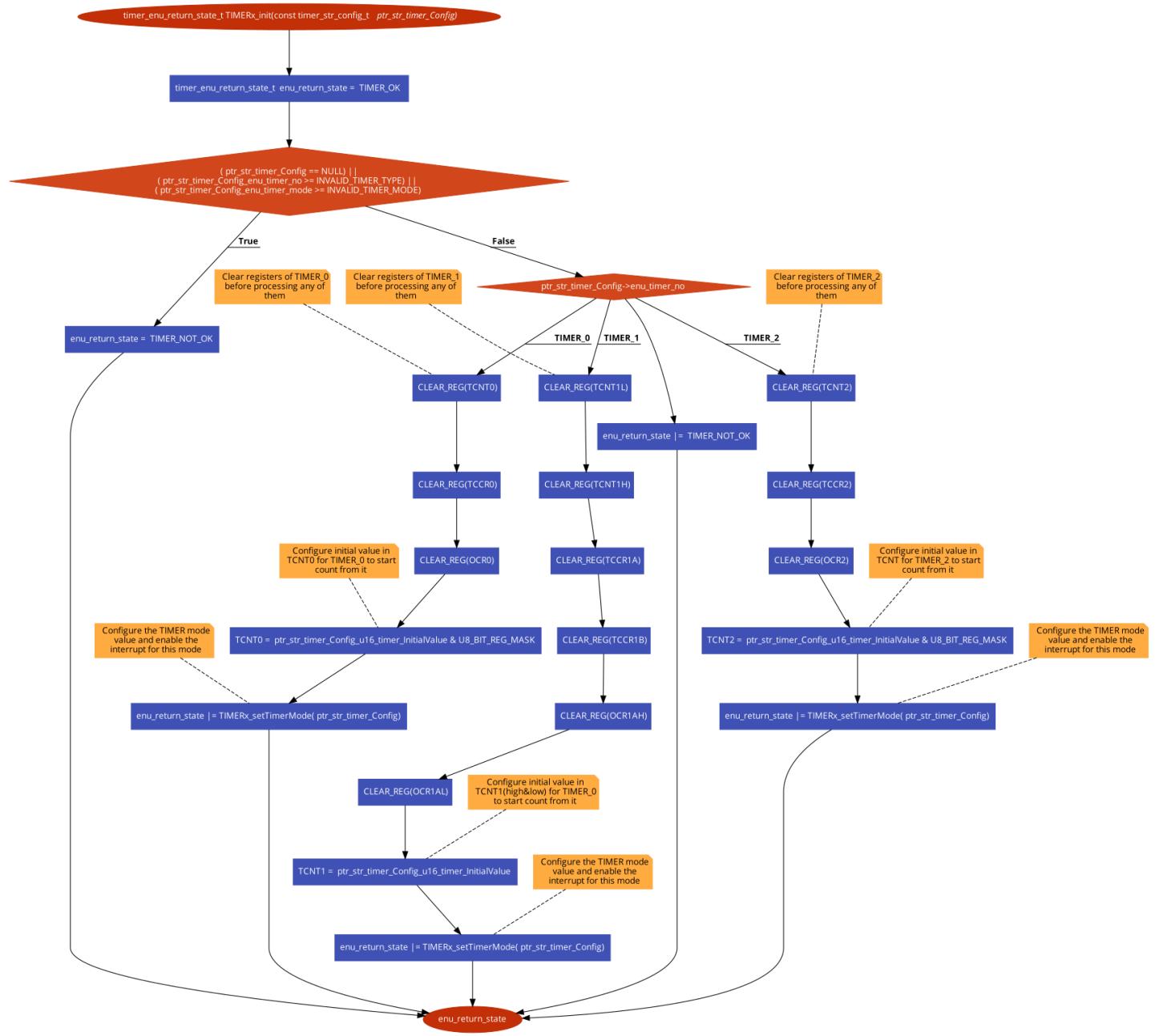


Figure 9 TIMERx_init

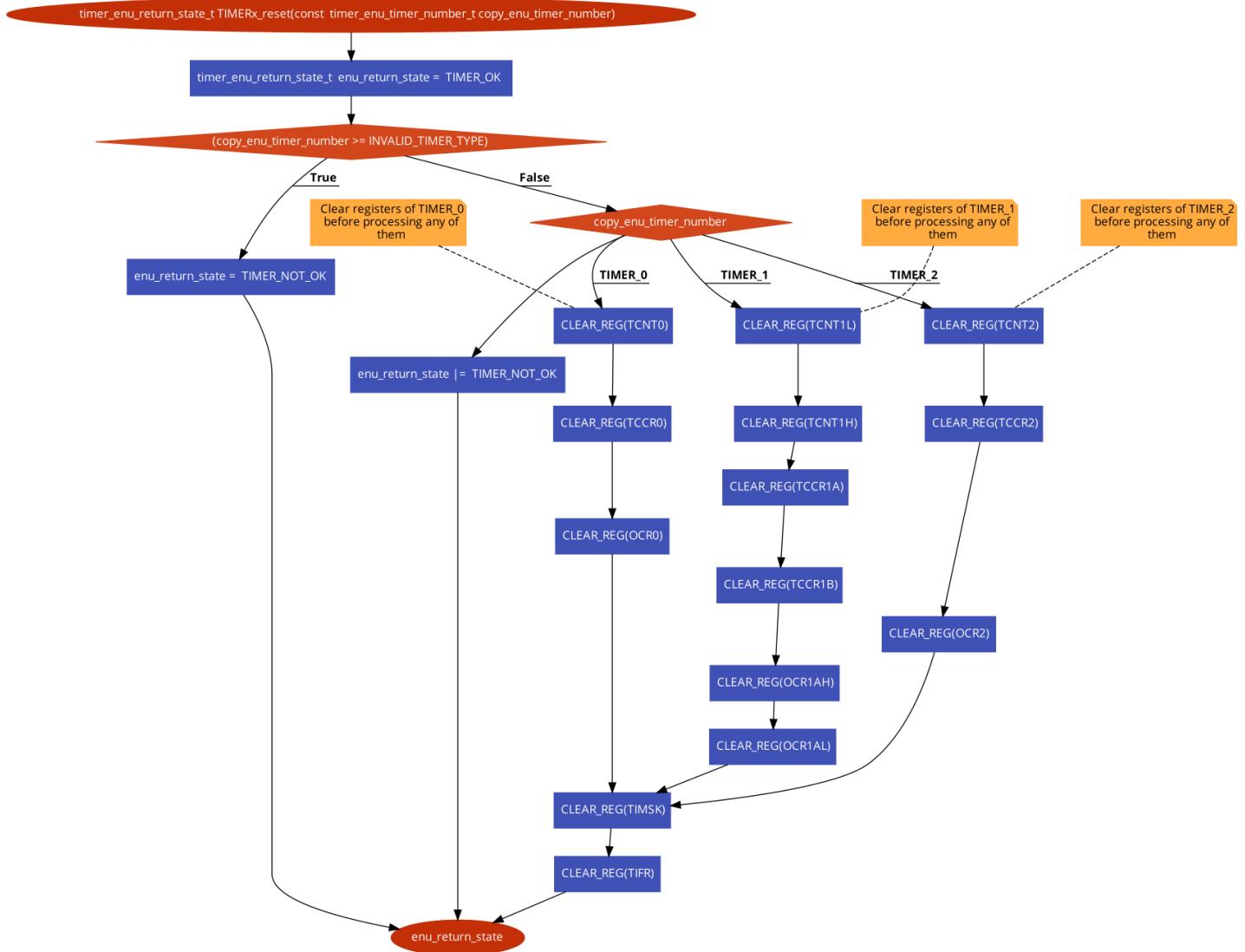


Figure 10 `TIMERx_reset()`

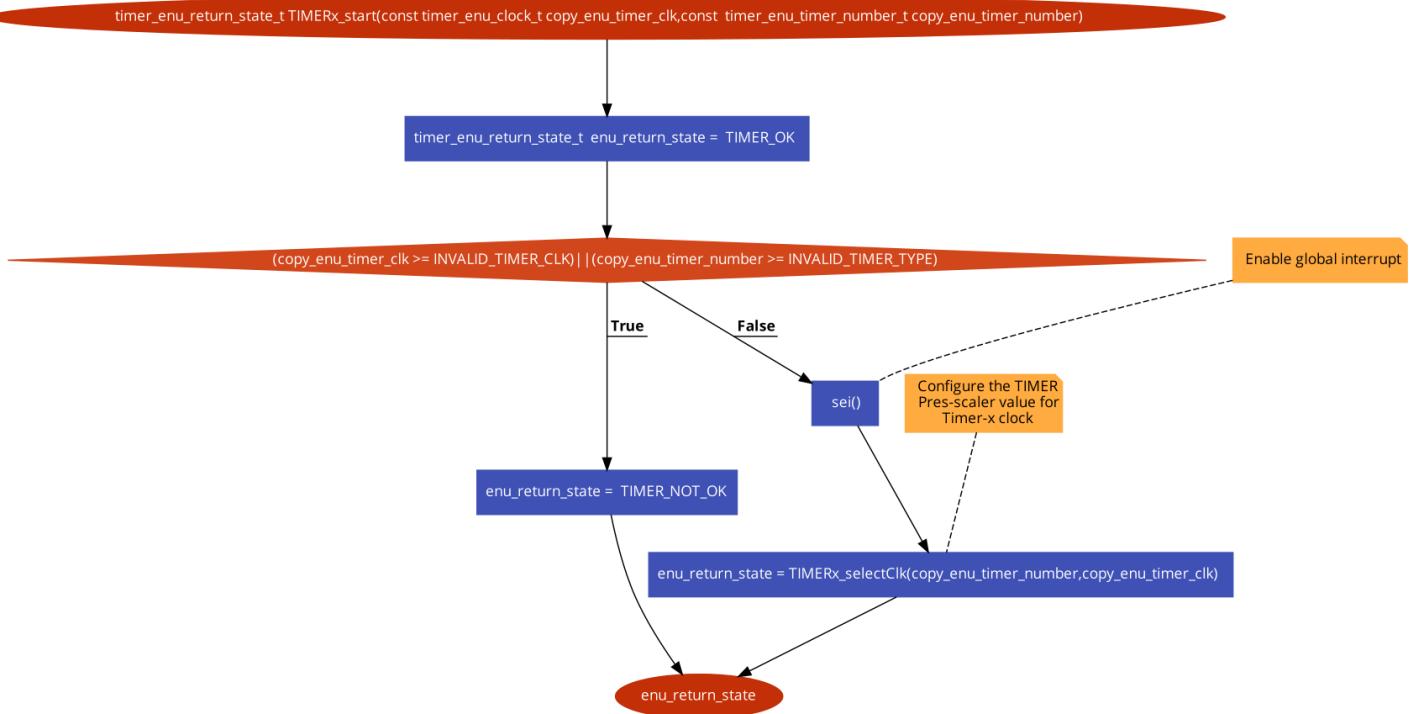


Figure 12 `TIMERx_start()`

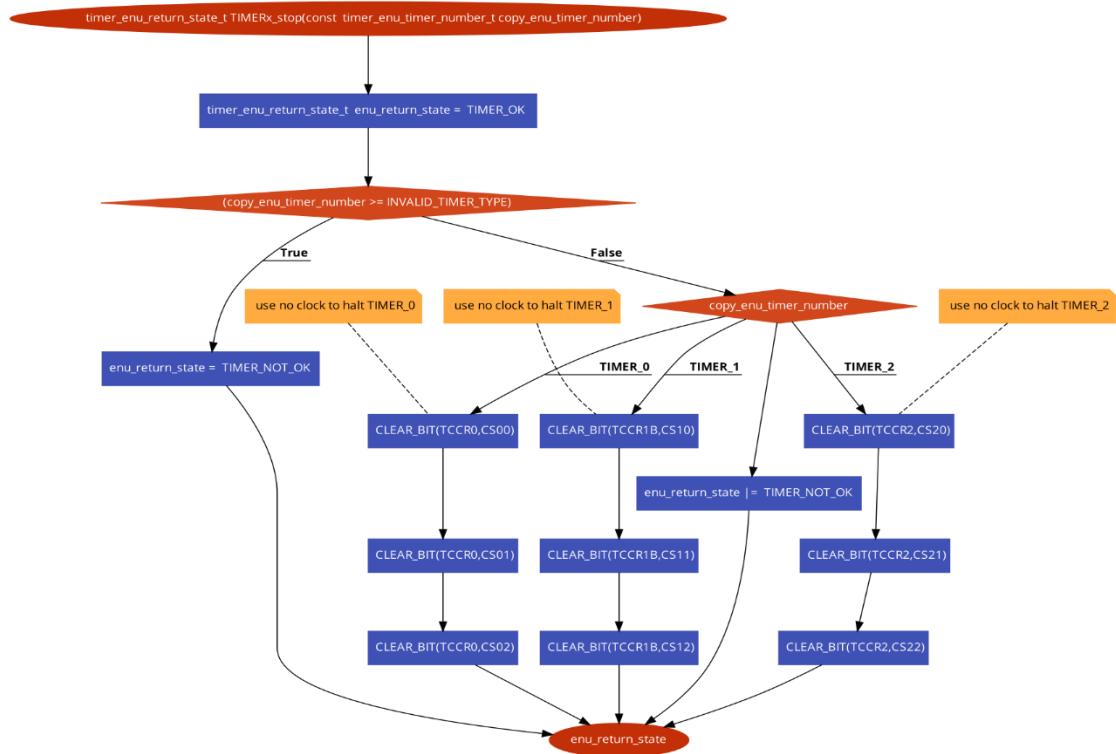


Figure 11 `TIMERx_stop()`

timer_enu_return_state_t TIMERx_setCallBack(ptr_to_v_fun_in_void_t ptr_v_fun_in_v, const timer_enu_timer_number_t copy_enu_timer_number)

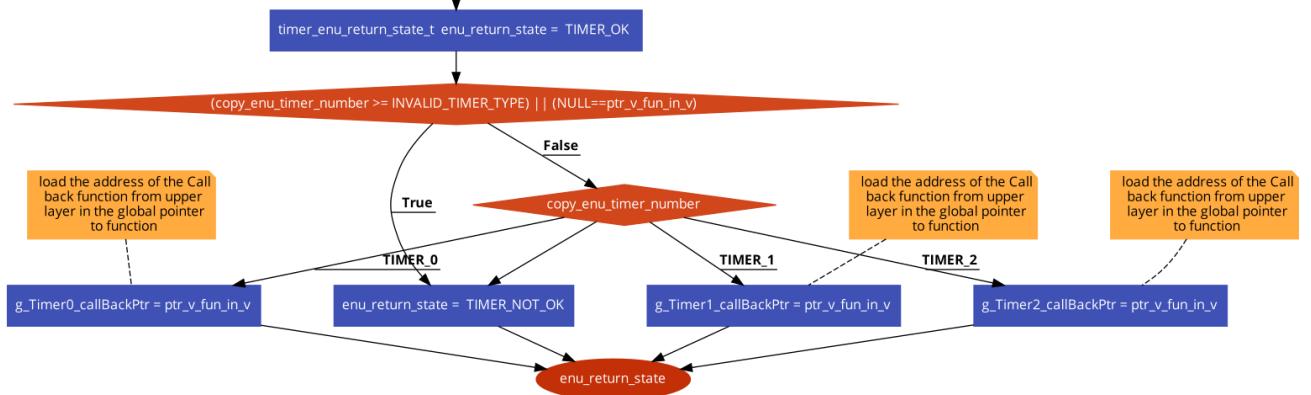


Figure 13 `TIMERx_setCallBack()`

timer_enu_return_state_t TIMERxSetValue(const timer_enu_timer_number_t copy_enu_timer_number, const uint16_t copy_u16_timer_init_value)

