Embedded System Design with MCU and FPGA

LAB02 0858706 黃品嫚

Goal

Let us know more about GPIO, Timer, and Clock.

Problems

1. Describe at least two methods that you used to blink the LED.

- There are three methods which could be used to blink the LED.

① **Waiting**

- Use delay() function, such as Fig 1., to blink the LED. (Here is the link of the experiment.)
- **delay()** is a blocking function.

```
1  void setup() {
2    DDRB |= (1 << 5);
3  }
4
5  void loop() {
6    PORTB |= (1<<5);
7    delay(1000);
8    PORTB &= ~(1<<5);
9    delay(1000);
10 }
```

Fig 1. using delay() to blink LED

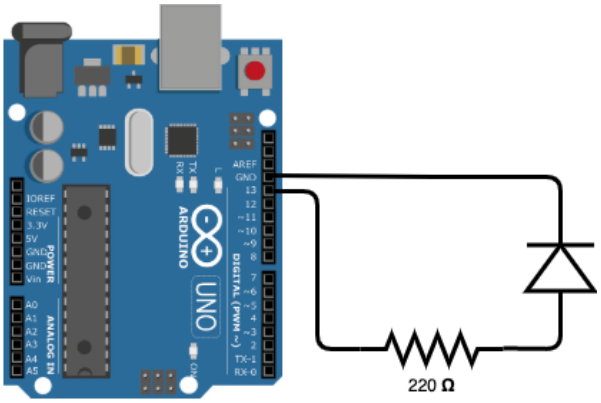


Fig 2. circuit diagram of blink LED

- Use **millis()** function, such as Fig 3., to blink the LED.

② **Interrupt**

- Use a **timer**, such as Fig 4., to blink the LED. (Here is the link of the experiment.)

```
1  const int ledPin = 13;
2  int ledState = LOW;
3  unsigned long previousMillis = 0;
4  const int interval = 1000;
5
6  void setup() {
7    pinMode(ledPin, OUTPUT);
8  }
9
10 void loop() {
11   unsigned long currentMillis = millis();
12   if (currentMillis - previousMillis > interval) {
13     previousMillis = currentMillis;
14     ledState = !ledState;
15     digitalWrite(ledPin, ledState);
16   }
17 }
```

Fig 3. using millis() to blink the LED

```
1  const int ledPin = 13;
2  void setup() {
3    pinMode(ledPin, OUTPUT);
4    TCCR1A = 0x00;
5    TCCR1B |= _BV(CS12);
6    TCCR1B &= ~_BV(CS11);
7    TCCR1B |= _BV(CS10);
8    TCNT1 = 0;
9  }
10
11 void loop() {
12   digitalWrite(ledPin, HIGH);
13   delay1s();
14   digitalWrite(ledPin, LOW);
15   delay1s();
16 }
17
18 void delay1s() {
19   while (TCNT1 < 15625/2)
20   ;
21   TCNT1 = 0;
22 }
```

Fig 4. using timer to blink the LED

- Use an **external interrupt wired** to an external source that toggles a logic state each second. (Any clock should be a good source.)

③ **Polling**

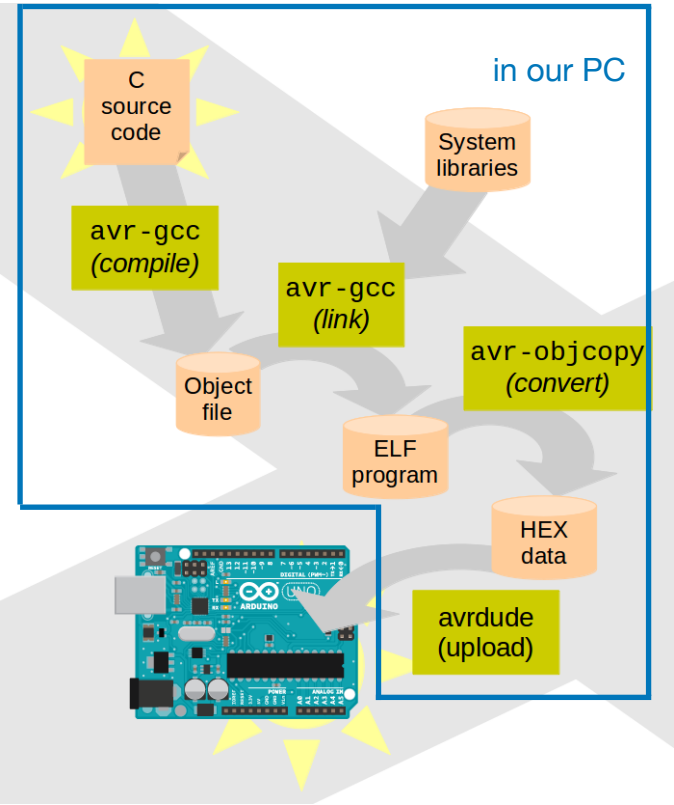
- Use a **library** like TimedAction, EventFuse, Scheduler or simply 'Blink without delay'

2. How many timer in ATmega 328p? What is the usage of timer0 in Arduino world.

- There are **three timer registers** in ATmega 328p. [3]
  - two 8-bit timers which are **timer0** and **timer2**
  - one 16-bit timers which is **timer1**
  - The most important difference between 8-bit and 16-bit timer is the timer resolution. 8bits means 256 values where 16bit means 65536 values.
- In the Arduino world, the usage of each timer:
  - timer0 is been used for the **timer functions, like delay(), millis() and micros()**
  - timer1 is been used in the Servo library
  - timer2 is been used in the tone() function

3. Explain why we should use the cross-compiler.

- A cross compiler is for cross-platform software development of machine code. [1]
- In the Arduino, direct compilation is infeasible, because it **doesn't contain operating system**. We could use the cross-compiler to compile the code on our computer, and generate the executable machine code for Arduino.



