LAB: GPIO Digital InOut

Date: 2022/10/01

Author/Partner: 21800447 Jeahyun Oh / 21800805 SeungEung Hwang

Github: https://github.com/Ohjeahyun1/EC-jeahyun-447.git

Demo Video: https://youtu.be/fjXTYepgAjg

Introduction

In this lab, you are required to create a simple program that toggle multiple LEDs with a push-button input. Create HAL drivers for GPIO digital in and out control and use your library.

Requirement

Hardware

- MCU
  - NUCLEO-F401RE
- Actuator/Sensor/Others:
  - LEDs x 3
  - Resistor 330 ohm x 3, breadboard

Software

- Keil uVision, CMSIS, EC_HAL library

Problem 1: Create EC_HAL library

Procedure

Create the library directory `\repos\EC\lib\`.

ecRCC.h (provided)

```
void RCC_HSI_init(void);

void RCC_GPIOA_enable(void);

void RCC_GPIOB_enable(void);

void RCC_GPIOC_enable(void);
```

ecGPIO.h

```
void GPIO_init(GPIO_TypeDef *Port, int pin,  int mode);

void GPIO_write(GPIO_TypeDef *Port, int pin,  int output);

int  GPIO_read(GPIO_TypeDef *Port, int pin);

void GPIO_mode(GPIO_TypeDef* Port, int pin, int mode);

void GPIO_ospeed(GPIO_TypeDef* Port, int pin,  int speed);

void GPIO_otype(GPIO_TypeDef* Port, int pin,  int type);

void GPIO_pupd(GPIO_TypeDef* Port, int pin,  int pupd);
```

ecGPIO.c

```
//GPID port mode register
// Input(00), Output(01), AlterFunc(10), Analog(11)
void GPIO_mode(GPIO_TypeDef *Port, int pin, int mode){
    Port->MODER &= ~(3UL<<(2*pin));       // clear bit
    Port->MODER |= mode<<(2*pin);         // set bit
}
void GPIO_init(GPIO_TypeDef *Port, int pin, int mode){
    // mode  : Input(0), Output(1), AlterFunc(2), Analog(3)
    if (Port == GPIOA)
        RCC_GPIOA_enable();
    if (Port == GPIOB)
        RCC_GPIOB_enable();
    if (Port == GPIOC)
        RCC_GPIOC_enable();

    //[TO-DO] YOUR CODE GOES HERE
    // Make it for GPIOB, GPIOD..GPIOH
 /* if (Port == GPIOD)
        RCC_GPIOD_enable();
     if (Port == GPIOE)
        RCC_GPIOE_enable();
      if (Port == GPIOF)
        RCC_GPIOF_enable();
         if (Port == GPIOG)
        RCC_GPIOG_enable();
         if (Port == GPIOH)
        RCC_GPIOH_enable();*/
    // You can also make a more general function of
    // void RCC_GPIO_enable(GPIO_TypeDef *Port);

    GPIO_mode(Port, pin, mode);

}
//write the number of pin on the Port
void GPIO_write(GPIO_TypeDef *Port, int pin,  int output){
    if(output == 0)
      (Port ->ODR) &=  ~(1UL << pin);     // input 0 -> output 0
    else
        (Port ->ODR) |=  (1UL << pin);      // input 1 -> output 1
```

```
}
//read the number of pin on the Port
int  GPIO_read(GPIO_TypeDef *Port, int pin){
     return (Port->IDR) & (1UL << pin);   // read the port
}

//GPIO port output speed register
// Low speed(00), Medium speed(01), Fast speed(10), High speed(11)
void GPIO_ospeed(GPIO_TypeDef* Port, int pin,  int speed){
     Port->OSPEEDR &= ~(3UL<<(2*pin));        // clear bit
   Port->OSPEEDR |= speed<<(2*pin);         // set bit
}
//GPIO port output type register
//0: Output push-pull (reset state), 1: Output open-drain
void GPIO_otype(GPIO_TypeDef* Port, int pin,  int type){
    Port->OTYPER &= ~(1UL<<(pin));            // clear bit
    Port->OTYPER |= type<<(pin);                // set bit
    //Port->OTYPER &= ~((~type)<<(2*pin));
}
//GPIO port pull-up/pull-down register
//00: No pull-up, pull-down , 01: Pull-up ,10: Pull-down , 11: Reserved
void GPIO_pupdr(GPIO_TypeDef* Port, int pin,  int pupdr){
      Port->PUPDR &=    ~(3UL<<(2*pin));          // clear bit
        Port->PUPDR  |= pupdr<<(2*pin);         // set bit
}
```

Problem 2: Toggle LED with Button

Procedure

1. Create a new project under the directory `\repos\EC\LAB\`

- The project name is "**LAB_GPIO_DIO_LED".**
- Name the source file as "**LAB_GPIO_DIO_LED.c"**
- Use the example code provided here.

