陽明交大
NYCU

物聯網裝置與平台
IoT Devices and Platforms

曾煜棋、吳昆儒

National Yang Ming Chiao Tung University

Ch 3, Arduino Digital - Summary

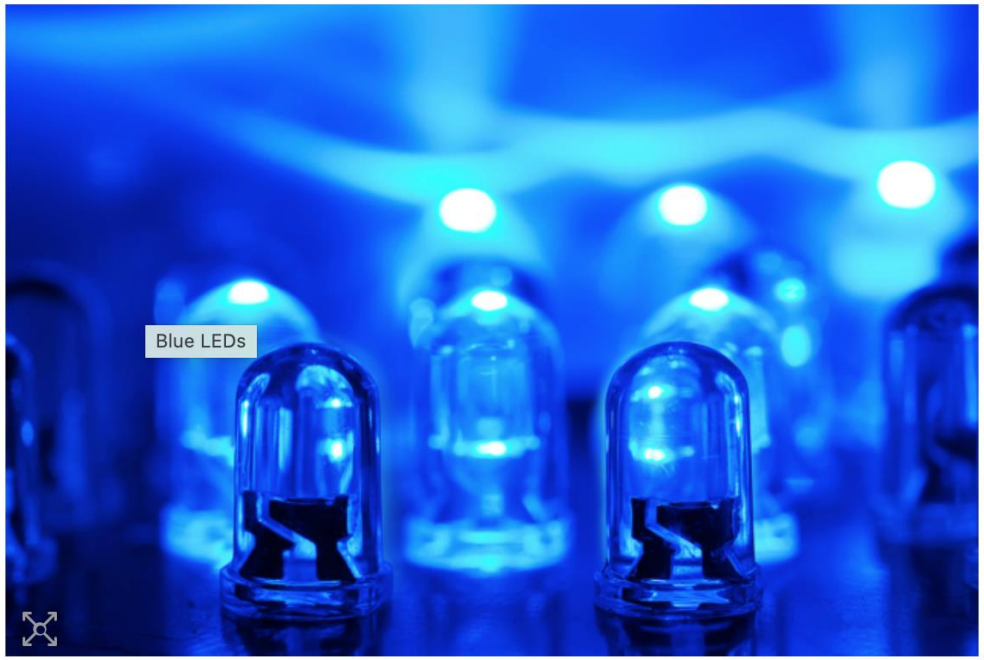
- Get your course package (Arduino + sensors)
- Arduino IDE and how it interacts with the external world
  - LED (signaling, binary)
  - Button (receiving)
  - Debounce (filter noise)
- Understanding this course:
  - Discussion & quiz
  - Interact with TA by Discord

Ch 3, Highlight
(Blue Light LED, Nobel Prize 2014)

HISTORY | BLOG

My favourite Nobel prize: the blue LED that lights up the modern world

27 Sep 2019 Margaret Harris



A new hue: blue LEDs are lighting the 21st century. (Courtesy: iStock/damarco-media)

- Of all the physics Nobel prizes awarded in the past 60 years, the one with the greatest impact on everyday life is undoubtedly the one shared by Isamu Akasaki, Hiroshi Amano and Shuji Nakamura in 2014. As the co-inventors of the blue light-emitting diode (LED), this Japanese-born trio set in motion a dramatic shift in how we see the world. You would have to go back to the 1956 prize – shared by transistor inventors John Bardeen, Walter Brattain and William Shockley – to find a Nobel-laureated discovery that sparked an equivalent transformation.
- LEDs are amazingly efficient, requiring around 90% less energy than incandescent bulbs to produce the same amount of light. Although some of these energy savings are being cancelled out as it becomes financially feasible to illuminate once-dark areas, the LED's efficiency is still a major step forward when you consider that 20-30% of the electricity consumed in industrial societies goes towards lighting. But the emergence of LED lighting as a mainstream commercial product would never have come about were it not for the work of Akasaki, Amano and Nakamura. By developing a blue LED to go with the red and green versions invented decades before, the 2014 laureates made it possible to create white-light LEDs – turning a niche technology into a ubiquitous one at a stroke.

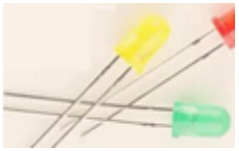
- Not that it was easy. Akasaki, Amano and Nakamura began their groundbreaking work in the 1980s, when Akasaki and Amano were researchers at Nagoya University and Nakamura was working at a small company called Nichia Chemicals. For years, it seemed their chosen material – a crystalline semiconductor called gallium nitride, or GaN – was not going to cooperate. Even getting the crystals to grow took effort. Then, in the early 1990s, first Akasaki and Amano, and then Nakamura, used a technique called metalorganic vapour phase epitaxy to deposit thin films of high-purity GaN onto substrates. After that, the next challenge was to introduce p-doping in order to produce the requisite p-n junction. Once this hurdle was surmounted, the first high-brightness blue LED followed.
- By the mid-2000s, phosphor-coated bulbs that turned blue LED light into a somewhat "cold" white light were readily available in shops. Later, "warmer" versions have now superseded not only incandescent bulbs, but also halogen and compact fluorescent devices. As the Nobel Prize website observes, the 20th century was lit by incandescent bulbs; thanks to Akasaki, Amano and Nakamura; the 21st century is being lit by LED lamps. And how many other Nobel-prize-winning discoveries can you buy for less than a tenner at your local corner shop?

