<u>การทดลองที่ 2</u> การใช้งาน GPIO

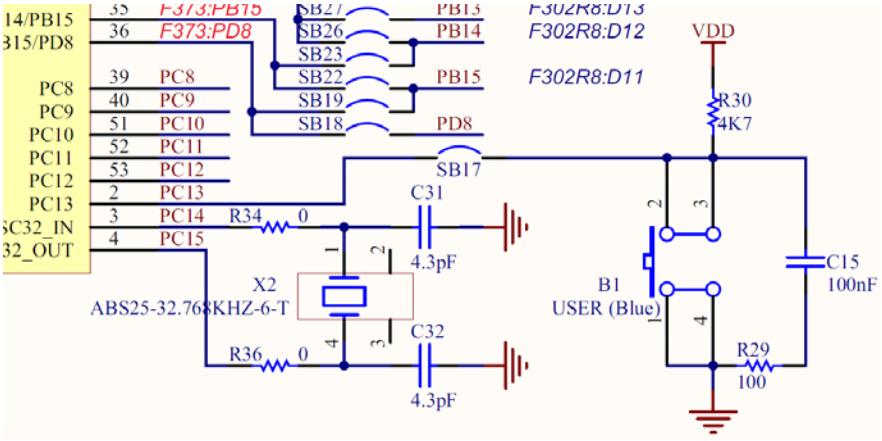
วัตถุประสงค์

1 เพื่อให้นักศึกษาเขียนโปรแกรมควบคุมการทำงานของ GPIO ได้

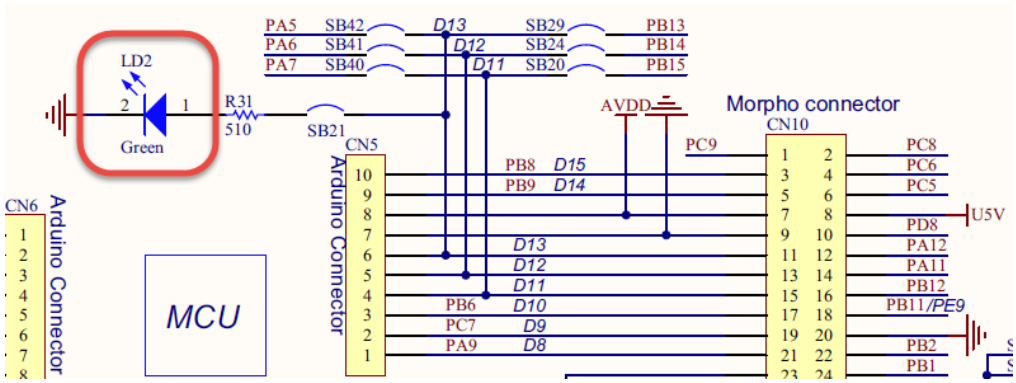
2 เพื่อให้นักศึกษาเขียนโปรแกรมเพื่อรับอินพุตจากสวิตช์บนบอร์ดได้

1. Switch และ LED

บนบอร์ด Nucleo-F411RE มีสวิตช์ 1 ปุ่มและ LED 1 ดวงที่ผู้ใช้สามารถใช้งานได้ ได้แก่สวิตช์ B1 สีน้ำเงินซึ่งเชื่อมต่อกับขา PC13 เมื่อกดจะมีสถานะลอจิก 0 เมื่อปล่อยจะมีสถานะลอจิก 1 ดังรูปที่ 1.1 และ LD2 ซึ่งเป็น LED สีเขียวที่จะติดเมื่อป้อนลอจิก 1 ทางขา PA5 ดังรูปที่ 1.2