2. Include your library **ecGPIO.h, ecGPIO.c** in `\repos\EC\lib\`.

> You MUST write your name in the top of the source file, inside the comment section.

3. Toggle the LED by pushing button.

- Pushing button (LED ON), Pushing Button (LED OFF) and repeat

Configuration

Button (B1)	**LED**
Digital In	Digital Out
GPIOC, Pin 13	GPIOA, Pin 5
PULL-UP	Open-Drain, Pull-up, Medium Speed

Code

Your code goes here: https://github.com/Ohjeahyun1/EC-jeahyun-447/blob/67f62d748be890beada7c419256dfe85ad8bd35c/lab/LAB_GPIO_DIO_LED.c

https://github.com/Ohjeahyun1/EC-jeahyun-447/blob/67f62d748be890beada7c419256dfe85ad8bd35c/include/ecGPIO.c

Explain your source code with necessary comments.

Description with Code

- Description 1

```
//define the led pin number and button pin number
#define LED_PIN     5
#define BUTTON_PIN 13

void setup(void);

int main(void) {
    // Initialiization --------------------------------------------------------
    setup();
    // Inifinite Loop ---------------------------------------------------------
    //when button pressed LED toggle
    while(1){

    if( GPIO_read(GPIOC,BUTTON_PIN) == 0){ //when button pressed
        bittoggle(GPIOA,LED_PIN);          // bit toggle function
    }
      delay_ms(50);                    //delay for debouncing
    }
}


// Initialiization
void setup(void)
{
    RCC_HSI_init();
  SysTick_init();                          // for delay
    GPIO_init(GPIOC, BUTTON_PIN, INPUT);  // calls RCC_GPIOC_enable() and button
pin mode -> input
    GPIO_init(GPIOA, LED_PIN, OUTPUT);    // calls RCC_GPIOA_enable() and LED
pin mode -> output
    GPIO_pupdr(GPIOA, LED_PIN, EC_PU);    // GPIOA LED pin pupdr -> pull up
    GPIO_pupdr(GPIOC, BUTTON_PIN, EC_PU); // GPIOC button pin pupdr -> pull up
    GPIO_otype(GPIOA, LED_PIN, PP);       // GPIOA LED pin otype -> push-pull
    GPIO_ospeed(GPIOA,LED_PIN,SMED);      // GPIOA LED pin ospeed -> Medium
speed
}
```

- Description 2

```
//LED bit toggle function
void bittoggle(GPIO_TypeDef* Port,int pin){
        (Port->ODR) ^= (1UL << pin);       // use XOR Gate
}
```

Problem 3: Toggle LED with Button

Procedure

1. Create a new project under the directory `\repos\EC\LAB\`

- The project name is "**LAB_GPIO_DIO_multiLED".**
- Name the source file as "**LAB_GPIO_DIO_multiLED.c"**

> You MUST write your name in the top of the source file, inside the comment section.

1. Include your library **ecGPIO.h, ecGPIO.c** in `\repos\lib\`.
2. Connect 3 LEDs externally with necessary load resistors.

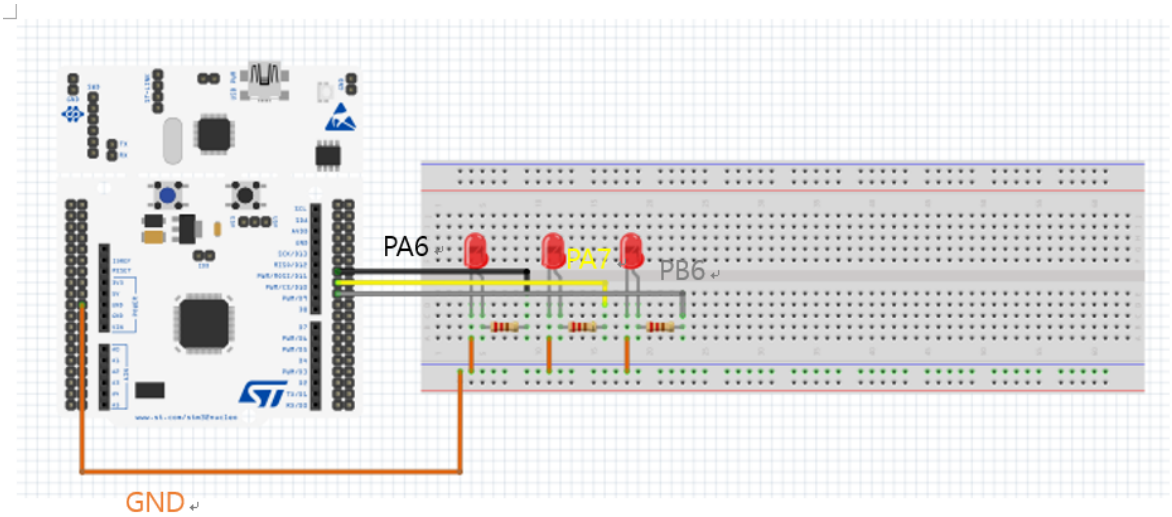
- As Button B1 is Pressed, light one LED at a time, in sequence.
- Example: LED0--> LED1--> ...LED3--> ...LED0....