Arduino

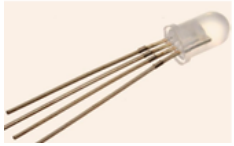
Arduino package

Arduino UNO

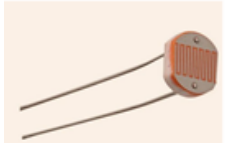
LED



RGB LED



Photocell
光敏電阻



Button Switch

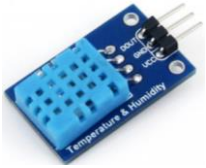
Potentiometer
可變電阻

Speaker

超音波 Ultrasonic
HC-SR04

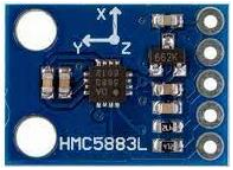
伺服馬達 Servo motor
SG-90

溫溼度
DHT11



氣壓計
BMP180

電子羅盤
HMC-5883L



加速度&陀螺儀
MPU-6050

人體紅外線
HC-SR505

心率感測
xd-58c

聲音感測器
KY-038

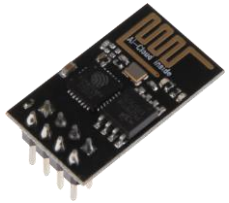
細懸浮微粒感測
PPD42NS

水位感測器

通訊模組 Bluetooth
HM-10

通訊模組 LoRa

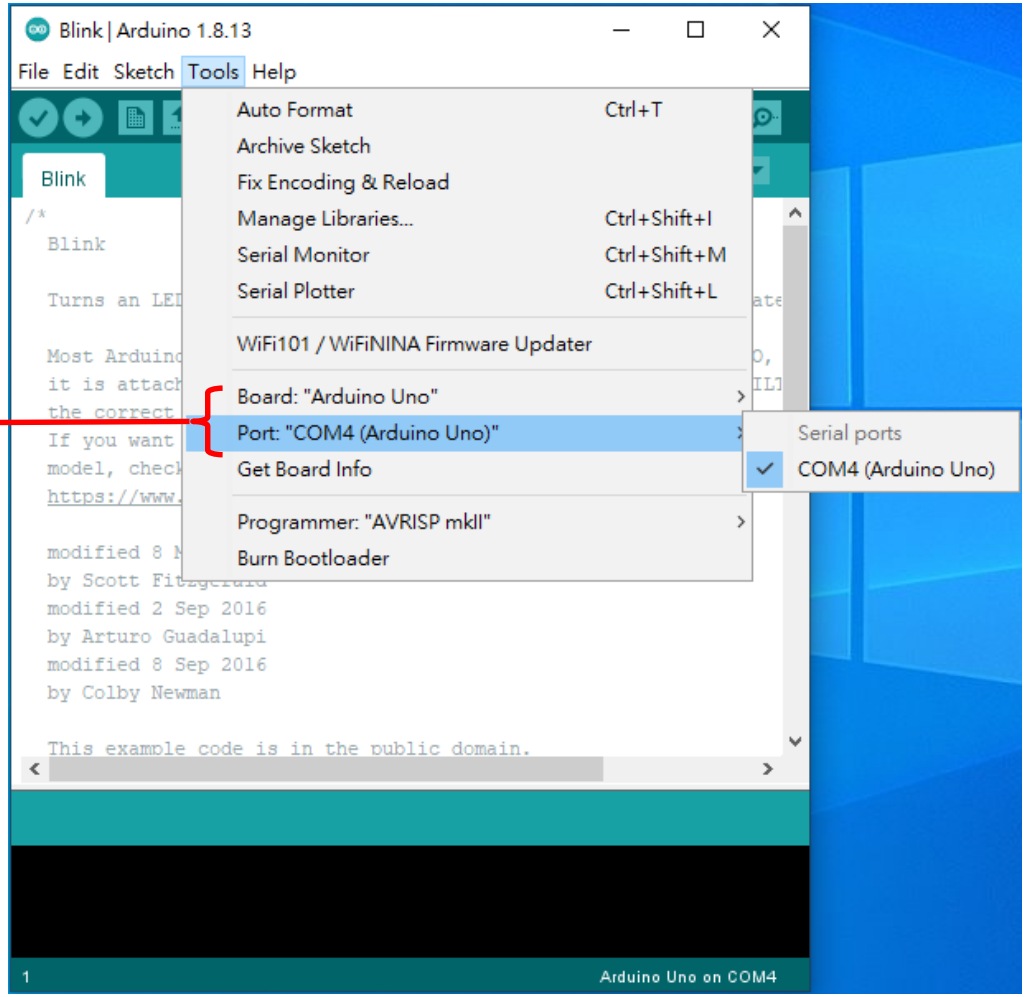
通訊模組 WiFi
ESP8266



Arduino IDE

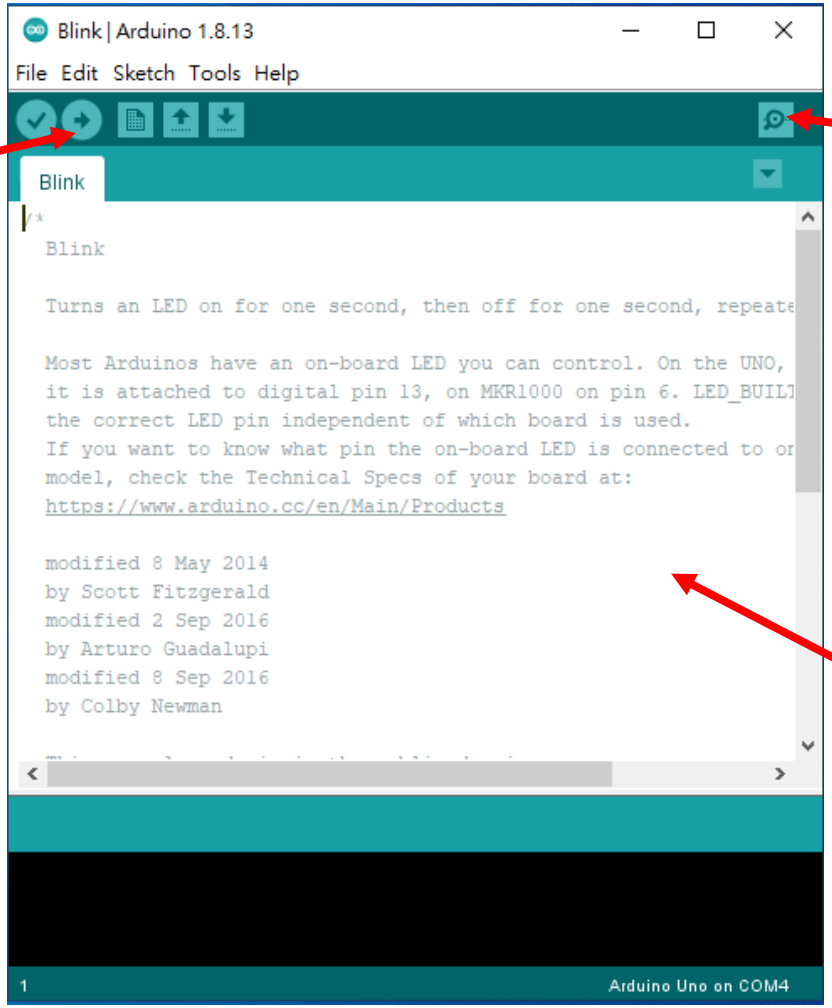
Arduino IDE

Set “Arduino Uno”!!

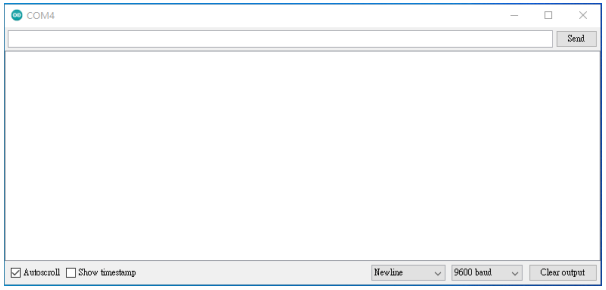


Arduino IDE

Upload code



Open terminal
(Serial Monitor)



Write your code

Labs (Last week)

- After installing Arduino IDE (with Arduino board driver)

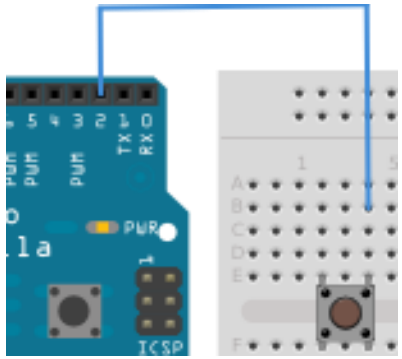
1. **Blink: Turn an LED on and off.**
2. **DigitalReadSerial: Read a switch, print the state to serial monitor in Arduino IDE.**
3. AnalogReadSerial: Read a potentiometer, print its state to the serial monitor.
4. ReadAnalogVoltage: Reads an analog input and prints the voltage to the serial monitor
5. Fade: Turn a analog pin on and off very quickly with different ratio between on and off