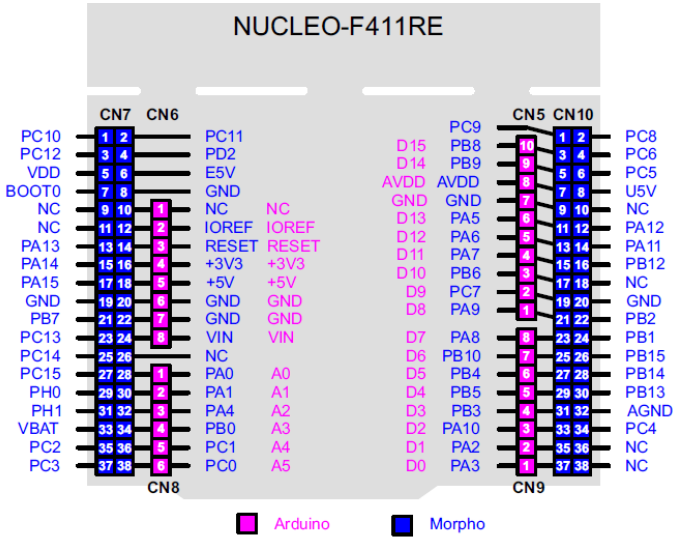


รูปที่ 1.1 การเชื่อมต่อสวิตช์เข้ากับขา PC13



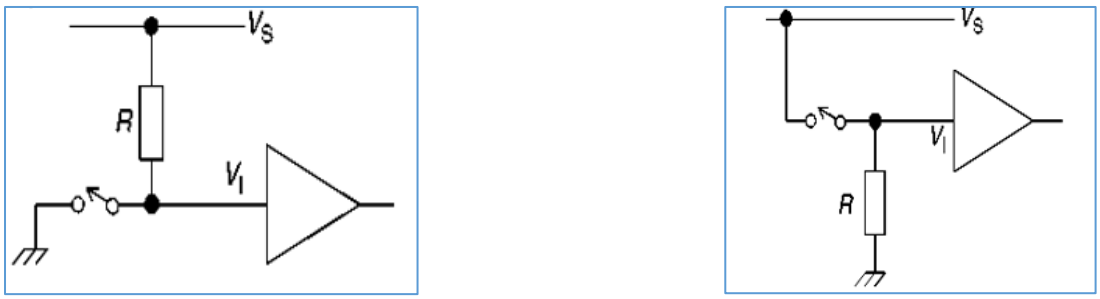
รูปที่ 1.2 การเชื่อมต่อ LED เข้ากับขา PA5

นอกจากสวิตช์และ LED ที่มีมาให้บนบอร์ดอยู่แล้ว ผู้ใช้สามารถเชื่อมต่อสวิตช์และ LED เพิ่มเติมได้ทางตัวเชื่อมต่อบนบอร์ด ซึ่งเชื่อมต่อกับขา GPIO ของไมโครคอนโทรลเลอร์ดังรูปที่ 1.3



รูปที่ 1.3 การเชื่อมต่อขา GPIO ของไมโครคอนโทรลเลอร์ไปยังตัวเชื่อมต่อตำแหน่งต่างๆ

การเชื่อมต่อสวิตช์ต้องต่อตัวต้านทานเพื่อจำกัดกระแส โดยต่อได้ 2 แบบ คือ แบบ pull-up ดังรูปที่ 1.4 (a) และแบบ pull-down ดังรูปที่ 1.4 (b)



(a) (b)