Configuration

Button	**LED**
Digital In	Digital Out
GPIOC, Pin 13	PA5, PA6, PA7, PB6
PULL-UP	Push-Pull, Pull-up, Medium Speed

Circuit Diagram

Circuit diagram



Code

Your code goes here: https://github.com/Ohjeahyun1/EC-jeahyun-447/blob/67f62d748be890beada7c419256dfe85ad8bd35c/include/ecGPIO.c

https://github.com/Ohjeahyun1/EC-jeahyun-447/blob/67f62d748be890beada7c419256dfe85ad8bd35c/lab/LAB_GPIO_DIO_multiLED.c

Explain your source code with necessary comments.

Description with Code

- Description 1

```
// 4 LEDs lit sequentially
void multled(int state){
    // 4 LEDs HIGH,LOW state definition
    int muled[4][4] ={
            //row - led1,led2,led3,led4, col- state
                 //led1,led2,led3,led4
                      {1,0,0,0},          //state zero
                    {0,1,0,0},          //state one
                    {0,0,1,0},          //state two
                    {0,0,0,1}           //state three
  };
    //4 LEDS outpus
        GPIO_write(GPIOA,LED_PIN,muled[state][0]);
            GPIO_write(GPIOA,6,muled[state][1]);
            GPIO_write(GPIOA,7,muled[state][2]);
            GPIO_write(GPIOB,6,muled[state][3]);
}
```

