

# Moving Car Capstone Project

By Team 2

- Mohab Ahmed
- Anas Mahmoud
- Mustafa Mohammed
- Omar Taha

# Table of Contents

- Project Introduction
- Project flowchart
- Layered Architecture
- Modules/Drivers
- APIs
- APIs flowcharts

# Introduction

## Description :

### 1. Car Components:

1. Four motors (M1, M2, M3, M4)
2. One button to start (PB1)
3. One button for stop (PB2)
4. Four LEDs (LED1, LED2, LED3, LED4)

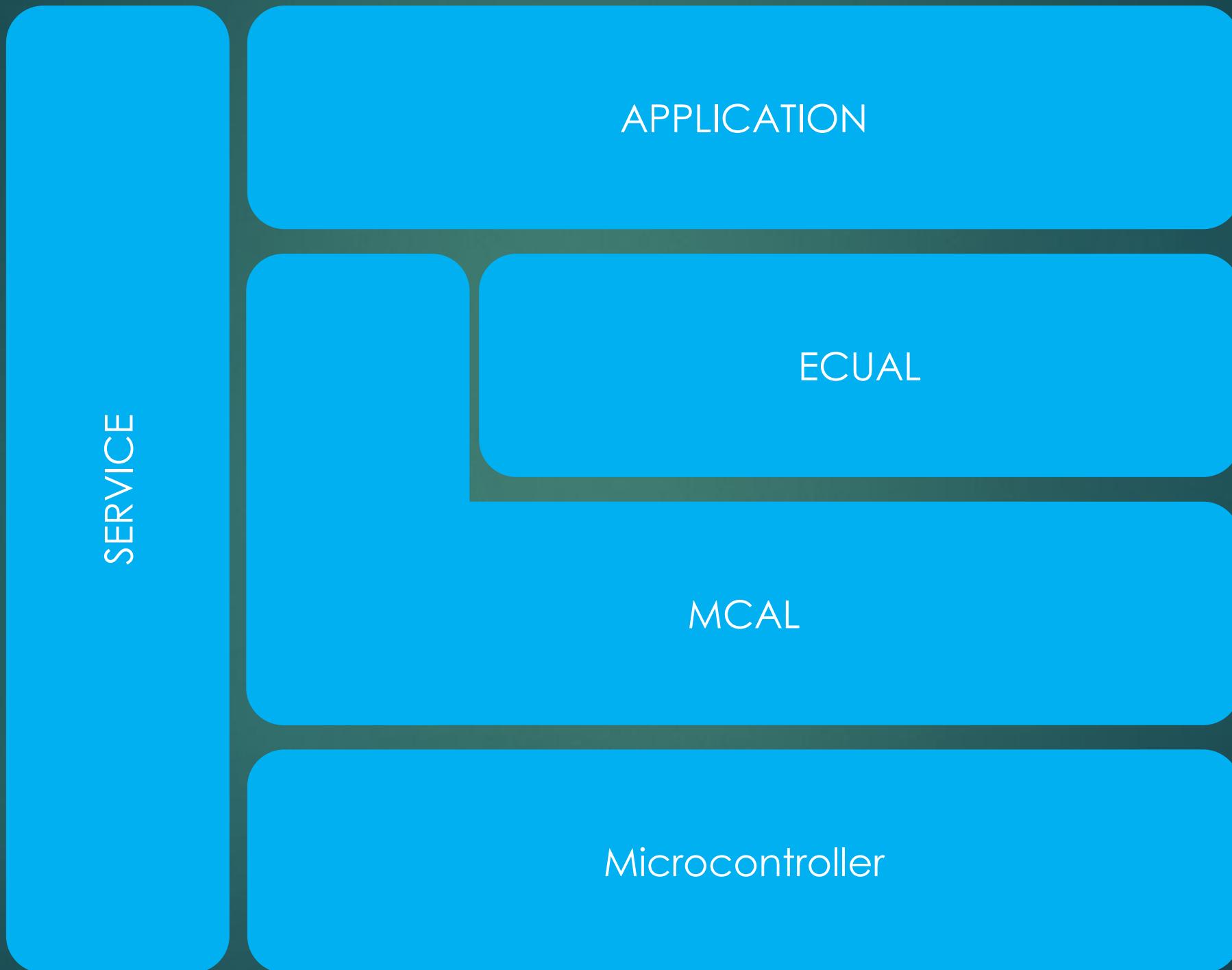
### 2. System Requirements:

1. The car starts initially from 0 speed
2. When PB1 is pressed, the car will move forward after 1 second
3. The car will move forward to create the longest side of the rectangle for 3 seconds with 50% of its maximum speed
4. After finishing the first longest side the car will stop for 0.5 seconds, rotate 90 degrees to the right, and stop for 0.5 second
5. The car will move to create the short side of the rectangle at 30% of its speed for 2 seconds
6. After finishing the shortest side the car will stop for 0.5 seconds, rotate 90 degrees to the right, and stop for 0.5 second
7. Steps 3 to 6 will be repeated infinitely until you press the stop button (PB2)
8. PB2 acts as a sudden break, and it has the highest priority

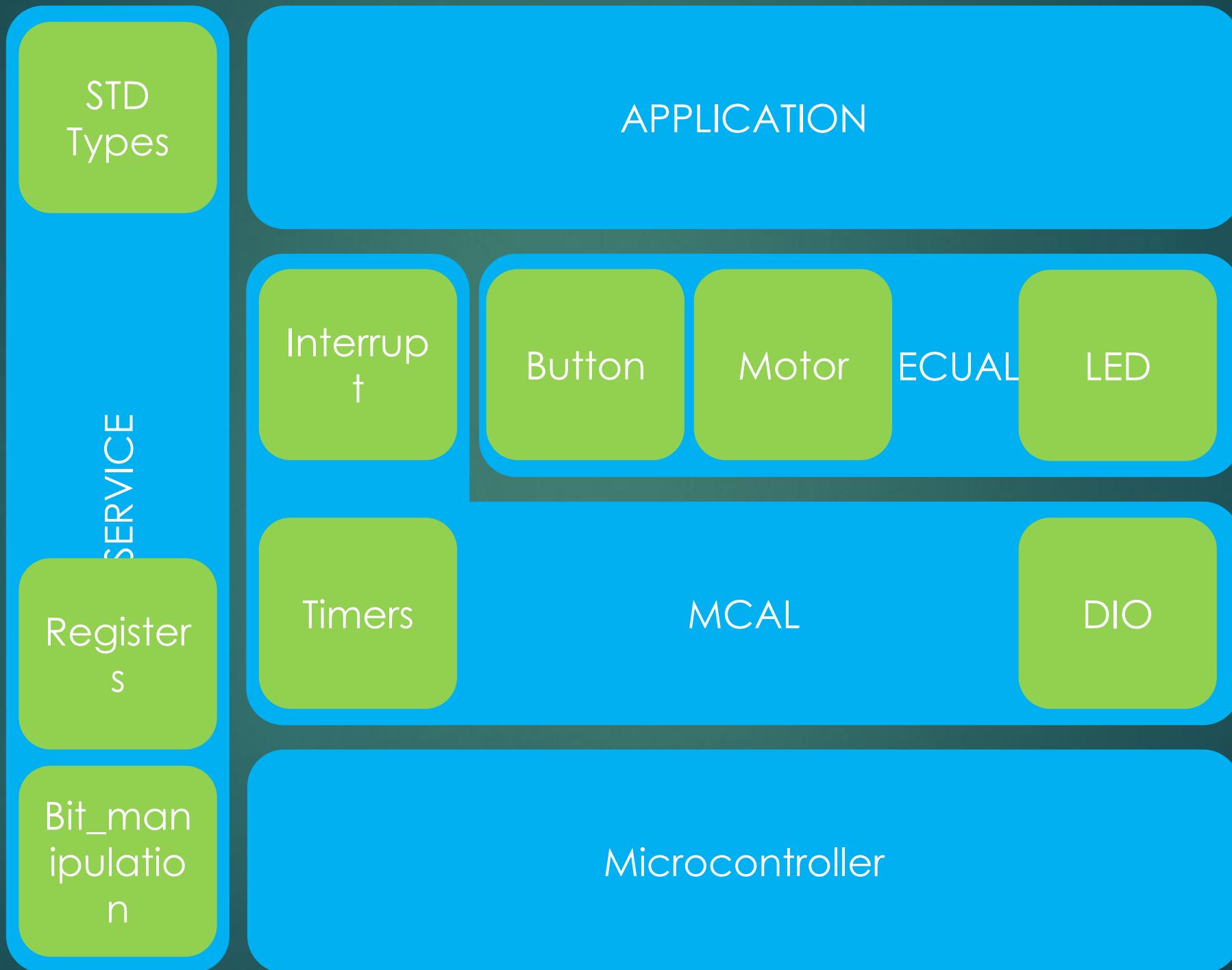
# Project flowchart



# Layered Architecture:



# Modules/Drivers:



# APIs:

## **Motor Driver:**

- ▶ err\_state MOTOR\_init(uint8\_t u8\_a\_pinNumber, uint8\_t u8\_a\_portNumber);
- ▶ err\_state MOTOR\_on(uint8\_t u8\_a\_mask, uint8\_t u8\_a\_portNumber);
- ▶ err\_state MOTOR\_off(uint8\_t u8\_a\_mask, uint8\_t u8\_a\_portNumber);
- ▶ err\_state MOTOR\_control(uint8\_t u8\_a\_mask, uint8\_t u8\_a\_portNumber, float f\_a\_speedPercentage);

## **Button Driver:**

- ▶ err\_state BUTTON\_init(uint8\_t u8\_a\_pinNumber, uint8\_t u8\_a\_portNumber);
- ▶ err\_state BUTTON\_read(uint8\_t u8\_a\_pinNumber, uint8\_t u8\_a\_portNumber, pin\_state \*en\_a\_value);

## **LED Driver:**

- ▶ err\_state LED\_init(uint8\_t u8\_a\_ledPin, uint8\_t u8\_a\_ledPort);
- ▶ err\_state LED\_on(uint8\_t u8\_a\_ledPin, uint8\_t u8\_a\_ledPort);
- ▶ err\_state LED\_off(uint8\_t u8\_a\_ledPin, uint8\_t u8\_a\_ledPort);
- ▶ err\_state LED\_toggle(uint8\_t u8\_a\_ledPin, uint8\_t u8\_a\_ledPort);
- ▶ err\_state LED\_blink(uint8\_t u8\_a\_ledPin, uint8\_t u8\_a\_ledPort, float f\_a\_onTime, float f\_a\_offTime);
- ▶ err\_state LED\_array\_blink(uint8\_t u8\_a\_mask, uint8\_t u8\_a\_ledPort, float f\_a\_onTime, float f\_a\_offTime);
- ▶ err\_state LED\_array\_on(uint8\_t u8\_a\_mask, uint8\_t u8\_a\_ledPort);
- ▶ err\_state LED\_array\_off(uint8\_t u8\_a\_mask, uint8\_t u8\_a\_ledPort);

