

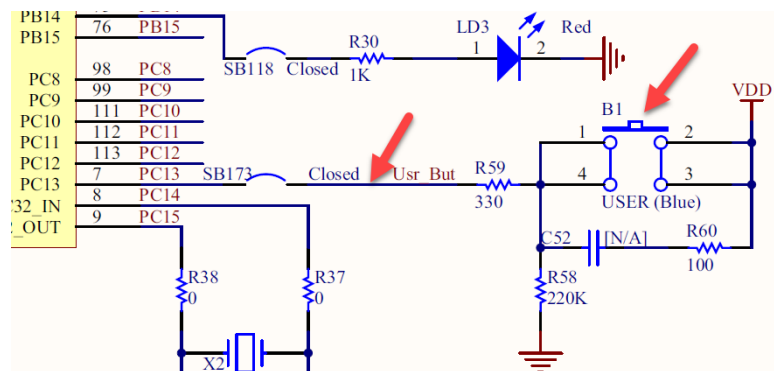
การทดลองที่ 2 การใช้ GPIO

วัตถุประสงค์

1. เพื่อให้นักศึกษาเขียนโปรแกรมควบคุมการทำงานของ GPIO ได้
2. เพื่อให้นักศึกษาเขียนโปรแกรมเพื่อรับอินพุตจากสวิตช์บนบอร์ดและสวิตช์ภายนอกได้

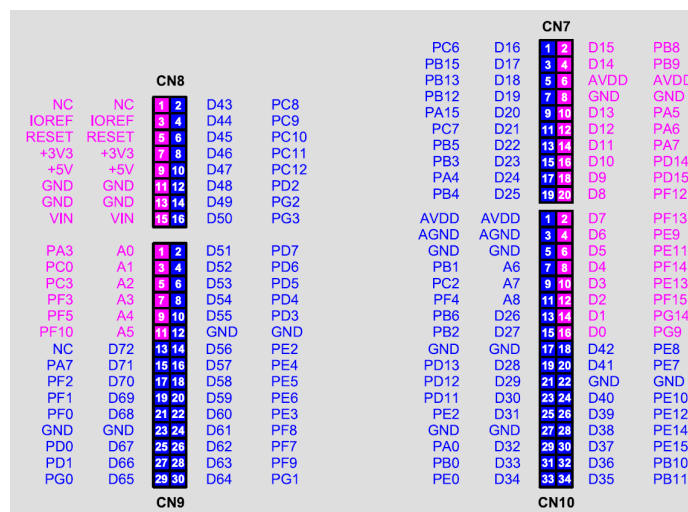
1. Switch และ LED

บนบอร์ด Nucleo-F767ZI มีสวิตช์ 1 ปุ่ม และ LED 3 ดวง ที่ผู้ใช้สามารถใช้งานได้ สวิตช์ B1 สีน้ำเงินซึ่งเชื่อมต่อกับขา PC13 เมื่อกดจะมีสถานะลอจิก 1 เมื่อปล่อยจะมีสถานะลอจิก 0 ดังรูปที่ 1.1 ส่วน LED 3 ดวง ได้แก่ LD1 LD2 และ LD3 เชื่อมต่อกับขา PB0 PB7 และ PB14 ตามลำดับ โดยจะติดสว่างเมื่อป้อนลอจิก 1



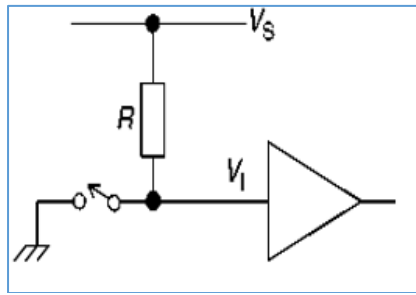
รูปที่ 1.1 การเชื่อมต่อสวิตช์เข้ากับขา PC13

