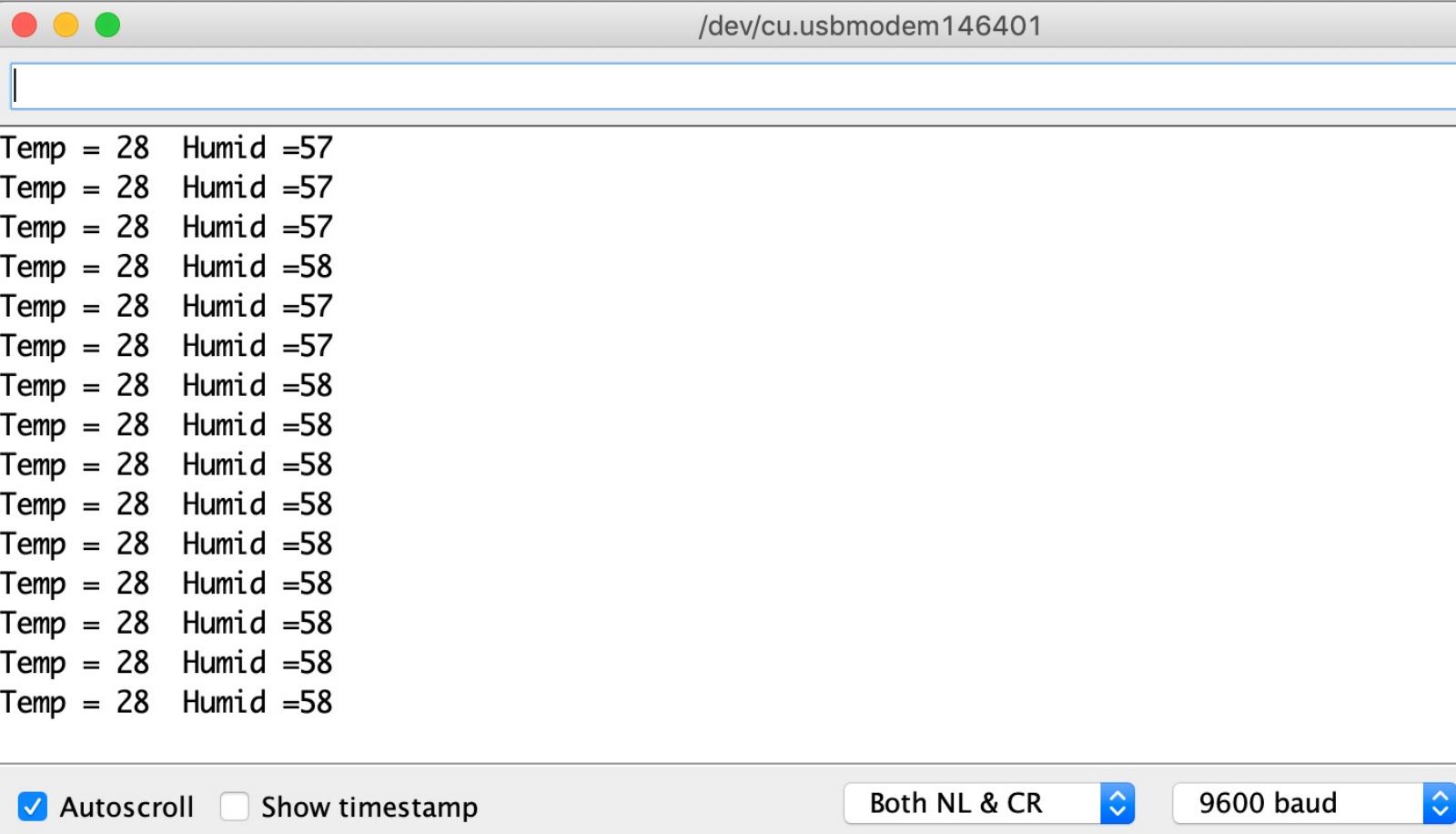




The screenshot shows the Arduino IDE interface with a sketch titled "Simple_DHT11". The code is written in C++ and uses the SimpleDHT library to read data from a DHT11 sensor connected to pin D2. The sketch includes setup and loop functions to initialize the serial port, read data, and print it to the Serial Monitor.

```
1 # include <SimpleDHT.h> // DHT at D2
2
3
4 SimpleDHT11 dht11;
5 int pinDHT11 = 2;
6
7 void setup() {
8   Serial.begin(9600);
9 }
10
11 void loop() {
12
13   byte temperature = 0;
14   byte humidity = 0;
15
16   int result = dht11.read(pinDHT11, &temperature, &humidity, 0) ;
17
18   Serial.print("Temp = ");
19   Serial.print(temperature);
20   Serial.print("  Humid =");
21   Serial.println(humidity);
22   delay(1000); // DHT11 sampling rate is 1HZ.
23 }
```



A screenshot of a terminal window titled "/dev/cu.usbmodem146401". The window displays a series of temperature and humidity readings. The text in the terminal is as follows:

```
Temp = 28 Humid =57
Temp = 28 Humid =57
Temp = 28 Humid =57
Temp = 28 Humid =58
Temp = 28 Humid =57
Temp = 28 Humid =57
Temp = 28 Humid =58
```

At the bottom of the window, there are several configuration options:

- Autoscroll Show timestamp
- Both NL & CR
- 9600 baud

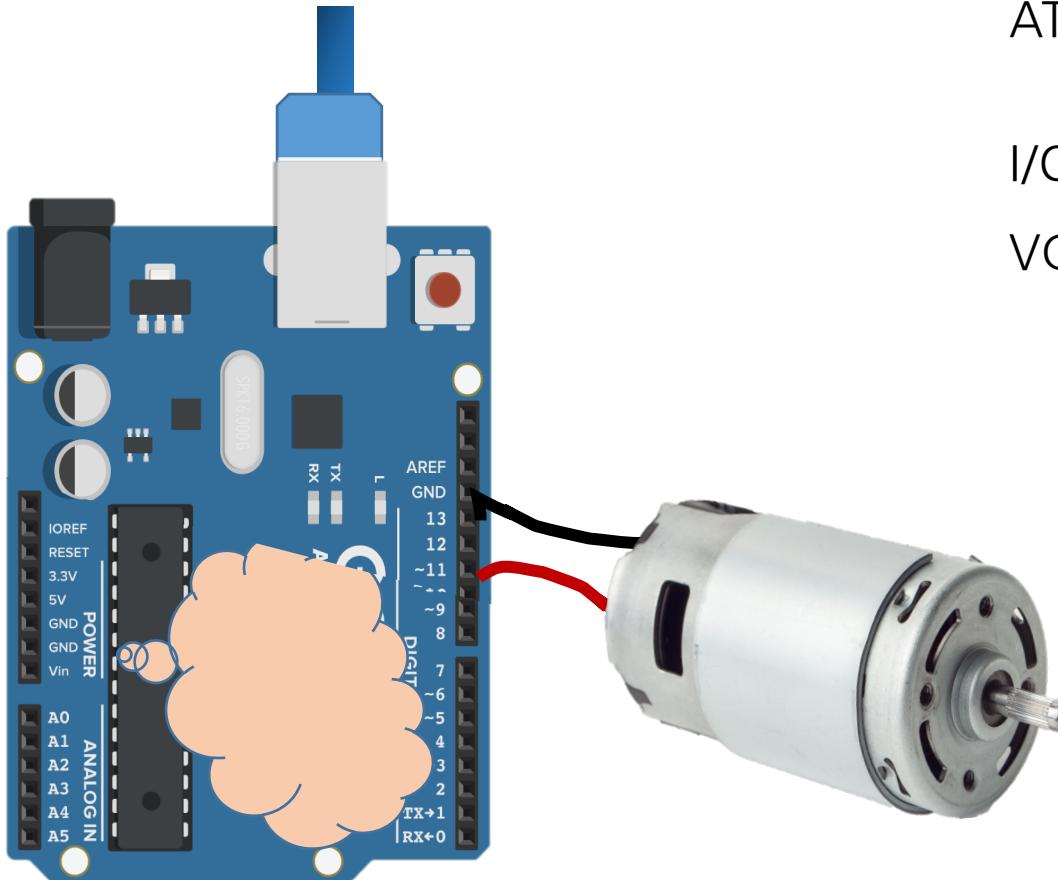
Ch5. Relay, Interrupt, Pullup, Debounce

양 세훈



1) Relay for high power control

BIG 모터 컨트롤 하려면 ?



ATmega328P Maximum Current

I/O pins = 40 mA

VCC and GND pins = 200 mA

1) Relay for high power control

릴레이 :

マイクロ 컨트롤러를 사용하여 큰 전기를 컨트롤 할때 사용된다.

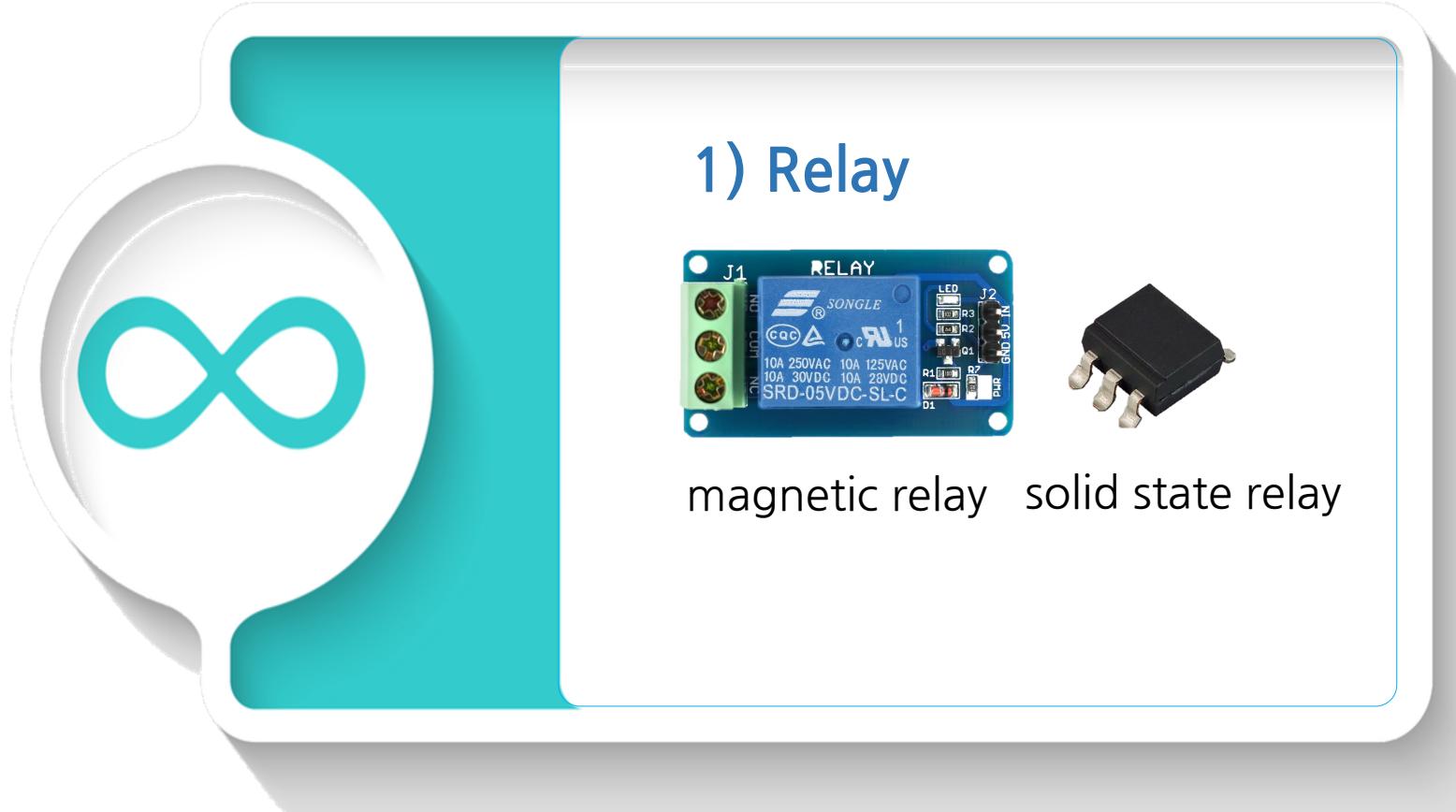
프로젝트 : 마그네틱 릴레이를 사용하여 모터를 구동 시키는 회로와 코딩

준비물 :