## **DIO Driver:**

- ▶ err\_state DIO\_init(uint8\_t u8\_l\_pinNumber, uint8\_t u8\_l\_portNumber, pin\_dir en\_l\_direction);
- ▶ err\_state DIO\_write(uint8\_t u8\_l\_pinNumber, uint8\_t u8\_l\_portNumber, pin\_state en\_l\_value);
- ▶ err\_state DIO\_toggle(uint8\_t u8\_l\_pinNumber, uint8\_t u8\_l\_portNumber);
- ▶ err\_state DIO\_read(uint8\_t u8\_l\_pinNumber, uint8\_t u8\_l\_portNumber, pin\_state \*en\_l\_value);  
err\_state DIO\_array\_write(uint8\_t mask, uint8\_t portNumber, pin\_state value);
- ▶ err\_state DIO\_array\_write(uint8\_t u8\_l\_mask, uint8\_t u8\_l\_portNumber, pin\_state en\_l\_value);

# APIs:

## Timers Driver:

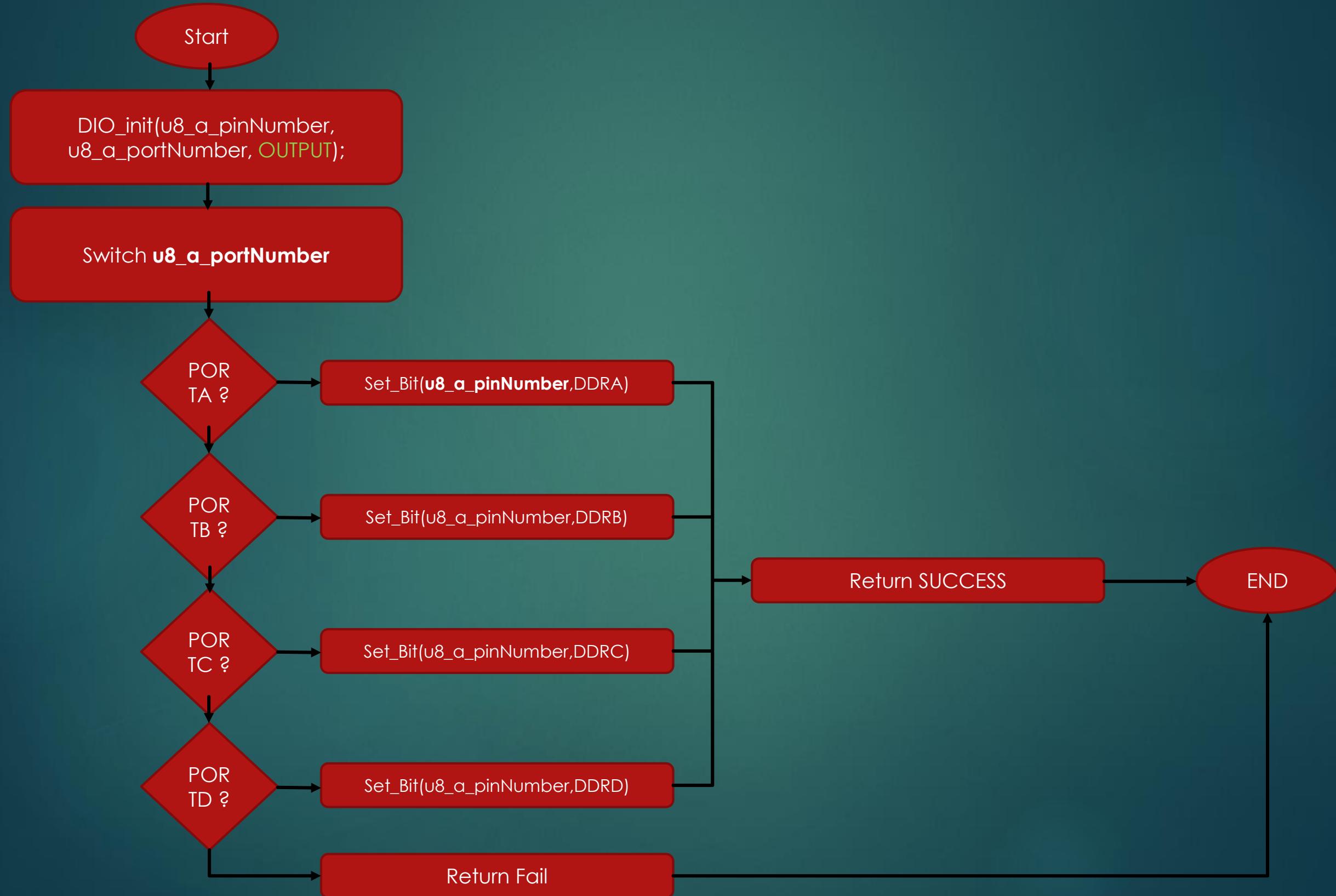
- ▶ err\_state TIMER0\_normalMode(void);
- ▶ err\_state TIMER0\_initialValue(uint8\_t value);
- ▶ err\_state TIMER0\_prescalerMode(unsigned int u16\_a\_prescaler)
- ▶ err\_state TIMER0\_delay(float f\_a\_delayInMillis);
- ▶ unsigned int TIMER0\_getInitialValue(float f\_a\_delayInMillis);
- ▶ err\_state TIMER2\_normalMode(void);
- ▶ err\_state TIMER2\_initialValue(uint8\_t value);
- ▶ err\_state TIMER2\_prescalerMode(unsigned int prescaler);
- ▶ err\_state TIMER2\_delay(float f\_a\_delayInMillis);
- ▶ unsigned int TIMER2\_getInitialValue(float f\_a\_delayInMillis);

## Interrupts Driver:

- ▶ err\_state BUTTON\_init(uint8\_t pinNumber, uint8\_t portNumber);
- ▶ err\_state BUTTON\_read(uint8\_t pinNumber, uint8\_t portNumber, uint8\_t \*value);

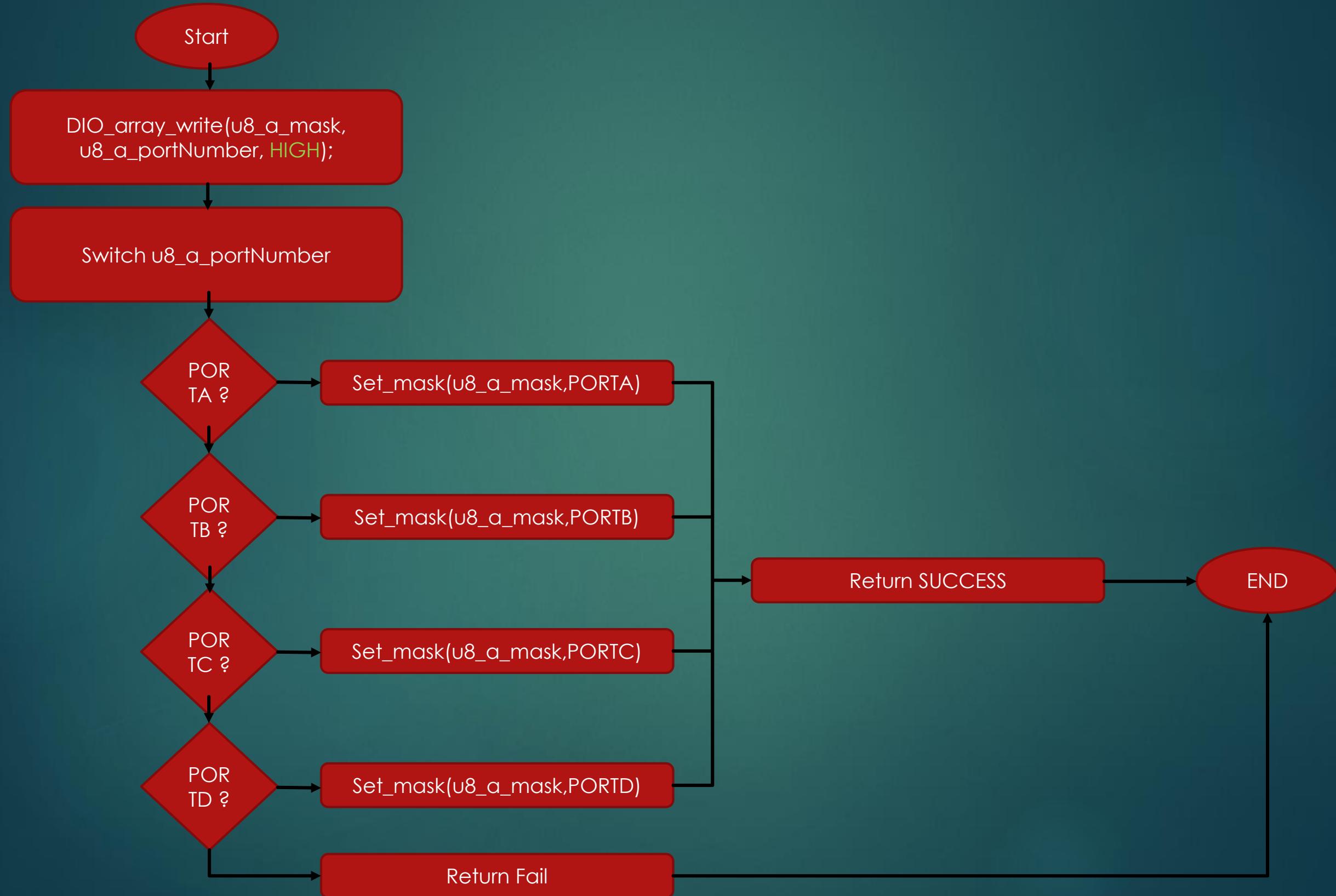
# APIs flowchart:

```
err_state MOTOR_init(uint8_t u8_a_pinNumber, uint8_t u8_a_portNumber);
```



