DESIGN DOCUMENT

RGB LED CONTROL

V1.0 DESIGN

Team2:

-Amr El-Abd

-Ahmed Aatef

Table of Contents: -

Project introduction:

This project aims to implement a button-driven control system using the TivaC board, where the state of the RGB LED changes based on the number of times the button (SW1) is pressed. Initially, the RGB LED is turned off. With each press of the button, a specific LED colour is illuminated: red, green, blue, and finally, all LEDs are turned on. On the fifth press, all LEDs are disabled, and on the sixth press, the cycle repeats. The project requires the development of drivers, including a GPIO driver to initialize, configure, and control pins, an LED driver to control the RGB LED, and a button driver to handle button press events.

Project description :-

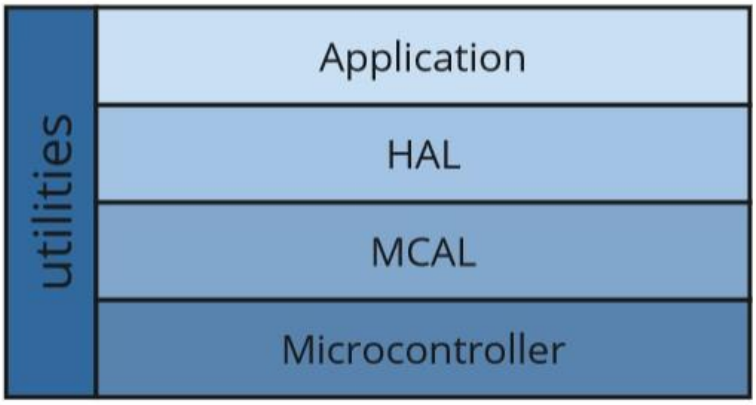
1. Hardware Requirements

1. Use the TivaC board.
2. Use SW1 as an input button.
3. Use the RGB LED.

2. Software Requirements

1. The RGB LED is OFF initially.
2. Pressing SW1:
    1. After the first press, the Red led is on
    2. After the second press, the Green Led is on
    3. After the third press, the Blue led is on
    4. After the fourth press, all LEDs are on
    5. After the fifth press, should disable all LEDs.
    6. After the sixth press, repeat steps from 1 to 6.

Layered Architectures:-



Application Layer: This is the topmost layer of the software stack, which contains the actual application logic. It interacts with the lower layers to perform its tasks. It is responsible for implementing the desired functionality of the system.

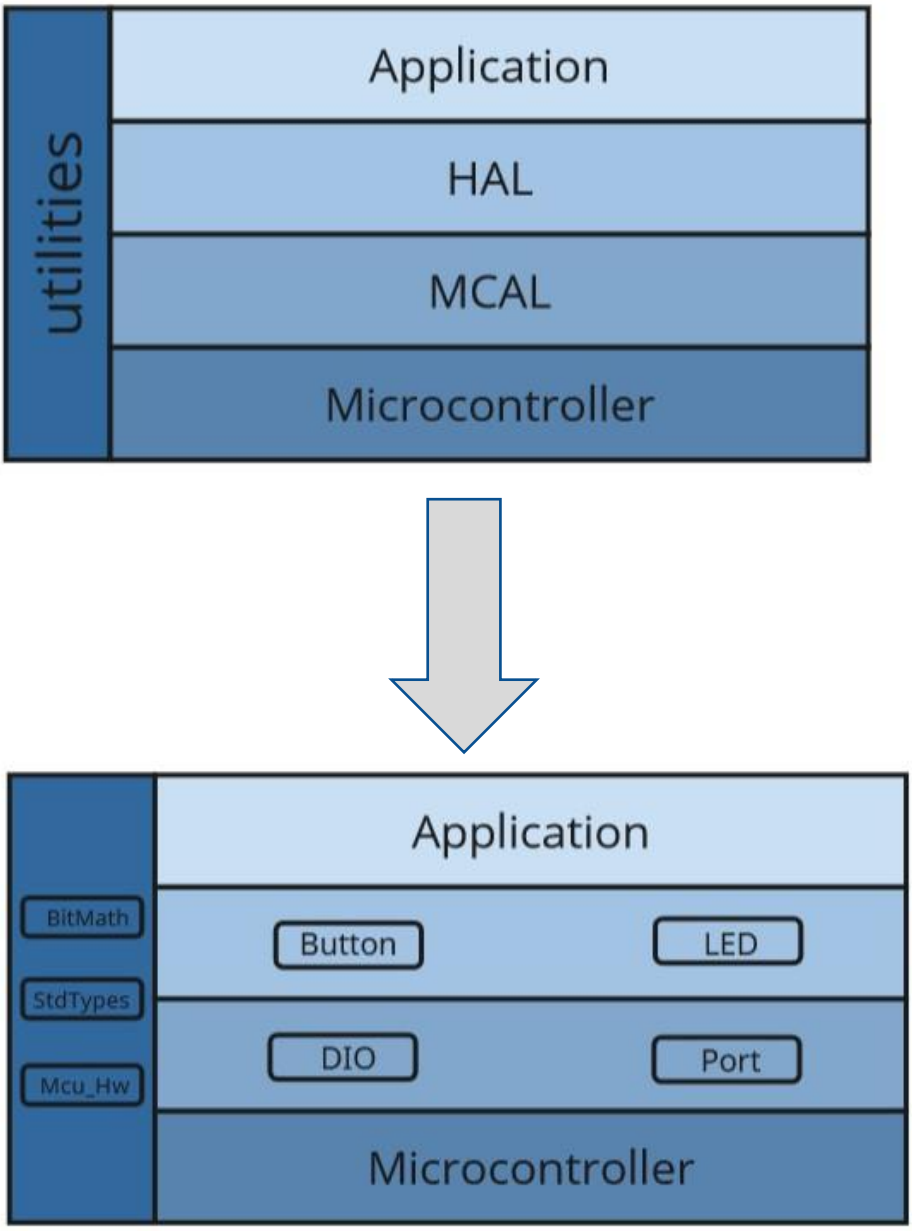
HAL Layer: This layer provides an abstraction for external devices connected to the microcontroller. The HAL layer provides interface to access external devices and hides the implementation details from the application layer.