รูปที่ 1.4 การต่อสวิตช์แบบต่างๆ

(a) Pull-up resistor

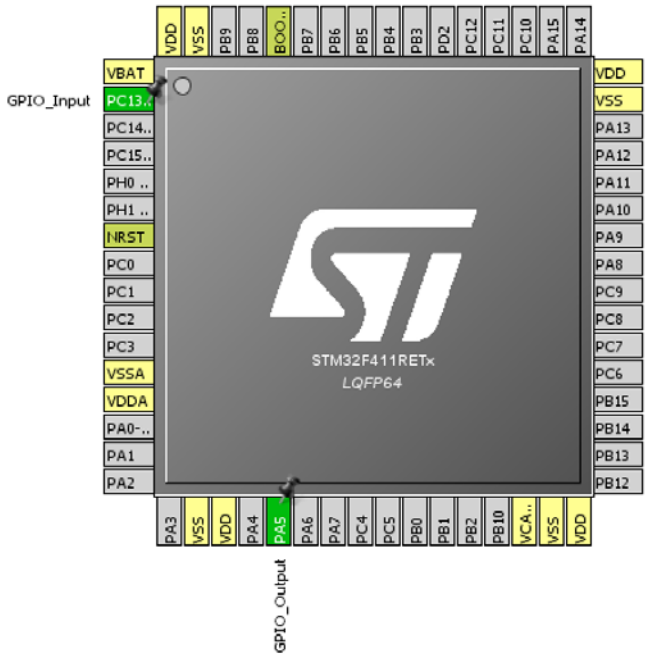
(b) Pull-down resistor

2. การอ่านค่าอินพุต

การเริ่มต้นใช้งาน GPIO นั้นต้องมีการจ่ายสัญญาณนาฬิกาไปยังพอร์ตที่ต้องการใช้งาน พร้อมทั้งกำหนดโหมดการทำงานให้กับแต่ละขาว่าจะให้เป็นอินพุตหรือเอาต์พุตประเภทใด อีกทั้งยังสามารถกำหนดความเร็ว (speed) เมื่อทำหน้าที่เป็นเอาต์พุตได้ว่าจะให้มีความเร็ว LOW, MEDIUM, HIGH หรือ VERY HIGH

หากต้องการเขียนโปรแกรมเพื่อตรวจสอบว่าเมื่อกดสวิตช์ B1 ทำให้ LD2 ติดนาน 1 วินาทีแล้วดับ มีขั้นตอนดังนี้

1) ใช้โปรแกรม STM32CubeMX สร้างโปรเจ็คขึ้นมาเช่นเดียวกับการทดลองที่ 1 แต่ใช้ขา GPIO ดังรูปที่ 2.1
2) พิมพ์โค้ดใน while loop ดังรูปที่ 2.2



(a)

GPIO Mode and Configuration

Configuration

Group By Peripherals

GPIO

Pin Name	Signal on Pin	GPIO output level	GPIO mode	GPIO Pull-up/P...	Maximum outpu...	User Label	Modified
PA5	n/a	Low	Output Push Pull	No pull-up and ...	Low		☐
PC13-ANTI_TA...	n/a	n/a	Input mode	No pull-up and ...	n/a		☐

(b)

รูปที่ 2.1 การกำหนด GPIO ที่ต้องการใช้งาน

(a) Pinout Configuration

(b) GPIO Configuration

```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
    //Check whether switch B1 is pressed
    if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
    {
      //Turn on LD2 at PA5
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);

      //Delay 1,000 millisecond
      HAL_Delay(1000);

      //Turn off LD2 at PA5
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);

      //Delay 1,000 millisecond
      HAL_Delay(1000);
    }
  }
  /* USER CODE END 3 */

}
```

รูปที่ 2.2 โค้ดใน while loop ในฟังก์ชัน main

3. อธิบายการทำงาน

ฟังก์ชัน MX_GPIO_init ( )

- เป็นฟังก์ชันที่โปรแกรม STM32CubeMX สร้างขึ้นมา เพื่อตั้งค่า GPIO บนไมโครคอนโทรลเลอร์ให้สอดคล้องกับที่กำหนดไว้ในโปรแกรม
- เริ่มต้นด้วยการ Enable สัญญาณนาฬิกาให้ GPIOC (สำหรับสวิตช์ B1) และ GPIOA (สำหรับ LED)

```
__HAL_RCC_GPIOC_CLK_ENABLE();
__HAL_RCC_GPIOA_CLK_ENABLE();
```

- กำหนดให้ PA5 ซึ่งเชื่อมต่อกับ LD2 เป็นเอาต์พุต Push Pull แบบ High

```
GPIO_InitStruct.Pin       = GPIO_PIN_5
GPIO_InitStruct.Mode      = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull      = GPIO_NOPULL;
GPIO_InitStruct.Speed     = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
```

- กำหนดให้ PC13 ซึ่งเชื่อมต่ออยู่กับสวิตช์ B1 เป็นอินพุตแบบ Floating

```
GPIO_InitStruct.Pin = GPIO_PIN_13;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_NOPULL;
HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
```

ฟังก์ชัน main ( )

- เริ่มต้นการทำงานด้วยฟังก์ชัน HAL_Init() , SystemClock_Config() และ MX_GPIO_Init();
- อ่านสถานะของสวิตช์ B1 โดยใช้คำสั่ง

  HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)
- ตรวจสอบเงื่อนไขเพื่อดูว่าสวิตช์ถูกกดหรือไม่ ด้วยคำสั่ง

  HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET

สามารถศึกษารายละเอียดฟังก์ชันที่สามารถใช้งานกับโมดูล GPIO เพิ่มเติมได้จากไฟล์เอกสาร HAL Drivers

4. การทดลอง

1. จงเขียนโปแกรมที่มีการทำงานดังนี้ **เมื่อสวิตช์ B1 ถูกกด ให้ LD2 ติดสว่างนาน 1 วินาที แล้วดับ**

2. จงต่อสวิตช์และ LED ภายนอก โดยเชื่อมต่อสวิตช์ที่**ขา PA0** แบบ pull down ดังรูปที่ 1.4 (b) และเชื่อมต่อ LED ที่**ขา PC0** โดยให้ย้อนกลับไปแก้ไขการตั้งค่าในโปรแกรม STM32CubeMX จากนั้น generate code ใหม่ แล้วจึงแก้ไขโค้ดให้สามารถทำงานได้ตามการทดลองข้อ 1

ห้ามสร้างโปรเจ็คใหม่ และควรปิดโปรแกรม Keil ก่อนการ generate code ซ้ำ

3. จงเขียนคำสั่งเพื่อตรวจสอบว่า LED ในการทดลองข้อ 2 **ติด**อยู่หรือไม่

```
HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_0) == GPIO_PIN_SET
```

4. จงต่อ LED เพิ่มอีก 2 ดวงเรียงกัน โดยให้เชื่อมต่อที่ขา PC1 ถึง PC2 แล้วเขียนโปรแกรมให้ทำงานดังตารางที่ 4.1

ตารางที่ 4.1 การแสดงผลด้วย LED เมื่อกดสวิตช์ต่างๆ

Switch	การทำงานของ LED
B1	เมื่อกดสวิตช์ B1 ให้ LED ติดค้างทีละดวง และเมื่อกดสวิตช์ B1 อีกครั้งให้ LED ดวงถัดไปติดแทน ถ้าหากไม่มี LED ดวงใดติดอยู่เลยให้เริ่มต้นที่ LED0 ที่ขา PC0 แล้วขยับไปยัง LED1 ที่ขา PC1 จนกระทั่งถึง LED2 ที่ขา PC2 แล้ววนไปที่ LED0 อีกครั้ง **เริ่มต้น** กดสวิตช์ B1 -> LED0 ติดค้าง -> กดสวิตช์ B1 -> LED0 ดับ และ LED1 ติดค้าง -> กดสวิตช์ B1 -> LED1 ดับ และ LED2 ติดค้าง . . .
PA0 Switch	LED จะไล่ติดแล้วดับทีละดวงเริ่มตั้งแต่ LED2 จนถึง LED0 โดยอัตโนมัติ 1 รอบ กำหนดให้ใช้ delay time 0.5 วินาที

ข้อ 1

```
if(HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET){
        HAL_GPIO_WritePin(GPIOC,GPIO_PIN_5,GPIO_PIN_SET);
        HAL_Delay(1000);
        HAL_GPIO_WritePin(GPIOC,GPIO_PIN_5,GPIO_PIN_RESET);
        HAL_Delay(1000);
}
```

ข้อ 2

```
if(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) == GPIO_PIN_SET){
        HAL_GPIO_WritePin(GPIOC,GPIO_PIN_5,GPIO_PIN_SET);
        HAL_Delay(1000);
        HAL_GPIO_WritePin(GPIOC,GPIO_PIN_5,GPIO_PIN_RESET);
        HAL_Delay(1000);
}
```

ข้อ 3 ไม่ต้องคอมไพล์

```
int LED_ON = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_0) == GPIO_PIN_SET
```

