LED sequence V3.0

Kareem Magdy Albolaqi

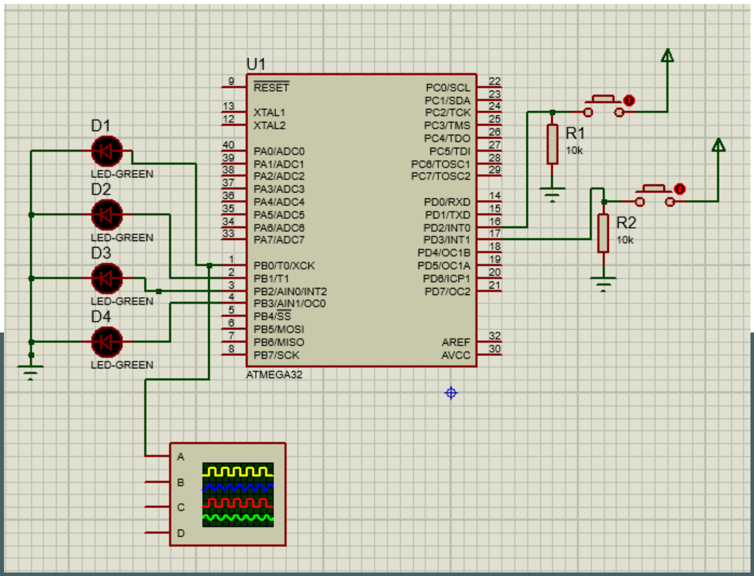
Sprint

3

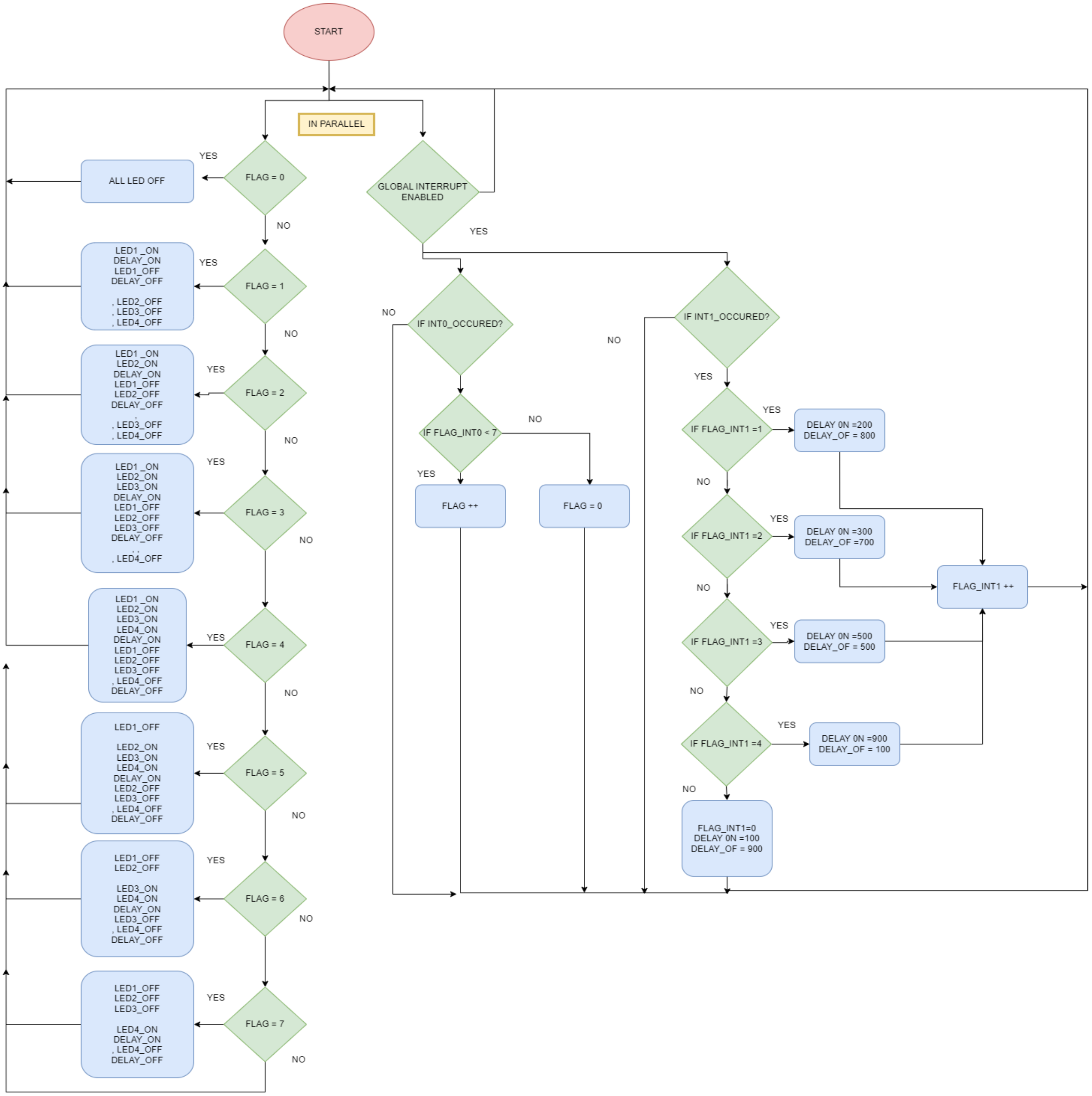
Project description

Software Requirements

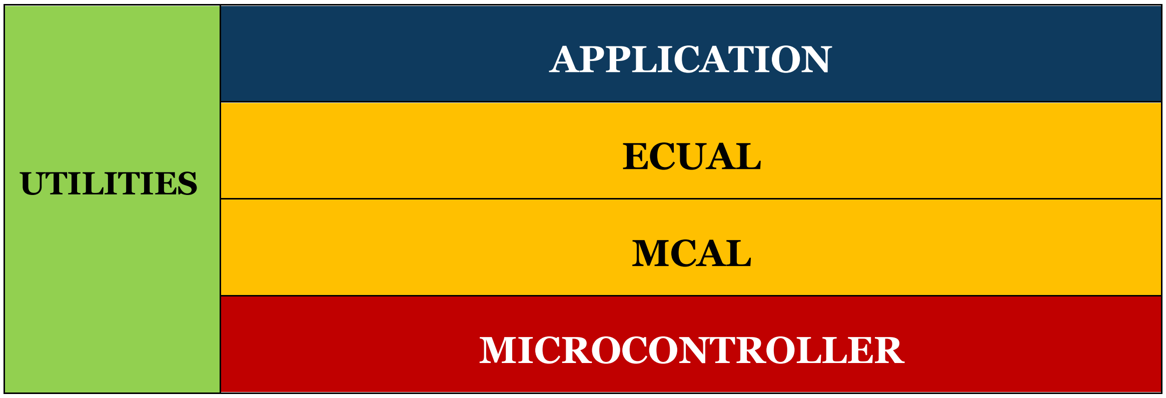
Initially, all LEDs are OFF
Once **BUTTON0** is pressed, **LED0** will blink with **BLINK_1** mode
Each press further will make another LED blinks **BLINK_1** mode
At the **fifth press**, **LED0** will changed to be **OFF**
Each **press further** will make only one LED is **OFF**
This will be repeated forever
The sequence is described below
Initially (OFF, OFF, OFF, OFF)
Press 1 (BLINK_1, OFF, OFF, OFF)
Press 2 (BLINK_1, BLINK_1, OFF, OFF)
Press 3 (BLINK_1, BLINK_1, BLINK_1, OFF)
Press 4 (BLINK_1, BLINK_1, BLINK_1, BLINK_1)
Press 5 (OFF, BLINK_1, BLINK_1, BLINK_1)
Press 6 (OFF, OFF, BLINK_1, BLINK_1)
Press 7 (OFF, OFF, OFF, BLINK_1)
Press 8 (OFF, OFF, OFF, OFF)
Press 9 (BLINK_1, OFF, OFF, OFF)
When BUTTON1 has pressed the blinking on and off durations will be changed
No press → **BLINK_1** mode (**ON**: 100ms, **OFF**: 900ms)
First press → **BLINK_2** mode (**ON**: 200ms, **OFF**: 800ms)
Second press → **BLINK_3** mode (**ON**: 300ms, **OFF**: 700ms)
Third press → **BLINK_4** mode (**ON**: 500ms, **OFF**: 500ms)
Fourth press → **BLINK_5** mode (**ON**: 800ms, **OFF**: 200ms)
Fifth press → **BLINK_1** mode
USE EXTERNAL INTERRUPTS