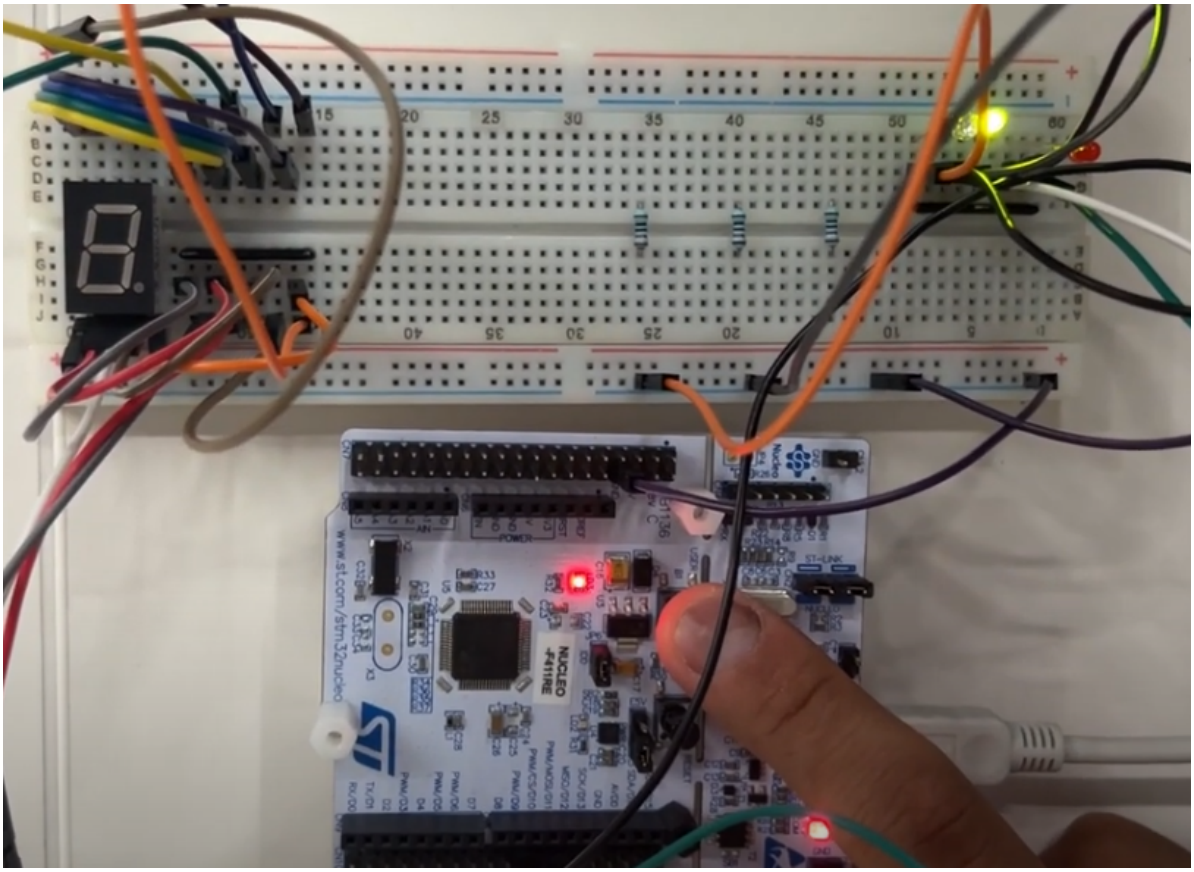
- Description 2

```
//define the led pin number and button pin number
#define LED_PIN     5
#define BUTTON_PIN 13

void setup(void);

int main(void) {
    // Initialiization --------------------------------------------------------
    setup();
    int state = 0;                          //state initalization
    // Inifinite Loop ---------------------------------------------------------
    while(1){
    //LED output according to state
        multled(state);
        if( GPIO_read(GPIOC,BUTTON_PIN) == 0){ //when button pressed
                state++;                          //state update
            if(state == 4){                     //There are only 0,1,2,3 states
            state =0;                           //state reset
           }
        }
        delay_ms(50);                          //delay for debouncing
    }
}


// Initialiization
void setup(void)
{
    RCC_HSI_init();
    SysTick_init();                     // for delay
```

```
    GPIO_init(GPIOC, BUTTON_PIN, INPUT);  // calls RCC_GPIOC_enable() and button
pin mode -> input
    GPIO_init(GPIOA, LED_PIN, OUTPUT);    // calls RCC_GPIOA_enable()   and LED
pin mode -> output
    GPIO_init(GPIOA, 6, OUTPUT);          // calls RCC_GPIOA_enable()   and 6
pin mode -> output
    GPIO_init(GPIOA, 7, OUTPUT);          // calls RCC_GPIOA_enable()   and 7
pin mode -> output
    GPIO_init(GPIOB, 6, OUTPUT);          // calls RCC_GPIOB_enable() and 6 pin
mode -> output

    GPIO_pupdr(GPIOC, BUTTON_PIN, EC_PU); // GPIOC button pin pupdr -> pull up
    GPIO_pupdr(GPIOA, LED_PIN, EC_PU);    // GPIOA LED pin pupdr -> pull up
    GPIO_pupdr(GPIOA, 6, EC_PU);          // GPIOA pin 6 pupdr -> pull up
    GPIO_pupdr(GPIOA, 7, EC_PU);          // GPIOA pin 7 pupdr -> pull up
    GPIO_pupdr(GPIOB, 6, EC_PU);          // GPIOB pin 6 pupdr -> pull up

    GPIO_otype(GPIOA, LED_PIN, PP);       // GPIOA LED pin otype -> push-pull
    GPIO_otype(GPIOA, 6, PP);             // GPIOA 6 pin otype -> push-pull
    GPIO_otype(GPIOA, 7, PP);             // GPIOA 7 pin otype -> push-pull
    GPIO_otype(GPIOB, 6, PP);             // GPIOB 6 pin otype -> push-pull

    GPIO_ospeed(GPIOA,LED_PIN,SMED);      // GPIOA LED pin ospeed -> Medium
speed
    GPIO_ospeed(GPIOA,6,SMED);            // GPIOA 6 pin ospeed -> Medium speed
    GPIO_ospeed(GPIOA,7,SMED);            // GPIOA 7 pin ospeed -> Medium speed
    GPIO_ospeed(GPIOB,6,SMED);            // GPIOB 6 pin ospeed -> Medium speed
}
```

Results

Experiment images and results



Pressing the button changes the state that determines the status of the 4 LEDs. The LEDs light up sequentially accordingly.