ข้อ 4

```
/* USER CODE BEGIN WHILE */
GPIO_PinState StatePin[3][4] = {   {GPIO_PIN_SET, GPIO_PIN_RESET, GPIO_PIN_RESET, GPIO_PIN_RESET},
                                   {GPIO_PIN_RESET, GPIO_PIN_SET, GPIO_PIN_RESET, GPIO_PIN_RESET},
                                   {GPIO_PIN_RESET, GPIO_PIN_RESET, GPIO_PIN_SET, GPIO_PIN_RESET} };
int b = 0;
while(1)
{
        if(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) == GPIO_PIN_SET){
                HAL_Delay(200);
                for(int a=0;a<5;a++){ // loop 4 cycle.
                        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_0, StatePin[2][a]);
                        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_1, StatePin[1][a]);
                        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_2, StatePin[0][a]);
                        HAL_Delay(500);
                }
        } else if(HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET){
                HAL_Delay(200);
                HAL_GPIO_WritePin(GPIOC, GPIO_PIN_0, StatePin[0][b]);
                HAL_GPIO_WritePin(GPIOC, GPIO_PIN_1, StatePin[1][b]);
                HAL_GPIO_WritePin(GPIOC, GPIO_PIN_2, StatePin[2][b]);
                if (b++ == 2) b=0;
        }
}
```

ใบตรวจการทดลองที่ 2

KU CSC Embedded System

วัน/เดือน/ปี _____ **กลุ่มที่** _____

1. รหัสนิสิต _____ ชื่อ-นามสกุล _____

2. รหัสนิสิต _____ ชื่อ-นามสกุล _____

3. รหัสนิสิต _____ ชื่อ-นามสกุล _____

ลายเซ็นผู้ตรวจ

การทดลองข้อ 2&3 ผู้ตรวจ _____ วันที่ตรวจ ☐ W ☐ W+1

การทดลองข้อ 4 ผู้ตรวจ _____ วันที่ตรวจ ☐ W ☐ W+1

คำถามท้ายการทดลอง

1. จงเขียนคำสั่งเพียง 1 คำสั่งที่ทำให้ LED0 LED1 และ LED2 ที่ขา PC0, PC1 และ PC2 ติดพร้อมกัน

HAL_GPIO_WritePin(GPIOC, (uint16_t)0x0007, GPIO_PIN_SET);

2. ภายหลังการต่อวงจรและการตั้งค่า GPIO ในการทดลองข้อ 4 หากโค้ดใน while loop เป็นดังโค้ดด้านล่าง แสดงว่าโปรแกรมนี้ทำงานอย่างไร

```
uint8_t n;

while (1)
  {
      for (n=0; n<=2; n++)
        {
          GPIOC -> BSRR = 0x00070000;
          HAL_Delay(500);

          GPIOC -> BSRR = 0x00000004 >> n;
          HAL_Delay(500);
        }
}
```

```
//      [16 - 32]              [0-15]
// 0000 0000 0000 0111 0000 0000 0000 0000
// ^      RESET      ^ ^       SET       ^

//     HEX          BINARY
// 0x00000004 == .... 0000 0100
// IF n = 0; .... 0000 0100 -> PC2
// IF n = 1; .... 0000 0010 -> PC1
// IF n = 2; .... 0000 0001 -> PC0
```

```
// 0000 0000 0000 0000 0000 0000 0000 0100
```

โปรแกรมจะวนลูปทั้งหมด 3 รอบ (0x00000004 >> n หมายถึง ขยับบิตไปด้านขวา n ครั้ง)

ในทุกๆรอบในส่วนโค้ด GPIOC -> BSRR = 0x00070000 ; เซ็ตค่า RESET ให้ PC0, PC1, PC2 แล้ว ดีเลย์ 500 ms

รอบที่ 1 n=0 จะได้ GPIOC -> BSRR = 0x00000004 >> 0 ; ทำให้โค้ดส่วนนี้ไปเซ็ตค่า SET ให้ PC2 แล้ว ดีเลย์ 500 ms

รอบที่ 2 n=1 จะได้ GPIOC -> BSRR = 0x00000004 >> 1 ; ทำให้โค้ดส่วนนี้ไปเซ็ตค่า SET ให้ PC1 แล้ว ดีเลย์ 500 ms

รอบที่ 3 n=2 จะได้ GPIOC -> BSRR = 0x00000004 >> 2 ; ทำให้โค้ดส่วนนี้ไปเซ็ตค่า SET ให้ PC0 แล้ว ดีเลย์ 500 ms