MCAL Layer (Microcontroller Abstraction Layer): This layer provides an abstraction for the microcontroller hardware. It includes low-level drivers for peripherals. It hides the hardware details and provides a uniform interface to the upper layers.

Utilities Layer: the utilities layer includes memory mapping, standard types, and utils.h. Memory mapping involves defining the memory layout and addresses for different components. Standard types provide a set of predefined data types that ensure consistency and portability across different platforms. The utils.h header file contains utility functions and macros that offer commonly used functionalities, such as bit manipulation.

Microcontroller: This layer represents the physical hardware layer consisting of the microcontroller chip. The microcontroller is responsible for executing the code stored in its memory and controlling the behavior of the system.

System modules:-



Drivers' documentation:-

Port driver:

Description: Driver to Setup the pin configuration:
- Setup the pin as Digital GPIO pin
- Setup the direction of the GPIO pin
- Set the passed initial values for the GPIO pin
- Setup the mode of the GPIO pin
- Setup the internal resistor for i/p pin
- Setup the output current in case of output pi

APIs:

```
void PortInit(const strPortConfig_t* ConfigPtr);
```

DIO driver:

Description: Driver for DIO to read/write/toggle Channel

APIs:

```
enuDioLevel_t DioReadChannel(DioChannel_t ChannelId);

void DioWriteChannel(DioChannel_t ChannelId, enuDioLevel_t Level);

enuDioLevel_t DioToggleChannel(DioChannel_t ChannelId);
```

LED driver:

Description: Driver to initialize/ turn on/ turn off/ toggle the connected channel

APIs:

```
void LedInit(void);

void LedTurnOn(LedChannel_t LedChannel);

void LedTurnOff(LedChannel_t LedChannel);

void LedToggle(LedChannel_t LedChannel);
```

Button driver:

Description: Driver to initialize and get the state of the connected Buttons

APIs:

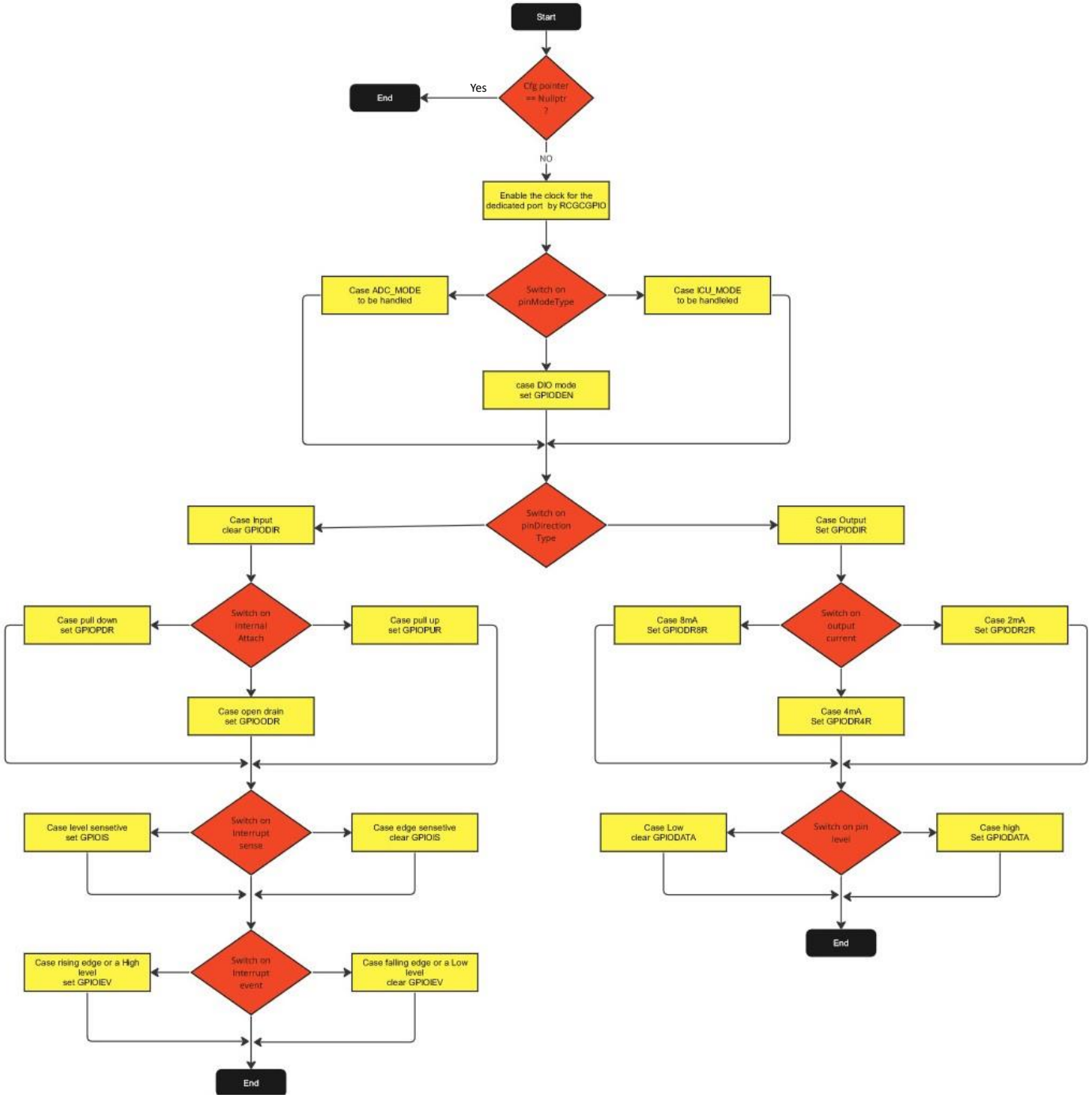
void ButtonInit(void);

enuButtonState_t ButtonGetState(ButtonChannel_t ButtonChannel, enuButtonAttach_t ButtonAttach);

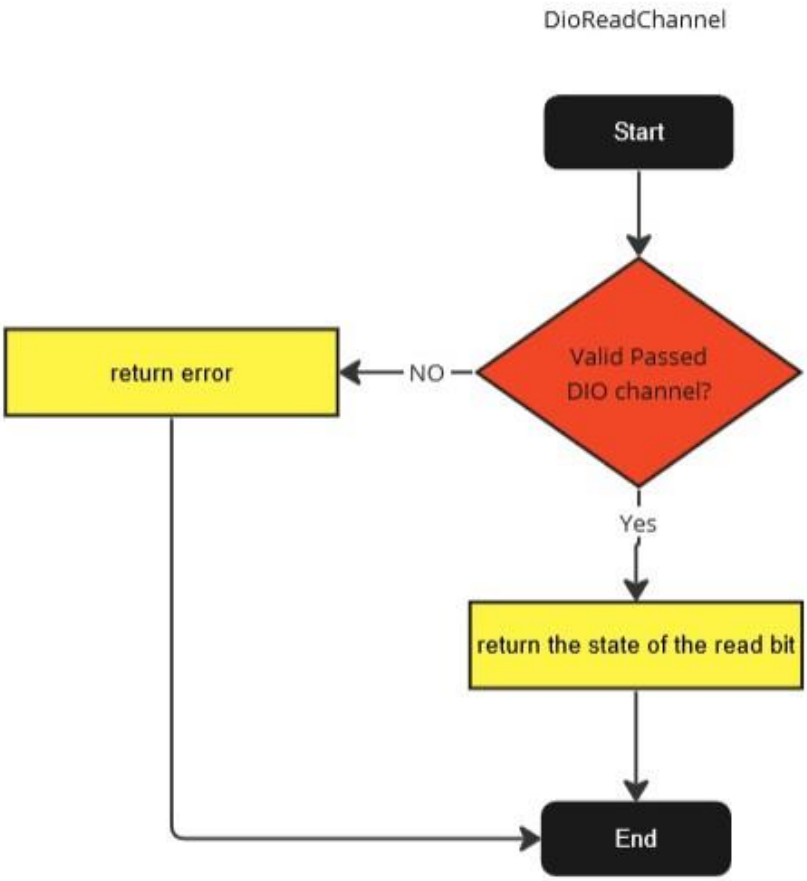
Flowcharts for Functions:

Port driver:

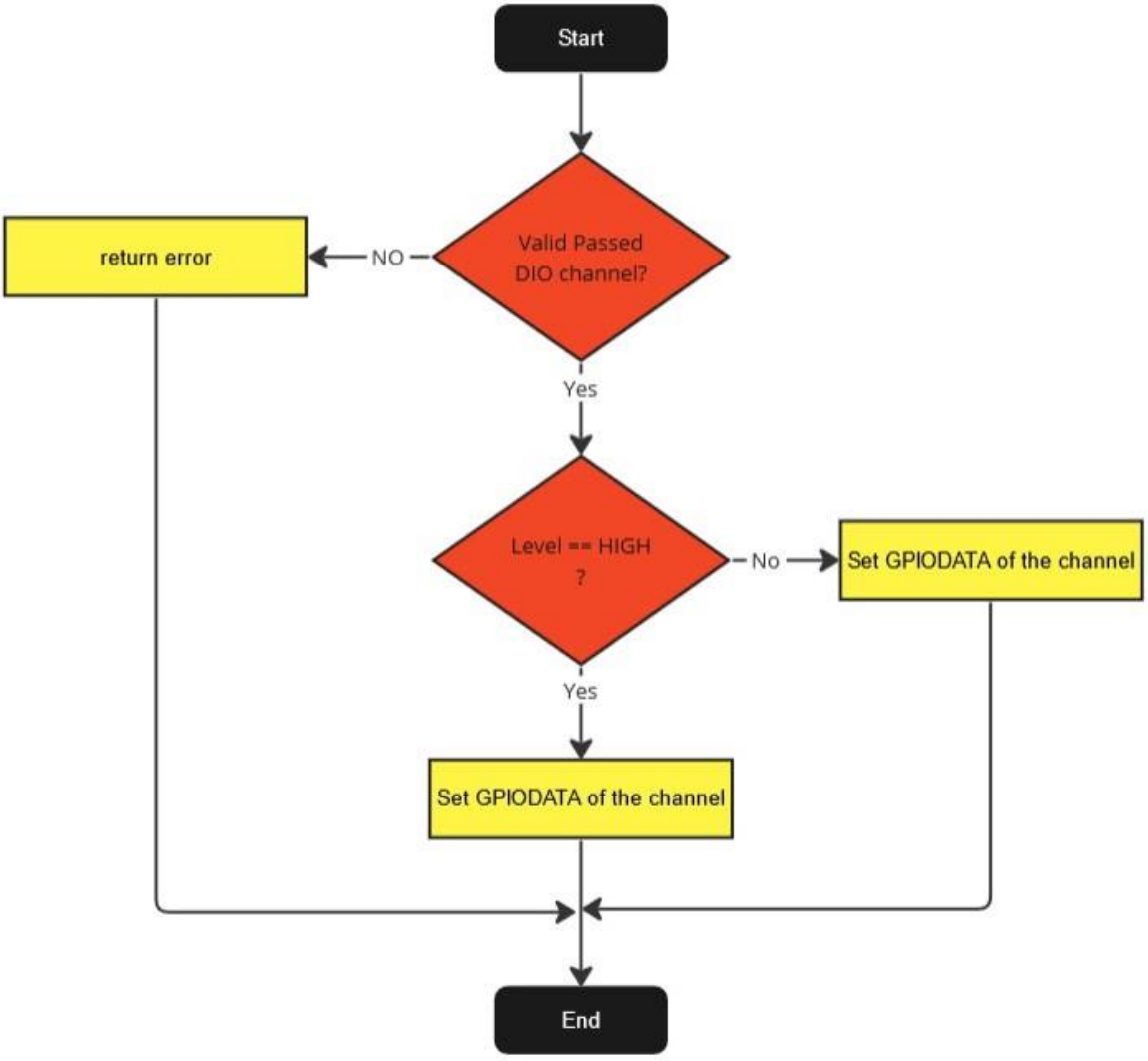
Port init :