Demo Video

Discussion

1.Find out a typical solution for software debouncing and hardware debouncing.

- What is bouncing?

The bounce of the switch refers to a phenomenon in which the metal contact vibrates as soon as the mechanical switch is pressed or released, and the contact point sticks and falls several times at a high speed.

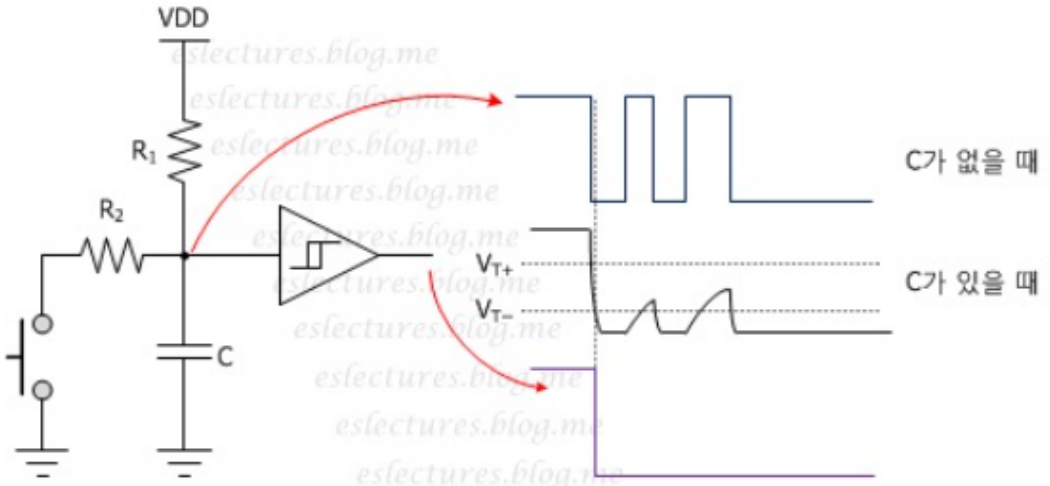
- Problems with bounce?

The microcontroller incorrectly determines that the switch has been pressed several times.

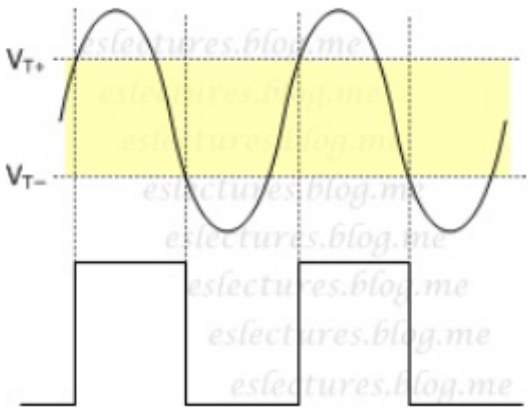
- Solution

Hardware debouncing

Use schmitt trigger and capacitor, resister



Use a capacitor to change the output of the LOW and HIGH straight lines into a curve as shown in the figure above. The schmitt trigger has a certain threshold value, so the values of LOW and HIGH are output according to the criteria. The figure below is an example.