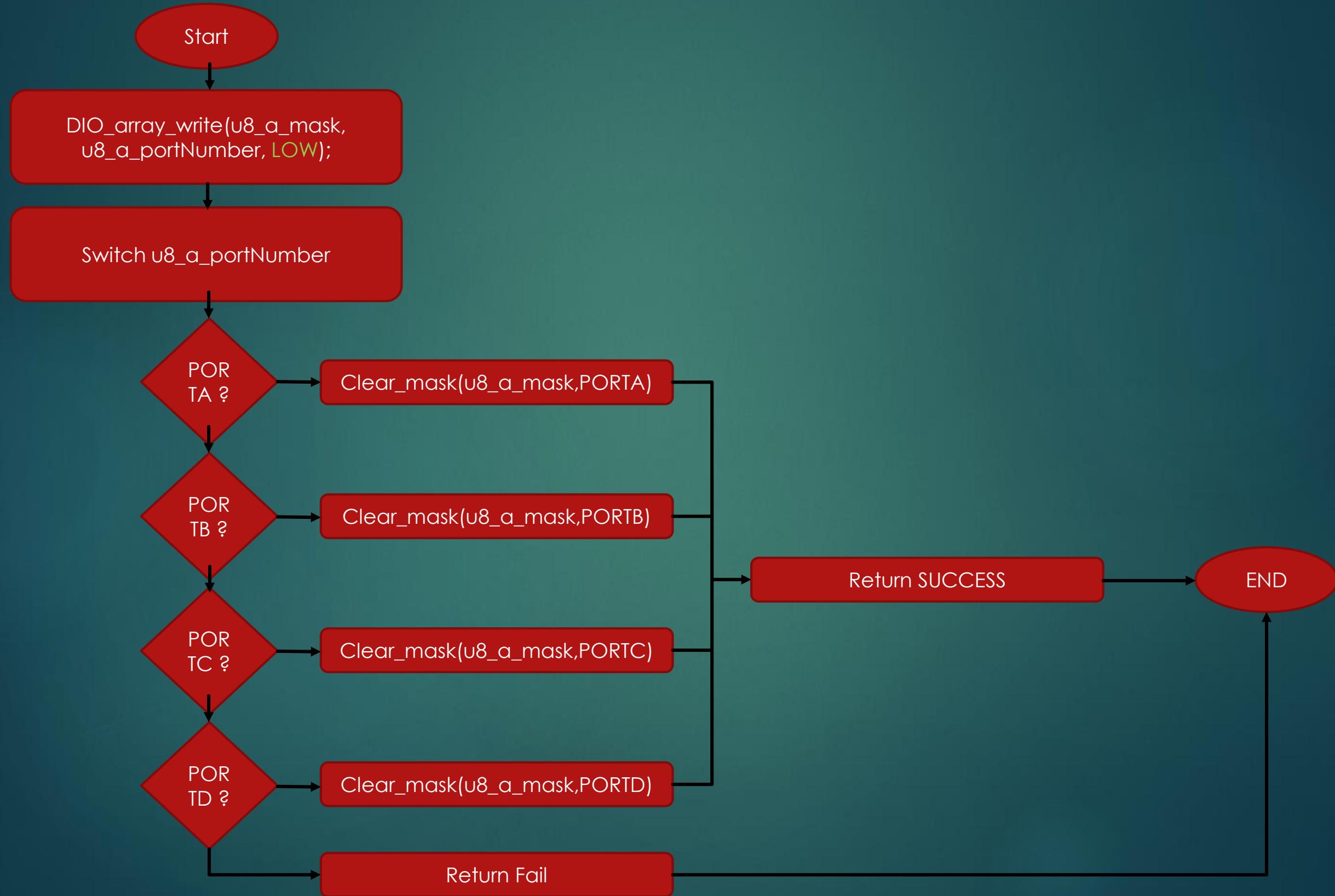
# APIs flowchart:

```
err_state MOTOR_on(uint8_t u8_a_mask, uint8_t u8_a_portNumber);
```



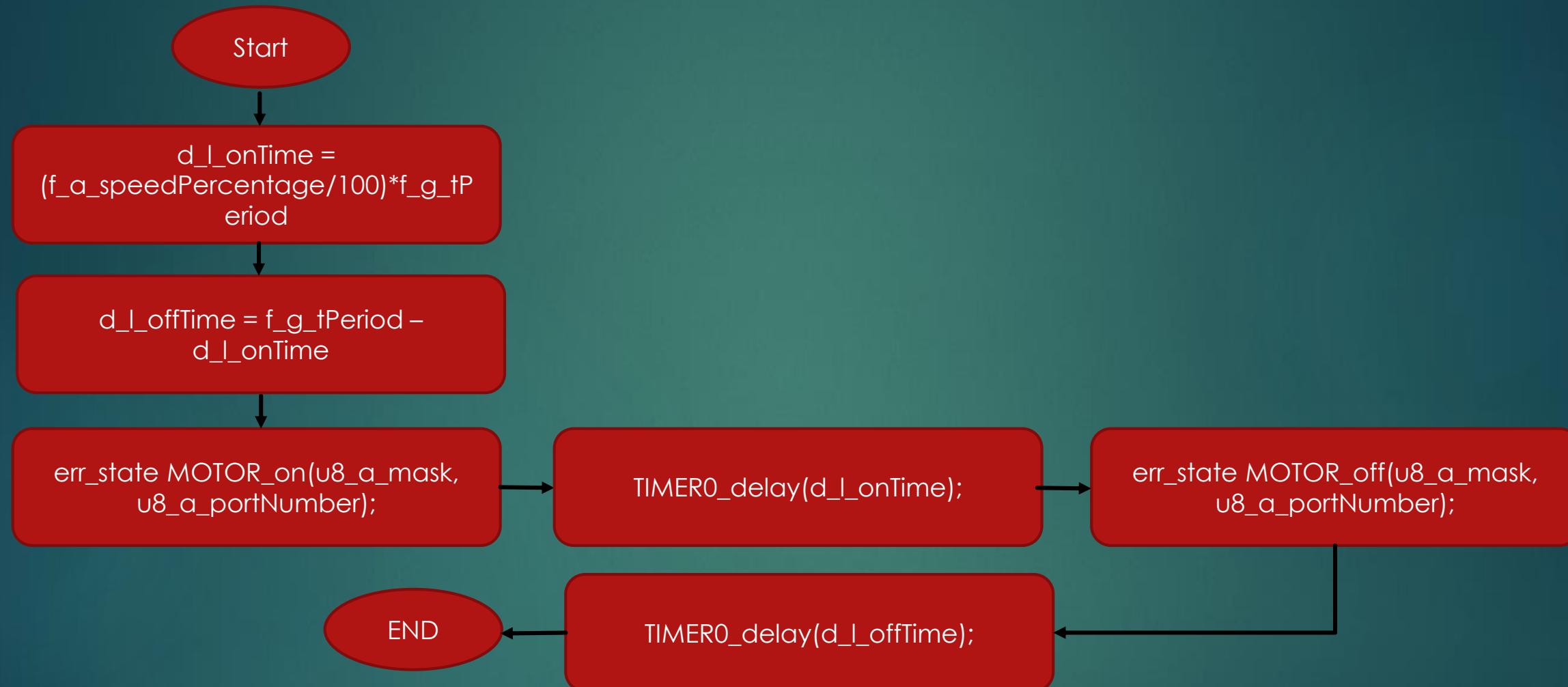
# APIs flowchart:

```
err_state MOTOR_off(uint8_t u8_a_mask, uint8_t u8_a_portNumber);
```



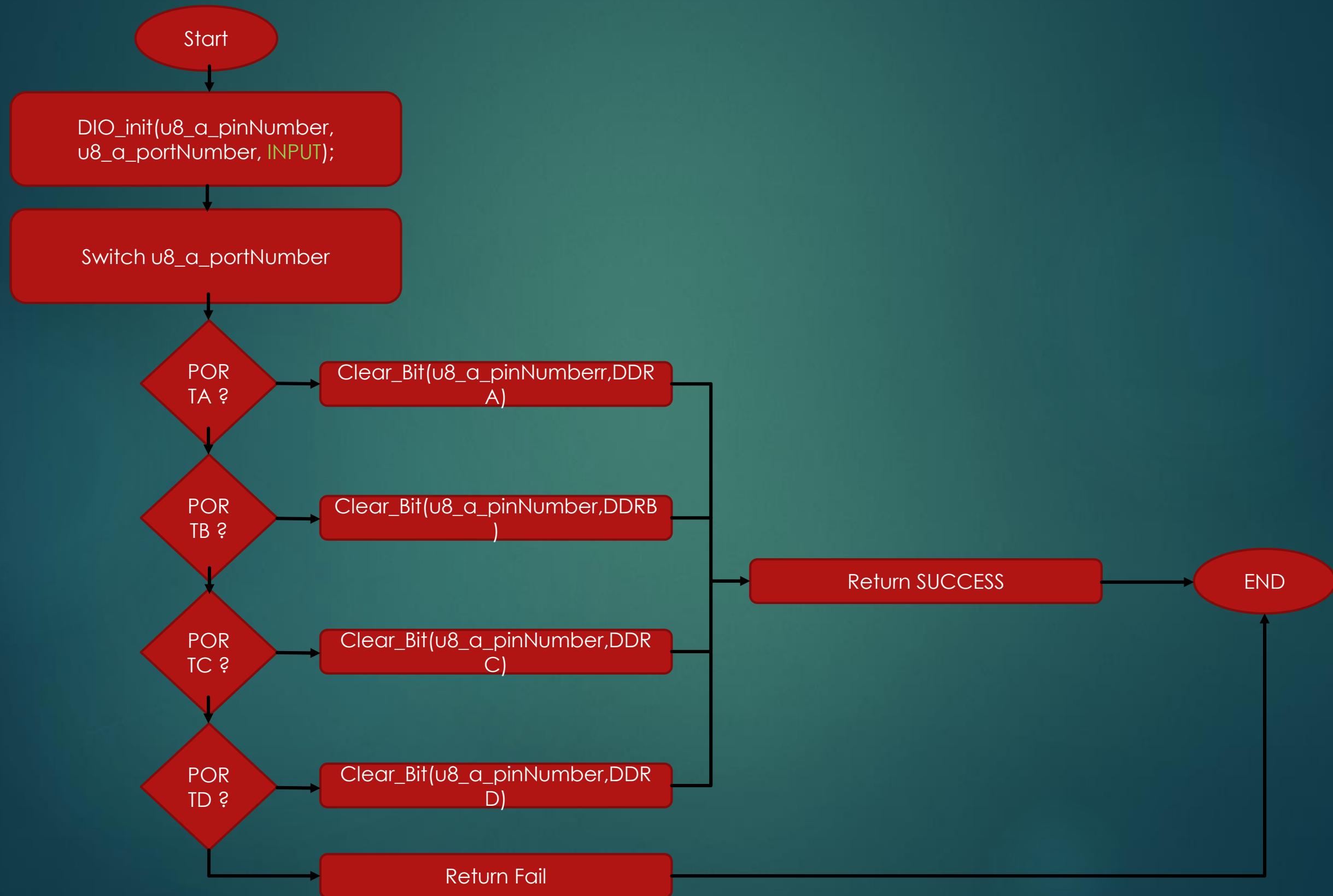
# APIs flowchart:

```
err_state MOTOR_control(uint8_t u8_a_mask, uint8_t u8_a_portNumber, float f_a_speedPercentage);
```