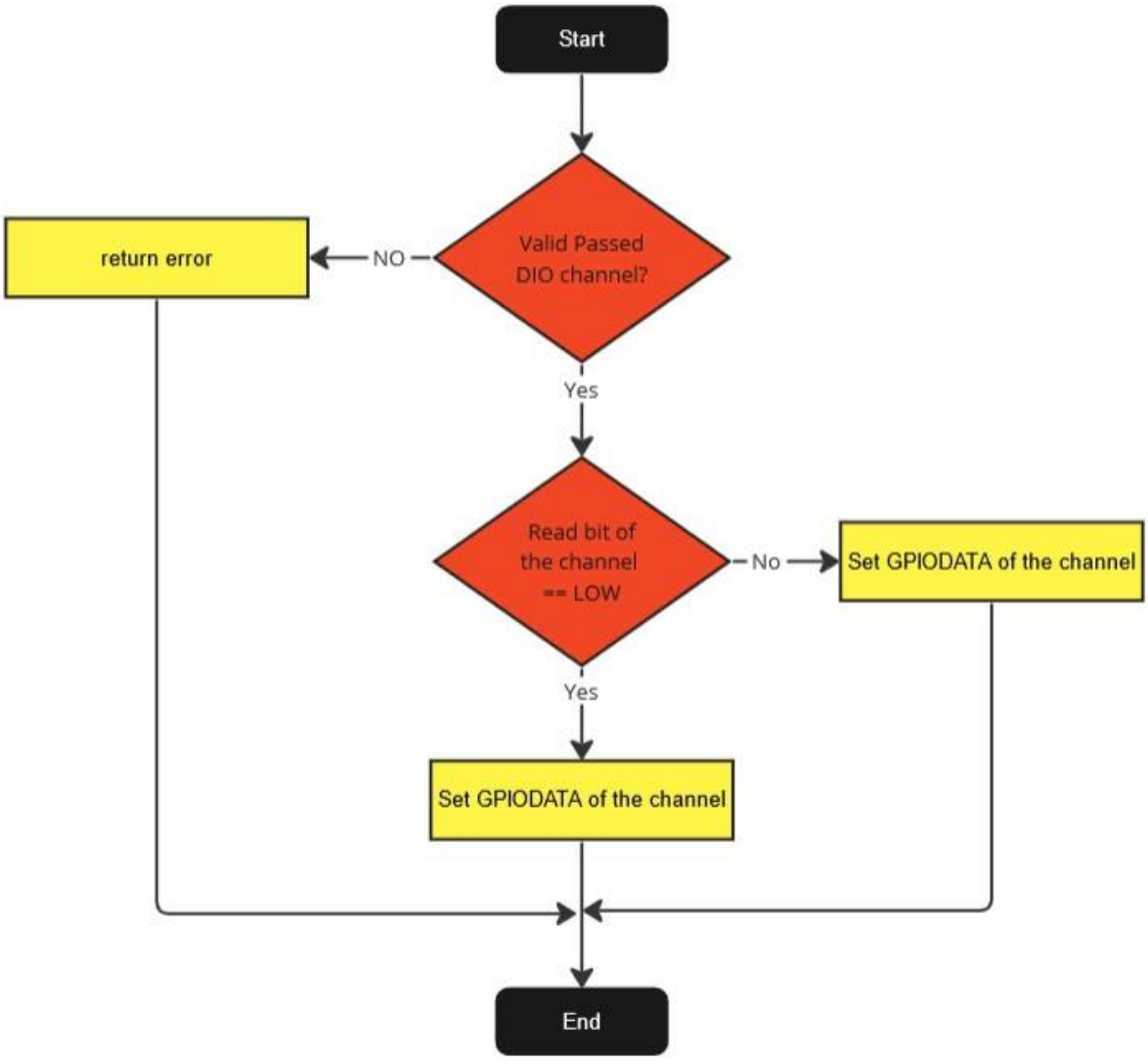
DIO driver:


DioReadChannel
Start
Valid Passed DIO channel?
NO
return error
Yes
return the state of the read bit
End