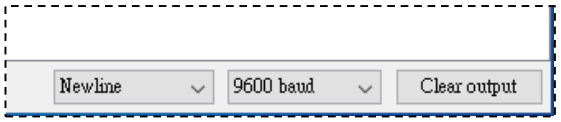
Blink Code

```
// give it a name:
int led = 12;          // change the digital output to pin 12

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 12 as an output.
  pinMode(led, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite (led, HIGH);      // turn the LED on (HIGH is the voltage level)
  delay(1000);                   // wait for a second
  digitalWrite (led, LOW);       // turn the LED off by making the voltage LOW
  delay(1000);                   // wait for a second
}
```

Button Code

Open--->File--->Examples--->Basic---> DigitalReadSerial

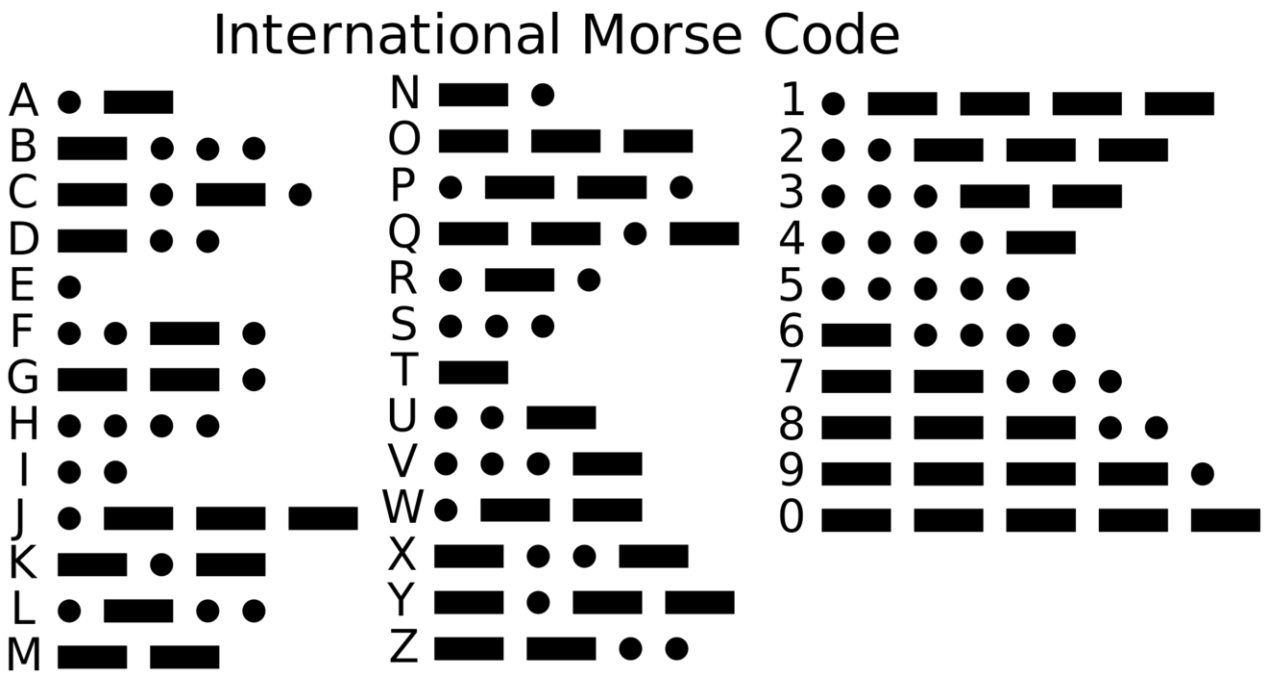





```
// digital pin 2 has a pushbutton attached to it. Give it a name:
int pushButton = 2;

void setup() {
  Serial.begin(9600);     // initialize serial communication at 9600 bits/s
  pinMode(pushButton, INPUT);    // make the pushbutton's pin an input
}

void loop() {
  int buttonState = digitalRead(pushButton);    // read the input pin
  Serial.println(buttonState);     // print out the state of the button:
  delay(1);       // delay in between reads for stability
}
```

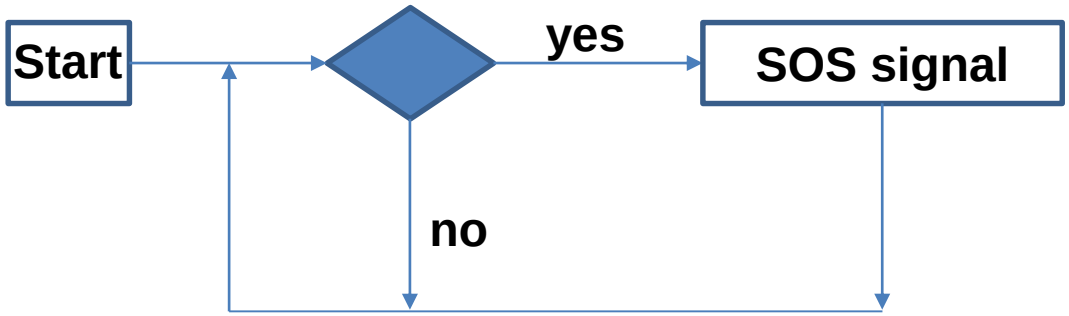
Quiz 1

- Make “SOS” signal with LED. (**Morse Code**)
  1. A dash is equal to three dots.
  2. The space between parts of the same letter is equal to one dot.
  3. The space between two letter is equal to three dots.
  4. The space between two words is equal to seven dots.



Quiz 2

- Press the button to show SOS signal **once**



Ref: https://www.arduino.cc/reference/en/language/structure/control-structure/if/

```
if (condition) {
  //statement(s)
}
```

Labs (This week)

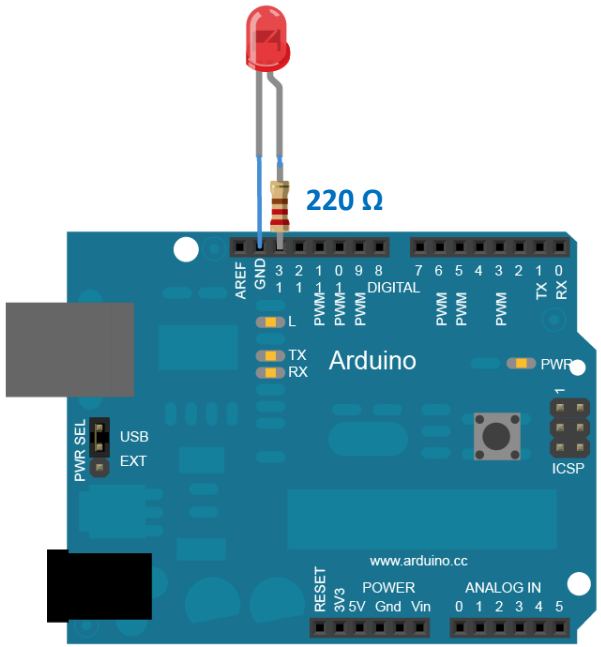
1. BlinkWithoutDelay: blinking an LED without using the delay() function.
2. StateChangeDetection: counting the number of button pushes.
3. Debounce: read a pushbutton, filtering noise.
4. DigitalInputPullup: Demonstrates the use of INPUT_PULLUP with pinMode().