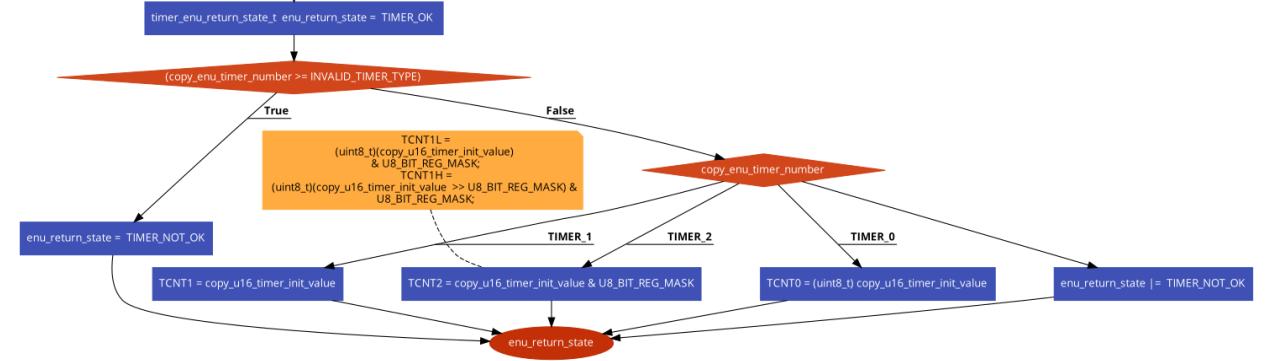


Figure 15 `TIMERx_SetValue`

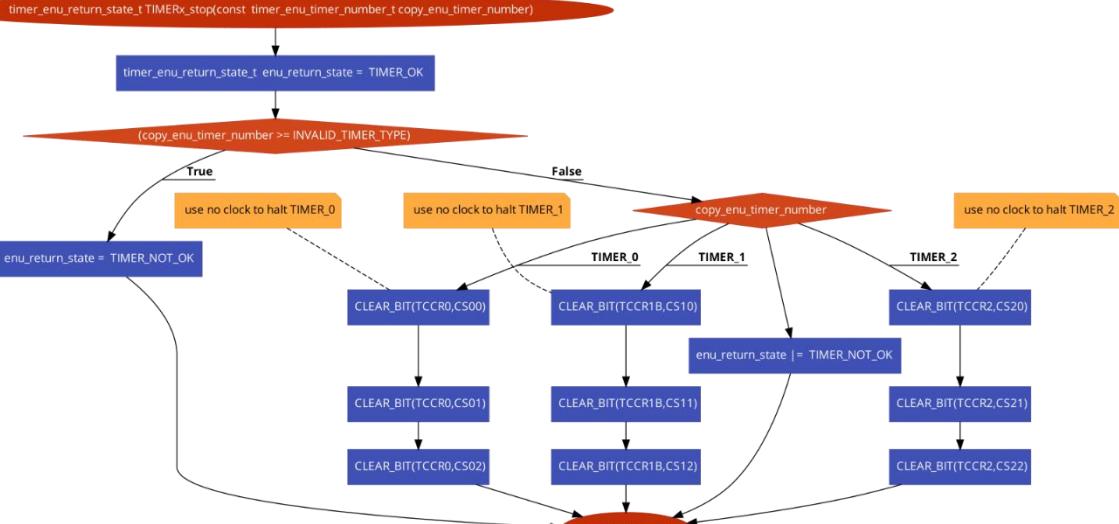


Figure 14 `TIMERx_SetValue()`

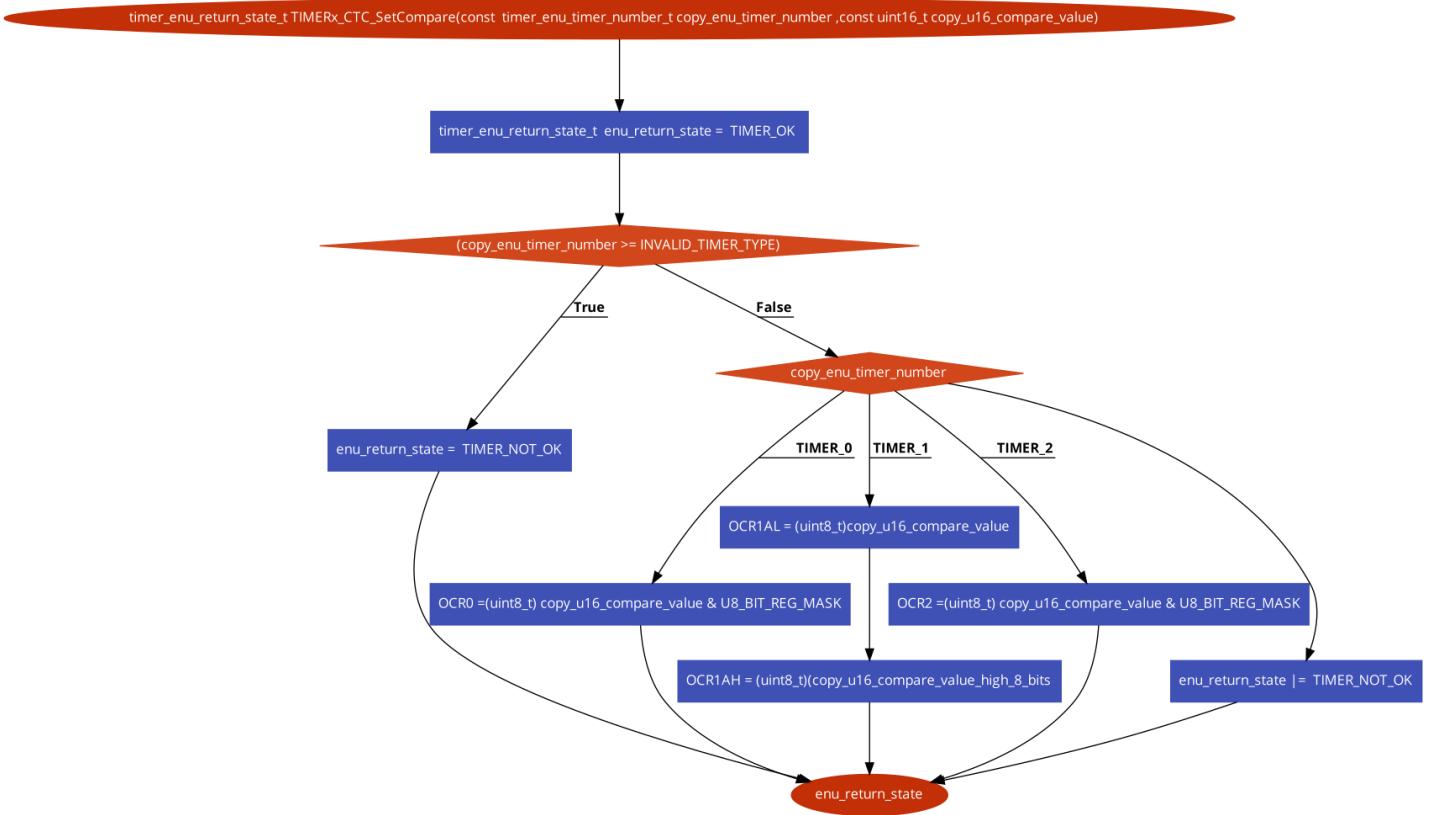


Figure 16 `TIMERx_CTC_SetCompare()`

HAL

BUZZER API:

Flowcharts:

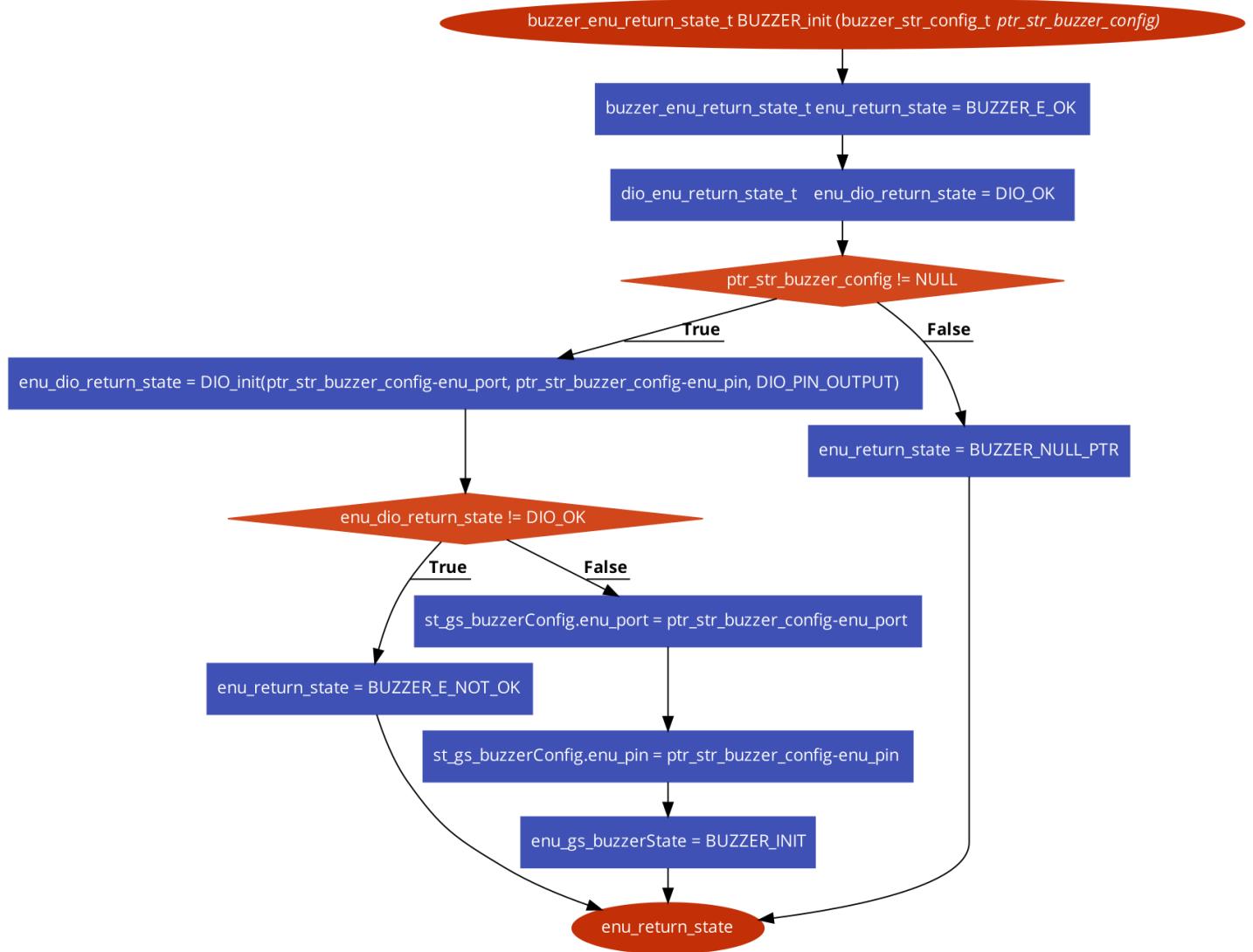


Figure 17 BUZZER_init()

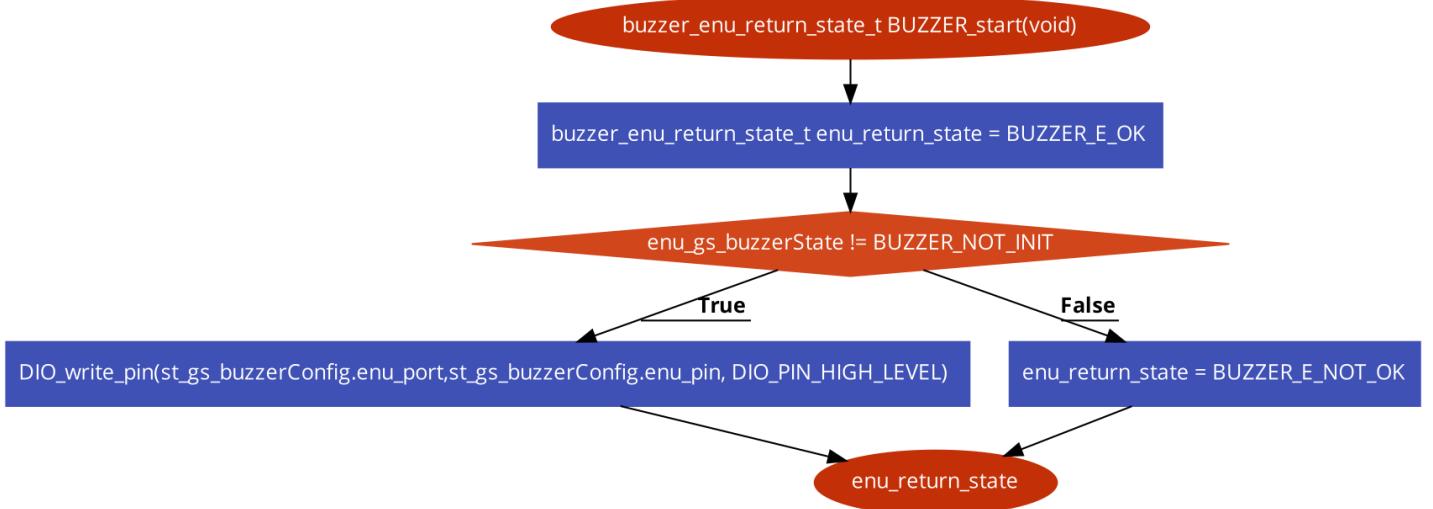


Figure 18 BUZZER_start()

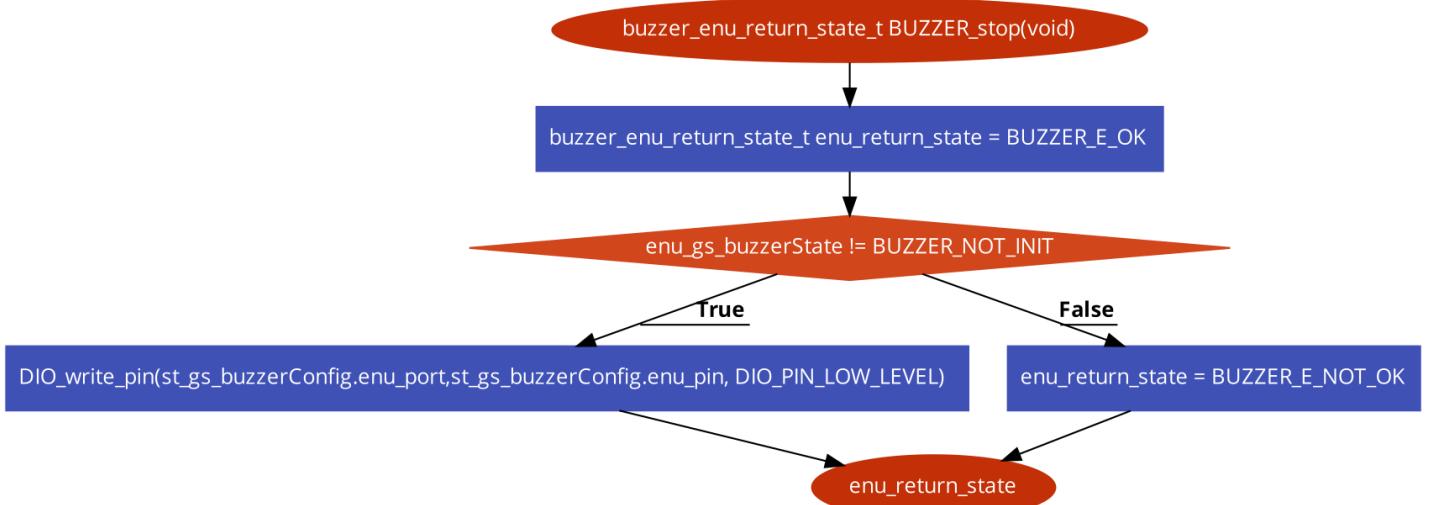


Figure 19 BUZZER_stop()

LCD API:

Flowcharts:

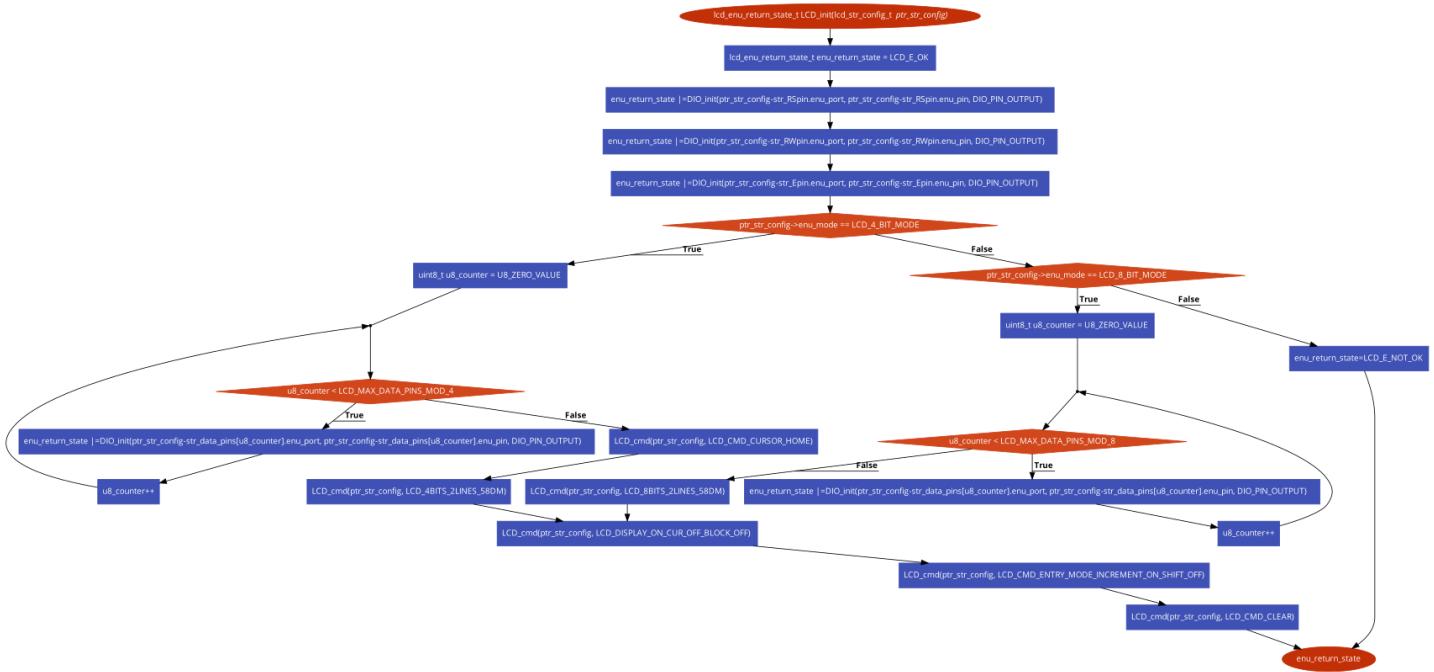


Figure 21 LCD_init()

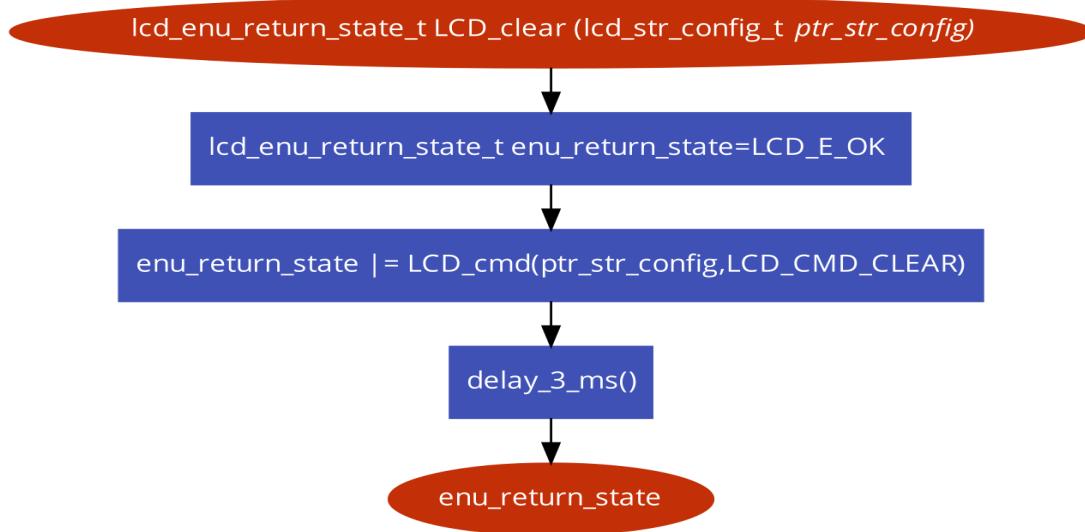


Figure 20 LCD_clear()

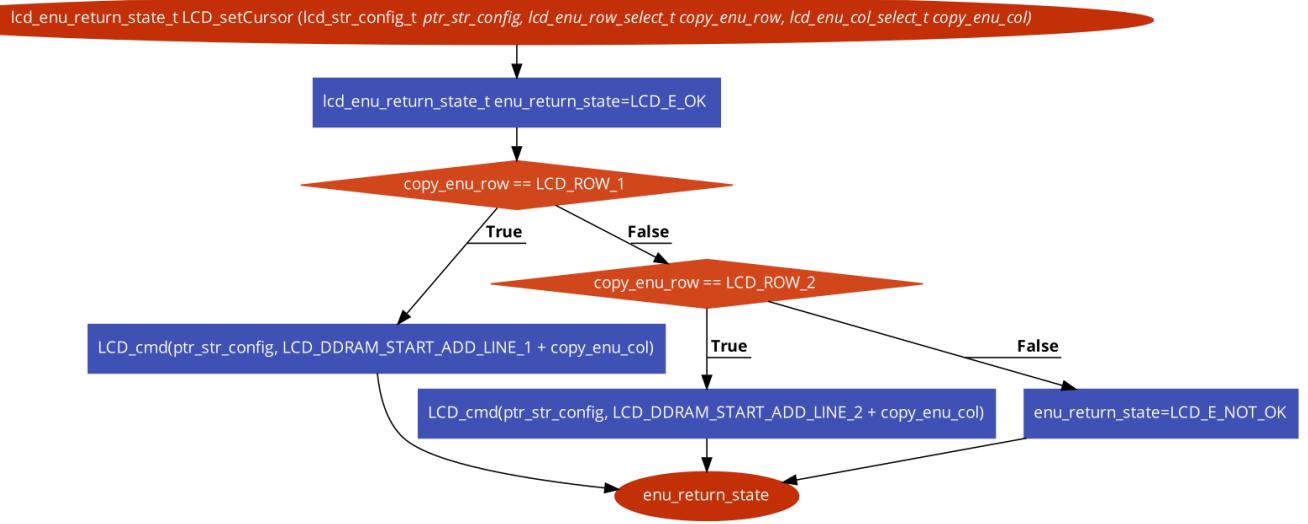


Figure 23 LCD_setCursor()

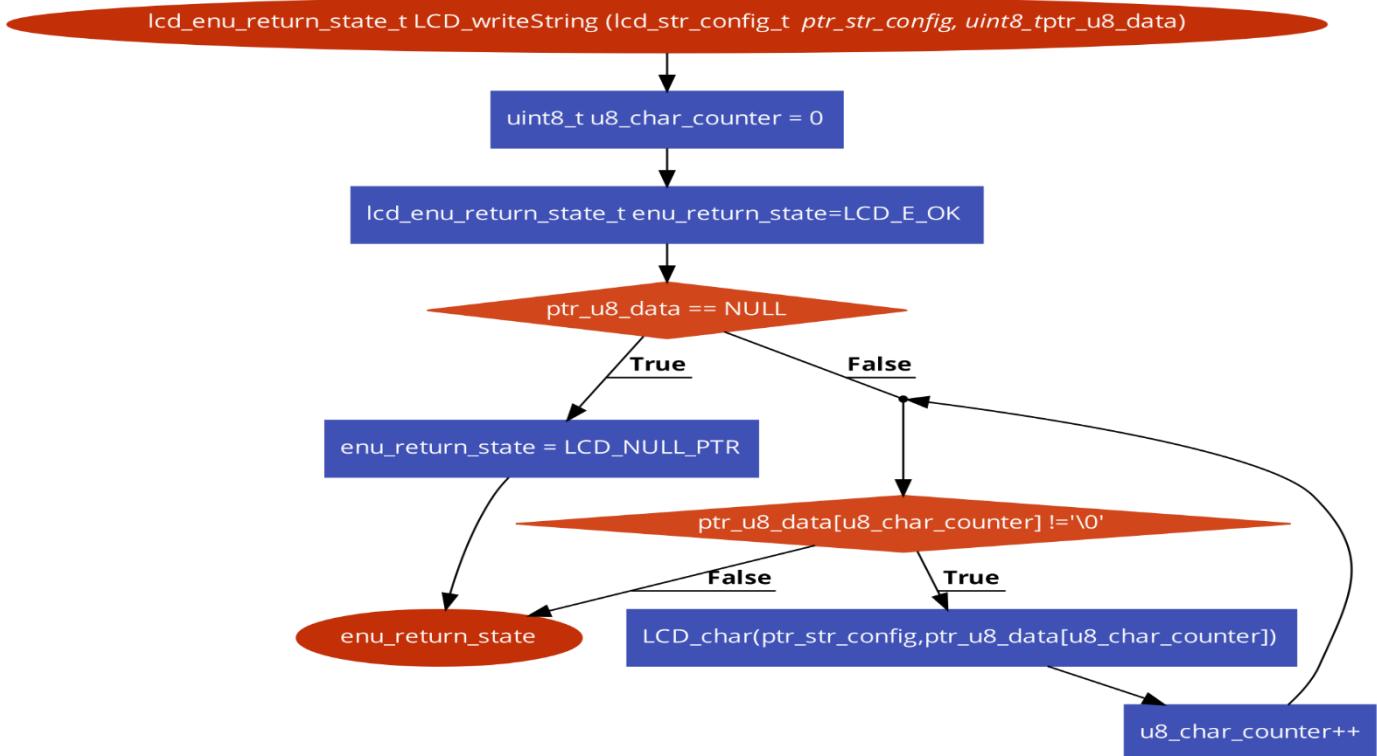


Figure 22 LCD_writeString()