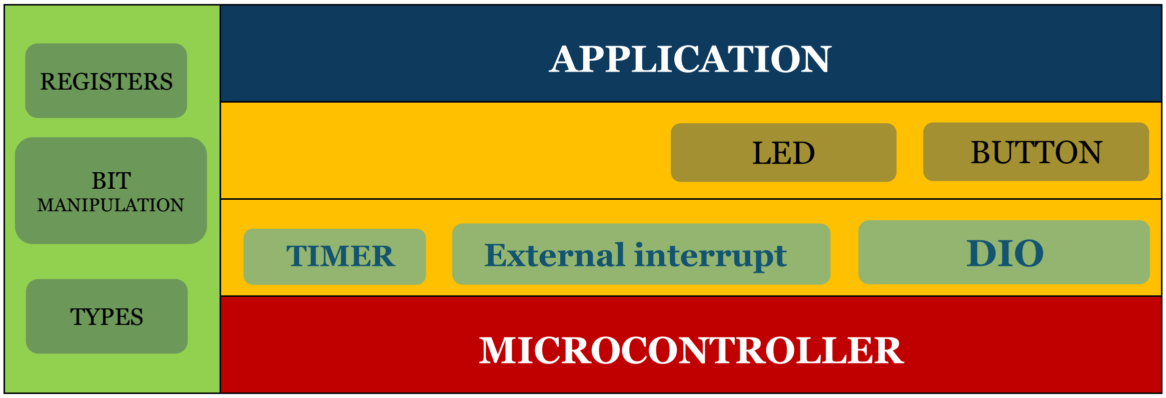


PROJECT FLOWCHART



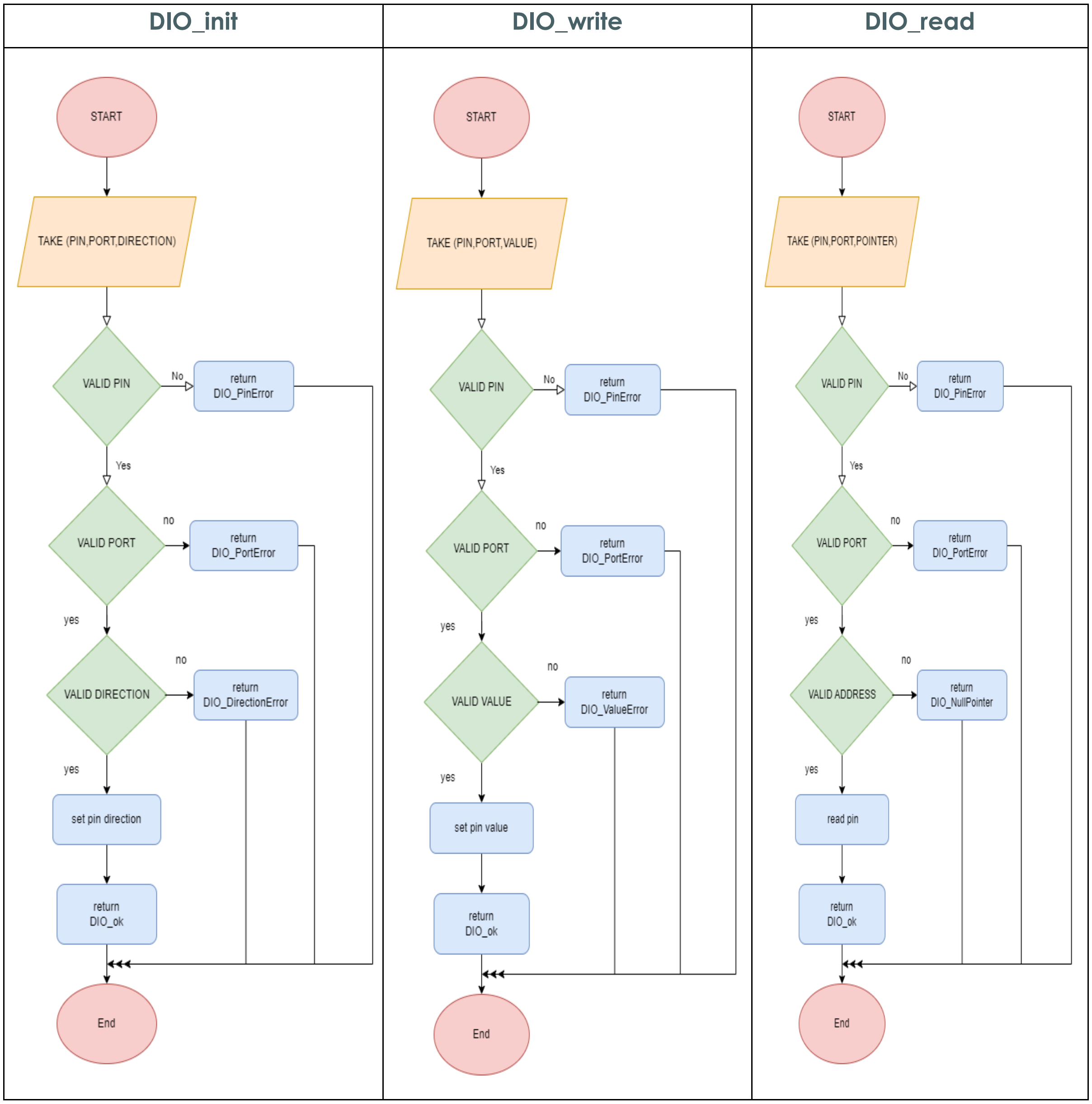
LAYERD ARCHTICTURE





DIO DRIVER

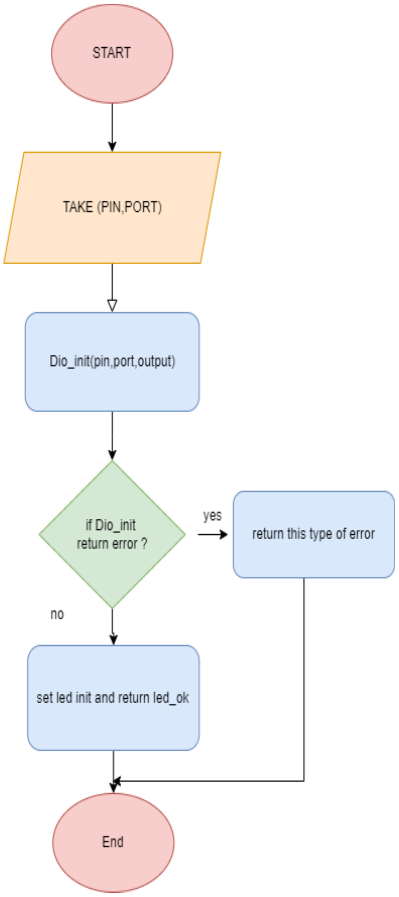
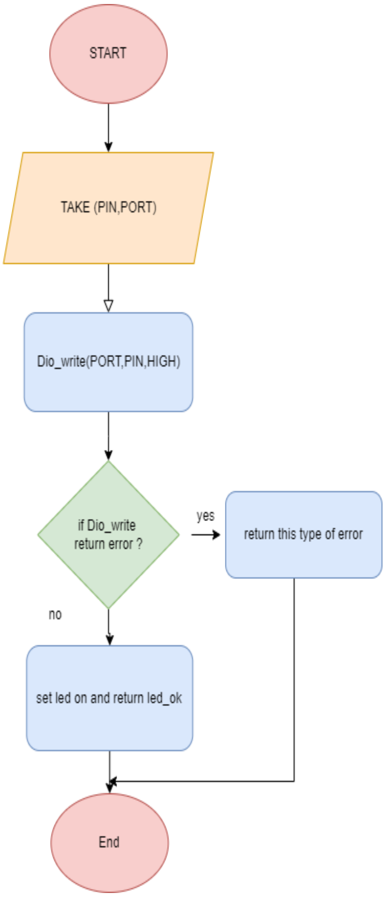
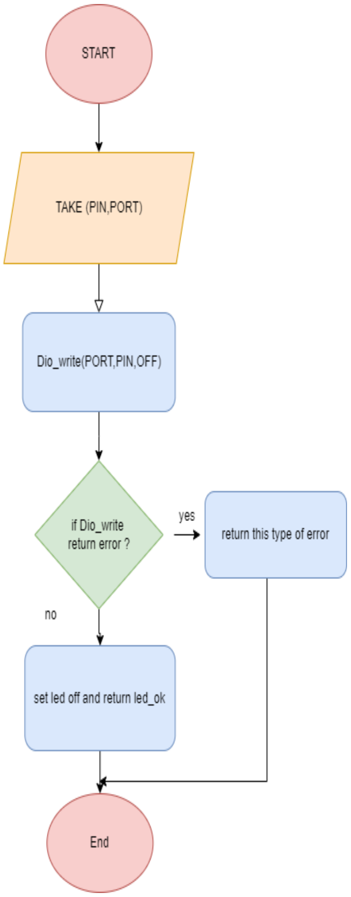
```
Dio_ErrorStatus DIO_init(PORT_NUM PortNum, PIN_NUM PinNum , PIN_DIR direction);
Dio_ErrorStatus DIO_write(PORT_NUM PortNum, PIN_NUM PinNum , uint8_t value);
Dio_ErrorStatus DIO_read(PORT_NUM PortNum, PIN_NUM PinNum , uint8_t* value);
```