Lab1. BlinkWithoutDelay :

blinking an LED without using the delay() function.

Lab 1. BlinkWithoutDelay

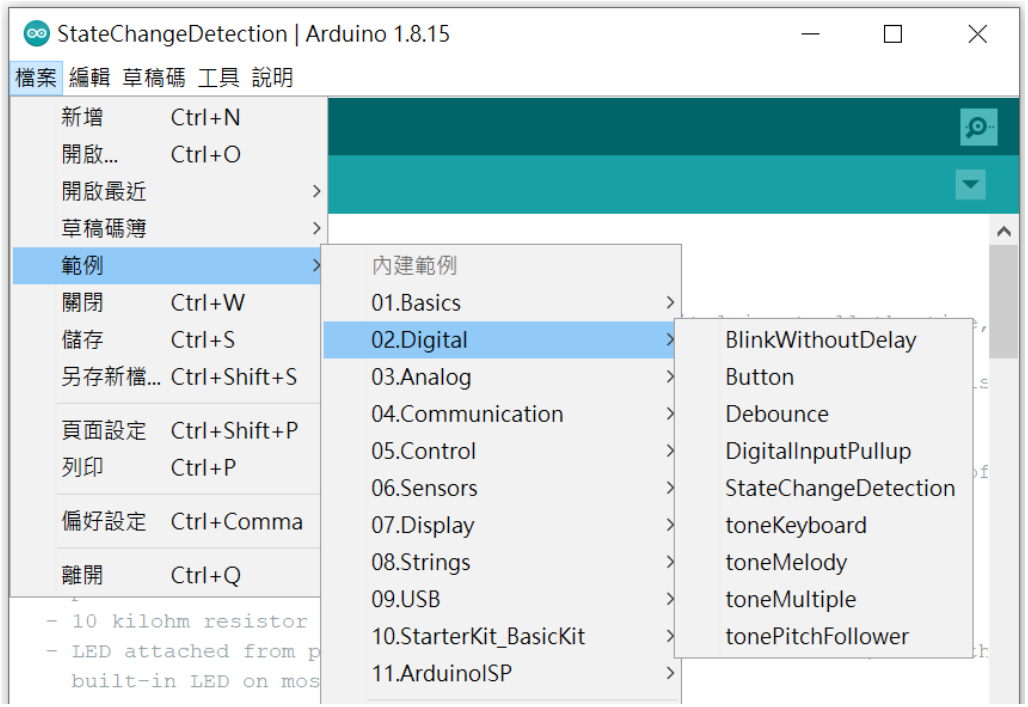
- Goal: You might want to blink an LED while reading a button press or other input. In this case, you can't use delay(); otherwise, you will stop everything else while the LED blinked.
- Hardware Required:
  - Arduino Board
  - LED
  - a 220 ohm resistor



Lab 1. BlinkWithoutDelay

Arduino IDE

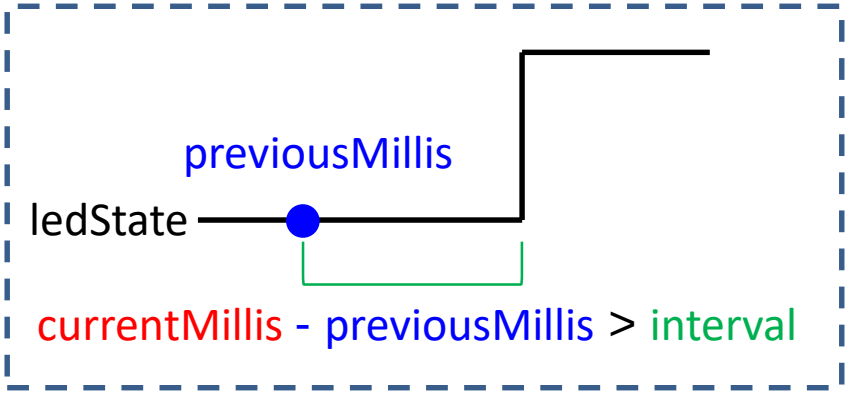
Open--->File--->Examples--->02.Digital--->BlinkWithoutDelay



Built-in Sample Code:

```
// constants won't change. Used here to set a pin number:
const int ledPin =  LED_BUILTIN;// the number of the LED pin

// Variables will change:
int ledState = LOW;             // ledState used to set the LED

// Generally, you should use "unsigned long" for variables that hold time
// The value will quickly become too large for an int to store
unsigned long previousMillis = 0;        // will store last time LED was updated

// constants won't change:
const long interval = 1000;           // interval at which to blink (milliseconds)

void setup() {
  // set the digital pin as output:
  pinMode(ledPin, OUTPUT);
}
```

```
void loop() {
  // here is where you'd put code that needs to be running all the time.

  // check to see if it's time to blink the LED; that is, if the difference
  // between the current time and last time you blinked the LED is bigger than
  // the interval at which you want to blink the LED.
  unsigned long currentMillis = millis();  // current time

  if (currentMillis - previousMillis >= interval) {
    // save the last time you blinked the LED
    previousMillis = currentMillis;

    // if the LED is off turn it on and vice-versa:
    if (ledState == LOW) {
      ledState = HIGH;
    } else {
      ledState = LOW;
    }

    // set the LED with the ledState of the variable:
    digitalWrite(ledPin, ledState);
  }
}
```



Lab 1. Syntax

- Syntax
  - millis()
- Returns
  - Returns the number of milliseconds **since the Arduino board began** running the current program.
  - This number will overflow (go back to zero), after approximately 50 days.
  - unsigned long, 32bits, range from 0 to 4,294,967,295 (2^32 - 1)
- Example
  - unsigned long time = millis();