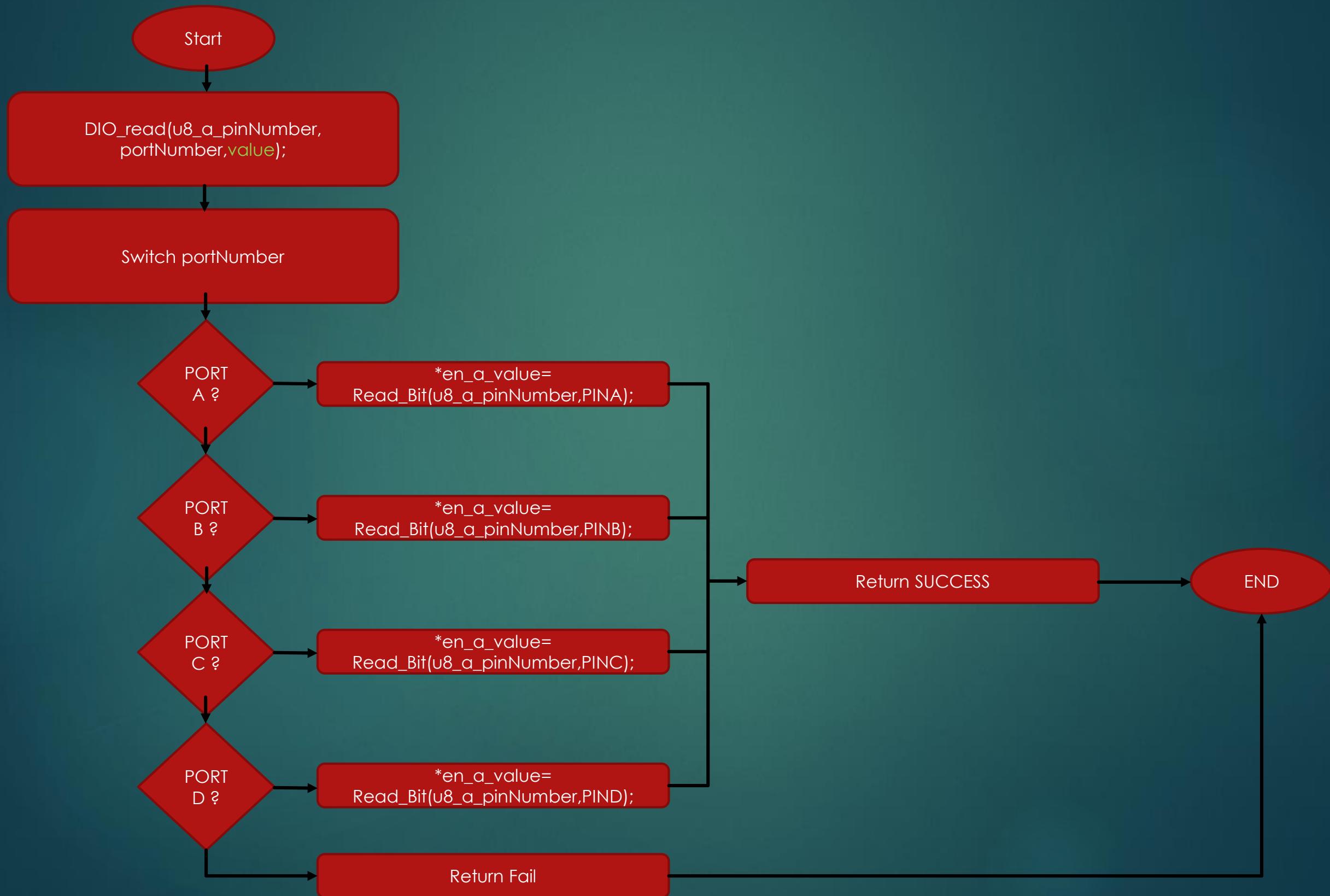
# APIs flowchart:

```
err_state BUTTON_init(uint8_t u8_a_pinNumber, uint8_t u8_a_portNumber);
```



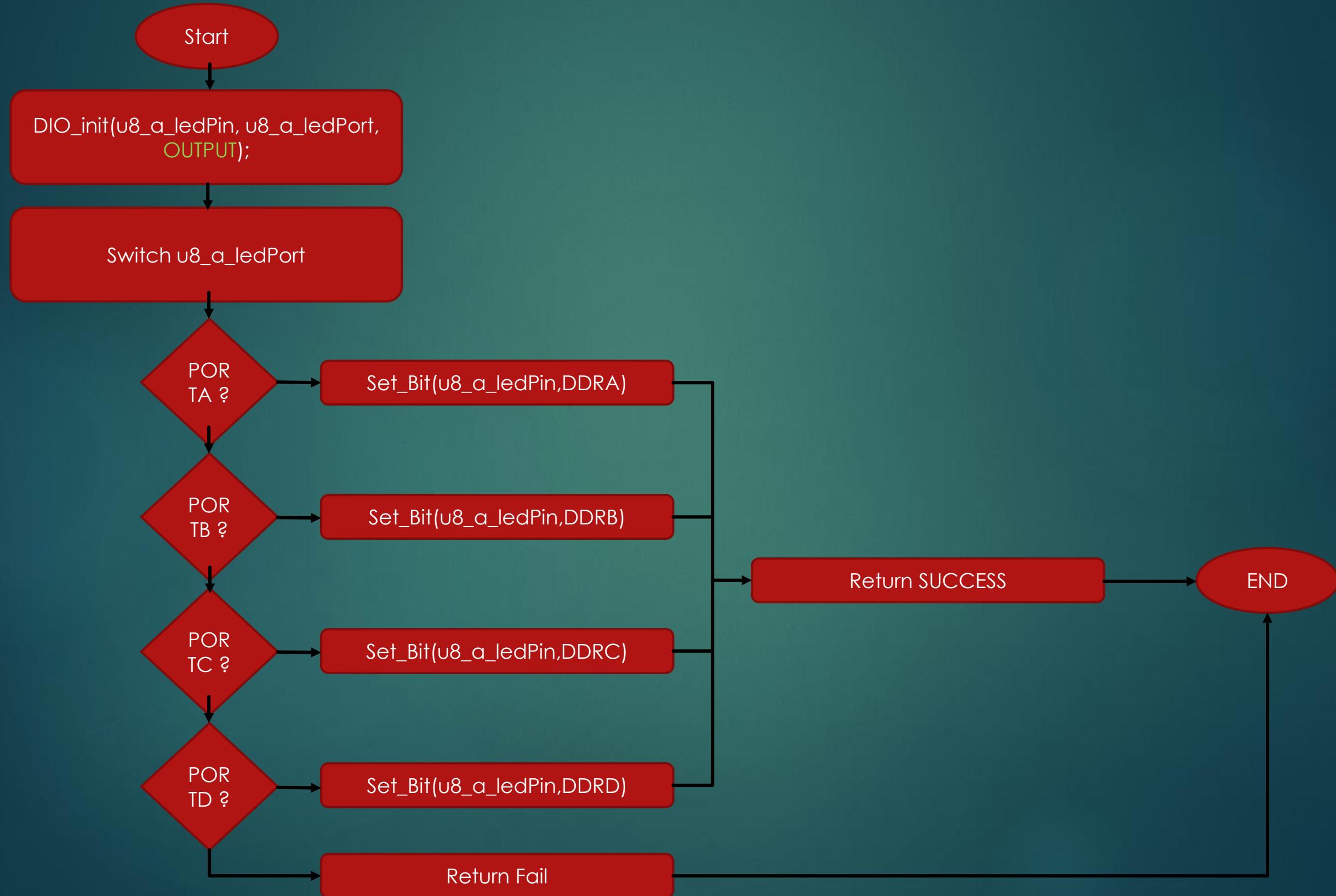
# APIs flowchart:

```
err_state BUTTON_read(uint8_t u8_a_pinNumber, uint8_t u8_a_portNumber, pin_state *en_a_value);
```



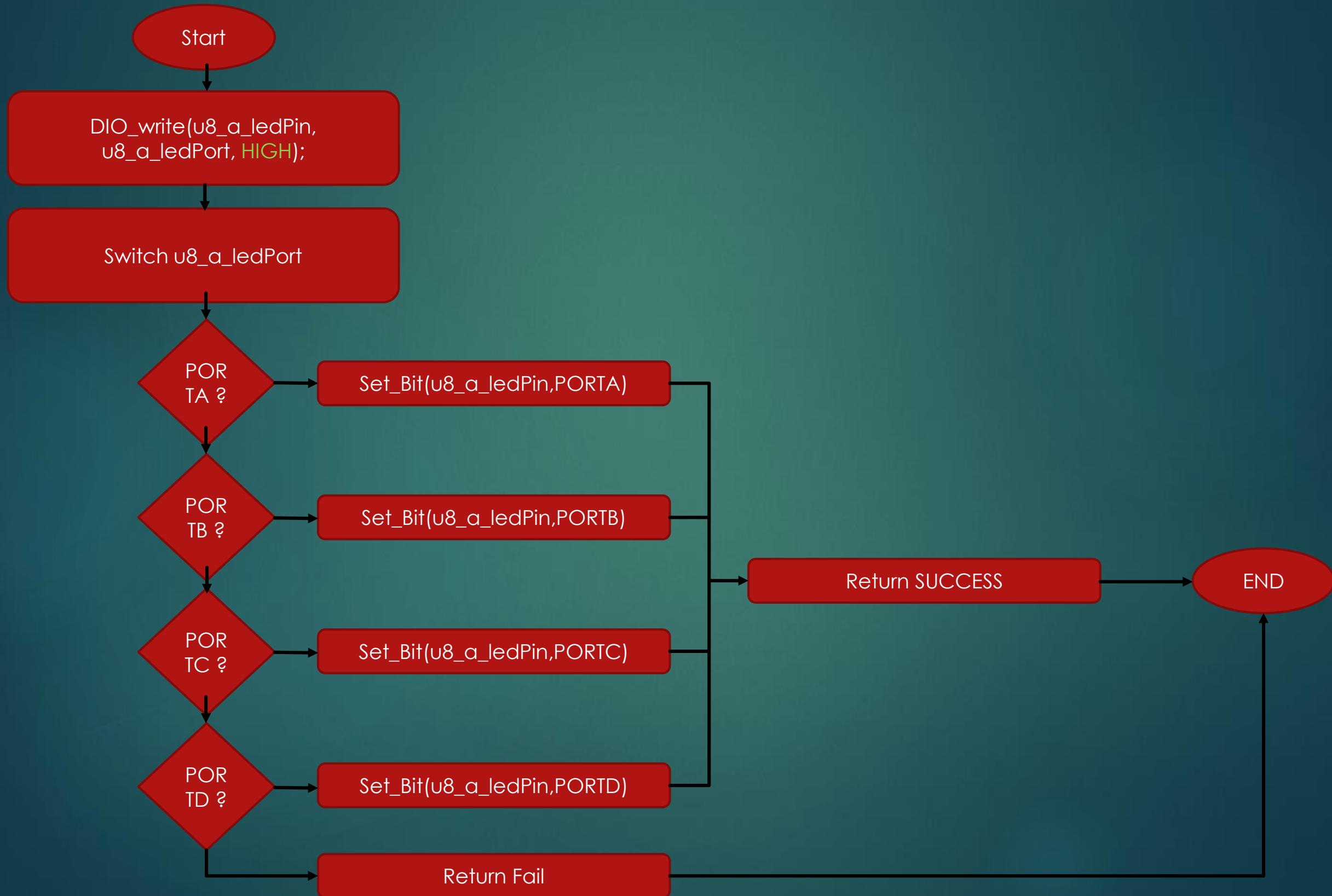
# APIs flowchart:

```
err_state LED_init(uint8_t u8_a_ledPin, uint8_t u8_a_ledPort);
```