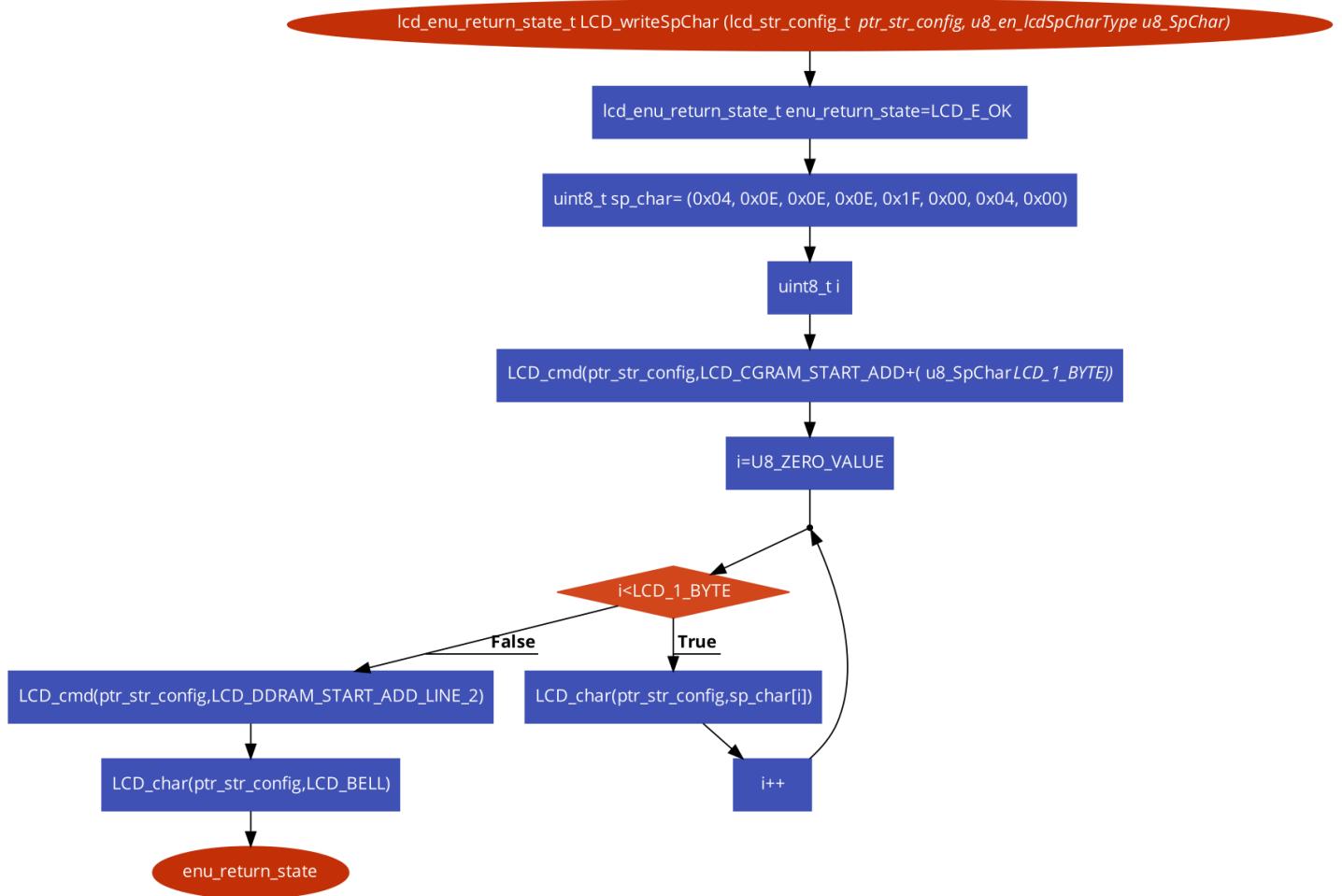


Figure 24 LCD_writeSpChar()

LM35 API :

Flowcharts:



Figure 25 LM35_INIT()

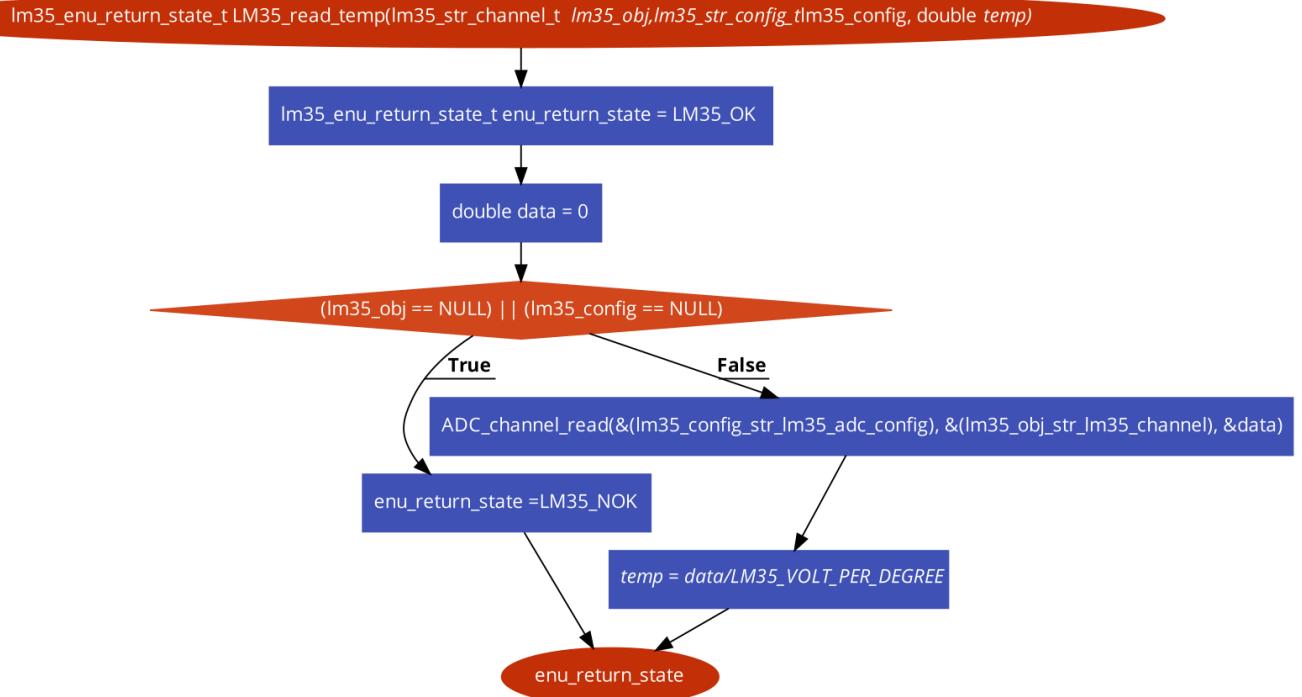


Figure 26 `LM35_read_temp()`

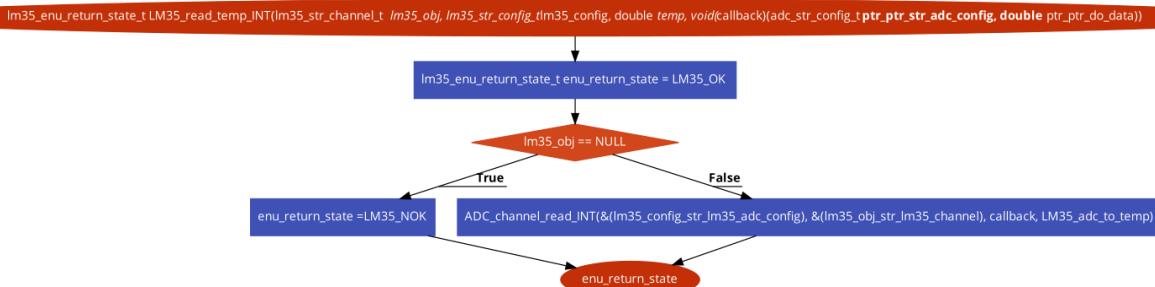


Figure 27 `LM35_read_temp_INT()`

Timer Manager API :

Flowcharts:

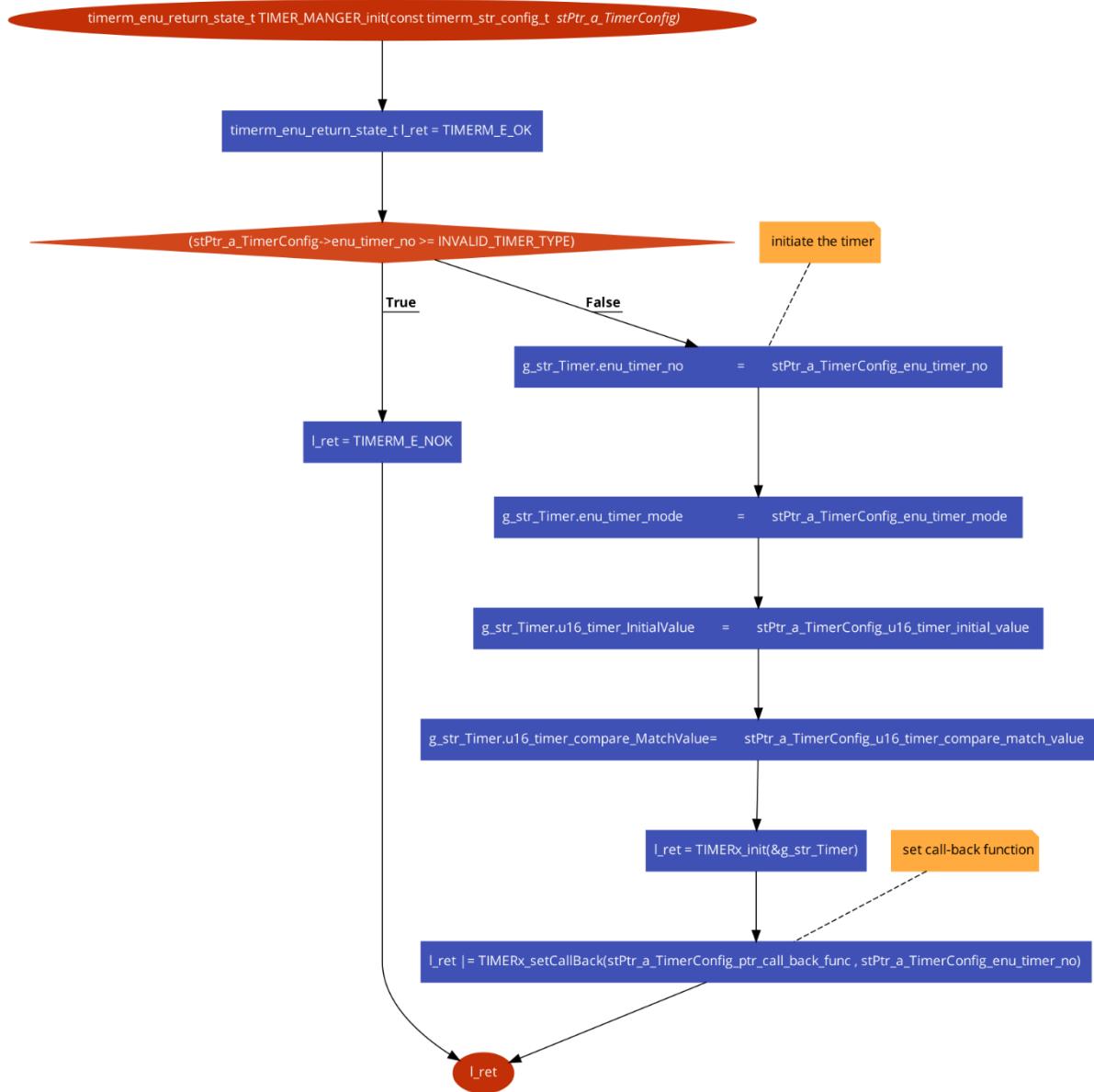


Figure 28 `TIMER_MANAGER_init()`

```
timerm_enu_return_state_t TIMER_MANGER_start(const timer_enu_clock_t copy_enu_timer_clock, const timer_enu_timer_number_t copy_enu_timer_num)
```

```
timerm_enu_return_state_t l_ret = TIMERM_E_OK
```

```
(copy_enu_timer_clock >= INVALID_TIMER_CLK) || (copy_enu_timer_num >= INVALID_TIMER_TYPE)
```