아두이노 우노 1개

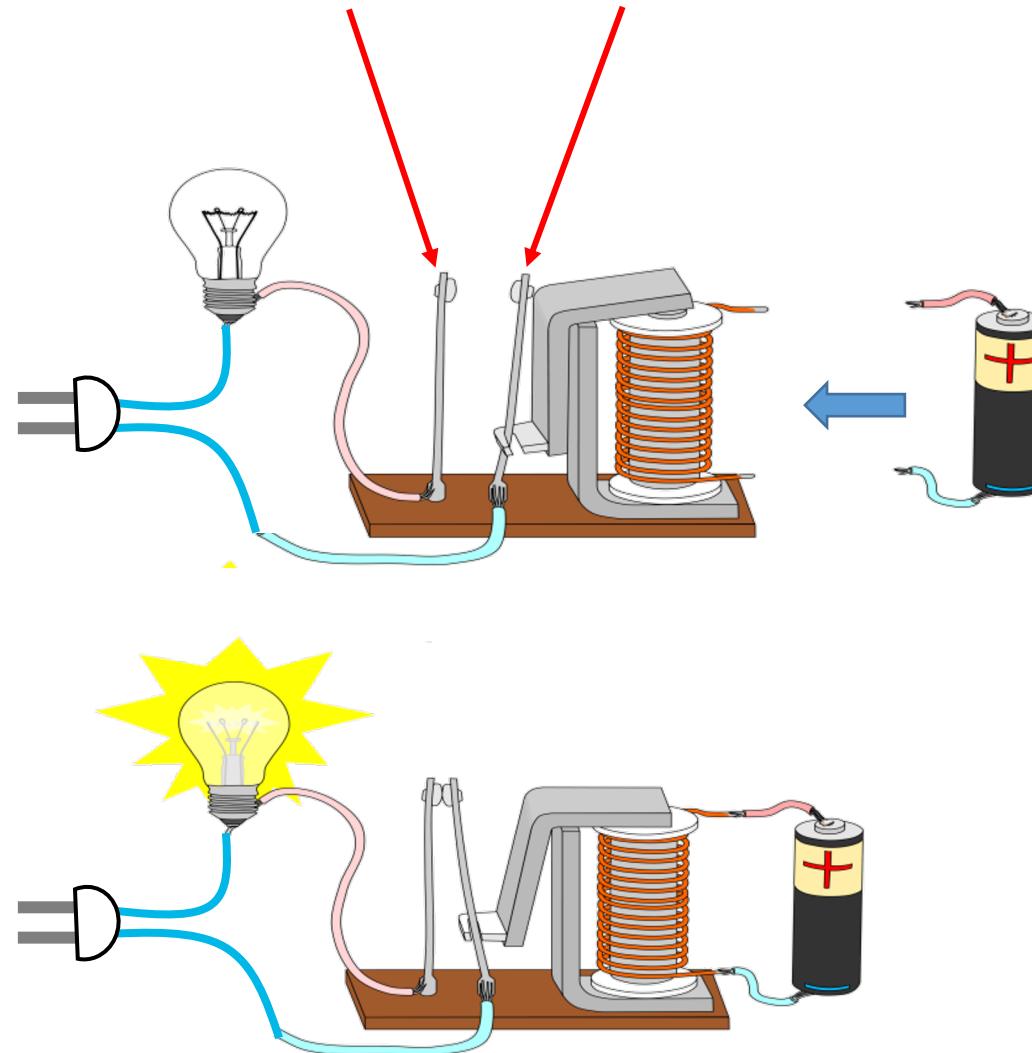
마그네틱 릴레이 1개

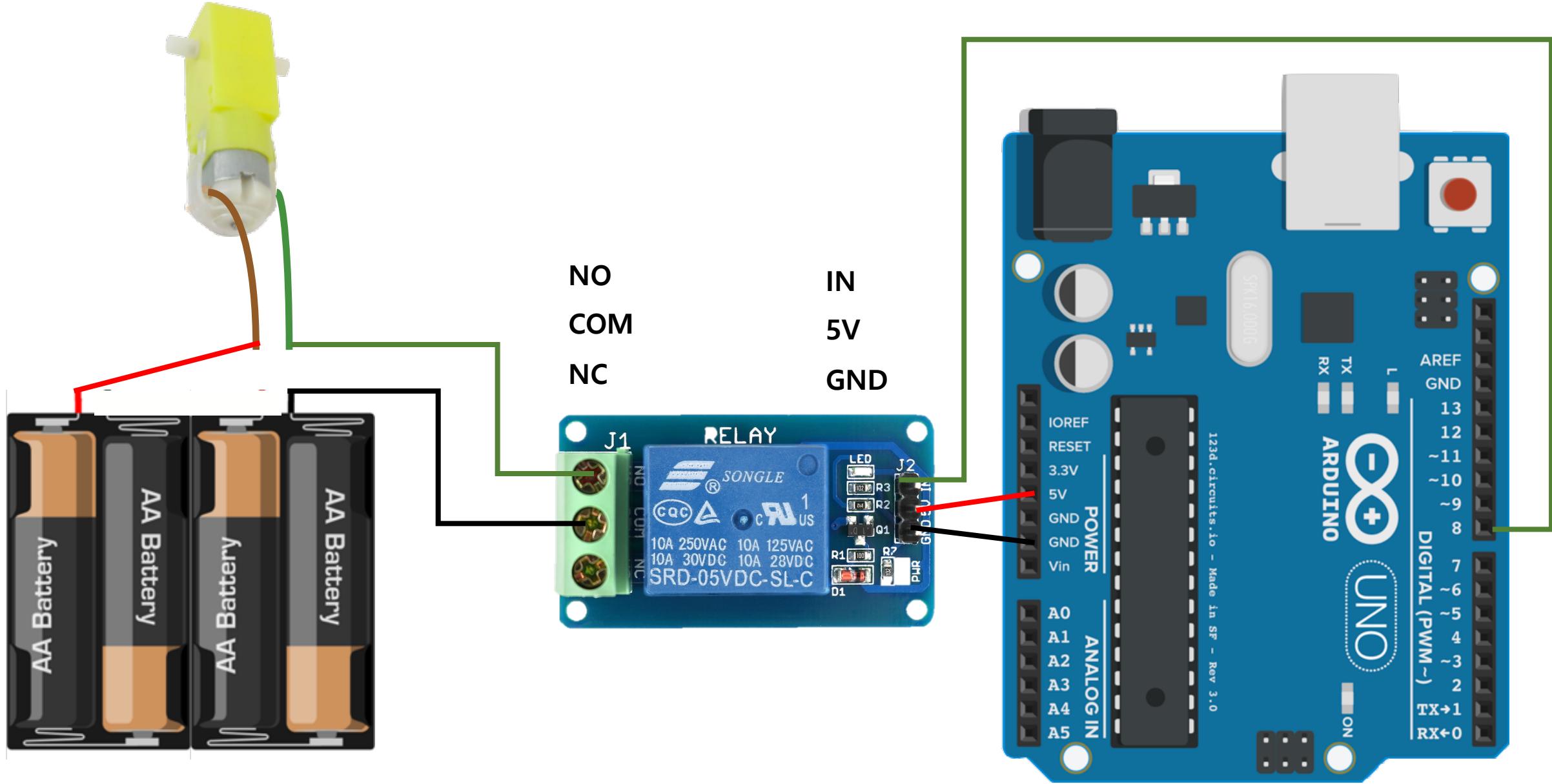
점퍼 케이블



1) Relay for high power control

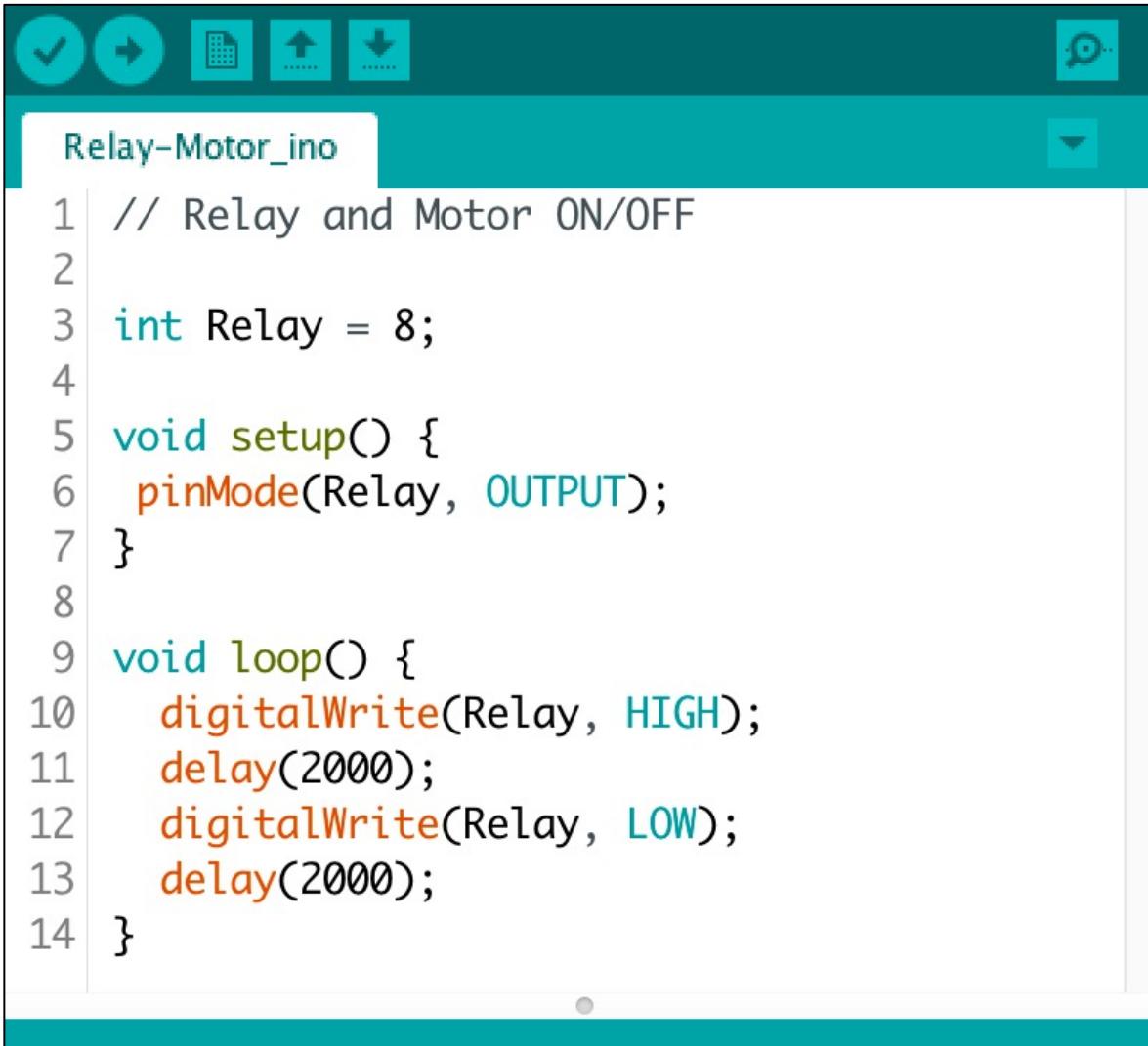
릴레이 구조 및 작동원리





1) Relay for high power control

Motor ON/OFF control through Relay Sketch

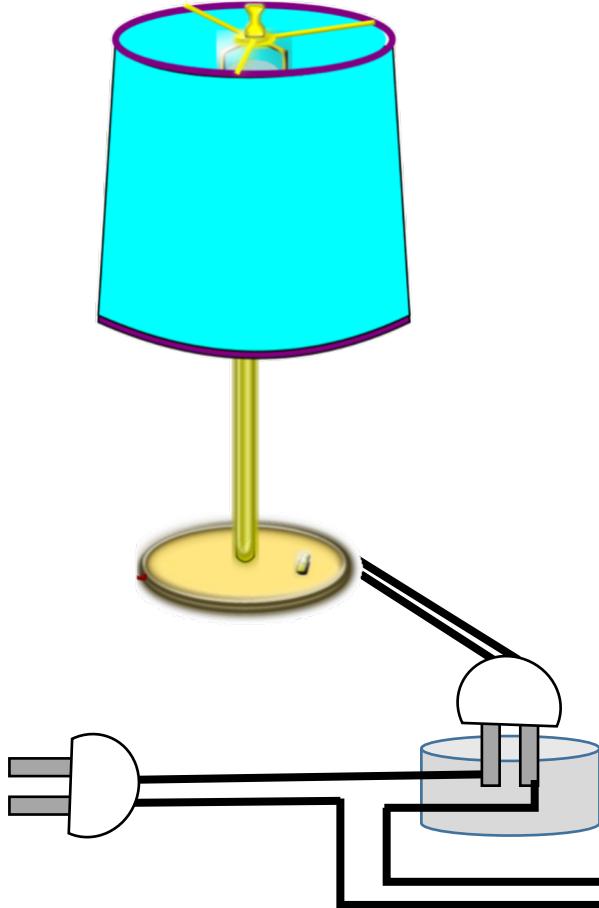