Discussion 1

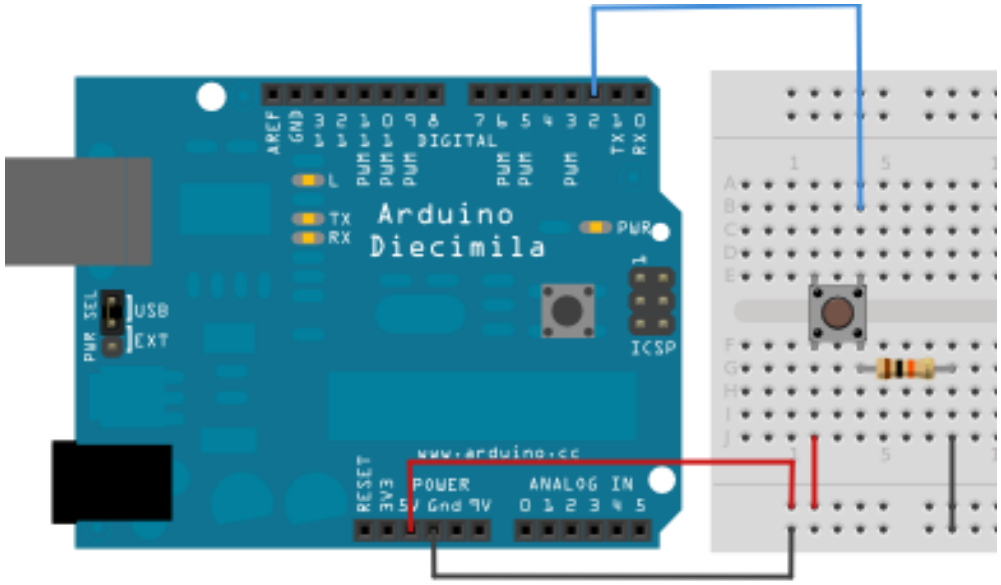
- What are the advantages of Blink without delay?

Lab2. StateChangeDetection:

Count the number of button pushes.

Lab 2. StateChangeDetection

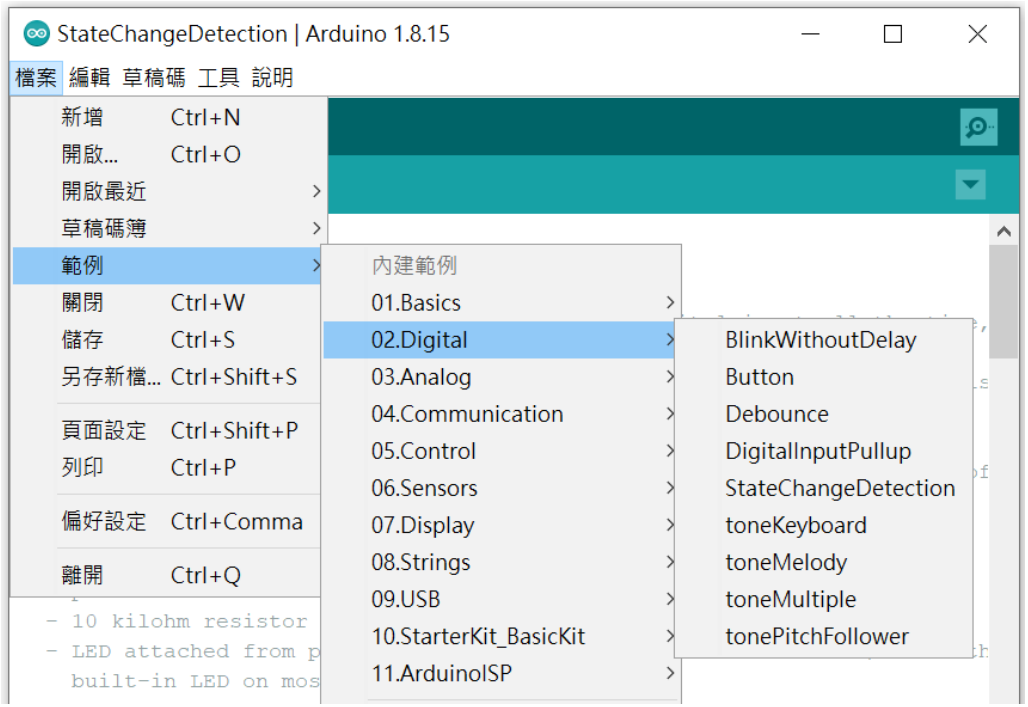
- Goal: Once you've got a pushbutton working, you often want to do action based on how many times the button is pushed. This lab teaches you to count the number of button pushes
- Hardware Required
  - Arduino Board
  - A button
  - 10K ohm resistor
  - breadboard
  - hook-up wire



Lab 2. State change detection

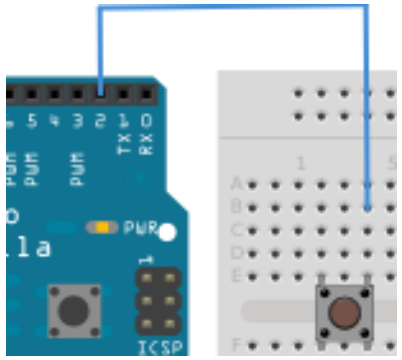
Arduino IDE

Open--->File--->Examples--->02.Digital--->State change detection



Built-in Sample Code:

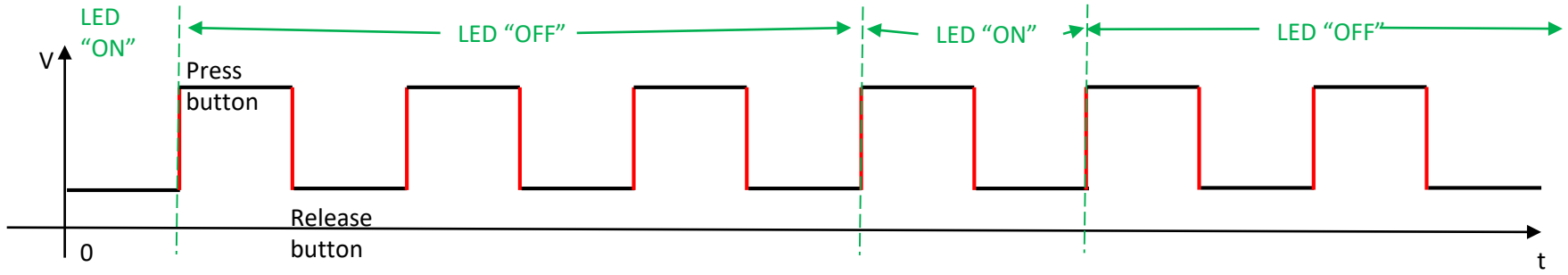
```
// this constant won't change:
const int  buttonPin = 2;    // the pin that the pushbutton is attached to
const int ledPin = 13;       // the pin that the LED is attached to

// Variables will change:
int buttonPushCounter = 0;   // counter for the number of button presses
int buttonState = 0;         // current state of the button
int lastButtonState = 0;     // previous state of the button

void setup() {
  // initialize the button pin as a input:
  pinMode(buttonPin, INPUT);
  // initialize the LED as an output:
  pinMode(ledPin, OUTPUT);
  // initialize serial communication:
  Serial.begin(9600);
}
```



```
void loop() {
  // read the pushbutton input pin:
  buttonState = digitalRead(buttonPin);

  // compare the buttonState to its previous state
  if (buttonState != lastButtonState) {
    // if the state has changed, increment the counter
    if (buttonState == HIGH) {
      // if the current state is HIGH then the button went from off to on:
      buttonPushCounter++;
      Serial.println("on");
      Serial.print("number of button pushes: ");
      Serial.println(buttonPushCounter);
    } else {
      // if the current state is LOW then the button went from on to off:
      Serial.println("off");
    }
    // Delay a little bit to avoid bouncing
    delay(50);
  }
  // save the current state as the last state, for next time through the loop
  lastButtonState = buttonState;
```

```
// turns on the LED every four button pushes by checking the modulo of the
// button push counter. the modulo function gives you the remainder of the
// division of two numbers:
if (buttonPushCounter % 4 == 0) {
  digitalWrite(ledPin, HIGH);
} else {
  digitalWrite(ledPin, LOW);
}

}
```