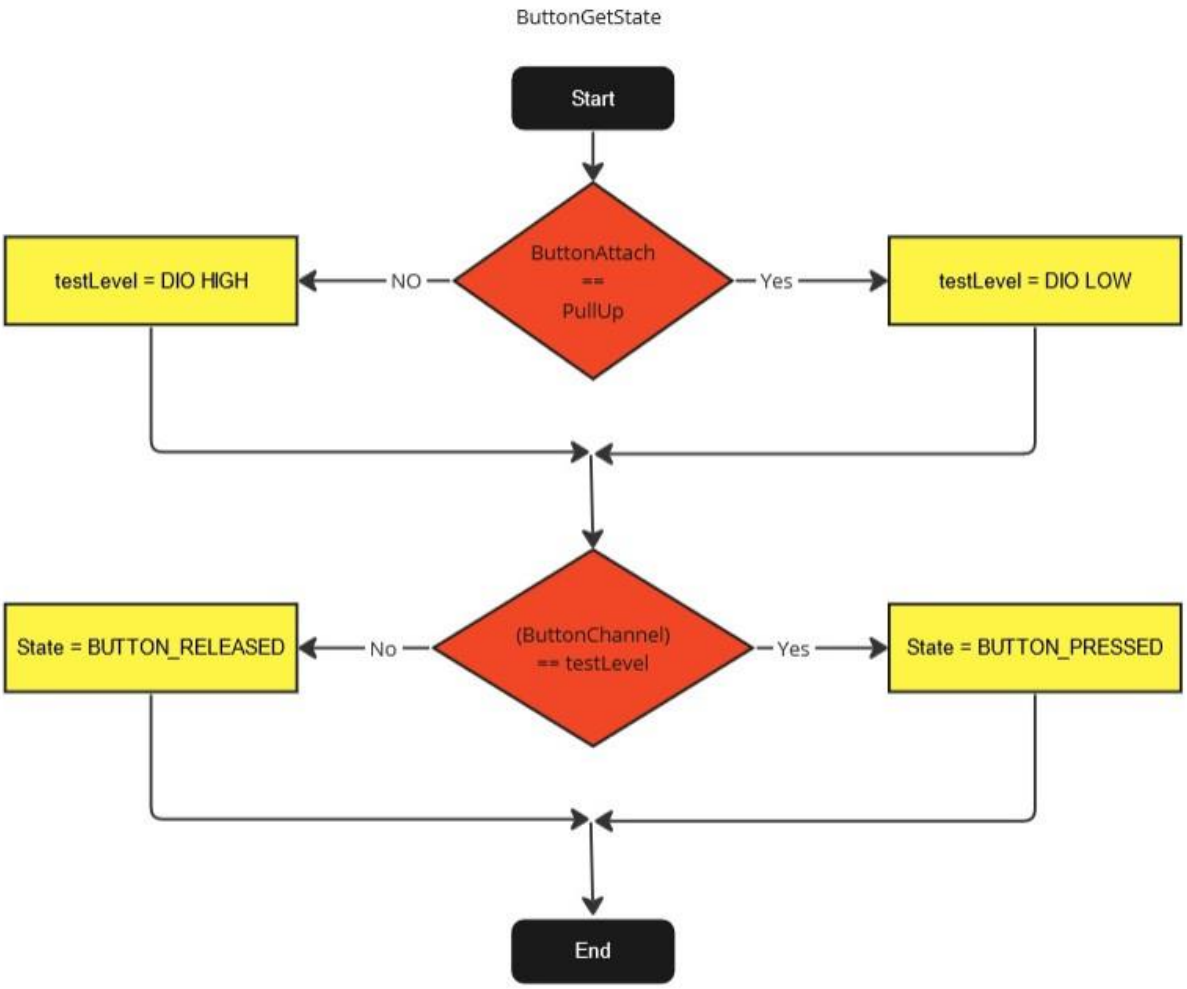
DioWriteChannel

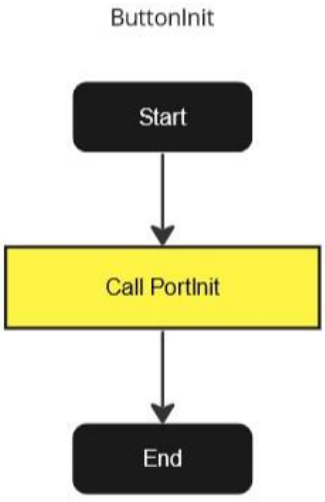


DioToggleChannel

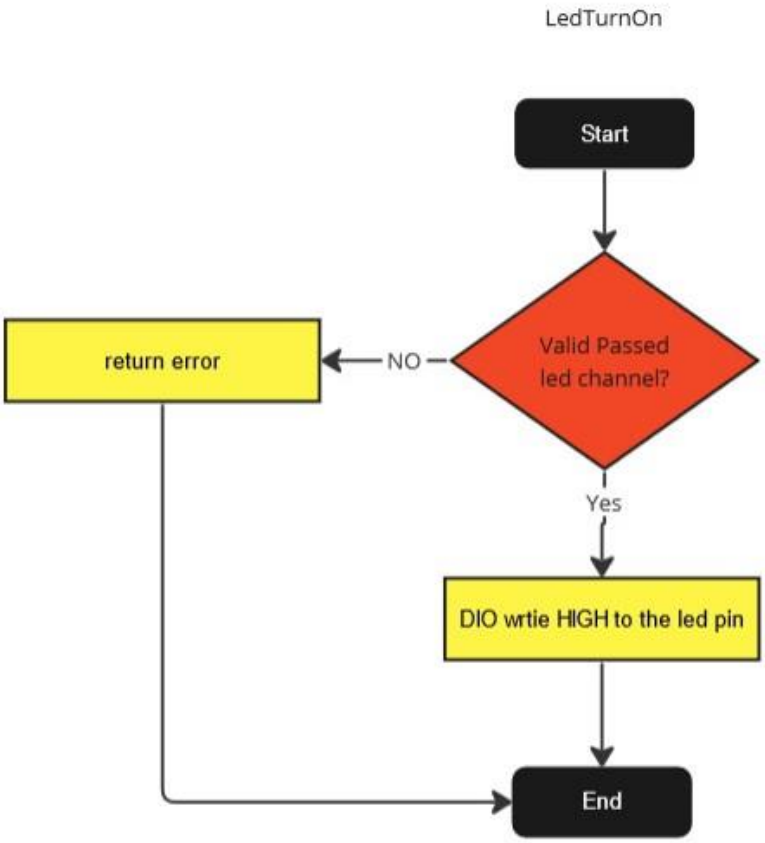


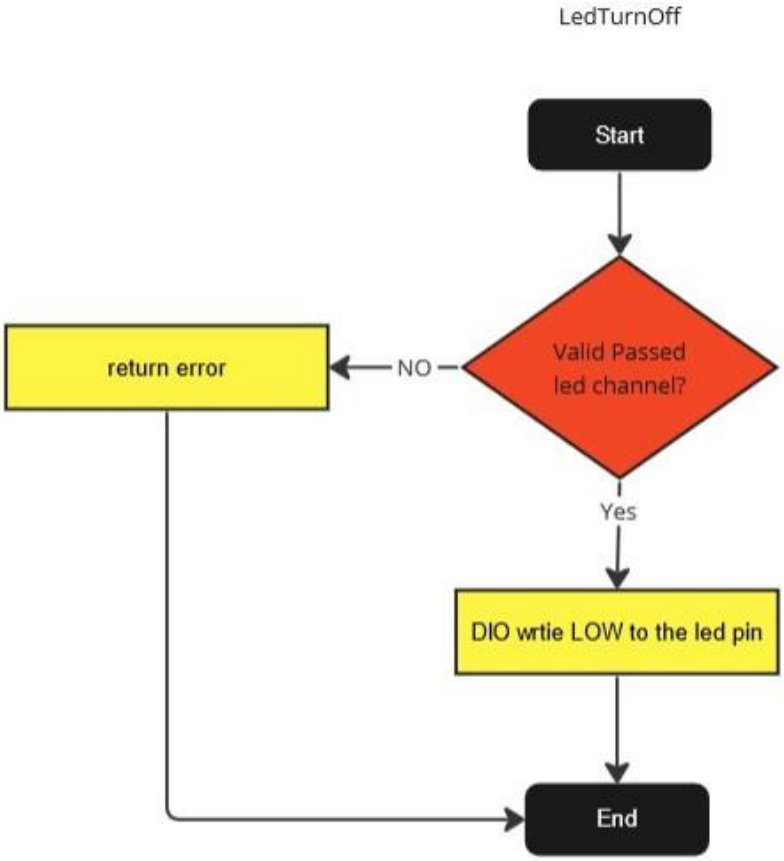