PROJECT MODULES APIS

LED DRIVER

```
Dio_ErrorStatus LED_init(PORT_NUM portNum , PIN_NUM pinNum);
Dio_ErrorStatus LED_on (PORT_NUM portNum , PIN_NUM pinNum);
Dio_ErrorStatus LED_off(PORT_NUM portNum , PIN_NUM pinNum);
```

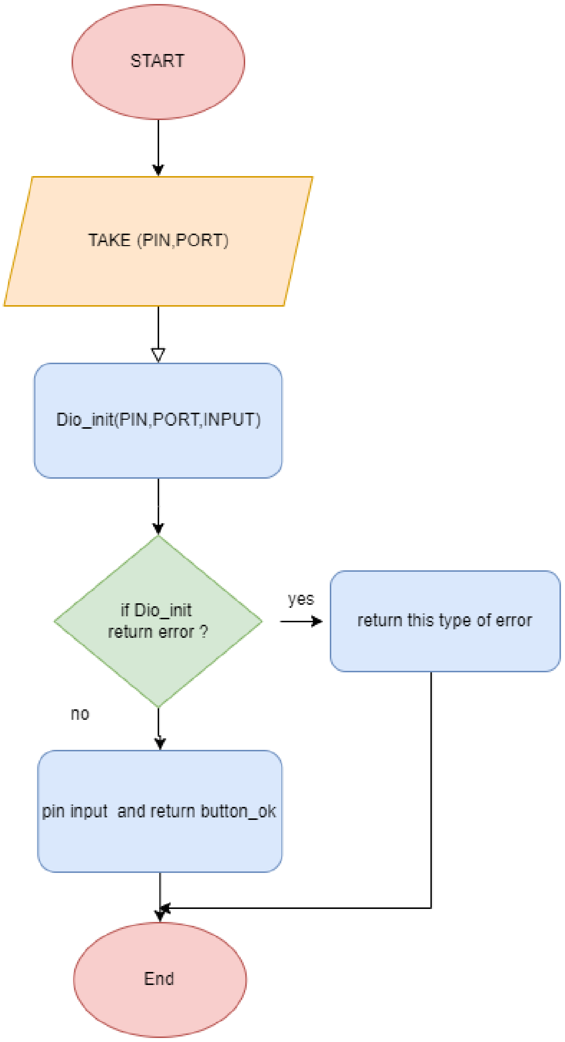
LED_init	LED_on	LED_off
 START TAKE (PIN,PORT) Dio_init(pin,port,output) if Dio_init return error ? yes return this type of error no set led init and return led_ok End	 START TAKE (PIN,PORT) Dio_write(PORT,PIN,HIGH) if Dio_write return error ? yes return this type of error no set led on and return led_ok End	 START TAKE (PIN,PORT) Dio_write(PORT,PIN,OFF) if Dio_write return error ? yes return this type of error no set led off and return led_ok End