นอกจากสวิตช์และ LED ที่มีมาให้บนบอร์ดอยู่แล้ว ผู้ใช้สามารถเชื่อมต่อสวิตช์และ LED เพิ่มเติมได้ทางตัวเชื่อมต่อ (Connector) บนบอร์ด ซึ่งเชื่อมต่อกับขา GPIO ของไมโครคอนโทรลเลอร์ดังรูปที่ 1.2

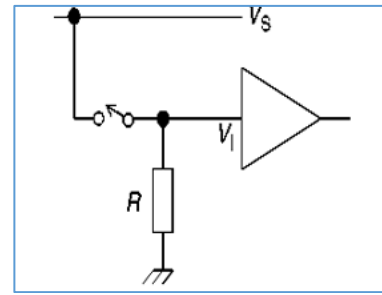


รูปที่ 1.2 การเชื่อมต่อขา GPIO ของไมโครคอนโทรลเลอร์ไปยังตัวเชื่อมต่อตำแหน่งต่างๆ

การเชื่อมต่อสวิตช์ต้องต่อตัวต้านทานเพื่อจำกัดกระแส โดยต่อได้ 2 แบบ คือ แบบ pull-up ดังรูปที่ 1.3 (a) และแบบ pull-down ดังรูปที่ 1.3 (b)



(a)



(b)

รูปที่ 1.3 การต่อสวิตช์แบบต่างๆ

(a) Pull-up resistor

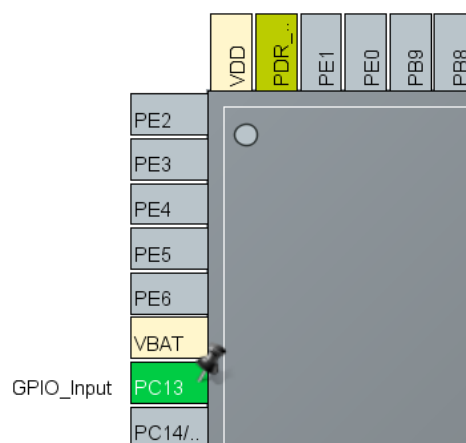
(b) Pull-down resistor

2. การอ่านค่าอินพุต

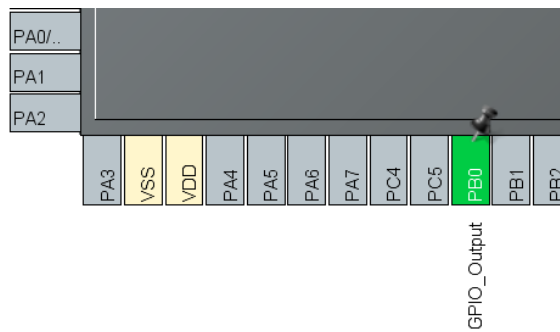
การเริ่มต้นใช้งาน GPIO นั้นต้องมีการจ่ายสัญญาณนาฬิกาไปยังพอร์ตที่ต้องการใช้งาน พร้อมทั้งกำหนดโหมดการทำงานให้กับแต่ละขาว่าจะให้เป็นอินพุตหรือเอาต์พุตประเภทใด อีกทั้งยังสามารถกำหนดความเร็ว (speed) เมื่อทำหน้าที่เป็นเอาต์พุตได้ว่าจะให้มีความเร็ว LOW, MEDIUM, HIGH หรือ VERY HIGH

หากต้องการเขียนโปรแกรมเพื่อตรวจสอบว่าเมื่อกดสวิตช์ B1 ทำให้ LD1 ติดนาน 1 วินาทีแล้วดับ มีขั้นตอนดังนี้