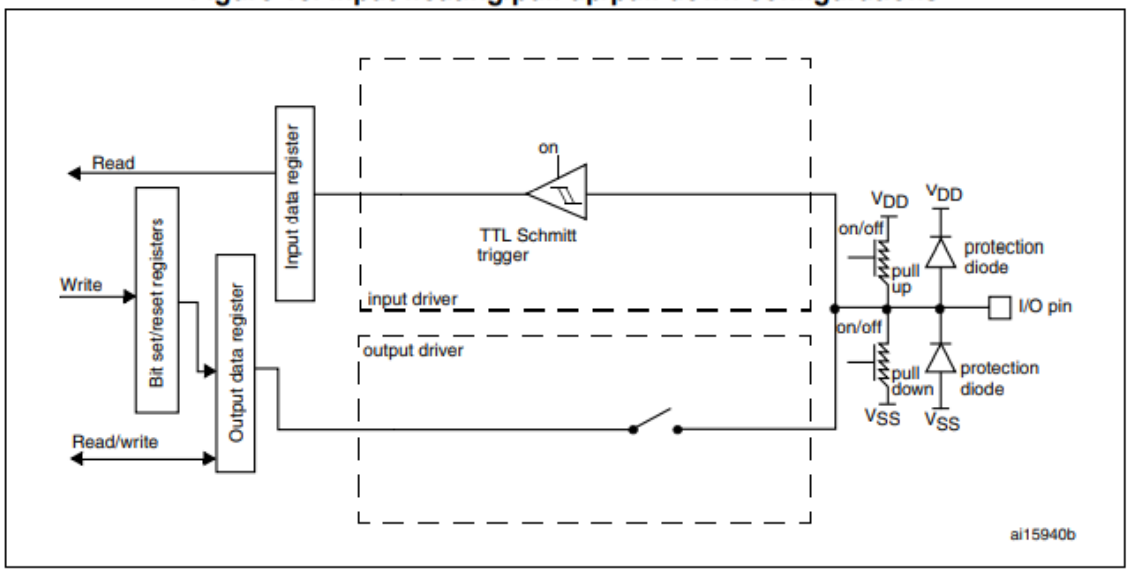


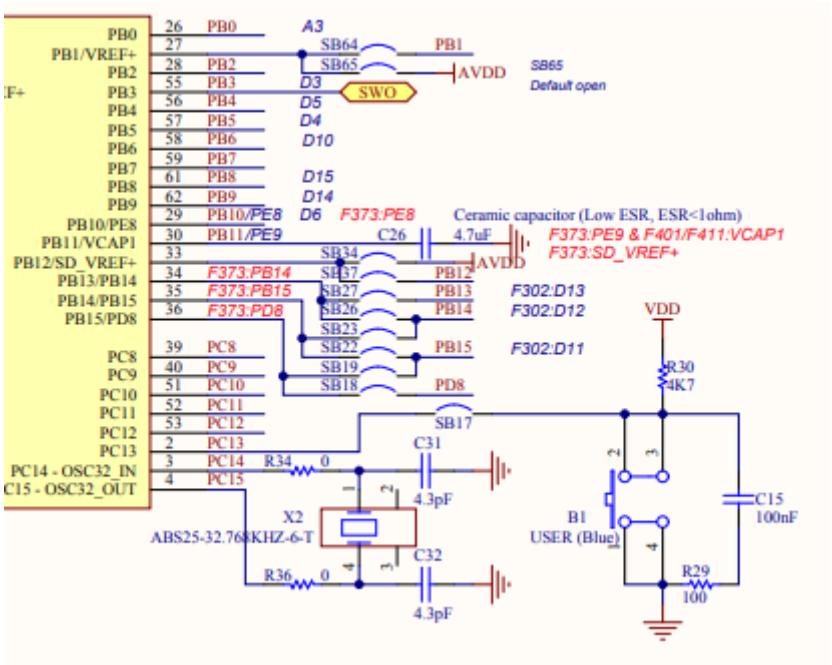
Software debouncing

It is a delay method that waits for a certain period of time to stabilize the sensor.

2.What method of debouncing did this NUCLEO board used for the push-button(B1)?

Figure 18. Input floating/pull up/pull down configurations





According to the stm32 reference, debouncing will be possible because there is a schmitt trigger in the input and an RC circuit in PC13, which is a button pin.

Personal study

GPIO port output type register

Difference between Open-drain and Push Pull

otype	Push-pull	Open-drain
output	MCU,IC internal power sources	External power supply

The open drain allows MCUs or ICs with low operating voltages to reliably control circuits that are higher or lower than their own operating voltages.

Reference

Complete list of all references used (github, blog, paper, etc)

http://choavrweb.blogspot.com/p/debounc-keypad.html

https://m.blog.naver.com/eslectures/80137812929

https://m.blog.naver.com/ansdbtls4067/220863464855

https://velog.io/@audgus47/Push-Pull%ED%91%B8%EC%89%AC-%ED%92%80-%EA%B3%BC-%EC%98%A4%ED%94%88-%EB%93%9C%EB%A0%88%EC%9D%B8Open-drain

stm32 reference.pdf

https://424033796-files.gitbook.io/~/files/v0/b/gitbook-x-prod.appspot.com/o/spaces%2F-MgmrEstOHxu62gXxq1t%2Fuploads%2Fgit-blob-be61a47040270d53ec6fe06231b76cc8c72c39f0%2FMB1136%5B1%5D.pdf?alt=media

Troubleshooting

(Option) You can write Troubleshooting section

Our group tried to experiment by connecting four LEDs externally, but it didn't work well. The PA5 was connected to the internal LED, so only the internal LED worked, and the external LED connected to the pin did not work. I think we need to use another pin for this experiment. Therefore, I think the configuration needs to be changed.