PROJECT MODULES APIS

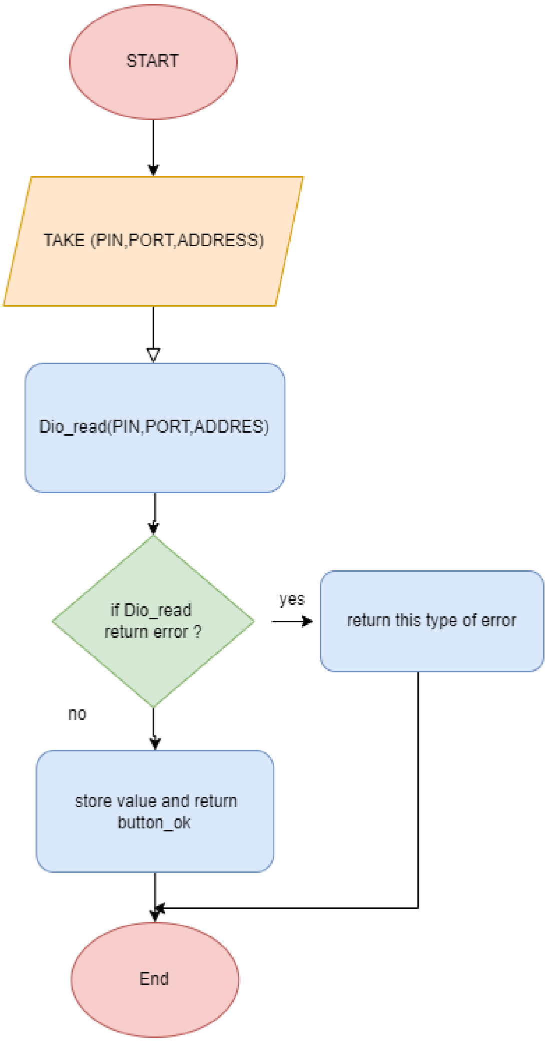
BUTTON DRIVER

```
        Dio_ErrorStatus BUTTON_init(PORT_NUM portnum ,PIN_NUM pinnum);
       Dio_ErrorStatus BUTTON_read(PORT_NUM portnum ,PIN_NUM pinnum, uint8_t *value);
```