The screenshot shows the Arduino IDE interface with a sketch titled "Relay-Motor_ino". The code is as follows:

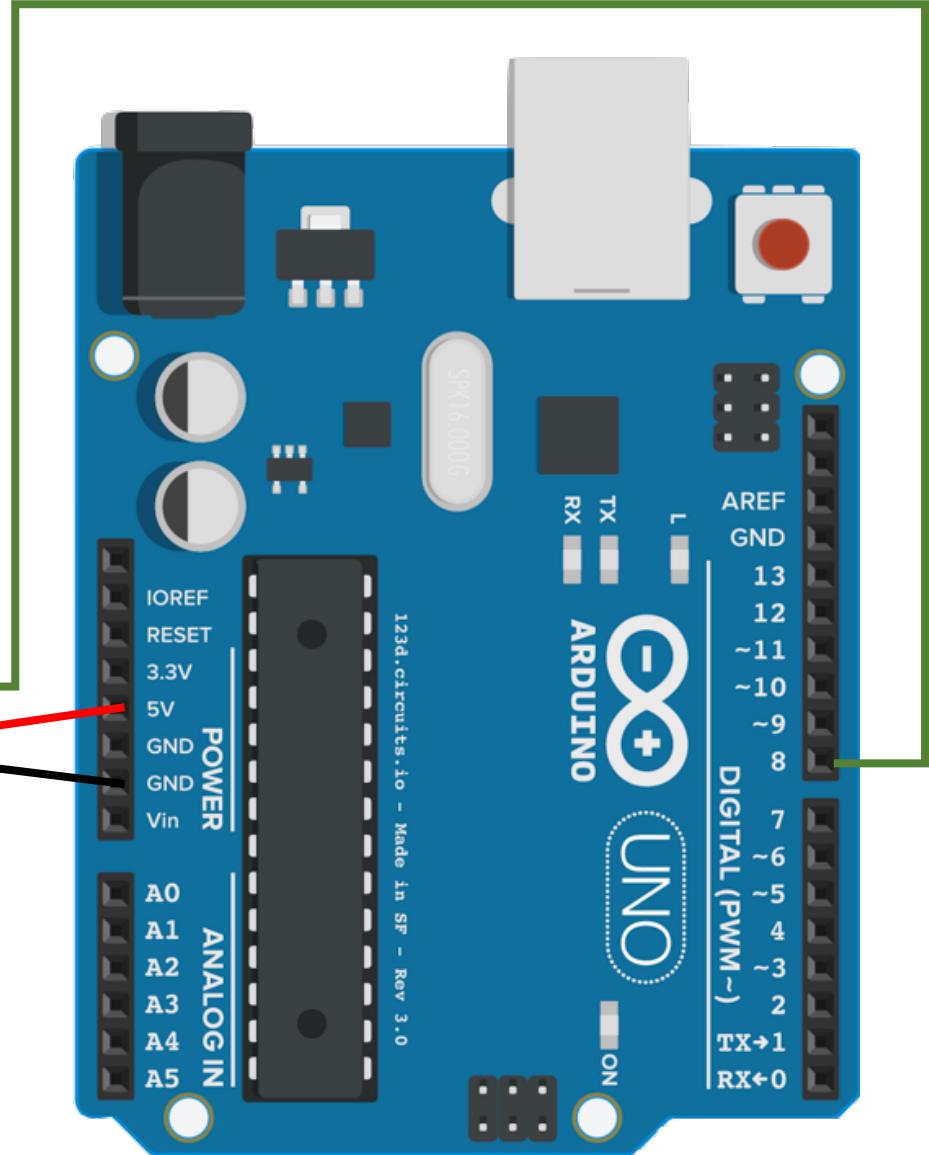
```
1 // Relay and Motor ON/OFF
2
3 int Relay = 8;
4
5 void setup() {
6     pinMode(Relay, OUTPUT);
7 }
8
9 void loop() {
10    digitalWrite(Relay, HIGH);
11    delay(2000);
12    digitalWrite(Relay, LOW);
13    delay(2000);
14 }
```