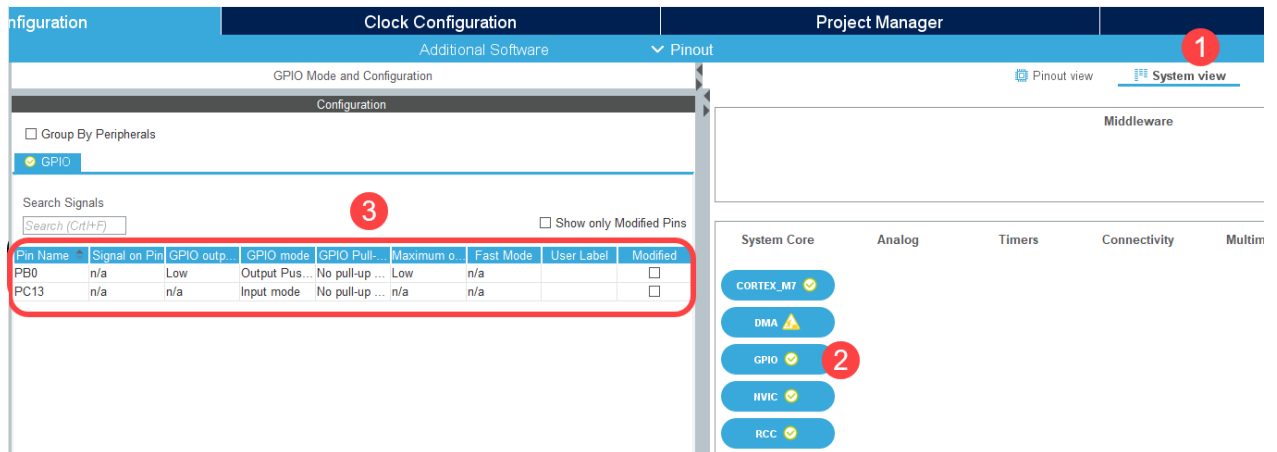
- 1) ใช้โปรแกรม STM32CubeMX สร้างโปรเจกต์ขึ้นมาเช่นเดียวกับการทดลองที่ 1 แต่ใช้ขา GPIO ดังรูปที่ 2.1
- 2) พิมพ์โค้ดใน while loop ดังรูปที่ 2.2



(a)



(b)



(c)

รูปที่ 2.1 การกำหนด GPIO ที่ต้องการใช้งาน

- (a) Pinout Configuration
- (b) Pinout Configuration
- (c) GPIO Configuration

```

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

    /* USER CODE BEGIN 3 */
    //Check whether button is pressed
    if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_SET)
    {
        //Turn on LD1 at PB0
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);

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

        //Turn off LD1 at PB0
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_RESET);
    }
}
/* USER CODE END 3 */

```

รูปที่ 2.2 โค้ดใน while loop ในฟังก์ชัน main

3. อธิบายการทำงาน

ฟังก์ชัน MX_GPIO_init ()

- เป็นฟังก์ชันที่โปรแกรม STM32CubeMX สร้างขึ้นมา เพื่อดำเนินการตั้งค่า GPIO บนไมโครคอนโทรลเลอร์ให้สอดคล้องกับที่กำหนดไว้ในโปรแกรม

- เริ่มต้นด้วยการ Enable สัญญาณนาฬิกาให้ GPIOC (สำหรับสวิตช์ B1) และ GPIOB (สำหรับ LED)

```
__HAL_RCC_GPIOC_CLK_ENABLE();  
__HAL_RCC_GPIOB_CLK_ENABLE();
```

- 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);
```

- กำหนดให้ PB0 ซึ่งเชื่อมต่อกับ LD1 เป็นเอาต์พุต Push Pull ที่มีความเร็วแบบ Low

```
GPIO_InitStruct.Pin = GPIO_PIN_0;  
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;  
GPIO_InitStruct.Pull = GPIO_NOPULL;  
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;  
HAL_GPIO_Init(GPIOB, &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

```
if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_SET)
```

สามารถศึกษารายละเอียดฟังก์ชันที่สามารถใช้งานกับโมดูล GPIO เพิ่มเติมได้จากไฟล์เอกสาร HAL Driver