BUTTON_init



BUTTON_read

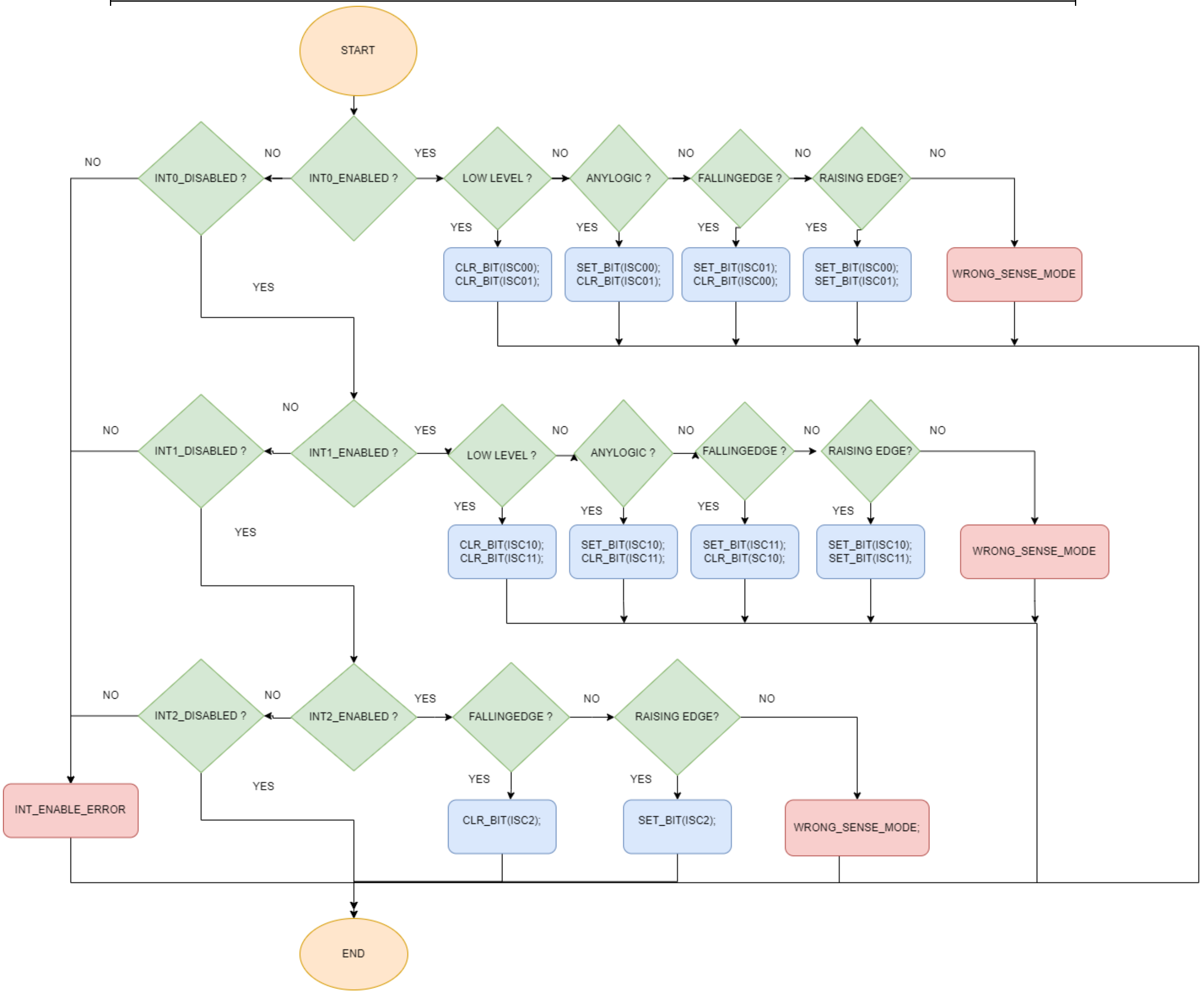


EXTERNAL INTERRUPT DRIVER

```
Ext_intErrorStatus ExtInt_init();
```

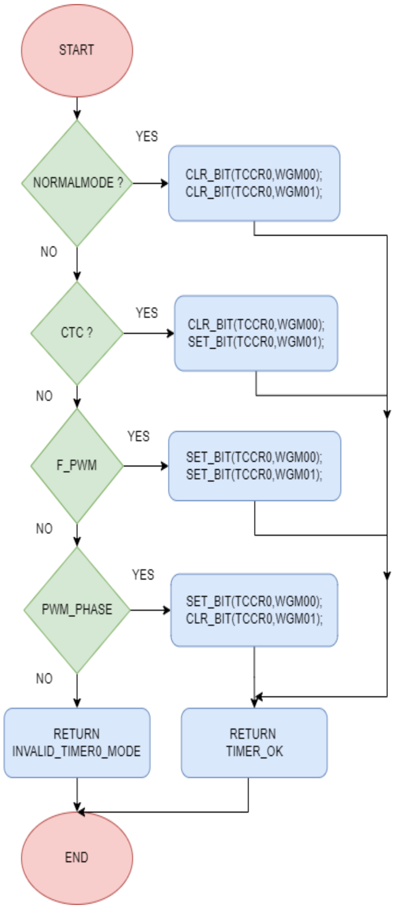
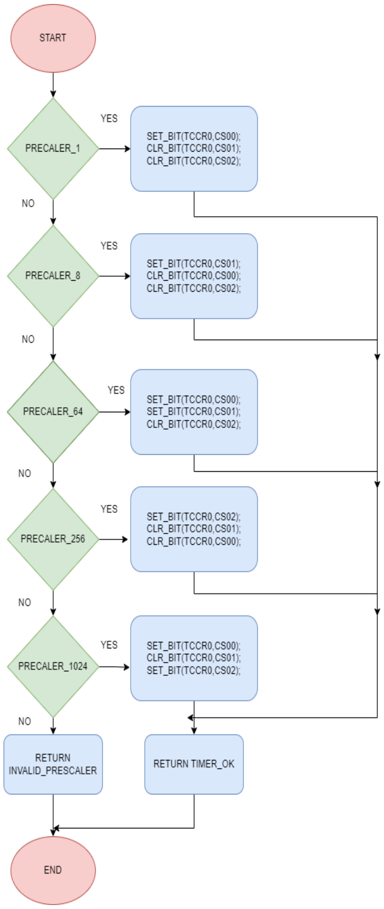
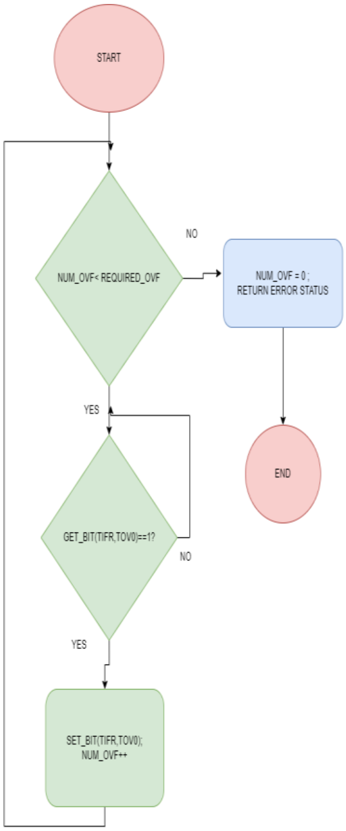
NOTE: ALL INTERRUPT CONFIGURATIONS ARE IN Ext_IntCnfg.h

ExtInt_init



TIMER DRIVER

```
Timer_ErrorStatus TIMER_init(Timer_Mode mode);
Timer_ErrorStatus TIMER_start(Timer_Prescaler prescaler);
void TIMER_stop(void);
Timer_ErrorStatus TIMER_setIntialValue(uint8_t value);
Timer_ErrorStatus TIMER_OvfNum(uint32 overflow);
\
```

TIMER_0_init	TIMER_0_start	TIMER_OvfNum
 START NORMALMODE ? YES CLR_BIT(TCCR0,WGM00); CLR_BIT(TCCR0,WGM01); NO CTC ? YES CLR_BIT(TCCR0,WGM00); SET_BIT(TCCR0,WGM01); NO