220 볼트를 다루는 일이기 때문에
전기에 대하여 잘 아는 분의 도움을 받으세요

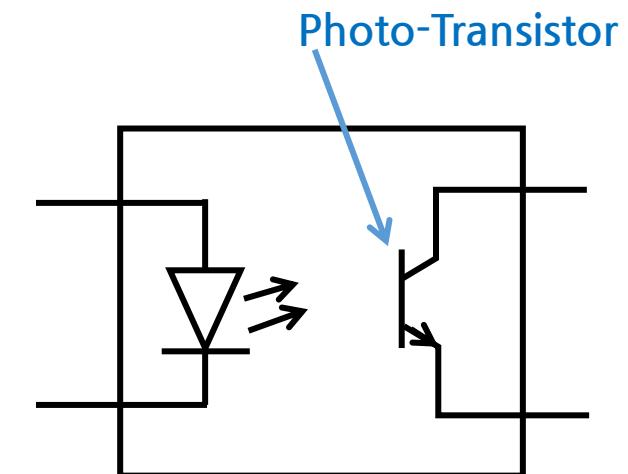
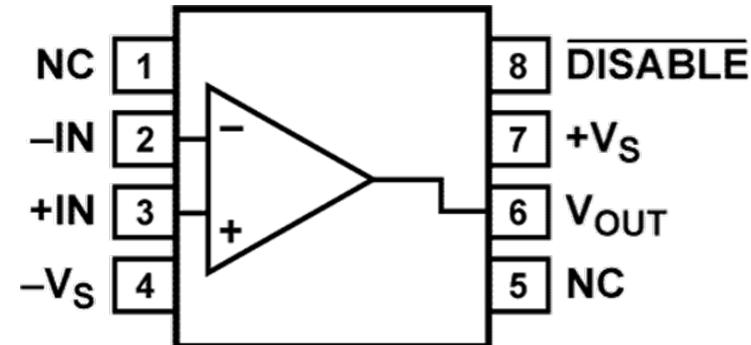
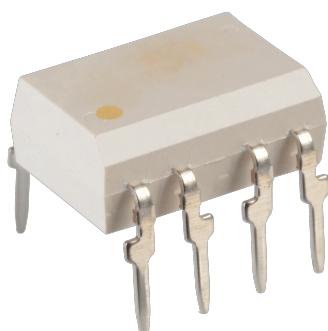
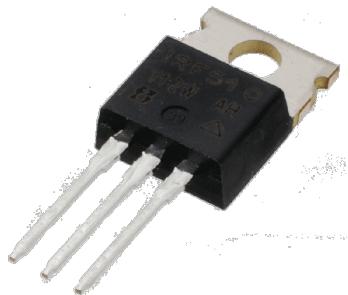