% : 餘數
1/4 = 0.....1



Lab 2. Syntax

- Syntax
  - %
  - Arithmetic operators > Modulo
- Returns
  - Modulo operation calculates the remainder when one integer is divided by another.
  - It is useful for keeping a variable within a particular range (e.g. the size of an array).
  - The % (percent) symbol is used to carry out modulo operation.
- Example
  - if (buttonPushCounter % 4 == 0)

Discussion 2

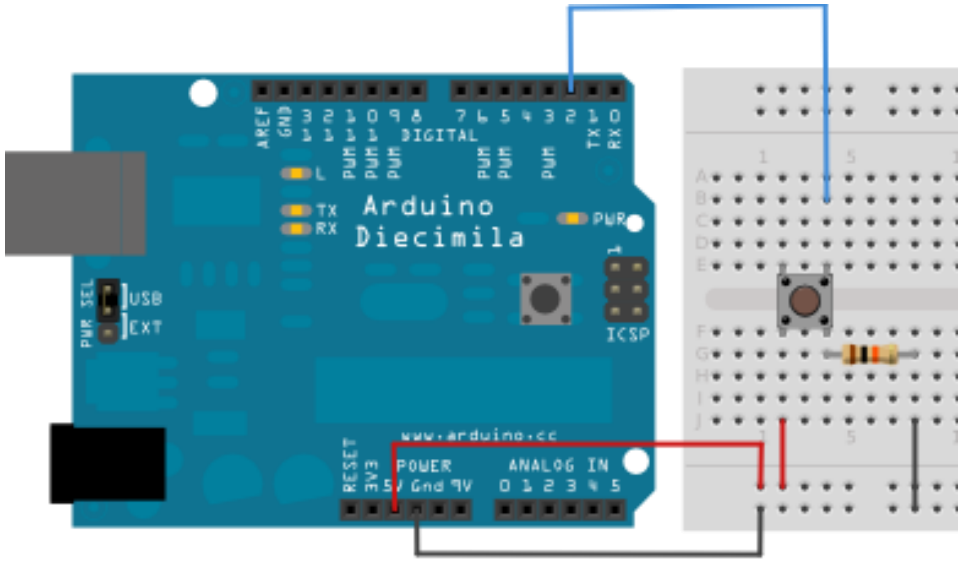
- Remove the “delay(50)” in the code. Play around by pushing the button several times.
- Observe the change of “buttonPushCounter”. Why do we see more than one push even if there is only one button state change?

Lab3. Debounce:

read a pushbutton, filtering noise.

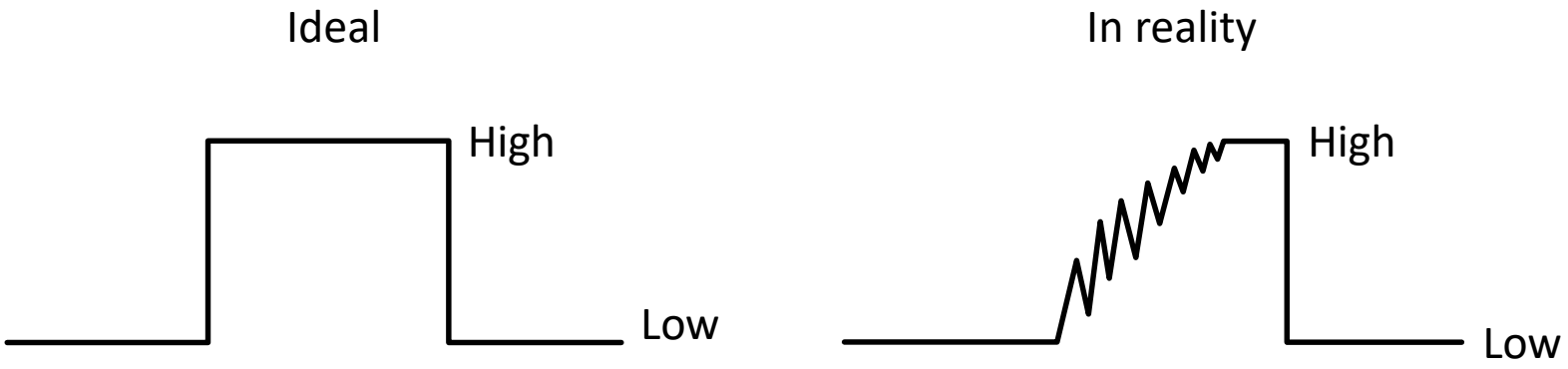
Lab 3. Debounce

- Goal: This example demonstrates how to debounce an input, which means checking twice in a short period of time to make sure that a button is definitely pressed.
- Hardware Required
  - Arduino Board
  - A button
  - 10K ohm resistor
  - breadboard
  - hook-up wire