F_PWM YES SET_BIT(TCCR0,WGM00); SET_BIT(TCCR0,WGM01); NO PWM_PHASE YES SET_BIT(TCCR0,WGM00); CLR_BIT(TCCR0,WGM01); NO RETURN INVALID_TIMER0_MODE RETURN TIMER_OK END	 START PRECALER_1 YES SET_BIT(TCCR0,CS00); CLR_BIT(TCCR0,CS01); CLR_BIT(TCCR0,CS02); NO PRECALER_8 YES SET_BIT(TCCR0,CS01); CLR_BIT(TCCR0,CS00); CLR_BIT(TCCR0,CS02); NO PRECALER_64 YES SET_BIT(TCCR0,CS00); SET_BIT(TCCR0,CS01); CLR_BIT(TCCR0,CS02); NO PRECALER_256 YES SET_BIT(TCCR0,CS02); CLR_BIT(TCCR0,CS01); CLR_BIT(TCCR0,CS00); NO PRECALER_1024 YES SET_BIT(TCCR0,CS00); CLR_BIT(TCCR0,CS01); SET_BIT(TCCR0,CS02); NO RETURN INVALID_PRESCALER RETURN TIMER_OK END	 START NUM_OVF< REQUIRED_OVF NO NUM_OVF = 0 ; RETURN ERROR STATUS END YES GET_BIT(TIFR,TOV0)==1? NO YES SET_BIT(TIFR,TOV0); NUM_OVF++