True

False

```
l_ret = TIMERM_E_NOK
```

```
l_ret = TIMERx_start(copy_enu_timer_clock, copy_enu_timer_num)
```

Configure the TIMER Prescaler value for Timer-x clock

Figure 29 TIMER_MANGER_start()

```
timerm_enu_return_state_t TIMER_MANGER_stop(const timer_enu_timer_number_t copy_enu_timer_num)
```

```
timerm_enu_return_state_t l_ret = TIMERM_E_OK
```

```
copy_enu_timer_num >= INVALID_TIMER_TYPE
```

True

False

```
l_ret = TIMERM_E_NOK
```

```
l_ret = TIMERx_stop(copy_enu_timer_num)
```

stop the clock for the specific timer

Figure 31 TIMER_MANGER_stop()

```
timerm_enu_return_state_t TIMER_MANGERSetValue(const timer_enu_timer_number_t copy_enu_timer_num, uint16_t u16_a_InitialValue)
```

```
timerm_enu_return_state_t l_ret = TIMERM_E_OK
```

```
copy_enu_timer_num >= INVALID_TIMER_TYPE
```

True

False

```
l_ret = TIMERM_E_NOK
```

```
l_ret = TIMERx_SetValue(copy_enu_timer_num, u16_a_InitialValue)
```

stop the clock for the specific timer

Figure 30 TIMER_MANGER_SetValue()

KEYPAD API:

Flowcharts:

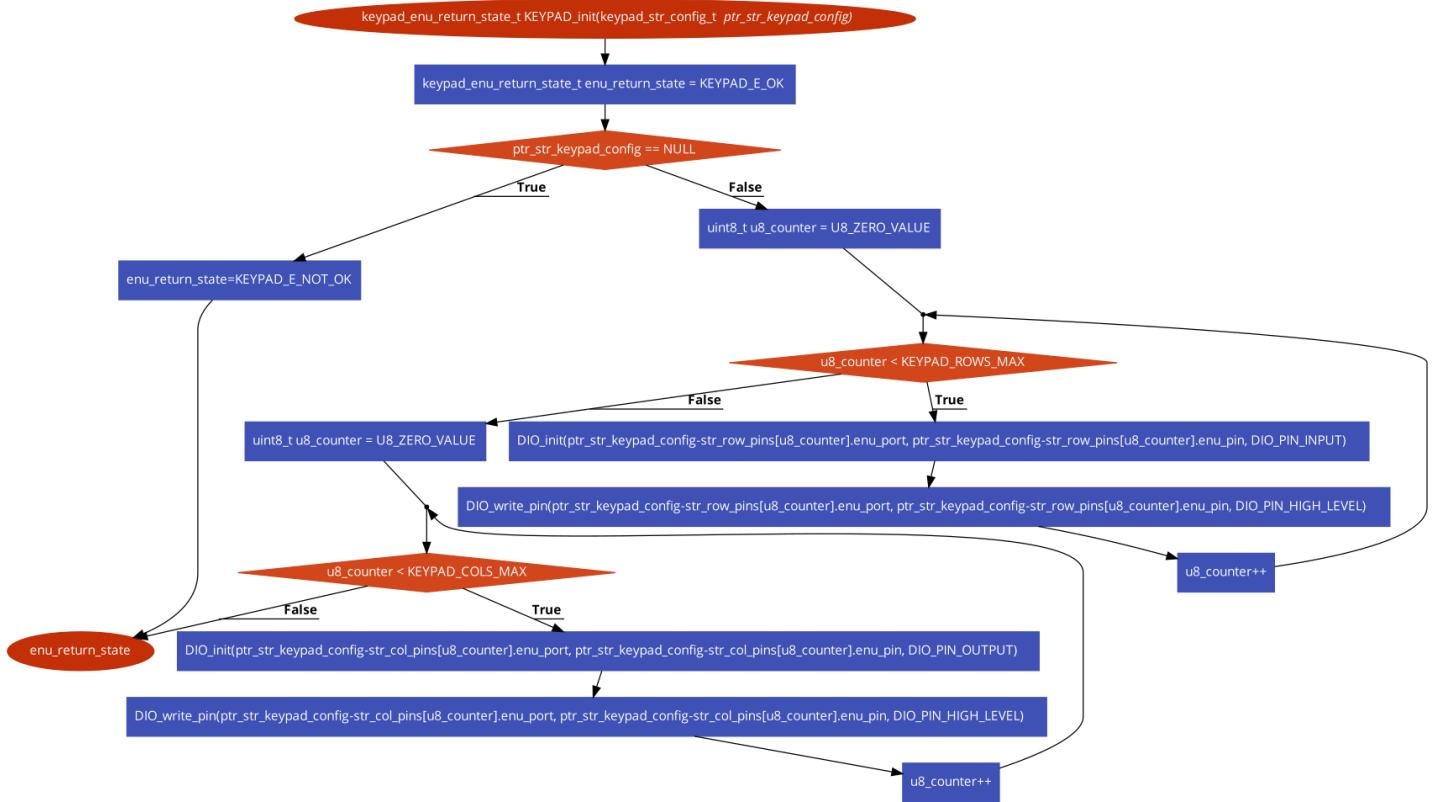


Figure 32 KEYPAD_init()*t*

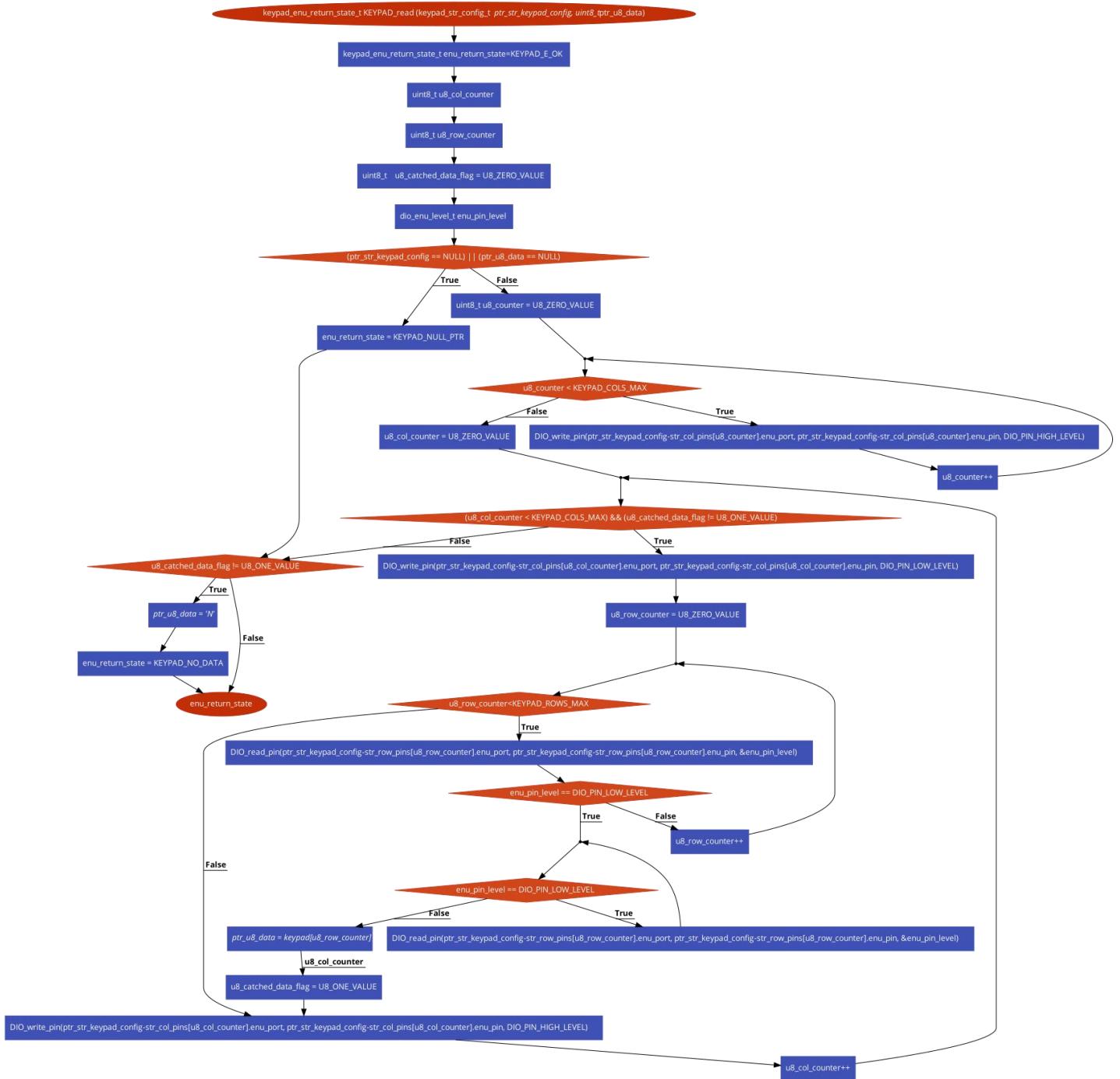


Figure 33 KEYPAD_read()

App

App API:

Flowcharts:

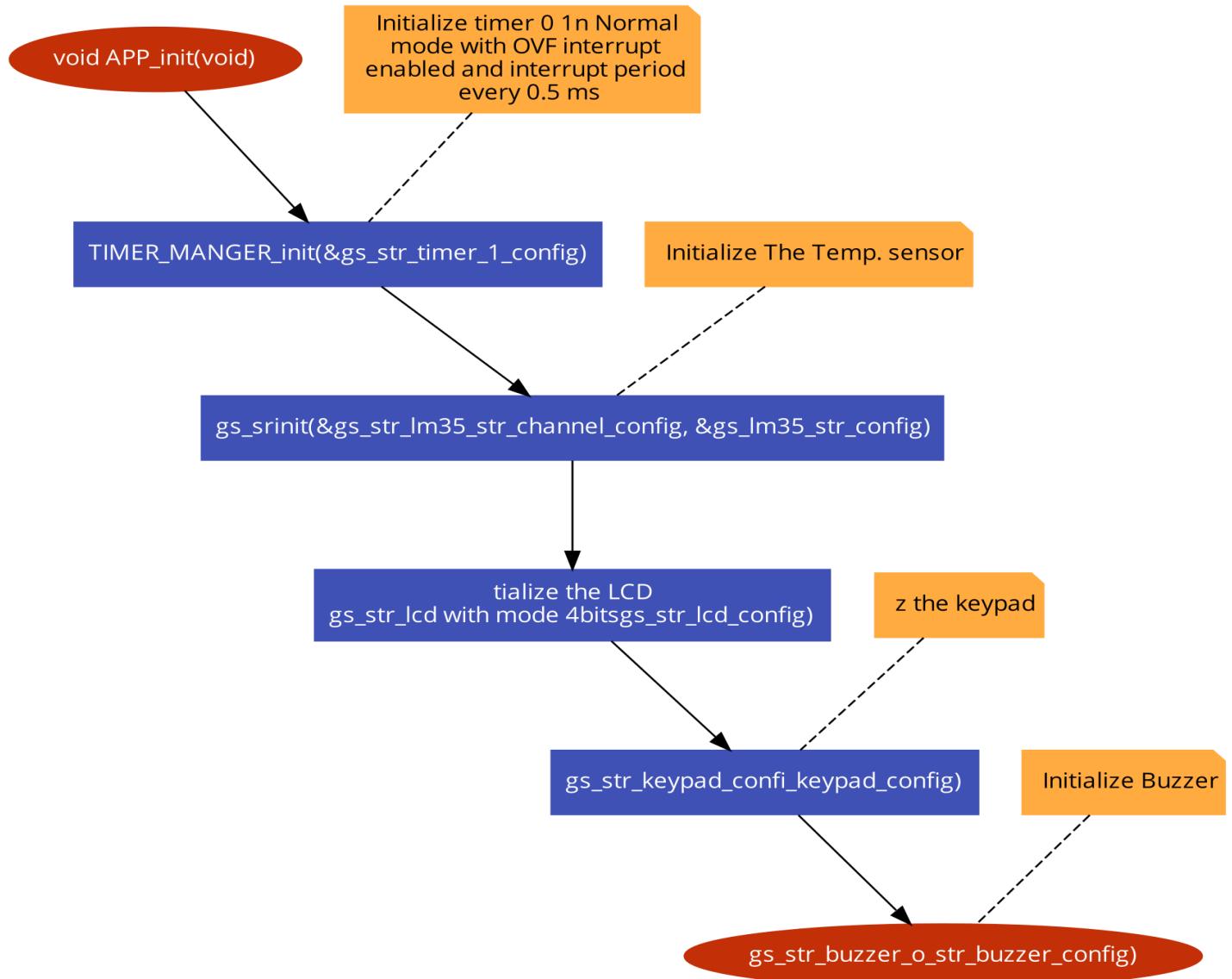


Figure 34 APP_init()

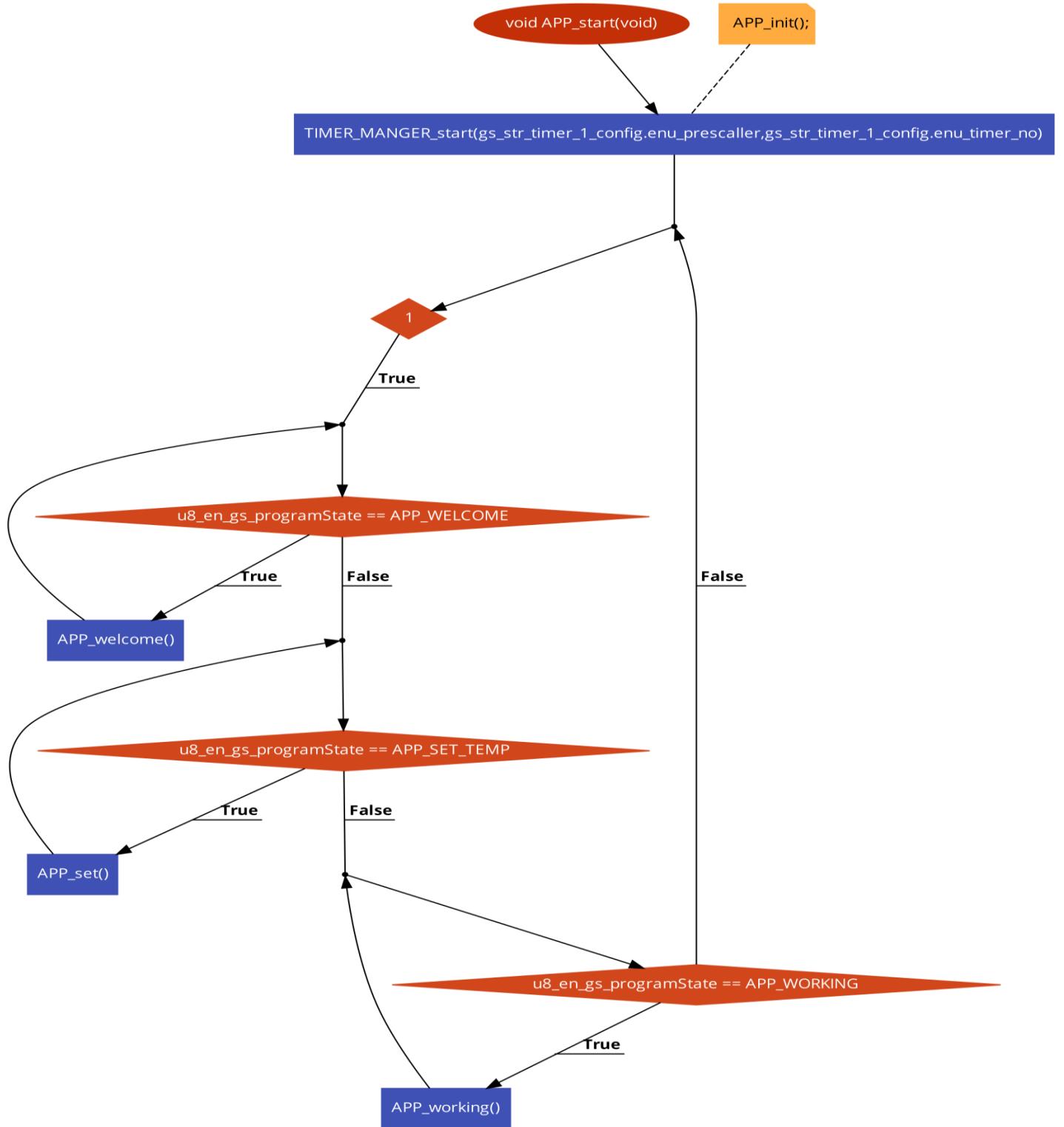


Figure 35 APP_start()

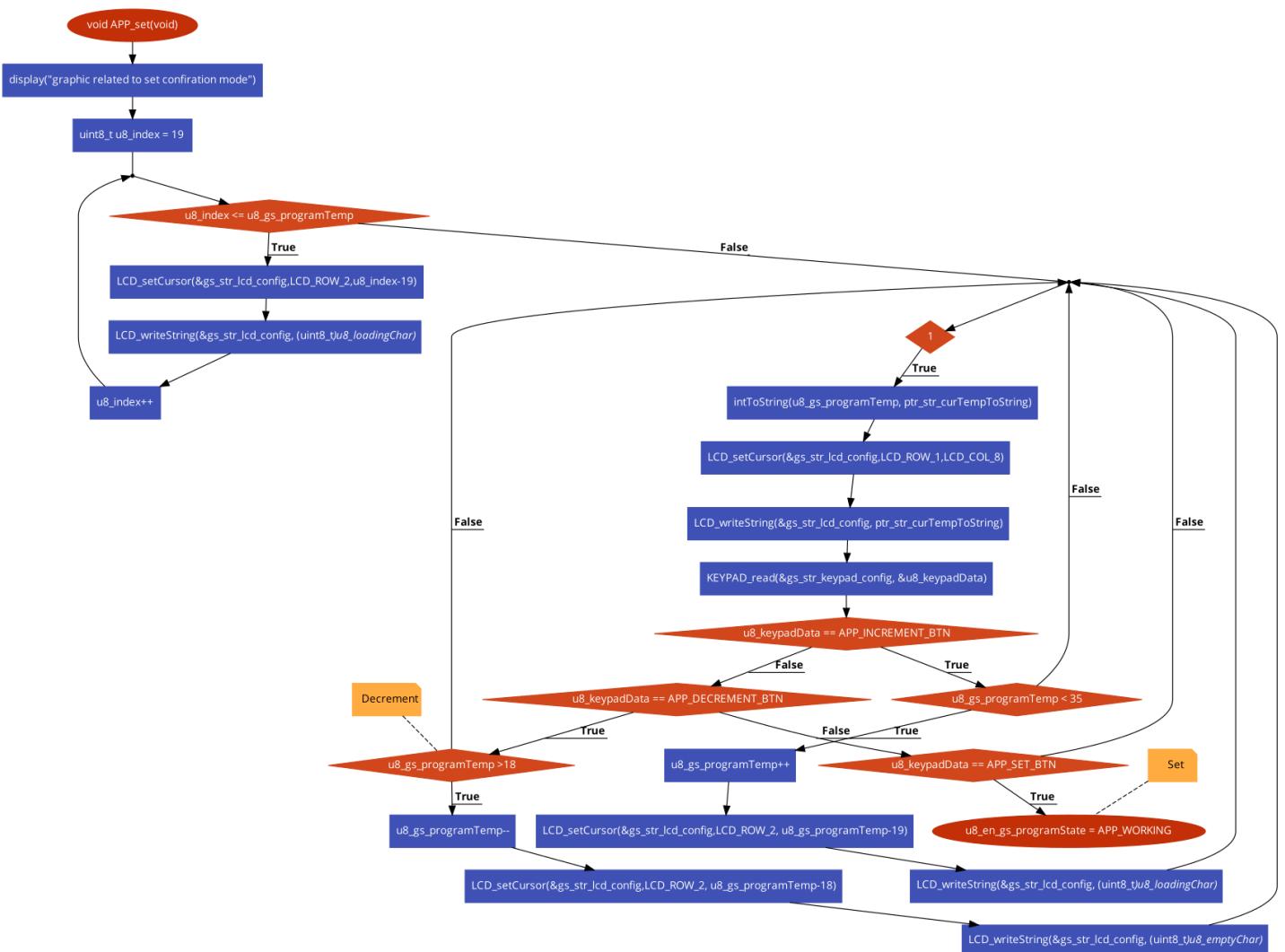


Figure 36 APP_set()