NO
COM
NC



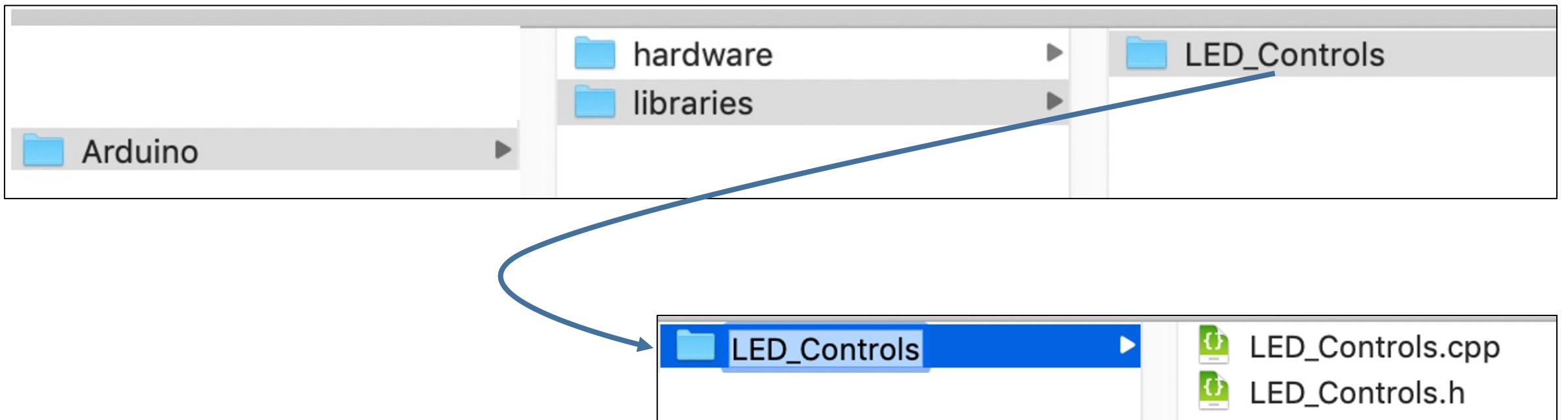
Transistor : BJT, MOSFET

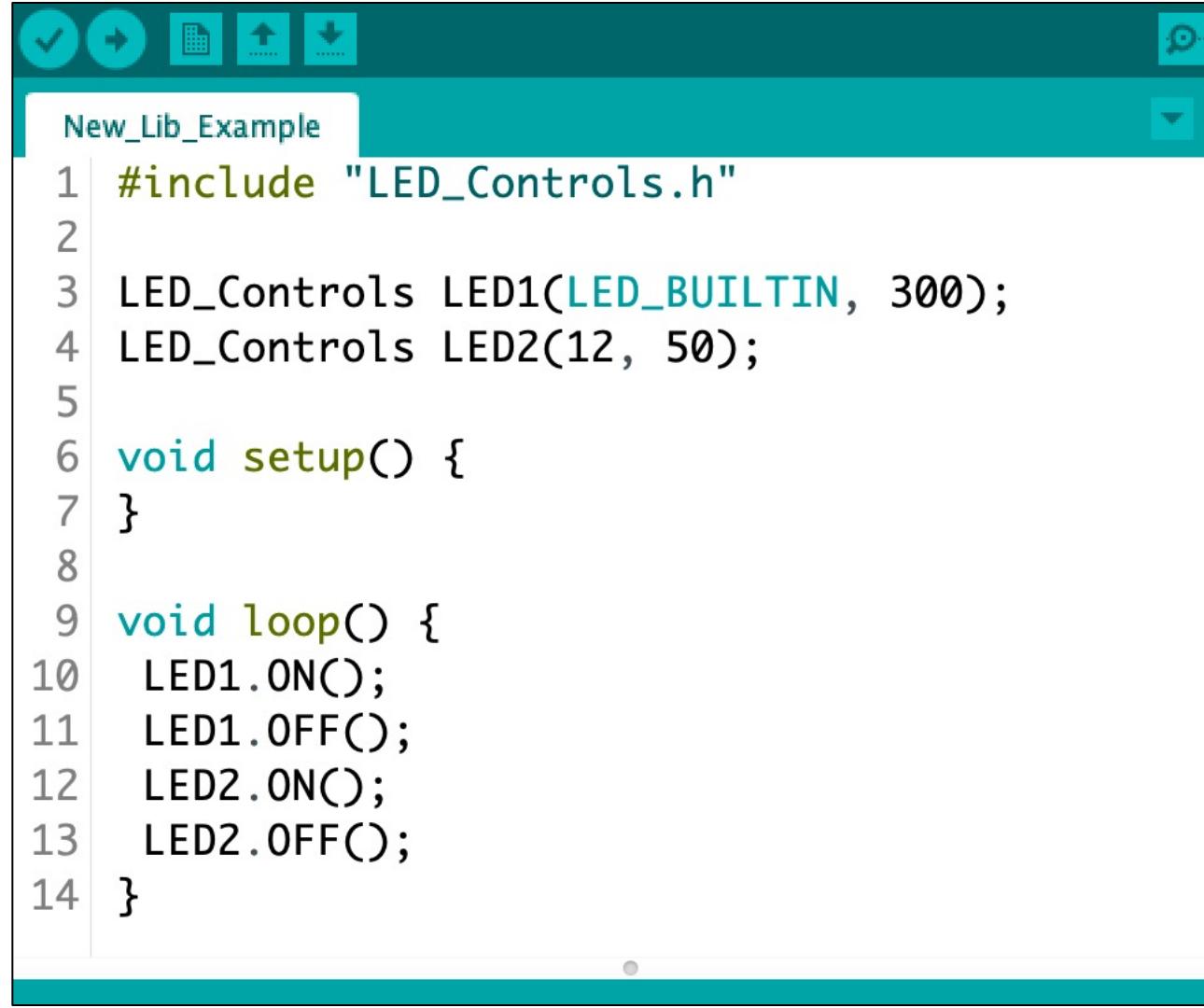
Opto-coupler(Opto isolator)



Arduino Library making







The screenshot shows a software interface for microcontroller development, likely a custom IDE or a modified version of the Arduino IDE. The window title is "New_Lib_Example". The code editor displays the following C++ code:

```
1 #include "LED_Controls.h"
2
3 LED_Controls LED1(LED_BUILTIN, 300);
4 LED_Controls LED2(12, 50);
5
6 void setup() {
7 }
8
9 void loop() {
10    LED1.ON();
11    LED1.OFF();
12    LED2.ON();
13    LED2.OFF();
14 }
```

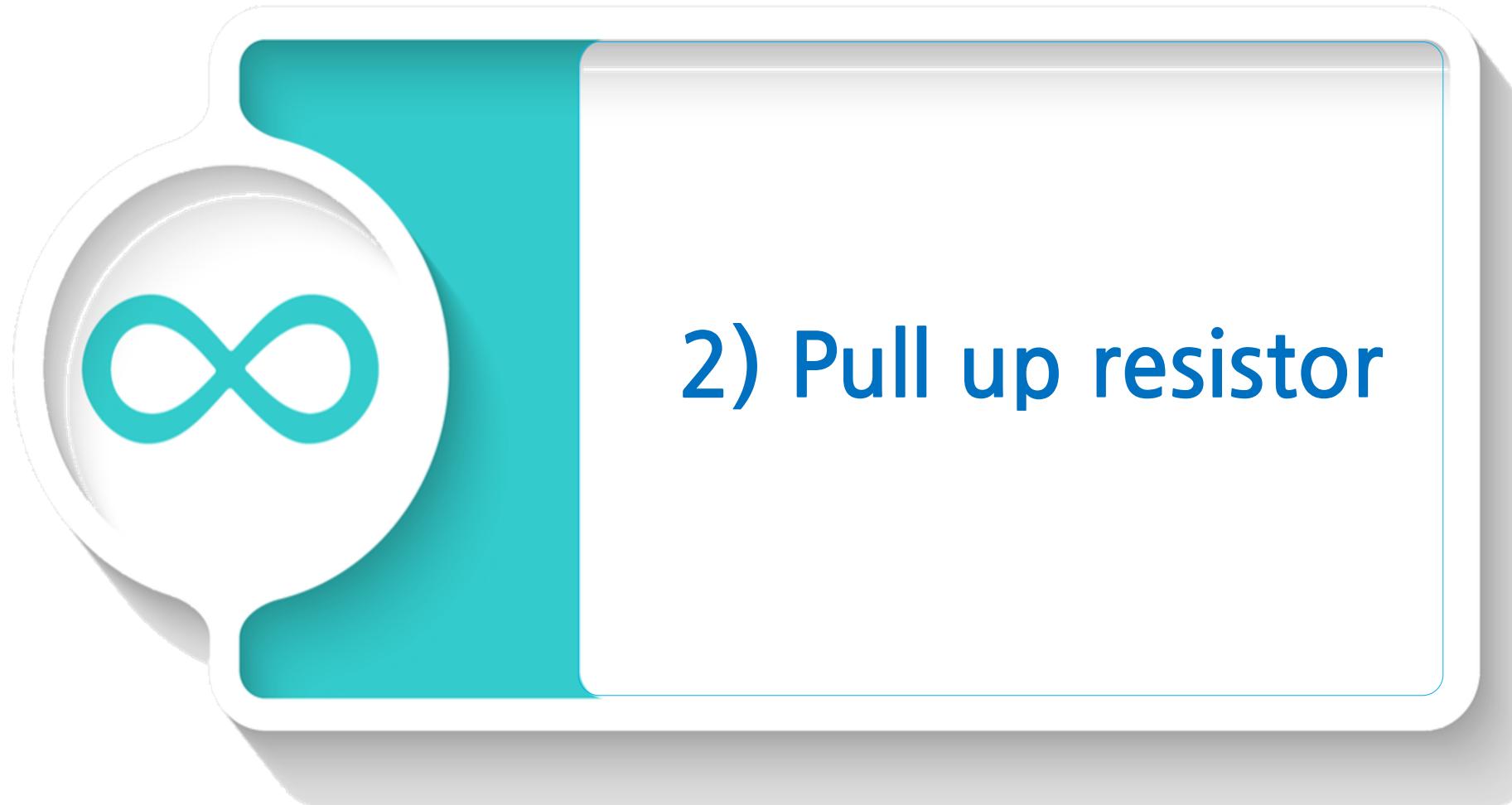
LED_Controls.h

```
1 #ifndef LED_Controls_H
2 #define LED_Controls_H
3 #include "arduino.h"
4
5 class LED_Controls {
6 public:
7     LED_Controls(int pin, int delay);
8     void ON();
9     void OFF();
10    int _pin;
11    int _delay;
12 };
13
14 #endif
15
```