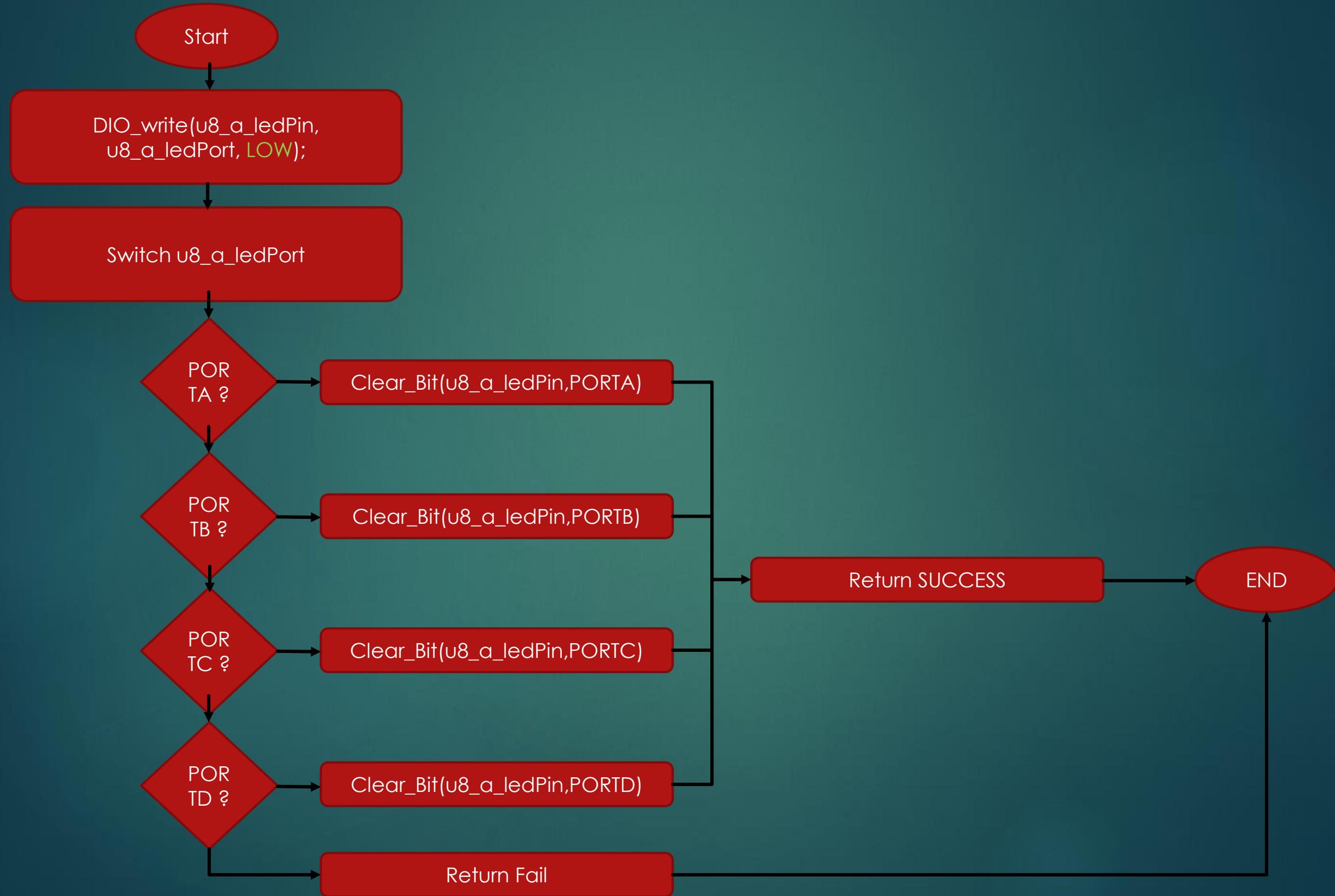
# APIs flowchart:

`err_state LED_on(uint8_t u8_a_ledPin, uint8_t u8_a_ledPort);`



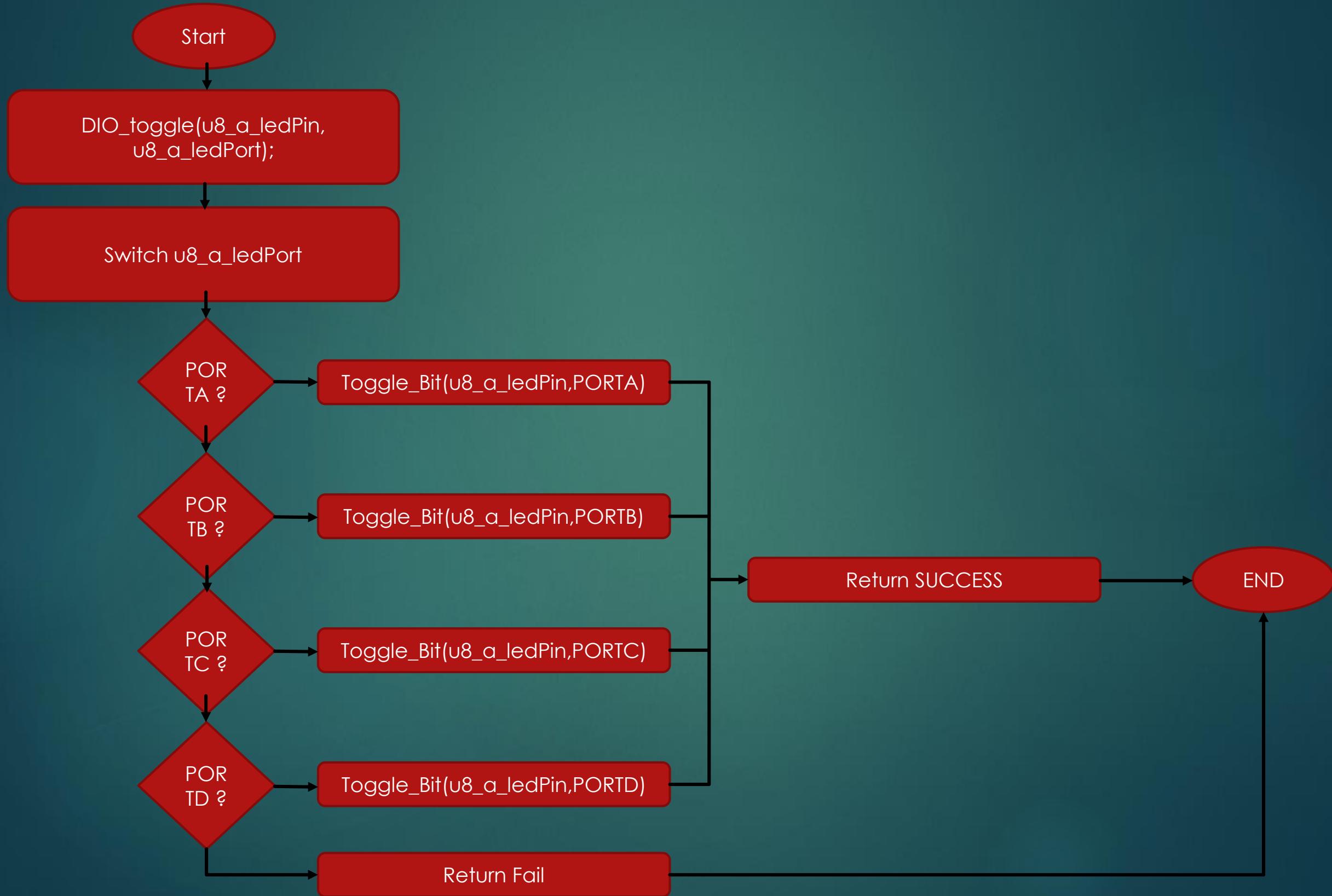
# APIs flowchart:

`err_state LED_off(uint8_t u8_a_ledPin, uint8_t u8_a_ledPort);`



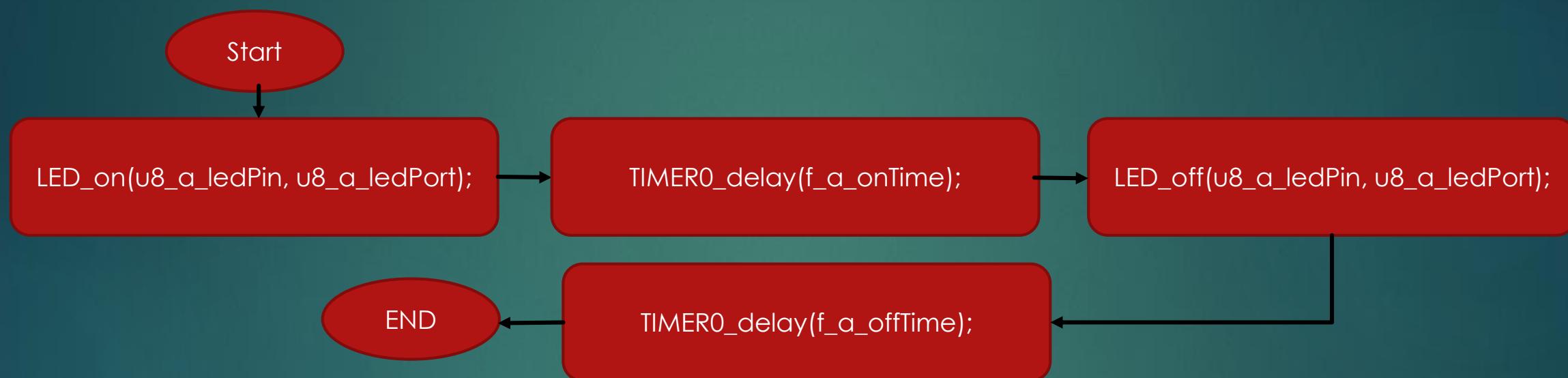
# APIs flowchart:

`err_state LED_toggle(uint8_t u8_a_ledPin, uint8_t u8_a_ledPort);`



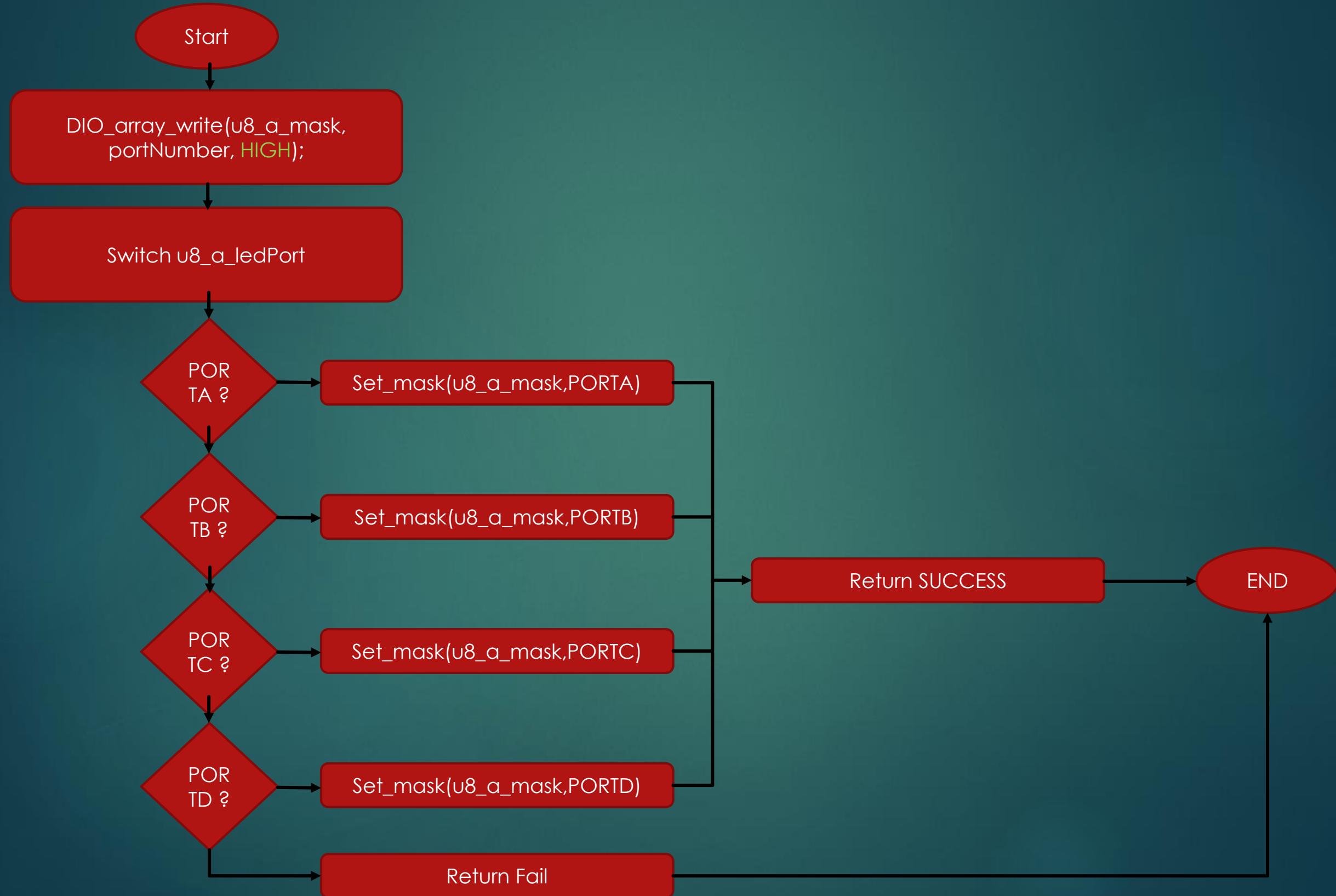
# APIs flowchart:

```
err_state LED_blink(uint8_t u8_a_ledPin, uint8_t u8_a_ledPort, float f_a_onTime, float f_a_offTime);
```



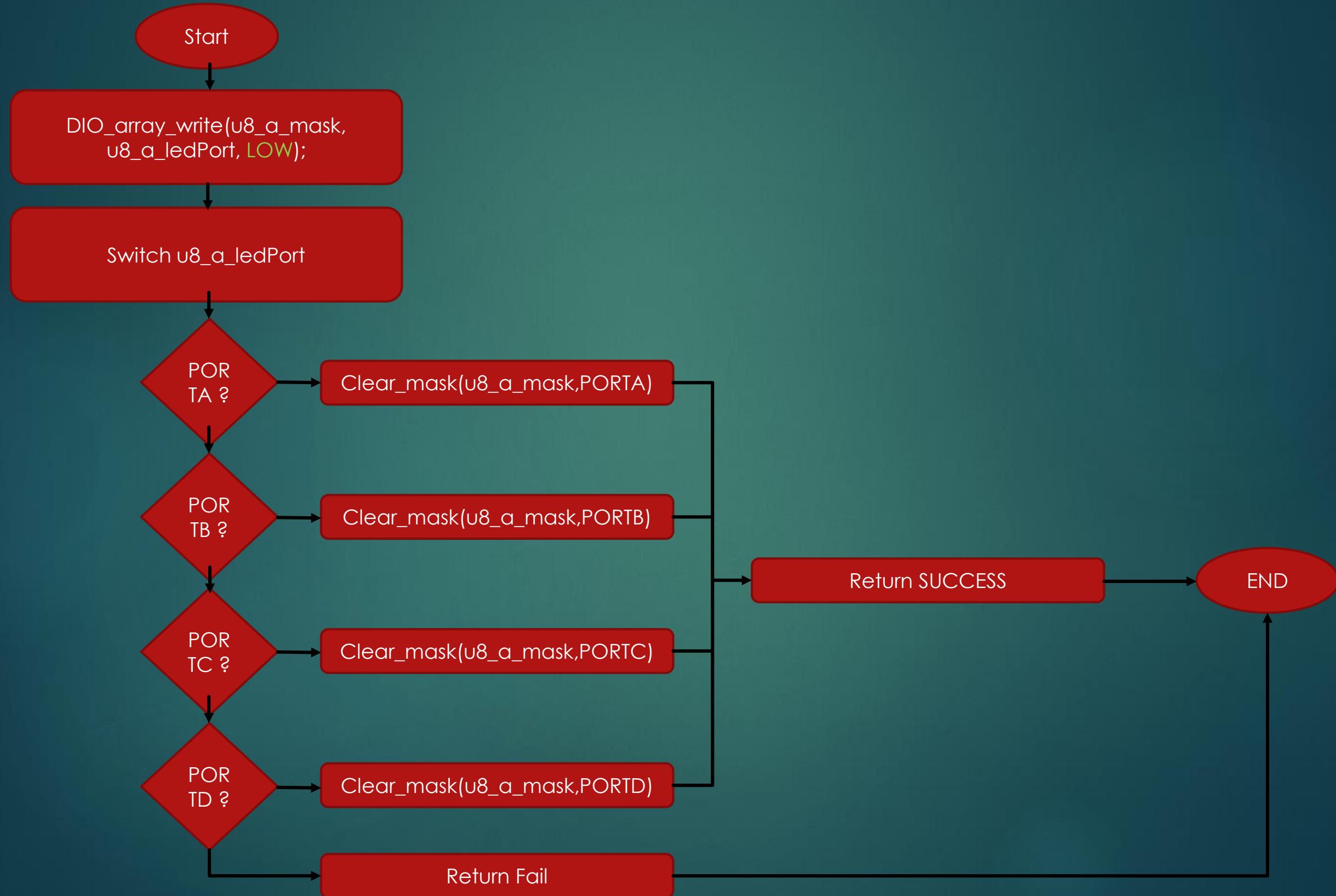
# APIs flowchart:

`err_state LED_array_on(uint8_t u8_a_mask, uint8_t u8_a_ledPort);`



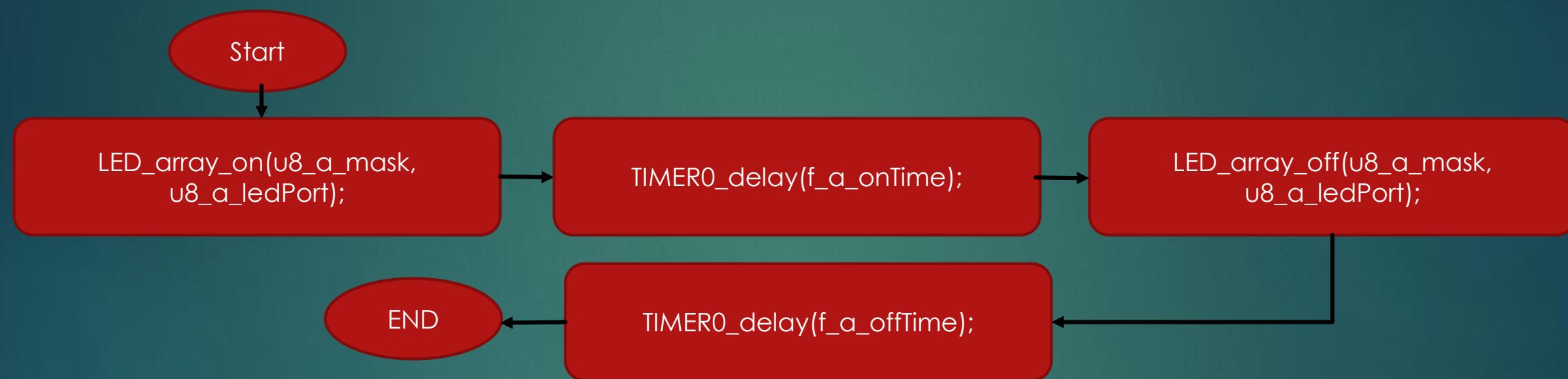
# APIs flowchart:

`err_state LED_array_off(uint8_t u8_a_mask, uint8_t u8_a_ledPort);`



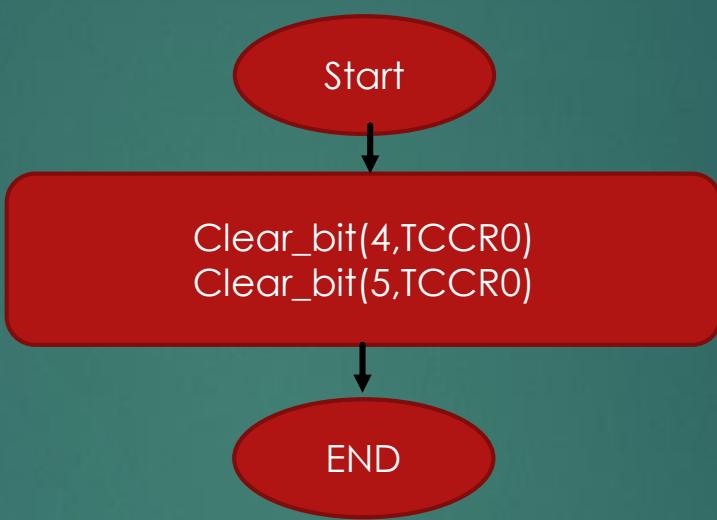
# APIs flowchart:

```
err_state LED_array_blink(uint8_t u8_a_mask, uint8_t u8_a_ledPort, float f_a_onTime, float f_a_offTime);
```



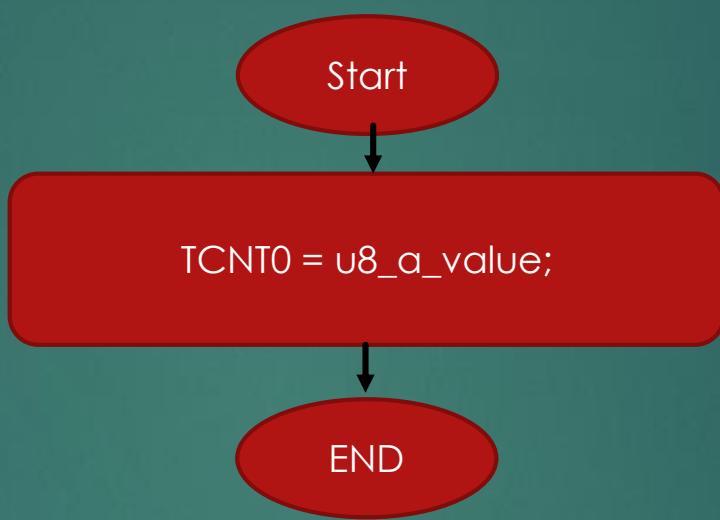
# APIs flowchart:

```
err_state TIMER0_normalMode(void);
```