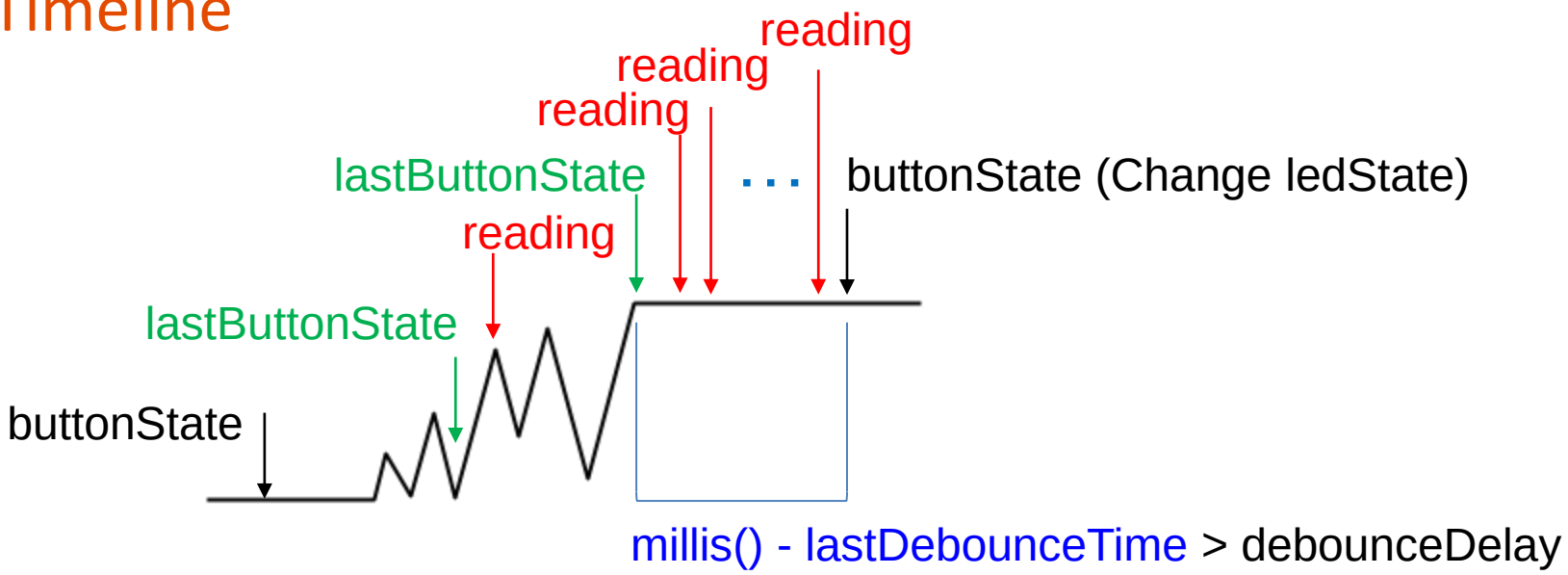
Lab 3. Debounce

- What is bounce?
  - Switches are usually made of springy metals.
  - When two contacts strike together, they may bounce apart one or more times before making steady contact.
  - The result is a rapid pulsed electric current change, instead of a clean transition from zero to full current.
- It can be solved by software or hardware.



Lab 3. Debounce

- Timeline

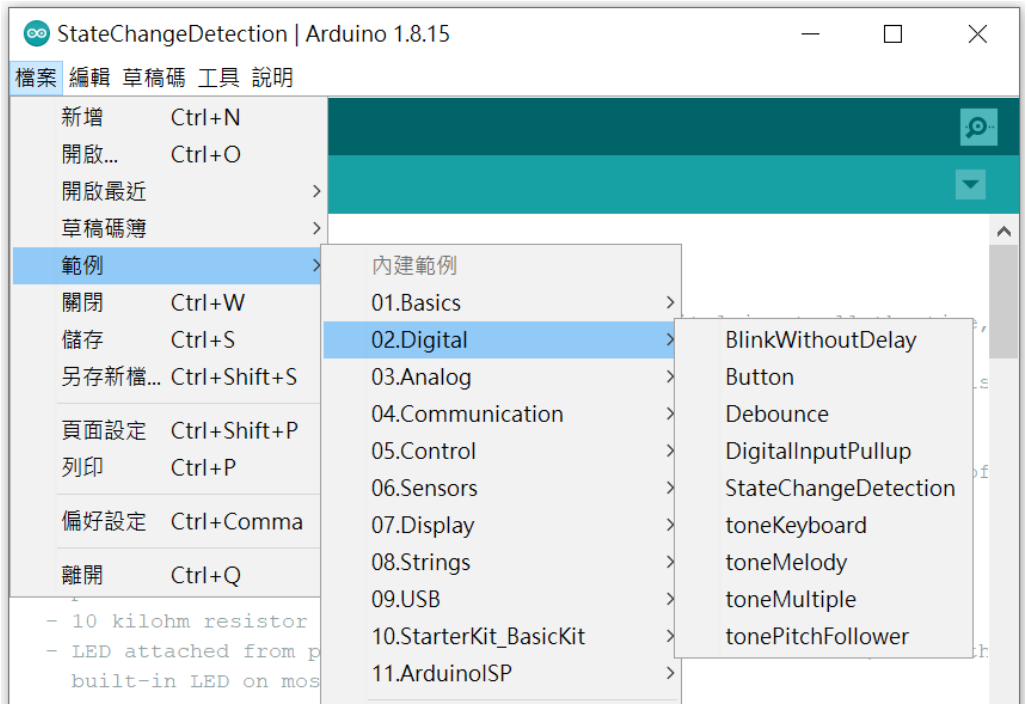


- Cases to be considered
  - Reading != lastButtonState
  - Reading == lastButtonState
  - millis() - lastDebounceTime > debounceDelay
  - millis() - lastDebounceTime <= debounceDelay

Lab 3. Debounce

Arduino IDE

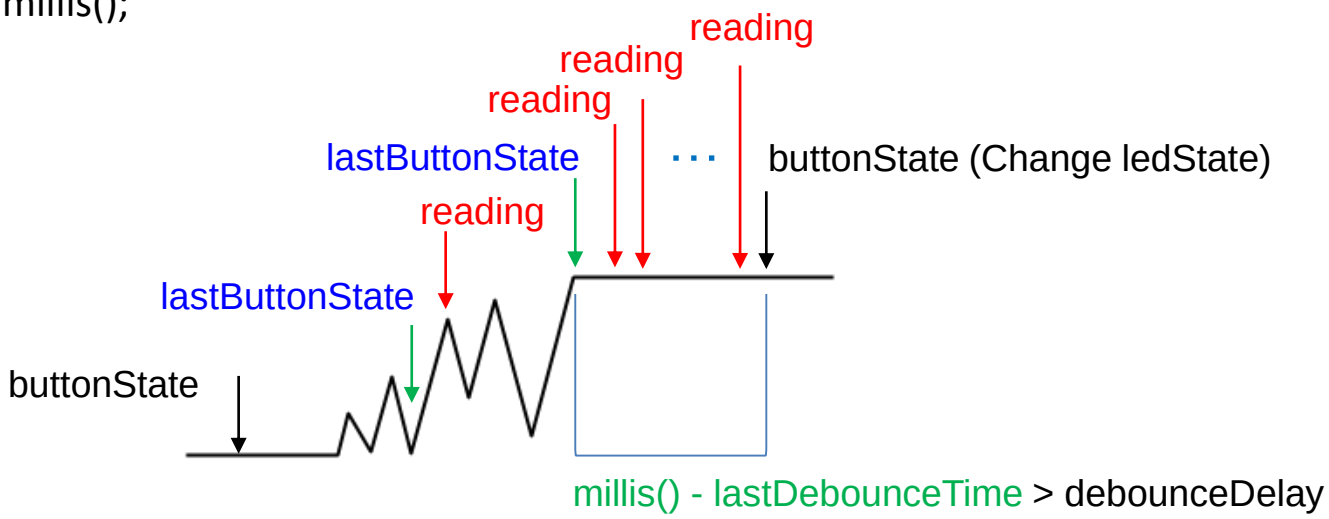
Open--->File--->Examples--->02.Digital--->Debounce



Built-in Sample Code:

```
// constants won't change. They're used here to set pin numbers:
const int buttonPin = 2;    // the number of the pushbutton pin
const int ledPin = 13;      // the number of the LED pin

// Variables will change:
int ledState = HIGH;            // the current LED state of the output pin
int buttonState;                // the current  button reading from the input pin
int lastButtonState = LOW;  // the previous button reading from the input pin

// the following variables are unsigned longs because the time, measured in
// milliseconds, will quickly become a bigger number than can be stored in an int.
unsigned long lastDebounceTime = 0;   // the last time the output pin was toggled
unsigned long debounceDelay = 50;     // the debounce time; increase if the output flickers

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

  // set initial LED state
  digitalWrite(ledPin, ledState);
}
```

```
void loop() {
  // read the state of the switch into a local variable:
  int reading = digitalRead(buttonPin);

  // check to see if you just pressed the button
  // (i.e. the input went from LOW to HIGH), and you've waited long enough
  // since the last press to ignore any noise:

  // If the switch changed, due to noise or pressing:
  if (reading != lastButtonState) {
    // reset the debouncing timer
    lastDebounceTime = millis();
  }

  // (cont.)
```



