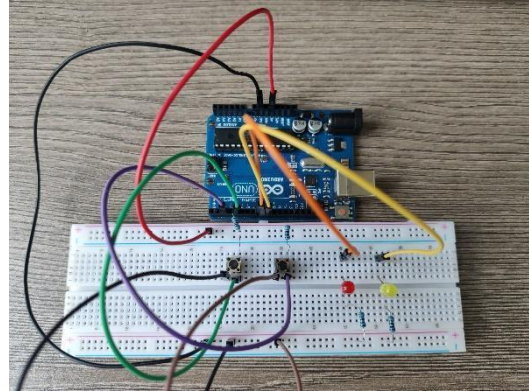
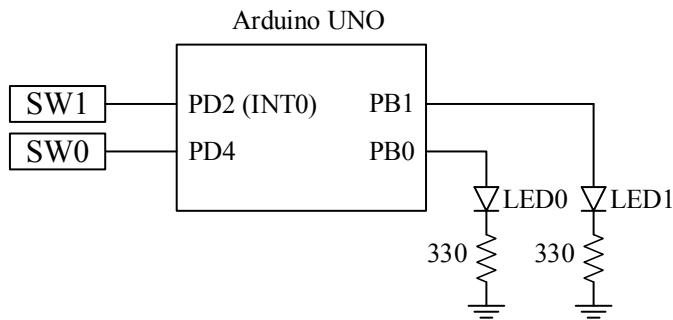


Exercise 4:

(Using the external interrupt INT0). Similar to Exercise 2 in Lab 7, we will use the external interrupt INT0 but in a C program. We have connected a circuit as shown below.



We would like to write a C program to do the following tasks.

- Task1 – The microcontroller monitors the status of the switch SW0: if we press the switch SW0, LED0 is on; if we do not press the switch SW0, LED0 is off.
- Task2 – The microcontroller monitors the status of the switch SW1: if we press and release the switch SW1, the LED1 will be toggled (on $\leftrightarrow$ off or off $\leftrightarrow$ on).

As a result, the microcontroller will do Task1 in the main program and do Task2 by using the external interrupt INT0 (falling-edge trigger). Recall the following related registers are set up as follows (from Exercise 2 in Lab 7):

- EIMSK = 0x01
- EICRA = 0x02

Answer the following questions.

- a) Similar to the structure of the Assembly program shown in Exercise 2 in Lab 7, we have the following C program to fulfill the tasks required above. Please fill in the blanks.

```
1  #include <avr/io.h>
2  #include <avr/interrupt.h>
3
4  void PIN_SETUP() {           //A function to set up the pin modes
5      DDRB |= (1<<0);          //Set PB0 as an output -> LED0
6      PORTB &= ~(1<<0);        //Set PB0 = 0
7      DDRB |= (1<<1);          //Set PB1 as an output -> LED1
8      PORTB &= ~(1<<1);        //Set PB1 = 0
9      DDRD &= ~(1<<4);          //Set PD4 as an input -> SW
10     PORTD |= (1<<4);           //Pull-up resistor for PD4
11     PORTD |= (1<<2);           //Pull-up resistor for PD2(INT0)
12 }
13
```

```

14 int main() {
15     PIN_SETUP();           //Call the PIN_SETUP function
16     EIMSK = 0x01;          //Enable the INT0 Interrupt
17     EICRA = 0x02;          //Set the falling-edge trigger
18     sei();                 //Enable the global interrupt
19     while (1) {
20         if ( ~PIND & (1<<4) ) { //Check PD4=0?
21             PORTB |= (1<<0);    //Yes, PB0=1 -> LED0 on
22         } else {
23             PORTB &= ~(1<<0);   //No, PB0=0 -> LED0 off
24         }
25     }
26     return (0);
27 }

28
29 ISR (INT0_vect) {          //ISR of INT0 interrupt
30     PORTB ^= (1<<1);       //and toggle PB1
31 }

```

- b) Upload your C program to your Arduino UNO board. Take a video to demonstrate your result. Name it “Ex4” and submit to the Google Classroom.

https://drive.google.com/drive/folders/1zQ05sgDaikkTC0O4dgxmKRmrSGcYfmPN?usp=share_link

```

1  #include <avr/io.h>
2  #include <avr/interrupt.h>
3
4  void PIN_SETUP() {
5      DDRB |= (1<<0); // Set PB0 as an output -> LED0
6      PORTB &= ~(1<<0); // Set PB0 = 0
7      DDRB |= (1<<1); // Set PB1 as an output -> LED1
8      PORTB &= ~(1<<1); // Set PB1 = 0
9      DDRD &= ~(1<<4); // Set PD4 as an input -> SW
10     PORTD |= (1<<4); // Pull-up resistor for PD4
11     PORTD |= (1<<2); // Pull-up resistor for PD2(INT0)
12 }
13
14
15 int main() {
16     PIN_SETUP();
17     EIMSK = 0x01; // Enable the INT0 Interrupt
18     EICRA = 0x02; // Set the falling-edge trigger
19     sei(); // Enable the global interrupt
20     while (1) {
21         if (~PIND & (1<<4)) { // Check PD4 = 0
22             PORTB |= (1<<0); // Yes, PB0=1 -> LED0 on
23         } else {
24             PORTB &= ~(1<<0); // No, PB0=0 -> LED0 off
25         }
26     }
27     return (0);
28 }
29
30 ISR(INT0_vect) { // ISR of INT0 interrupt
31     PORTB ^= (1<<1); // toggle PB1
32 }

```