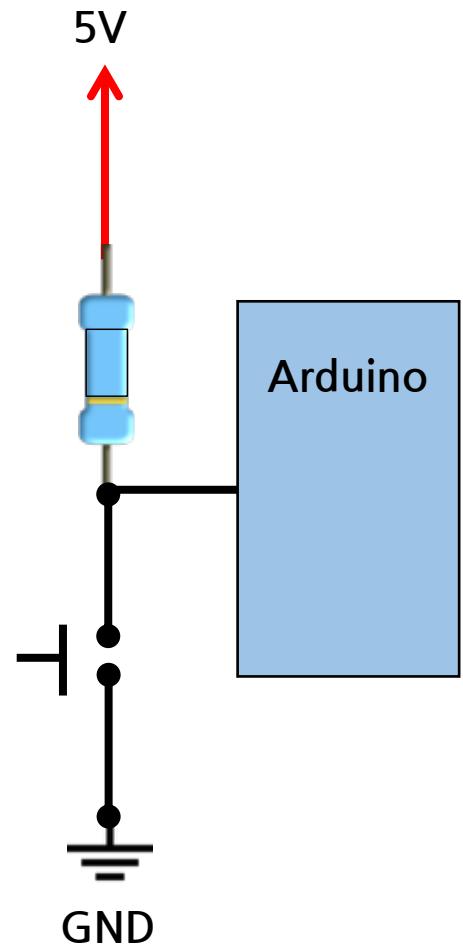
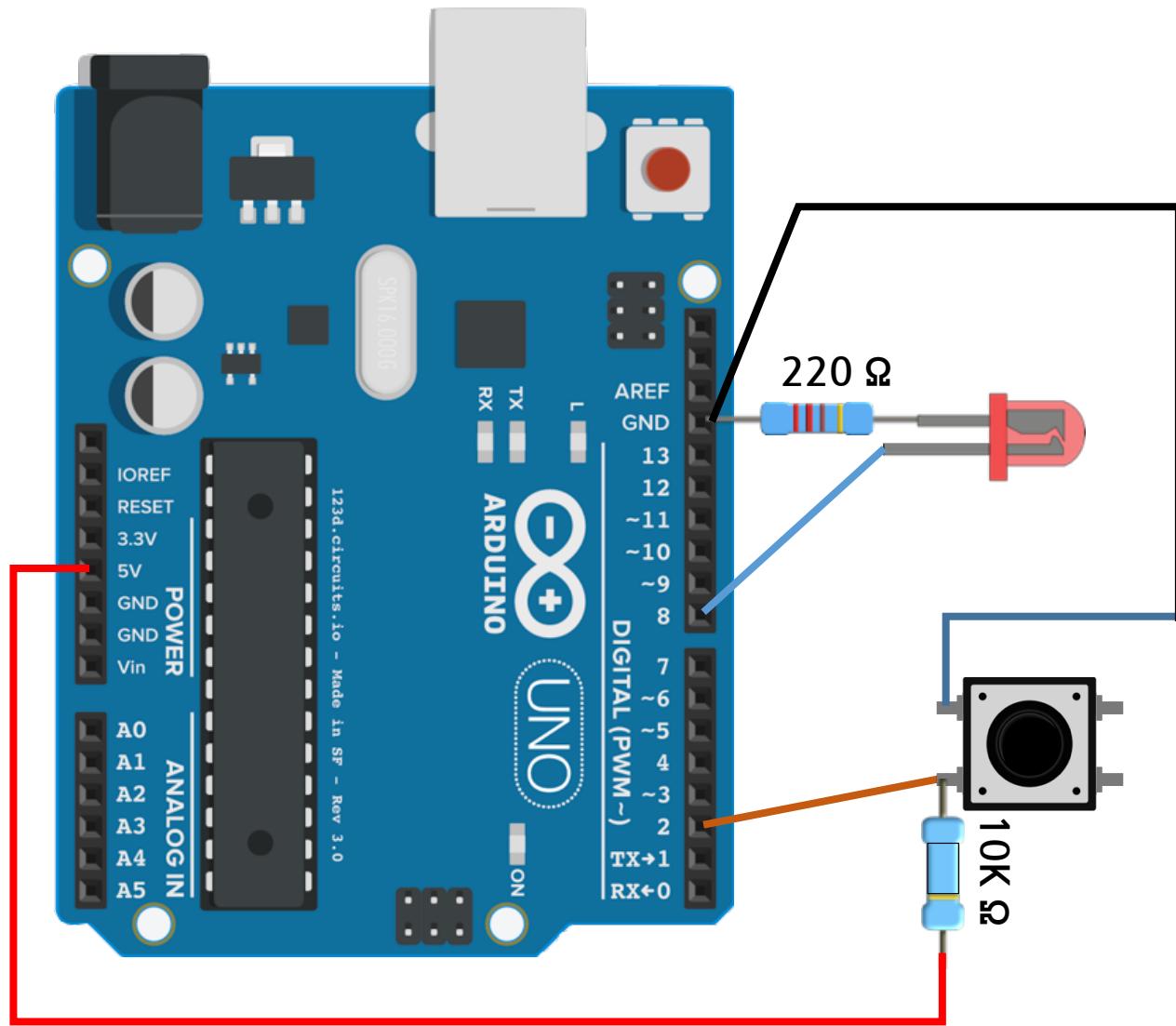
LED_Controls.cpp

```
1 #include "LED_Controls.h"
2
3 LED_Controls::LED_Controls(int pin, int led_delay){
4     pinMode(pin, OUTPUT);
5     _pin = pin;
6     _delay = led_delay;
7 }
8
9 void LED_Controls::ON(){
10    digitalWrite(_pin,HIGH);
11    delay(_delay);
12 }
13
14 void LED_Controls::OFF(){
15    digitalWrite(_pin,LOW);
16    delay(_delay);
17 }
18
```