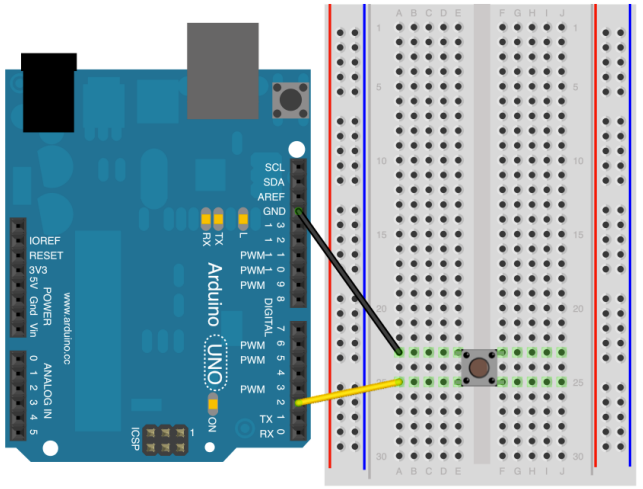
```
if ((millis() - lastDebounceTime) > debounceDelay) {
  // whatever the reading is at, it's been there for longer than the debounce
  // delay, so take it as the actual current state:

  // if the button state has changed:
  if (reading != buttonState) {
    buttonState = reading;

    // only toggle the LED if the new button state is HIGH
    if (buttonState == HIGH) {
      ledState = !ledState;  // store the inverted value of ledState
    }
  }
}

// set the LED:
digitalWrite(ledPin, ledState);

// save the reading. Next time through the loop, it'll be the lastButtonState:
lastButtonState = reading;

} // void loop() {
```

Lab4. DigitalInputPullup:

Demonstrates the use of INPUT_PULLUP with pinMode().

Lab 4. DigitalInputPullup

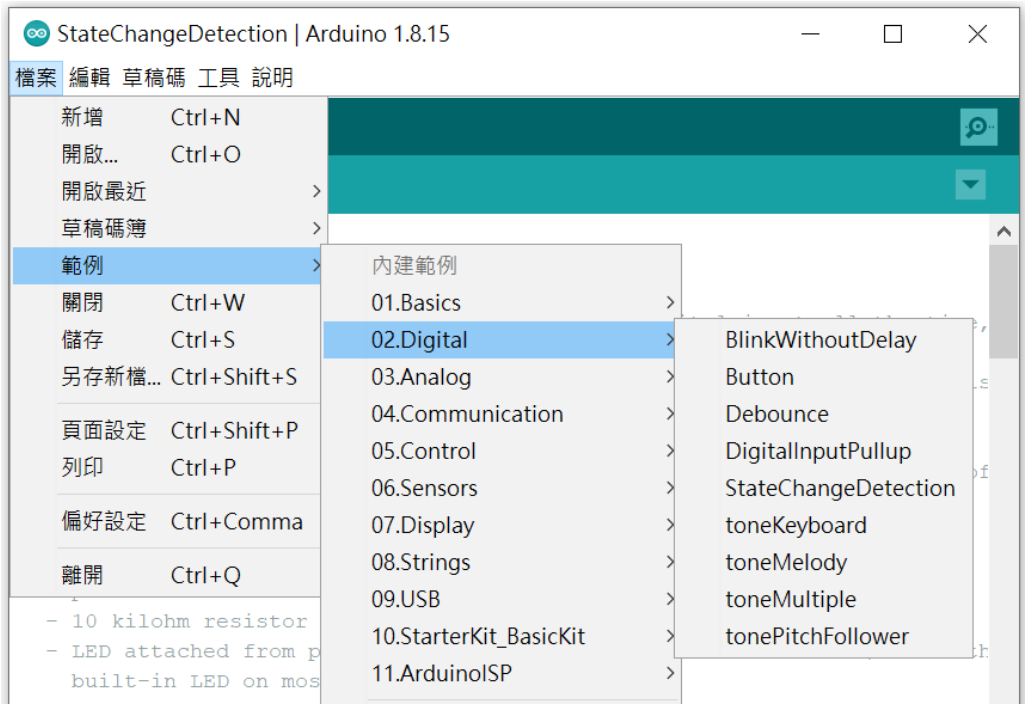
- Goal: There are 20K pullup resistors built into the Atmega chip that can be accessed from software.
  - When the pushbutton is open (released)
    - the internal pull-up on pin 2 is active and connected to 5V
    - we read HIGH value
  - When the button is closed (pressed)
    - we read LOW because a connection to ground is completed.
- Hardware Required
  - Arduino Board
  - A button
  - breadboard
  - hook-up wire



Lab 4. DigitalInputPullup

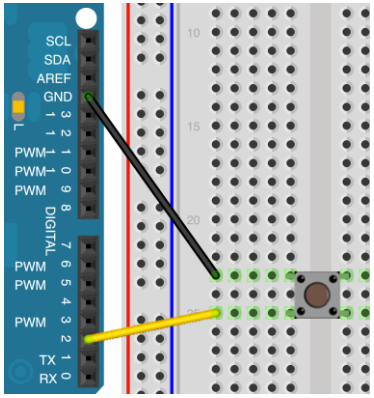
Arduino IDE

Open--->File--->Examples--->02.Digital--->DigitalInputPullup



Built-in Sample Code:

```
void setup() {
  //start serial connection
  Serial.begin(9600);
  //configure pin 2 as an input and enable the internal pull-up resistor
  pinMode(2, INPUT_PULLUP);
  pinMode(13, OUTPUT);
}

void loop() {
  //read the pushbutton value into a variable
  int sensorVal = digitalRead(2);
  //print out the value of the pushbutton
  Serial.println(sensorVal);

  // Keep in mind the pull-up means the pushbutton's logic is inverted. It goes
  // HIGH when it's open, and LOW when it's pressed. Turn on pin 13 when the
  // button's pressed, and off when it's not:
  if (sensorVal == HIGH) {
    digitalWrite(13, LOW);
  } else {
    digitalWrite(13, HIGH);
  }
}
```



Quiz

- Design a “reset” button for a device. If someone presses the reset button for **more than 5 seconds**, the device will **send a signal**.
- **How would you change the above codes to achieve this goal?**

Summary

Summary

- Practice Labs by yourself
- **Write Answers for Discussion 1 to 2**
  - Upload to e3 before next class
- **Quiz: Write code for quiz, then demonstrate to TAs**
  - 1. Make “SOS” signal with LED. (Morse Code)
  - 2. Press the button to show SOS signal once
  - 3. Design a “reset” button to send a signal.