Button driver



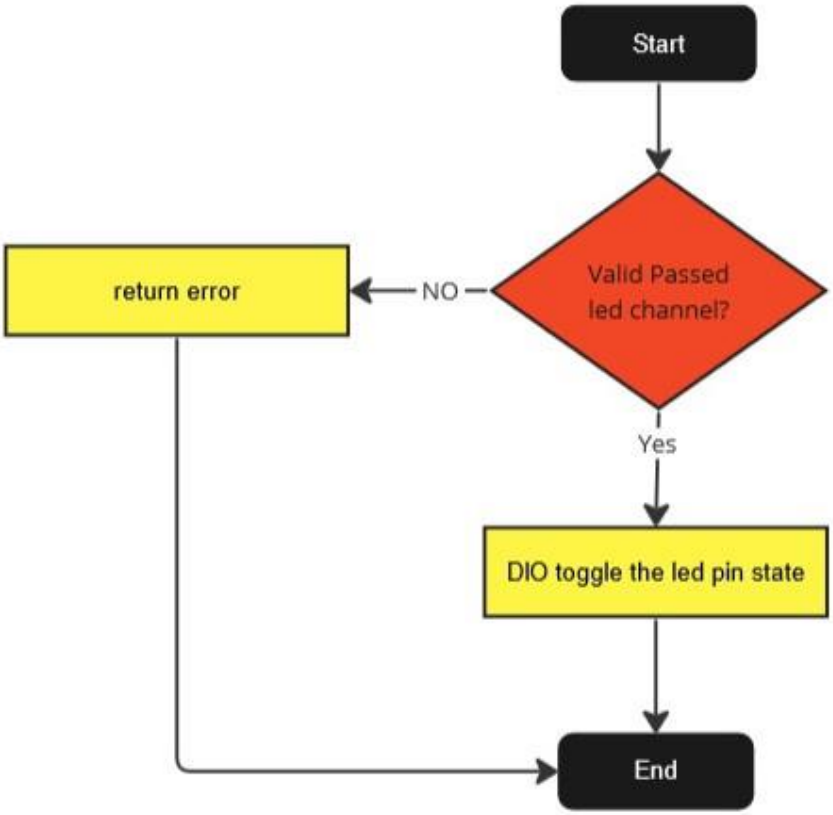


LED driver:

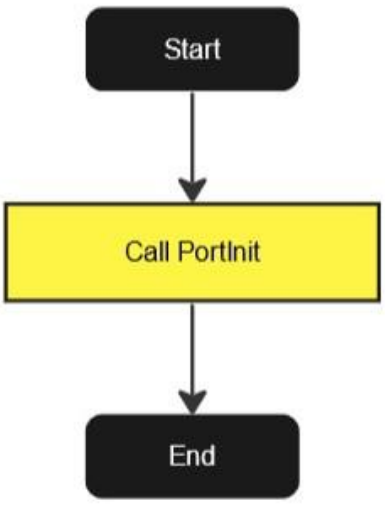

LedTurnOn
Start
Valid Passed
led channel?
NO
return error
Yes
DIO wrtie HIGH to the led pin
End