4. การทดลอง

1. จงเขียนโปรแกรมที่มีการทำงานดังนี้ เมื่อกดสวิตช์ B1 ถูกกด ให้ LED LD1 ติดสว่างนาน 1 วินาที แล้วดับ
2. จงแก้ไขโปรแกรมในการทดลองข้อ 1 ให้ทำงานได้ดังนี้ ต่อสวิตช์จากวงจรภายนอกตามรูปที่ 1.3 (a) เชื่อมต่อกับไมโครคอนโทรลเลอร์ที่ขา PA0 แทนการใช้สวิตช์ B1 และต่อ LED ภายนอกที่ขา PC8 แทนการใช้ LD2 จากนั้นให้เปิดโปรแกรม STM32CubeMX เพื่อแก้ไขการตั้งค่าให้สอดคล้องกับการทดลองในข้อนี้ สั่ง Generate Code ใหม่ แล้วแก้ไขโค้ดใน while loop ให้ถูกต้อง
3. จงเขียนคำสั่งเพื่อตรวจสอบว่า LED ในการทดลองข้อ 2 ติดอยู่หรือไม่

4. จงต่อ LED 8 ดวงเข้ากับไมโครคอนโทรลเลอร์ โดยให้เชื่อมต่อที่ขา D0 ถึง D7 แล้วเขียนโปรแกรมให้ทำงานดังตารางที่ 4.1 โดยเริ่มต้นไม่ให้มี LED ดวงไหนติดอยู่เลย

ตารางที่ 4.1 การแสดงผลด้วย LED เมื่อกดสวิตช์ต่างๆ

Switch	การทำงานของ LED
B1	<p>เมื่อกดสวิตช์ B1 ให้ LED ติดค้างทีละดวง และเมื่อกดสวิตช์ B1 อีกครั้งให้ LED ดวงถัดไปติดแทน ถ้าหากไม่มี LED ดวงใดติดอยู่เลยให้เริ่มต้นที่ LED ที่ขา D0 แล้วขยับไปยัง LED ที่ขา D1 จนกระทั่งถึง LED ที่ขา D7 แล้ววนไปที่ LED0 อีกครั้ง</p> <p>เริ่มต้น</p> <p>กดสวิตช์ B1 -> LED_D0 ติดค้าง</p> <p>กดสวิตช์ B1 -> LED_D0 ดับ และ LED_D1 ติดค้าง</p> <p>กดสวิตช์ B1 -> LED_D1 ดับ และ LED_D2 ติดค้าง</p> <p>...</p> <p>กดสวิตช์ B1 -> LED_D7 ดับ และ LED_D0 ติดค้าง</p>
PA0 Switch	<p>LED จะไล่ติดแล้วดับทีละดวงเริ่มตั้งแต่ LED ดวงที่ติดอยู่ วนไปทาง LED_D0 แล้วหยุดที่ LED ดวงเดิมที่ติดค้างอยู่</p> <p>หากไม่มี LED ติดอยู่เลย ให้เริ่มที่ LED_D7 วนไปทาง LED_D0 แล้วให้ทุกดวงดับ</p> <p>ใช้ delay time 0.3 วินาที</p>

ใบตรวจการทดลองที่ 2

Microcontroller Project 2568

วัน/เดือน/ปี _____ กลุ่มที่ _____

1. รหัสนักศึกษา _____ ชื่อ-นามสกุล _____

2. รหัสนักศึกษา _____ ชื่อ-นามสกุล _____

3. รหัสนักศึกษา _____ 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 PC8, PC9 และ PC10 ติดพร้อมกัน

.....
.....

2. หากเชื่อมต่อ LED จำนวน 3 ดวง ที่ขา PC10 PC9 และ PC8 หากโค้ดใน while loop เป็นดังโค้ดด้านล่าง แสดงว่าโปรแกรมนี้ทำงานอย่างไร

```
uint8_t n;
while (1)
{
    for (n=0; n<=2; n++)
    {
        GPIOC -> BSRR = 0x07000000;
        HAL_Delay(500);

        GPIOC -> BSRR = 0x00000400 >> n;
        HAL_Delay(500);
    }
}
```

.....
.....
.....
.....