4. Explain your plan of de-bouncing the push button.

- Use **millis() function** to de-bounce the push button. (Here is the link of the experiment.)
- In Fig 6., it is the flow chart I used to de-bounce the push button.

```ino
const int buttonPin = 2;
const int ledPin = 13;
int ledState = HIGH;
int buttonState;
int lastButtonState = LOW;

unsigned long lastDebounceTime = 0;
unsigned long debounceDelay = 50;

void setup() {
  pinMode(buttonPin, INPUT);
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, ledState);
}

void loop() {
  int reading = digitalRead(buttonPin);
  if (reading != lastButtonState) {
    lastDebounceTime = millis();
  }
  if ((millis() - lastDebounceTime) > debounceDelay) {
    if (reading != buttonState) {
      buttonState = reading;
      if (buttonState == HIGH) {
        ledState = !ledState;
      }
    }
  }
  digitalWrite(ledPin, ledState);
  lastButtonState = reading;
}
```

Fig 5. using millis() to de-bounce the push

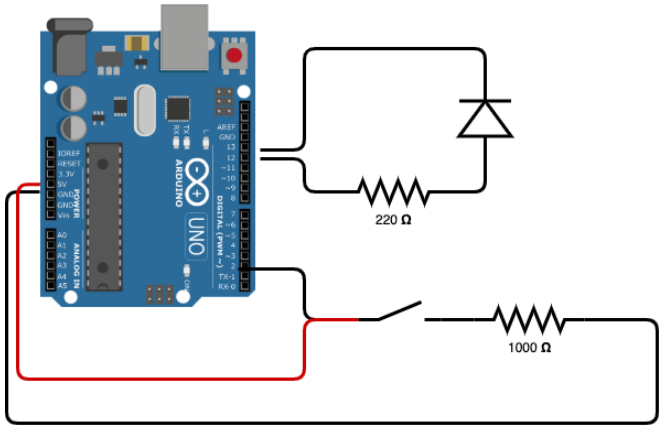


Fig 7. circuit diagram of de-bouncing the

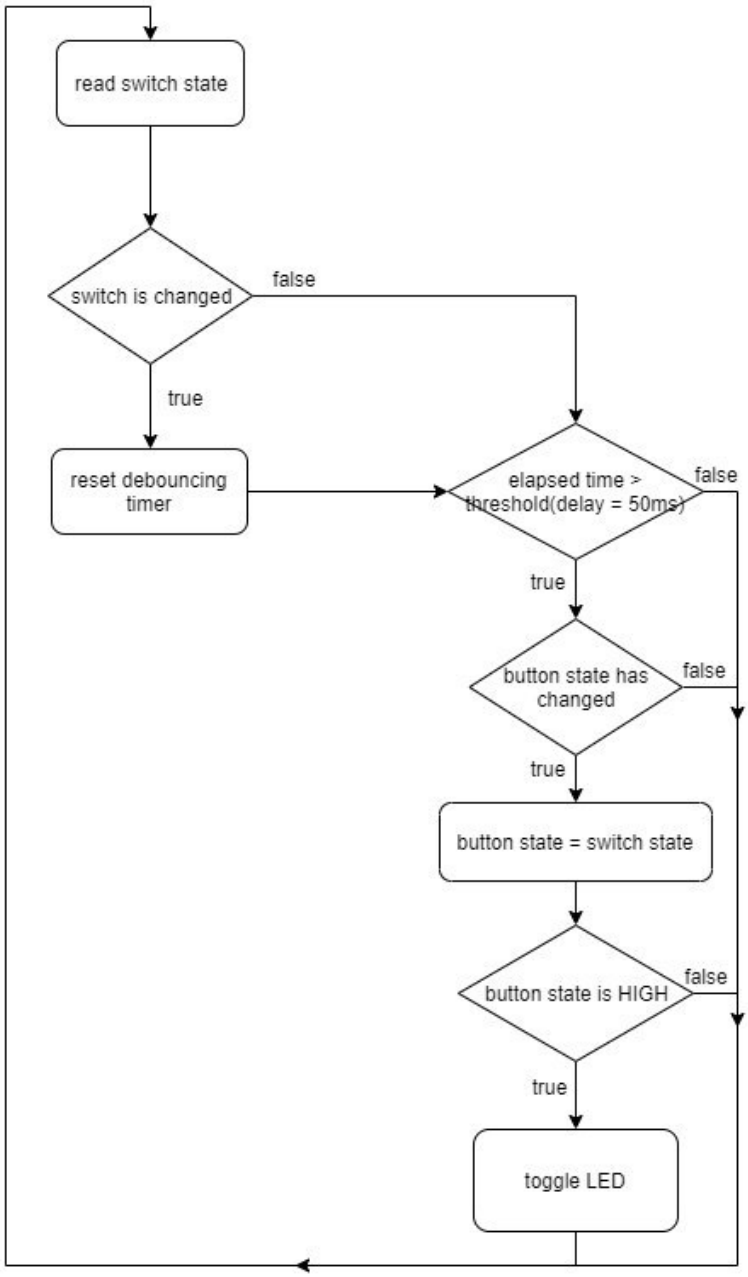


Fig 6. flow chart of de-bouncing the push button

Reference

[1] https://en.wikipedia.org/wiki/Cross_compiler

[2] http://www.avrbeginners.net/architecture/timers/timers.html

[3] https://www.arnabkumardas.com/online-courses/avr-timer-counter-programming-tutorial-atmega328p-avr-8-bit-arduino-uno/

* Note: The LAB exercises and answers in this report are discussed with another student 吳承霖(0858615).