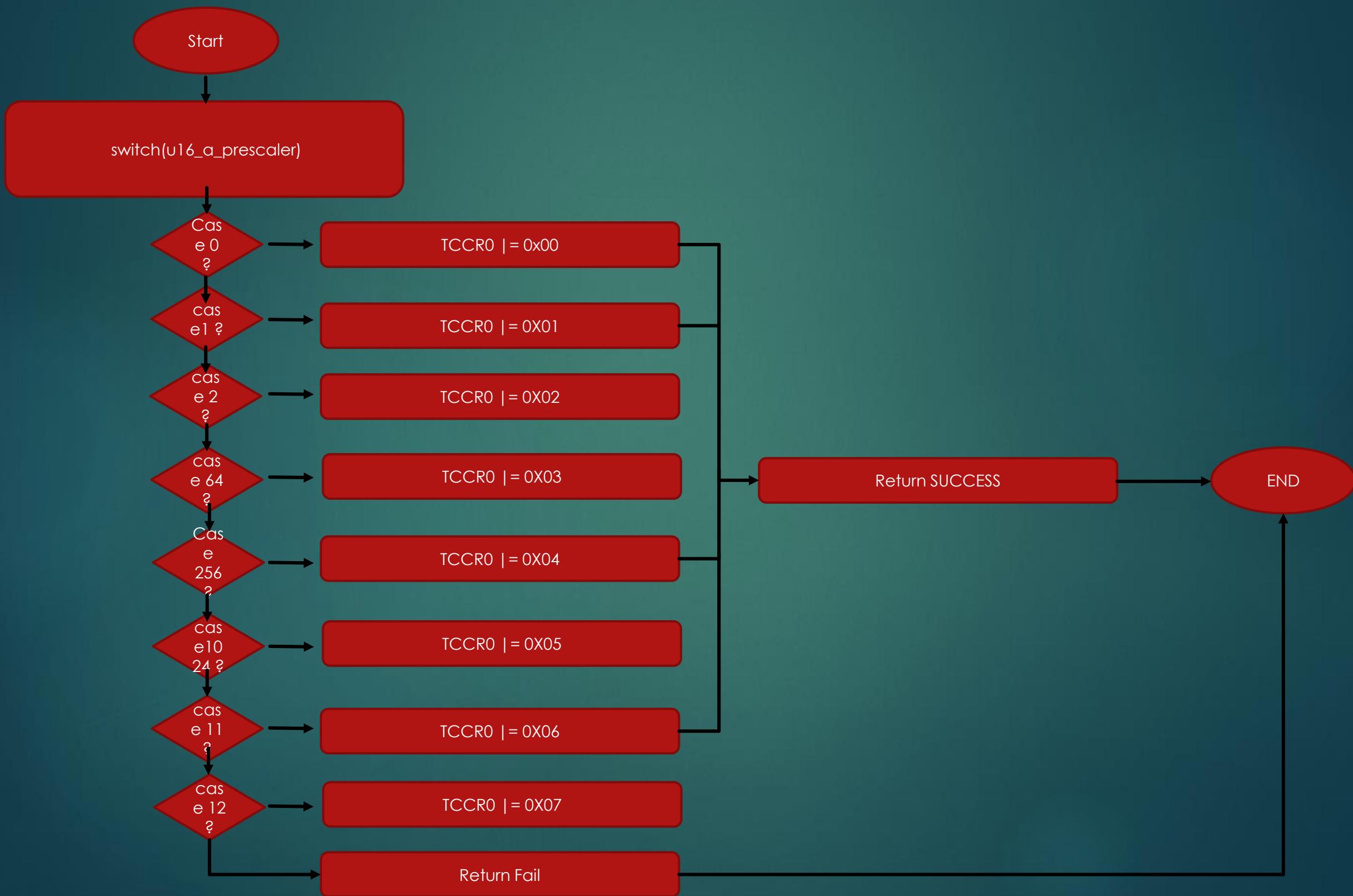
# APIs flowchart:

```
err_state TIMERO_initialValue(uint8_t value);
```



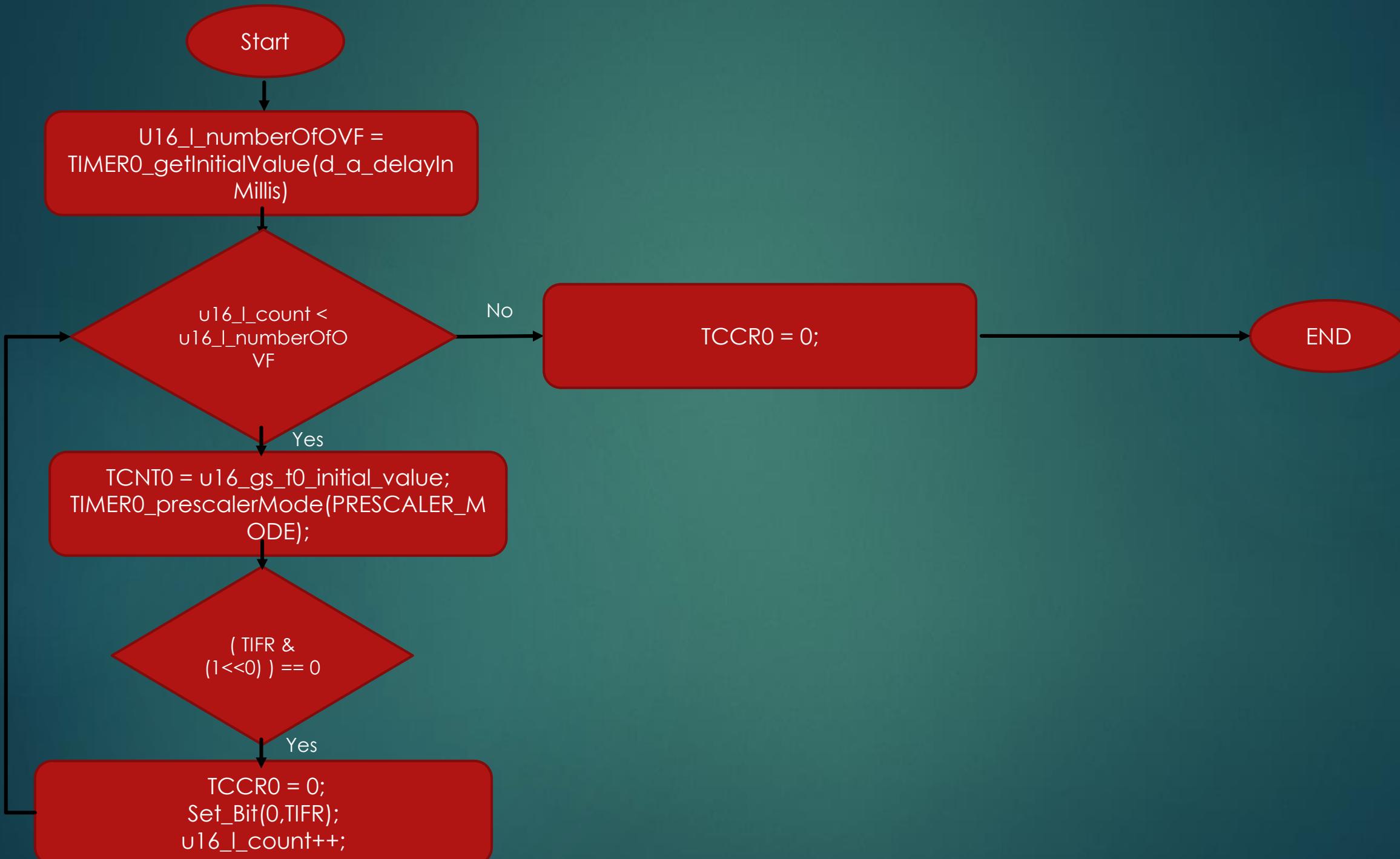
# APIs flowchart:

`err_state TIMER0_prescalerMode(unsigned int u16_a_prescaler)`



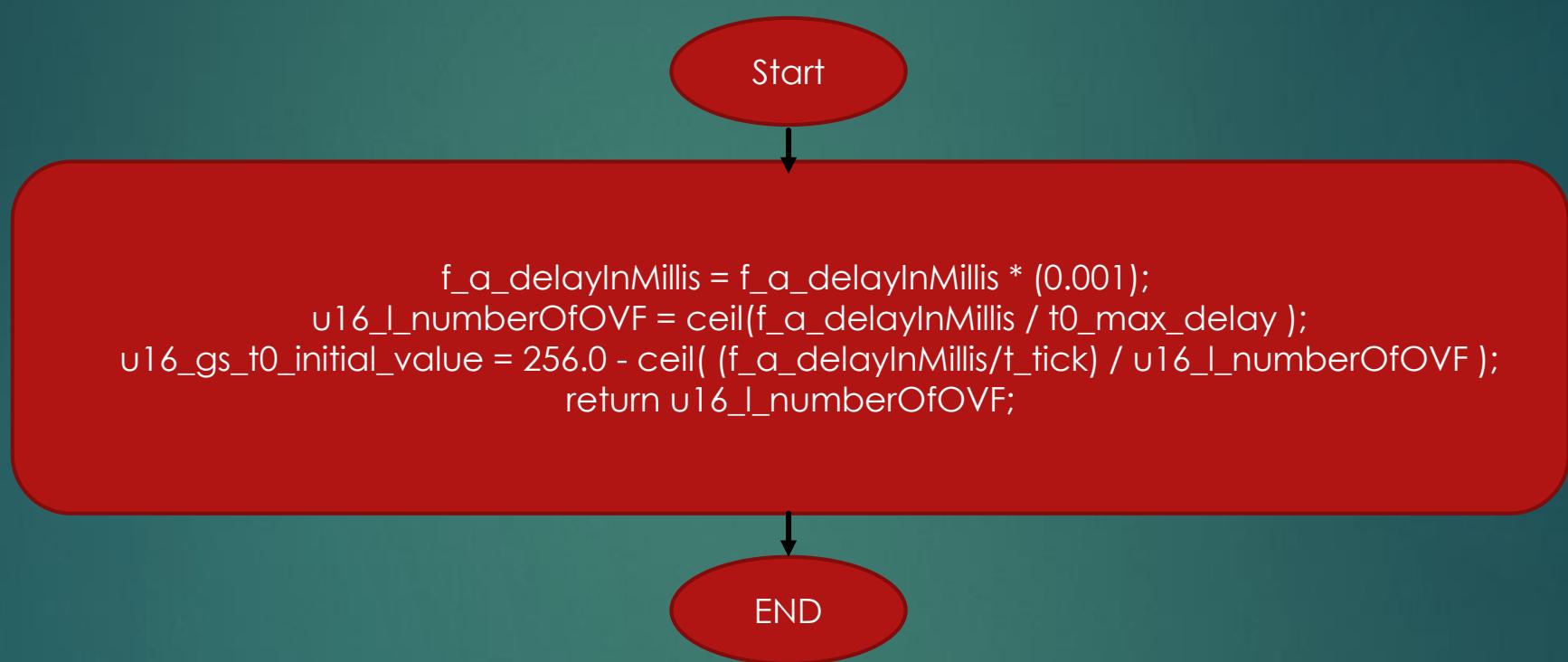
# APIs flowchart:

`err_state TIMER0_delay(float f_a_delayInMillis);`



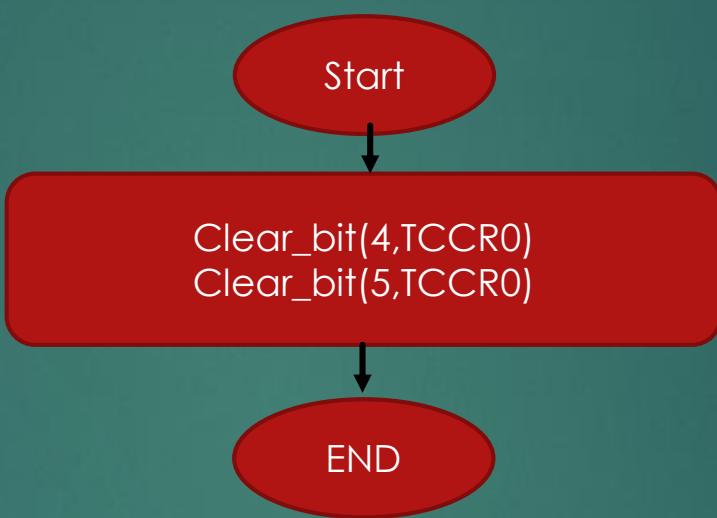
# APIs flowchart:

`unsigned int TIMER0_getInitialValue(float f_a_delayInMillis)`



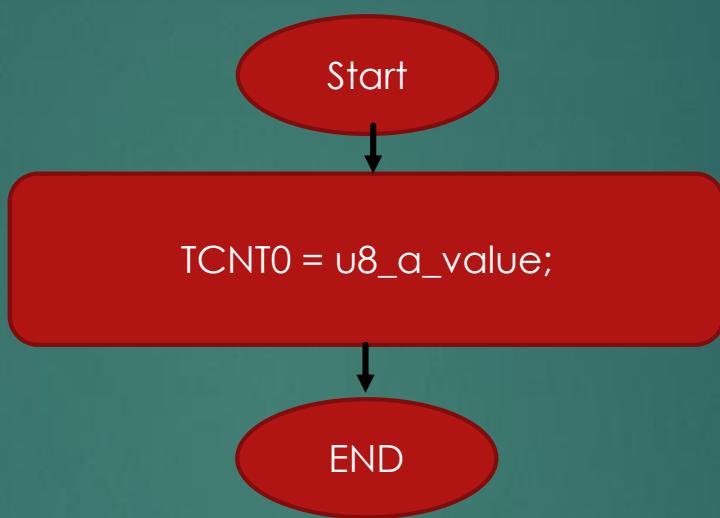
# APIs flowchart:

```
err_state TIMER2_normalMode(void);
```



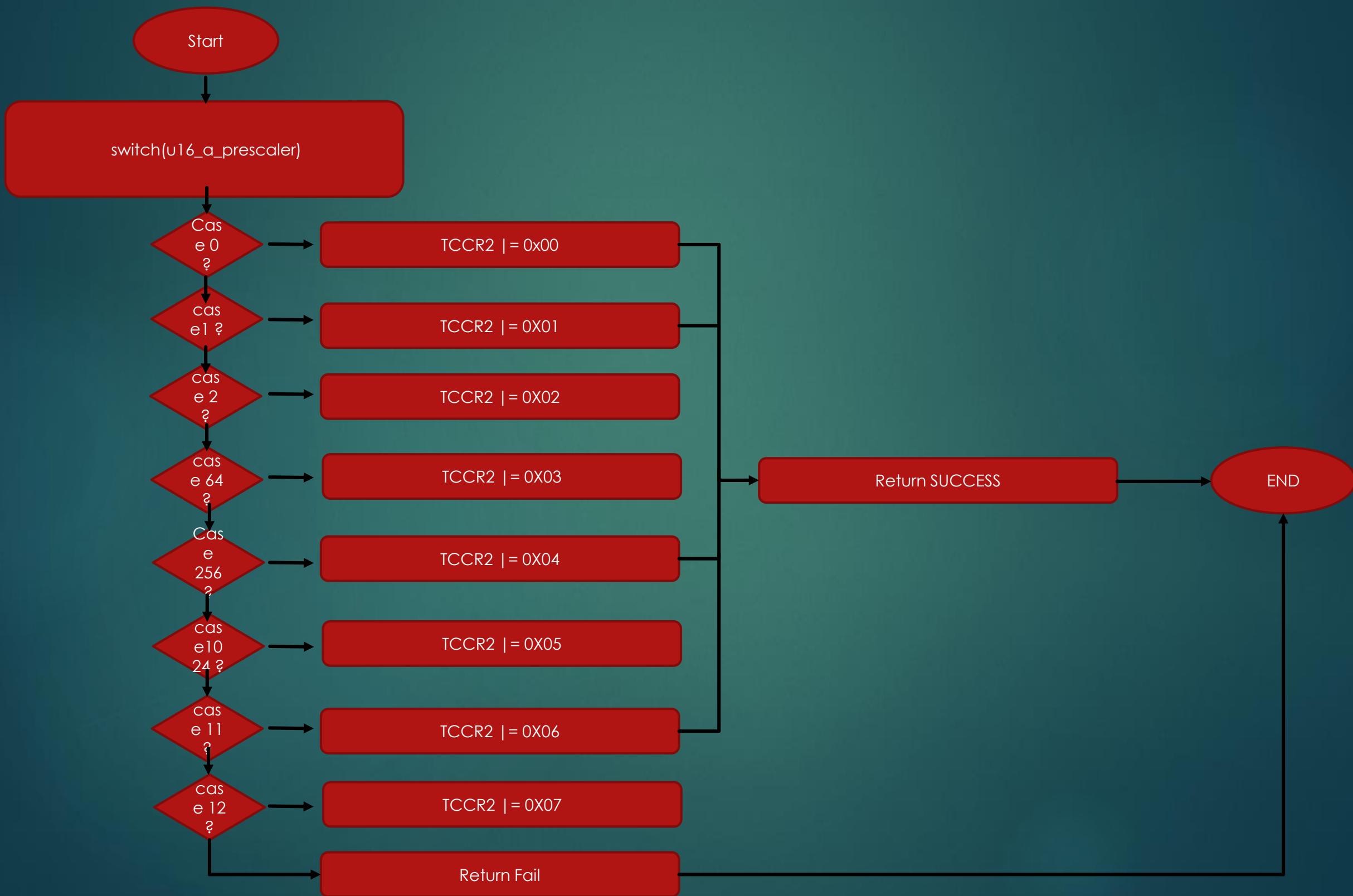
# APIs flowchart:

```
err_state TIMER2 initialValue(uint8_t value);
```



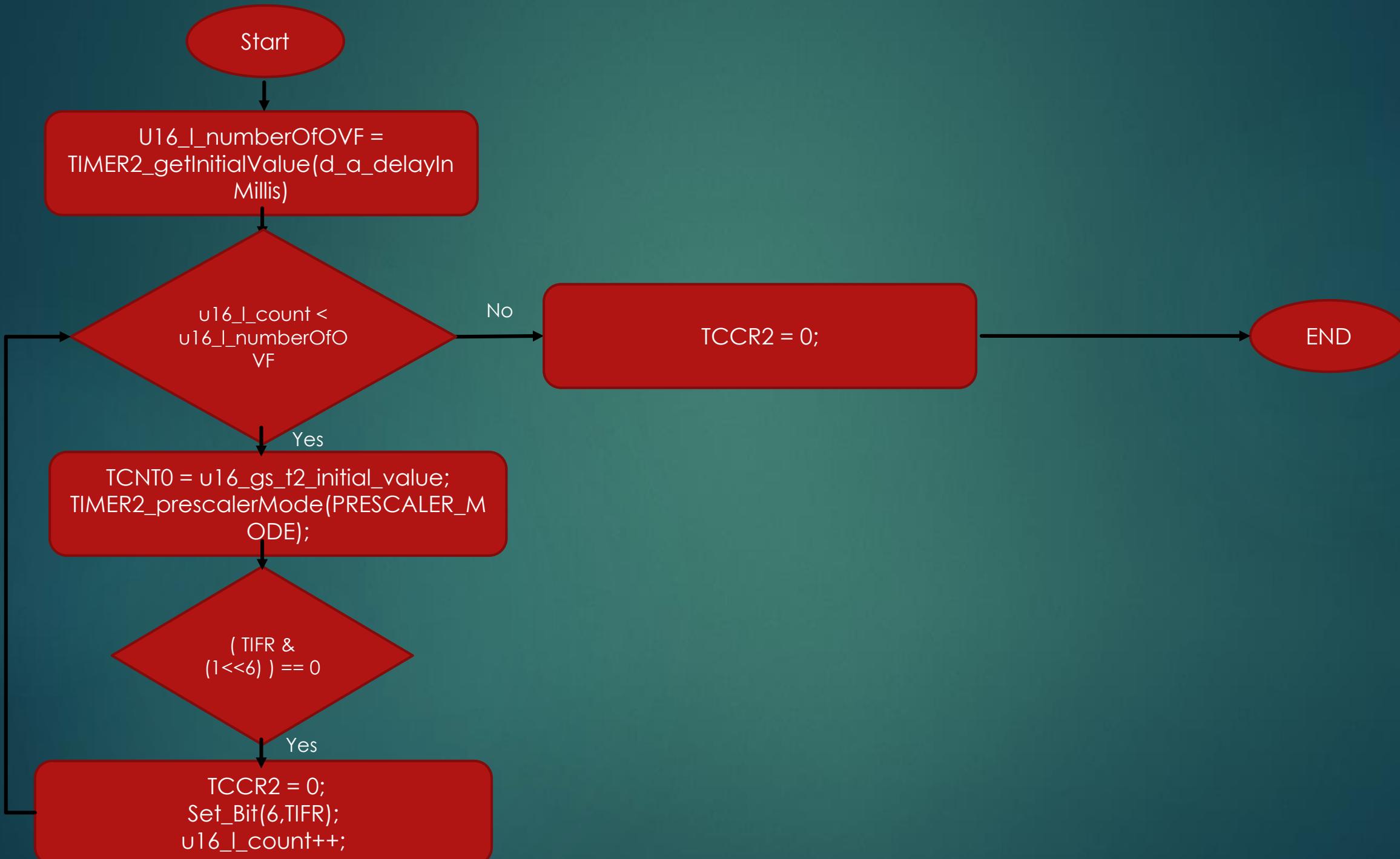
# APIs flowchart:

`err_state TIMER2_prescalerMode(unsigned int u16_a_prescaler)`



# APIs flowchart:

```
err_state TIMER2_delay(float f_a_delayInMillis);
```



# APIs flowchart:

```
unsigned int TIMER2_getInitialValue(float f_a_delayInMillis)
```