LedTurnOff
Start
Valid Passed
led channel?
NO
return error
Yes
DIO wrtie LOW to the led pin
End

LedToggle



LedInit



Precompiling and linking configurations :-

Port Lcfg:

```
const strPortConfig_t strPortConfig =
{
    /********************************** PORTA ***********************************/
    PORTA, PIN0, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTA, PIN1, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTA, PIN2, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTA, PIN3, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTA, PIN4, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTA, PIN5, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTA, PIN6, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTA, PIN7, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,

    /********************************** PORTB ***********************************/
    PORTB, PIN0, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTB, PIN1, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTB, PIN2, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTB, PIN3, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTB, PIN4, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTB, PIN5, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTB, PIN6, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTB, PIN7, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,

    /********************************** PORTC ***********************************/
    PORTC, PIN0, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTC, PIN1, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTC, PIN2, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTC, PIN3, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTC, PIN4, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTC, PIN5, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTC, PIN6, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
    PORTC, PIN7, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
```

```
/********************************** PORTD ***********************************/
PORTD, PIN0, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTD, PIN1, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTD, PIN2, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTD, PIN3, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTD, PIN4, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTD, PIN5, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTD, PIN6, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTD, PIN7, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,

/********************************** PORTE ***********************************/
PORTE, PIN0, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTE, PIN1, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTE, PIN2, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTE, PIN3, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTE, PIN4, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,
PORTE, PIN5, CHANNEL_DISABLED, INPUT, PIN_LEVEL_LOW, DIO_MODE, PULL_UP, DRIVE_2mA,Port_IntDisable,

/********************************** PORTF ***********************************/
PORTF, PIN0, CHANNEL_ENABLED, INPUT,     PIN_LEVEL_HIGH, DIO_MODE, PULL_UP,       DRIVE_2mA,Port_IntDisable,
PORTF, PIN1, CHANNEL_ENABLED, OUTPUT, PIN_LEVEL_LOW,  DIO_MODE, PULL_DOWN, DRIVE_2mA,Port_IntDisable,
PORTF, PIN2, CHANNEL_ENABLED, OUTPUT, PIN_LEVEL_LOW,  DIO_MODE, PULL_DOWN, DRIVE_2mA,Port_IntDisable,
PORTF, PIN3, CHANNEL_ENABLED, OUTPUT, PIN_LEVEL_LOW,  DIO_MODE, PULL_DOWN, DRIVE_2mA,Port_IntDisable,
PORTF, PIN4, CHANNEL_ENABLED, INPUT,     PIN_LEVEL_HIGH, DIO_MODE, PULL_UP,       DRIVE_2mA,Port_IntDisable,
};
```

DIO:

```
/* DIO Configured Port's ID  */
#define DIO_CONFIG_LED1_PORT                (enuDioPort_t)PORTF
#define DIO_CONFIG_LED2_PORT                (enuDioPort_t)PORTF
#define DIO_CONFIG_LED3_PORT                (enuDioPort_t)PORTF
#define DIO_CONFIG_SWITCH1_PORT             (enuDioPort_t)PORTF
#define DIO_CONFIG_SWITCH2_PORT             (enuDioPort_t)PORTF

/* DIO Configured Channel's ID */
#define DIO_CONFIG_LED1_CHANNEL             (enuDioPin_t)PIN1
#define DIO_CONFIG_LED2_CHANNEL             (enuDioPin_t)PIN2
#define DIO_CONFIG_LED3_CHANNEL             (enuDioPin_t)PIN3
#define DIO_CONFIG_SWITCH1_CHANNEL          (enuDioPin_t)PIN4
#define DIO_CONFIG_SWITCH2_CHANNEL          (enuDioPin_t)PIN0
```

DIO Lcfg :

```
const strDioConfig_t ConfigList =
{
 DIO_CONFIG_LED1_PORT, DIO_CONFIG_LED1_CHANNEL,          /* LED 1 @ PF1 */
 DIO_CONFIG_LED2_PORT, DIO_CONFIG_LED2_CHANNEL,          /* LED 2 @ PF2 */
 DIO_CONFIG_LED3_PORT, DIO_CONFIG_LED3_CHANNEL,          /* LED 3 @ PF3 */
 DIO_CONFIG_SWITCH1_PORT, DIO_CONFIG_SWITCH1_CHANNEL,    /* Switch 1 @ PF0 */
 DIO_CONFIG_SWITCH2_PORT, DIO_CONFIG_SWITCH2_CHANNEL     /* Switch 2 @ PF4 */
};
```