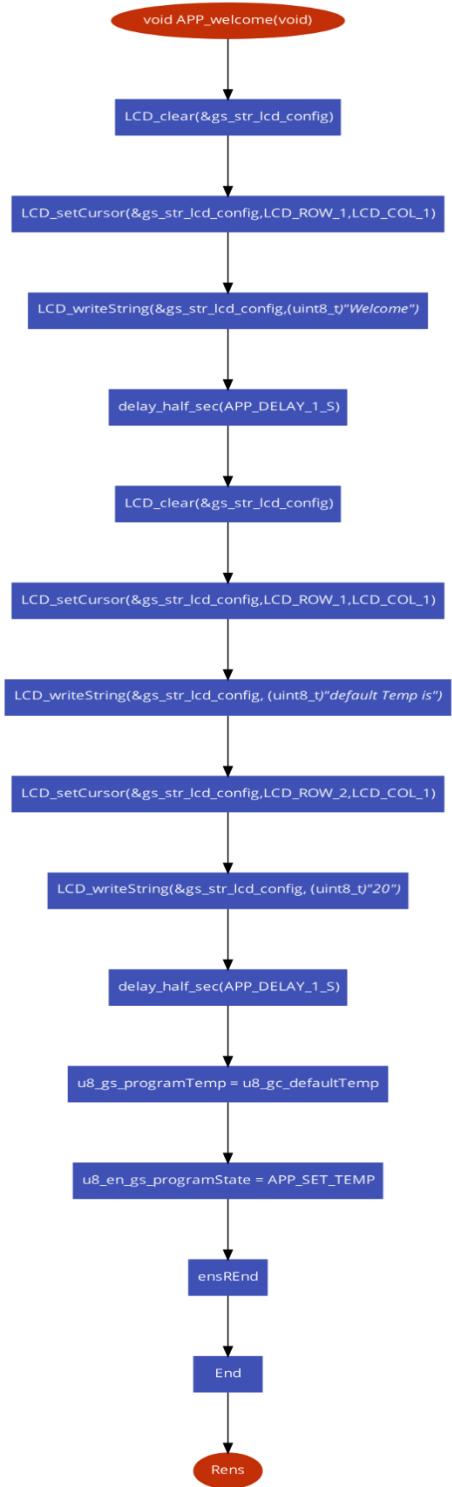


Figure 37 APP_welcome()

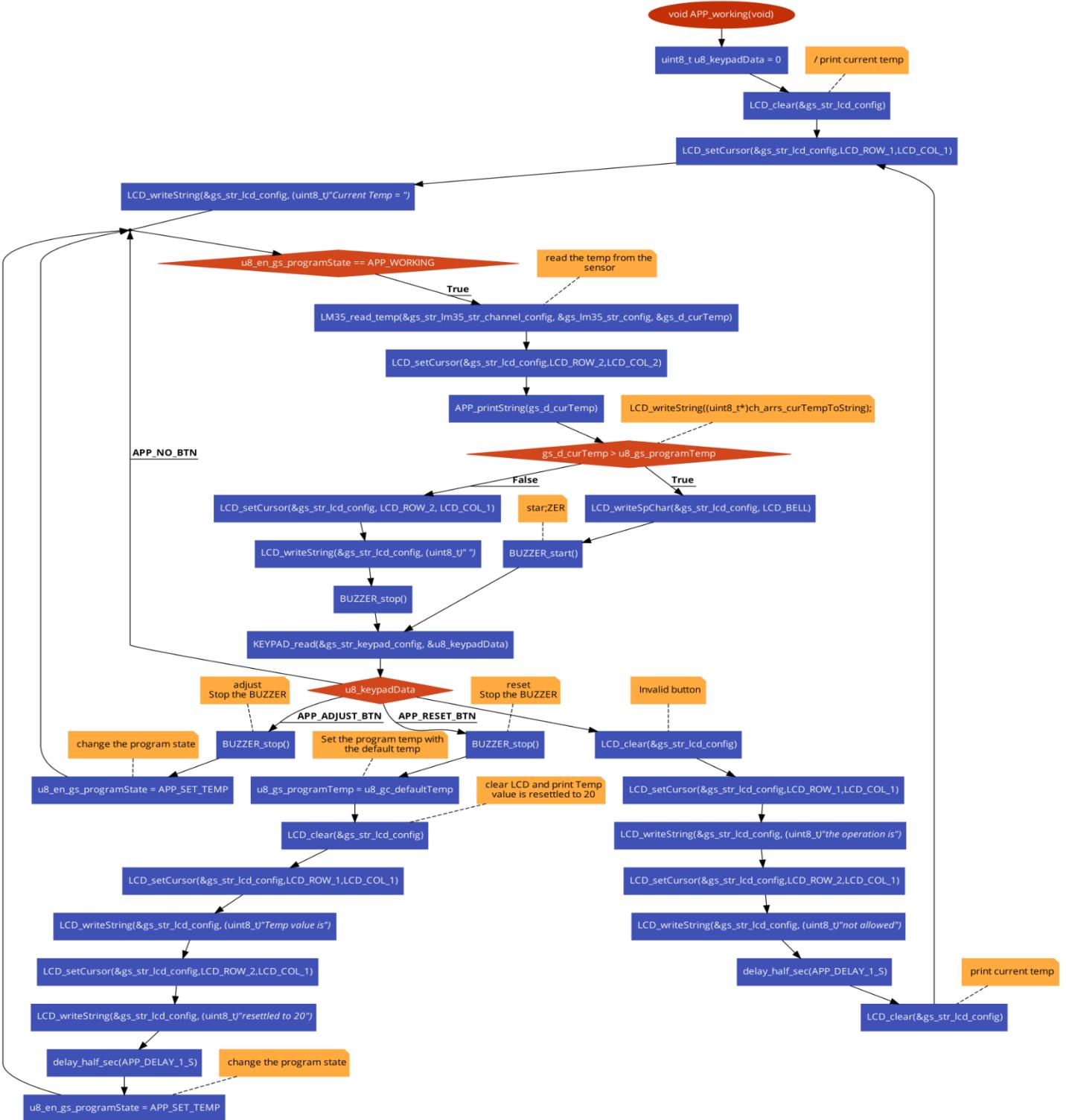


Figure 38 APP_working()

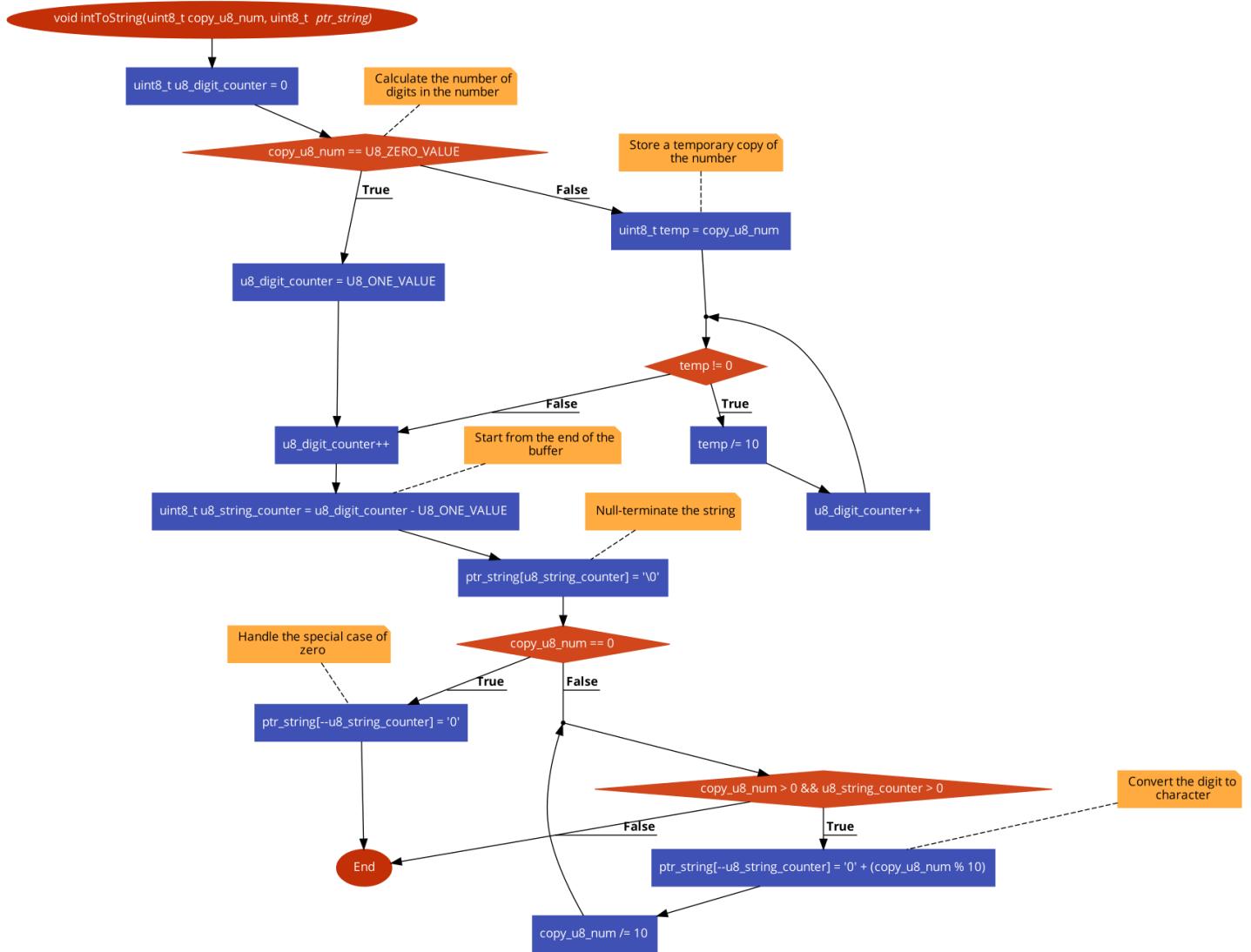


Figure 39 `intToString()`