2) Pull Up resistor



PULL UP circuit

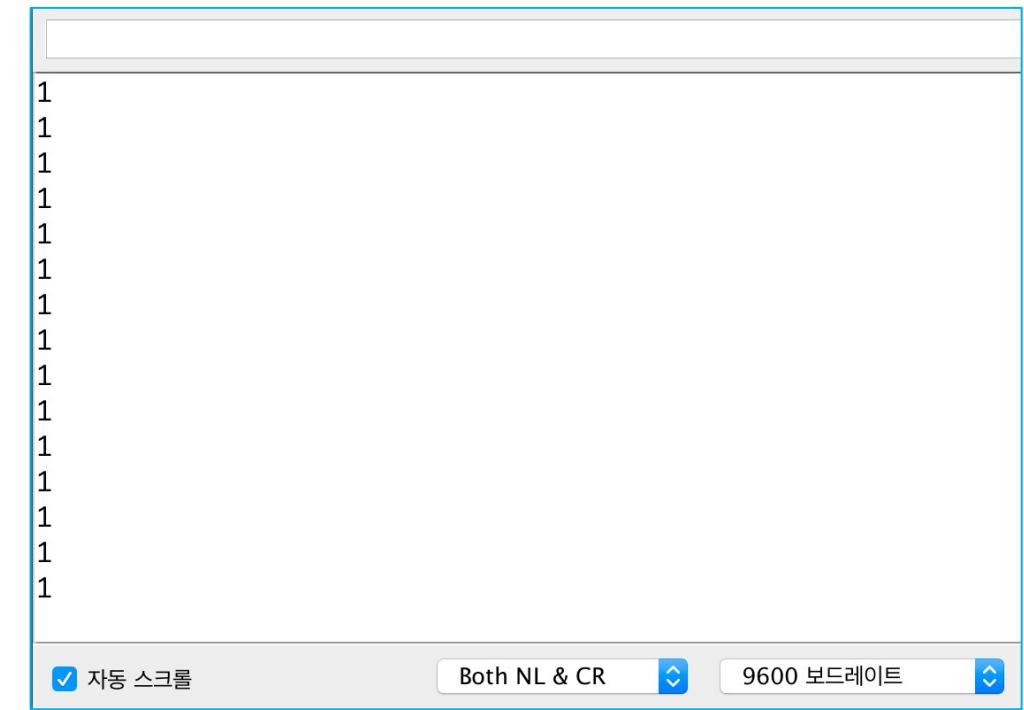


The screenshot shows the Arduino IDE interface with the following details:

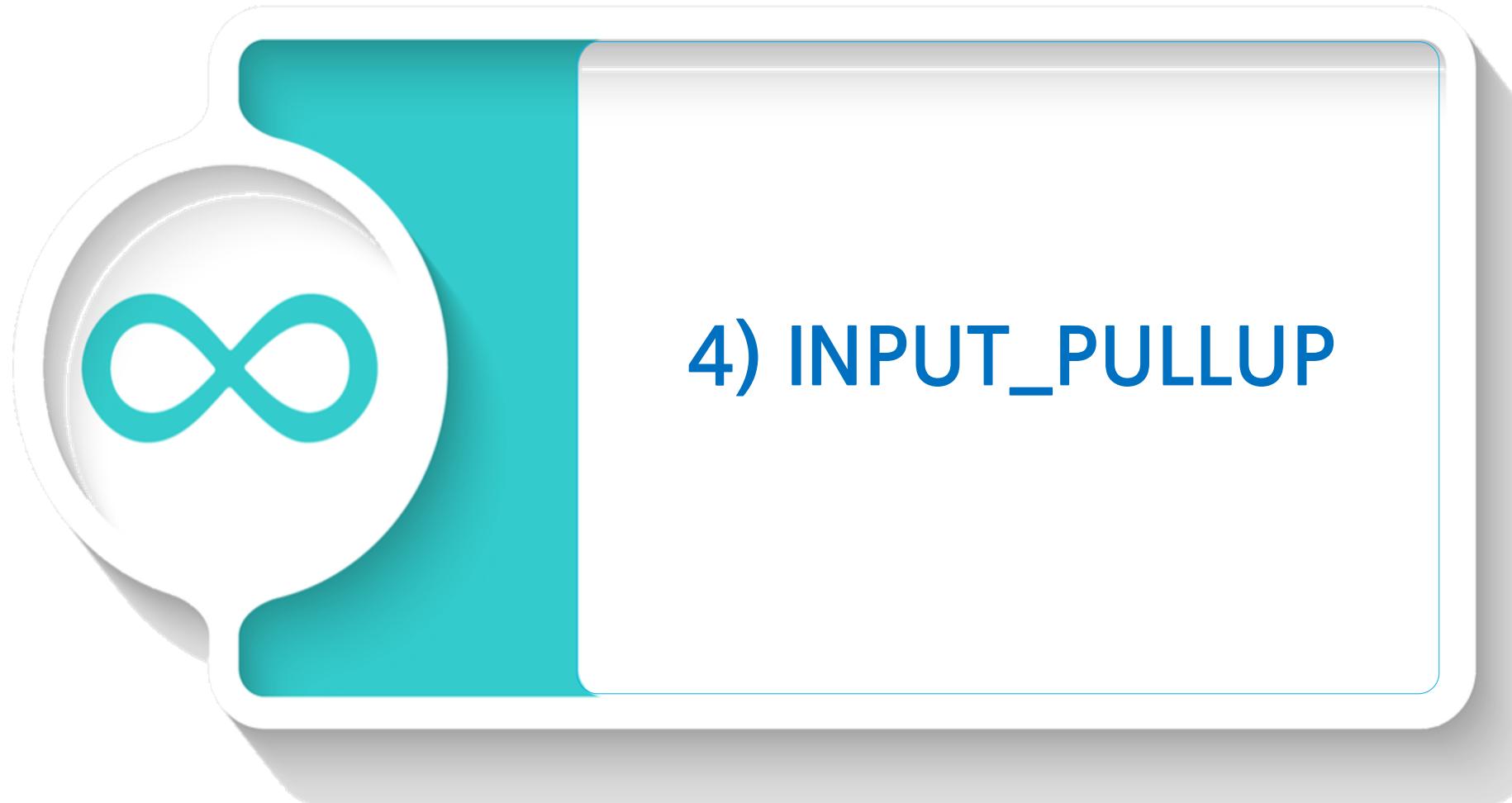
- Title Bar:** Push_button_PULLUP
- Code Area:** Displays the C++ code for a push button example. The code initializes pin 8 as an output for the LED and pin 2 as an input with internal pull-up resistor. It sets up the serial port at 9600 baud. In the loop, it reads the state of pin 2 and prints "1" to the serial monitor if the button is pressed (value == 1), or prints "0" if it is not.
- Serial Monitor:** Shows the output of the serial port. The text "1" is repeated 15 times, indicating the button is continuously pressed.
- Bottom Status Bar:** Includes checkboxes for "자동 스크롤" (Auto Scroll) and "Both NL & CR", and a dropdown for "9600 보드레이트" (Board Rate).

```
Push_button_PULLUP

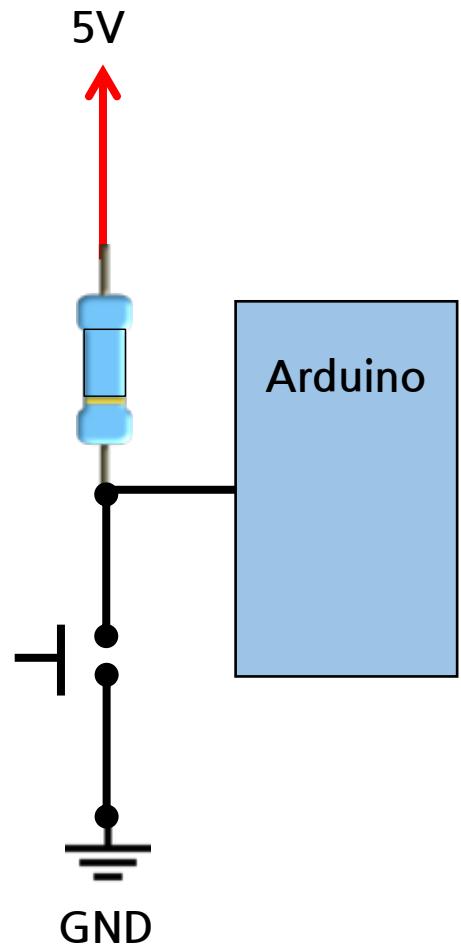
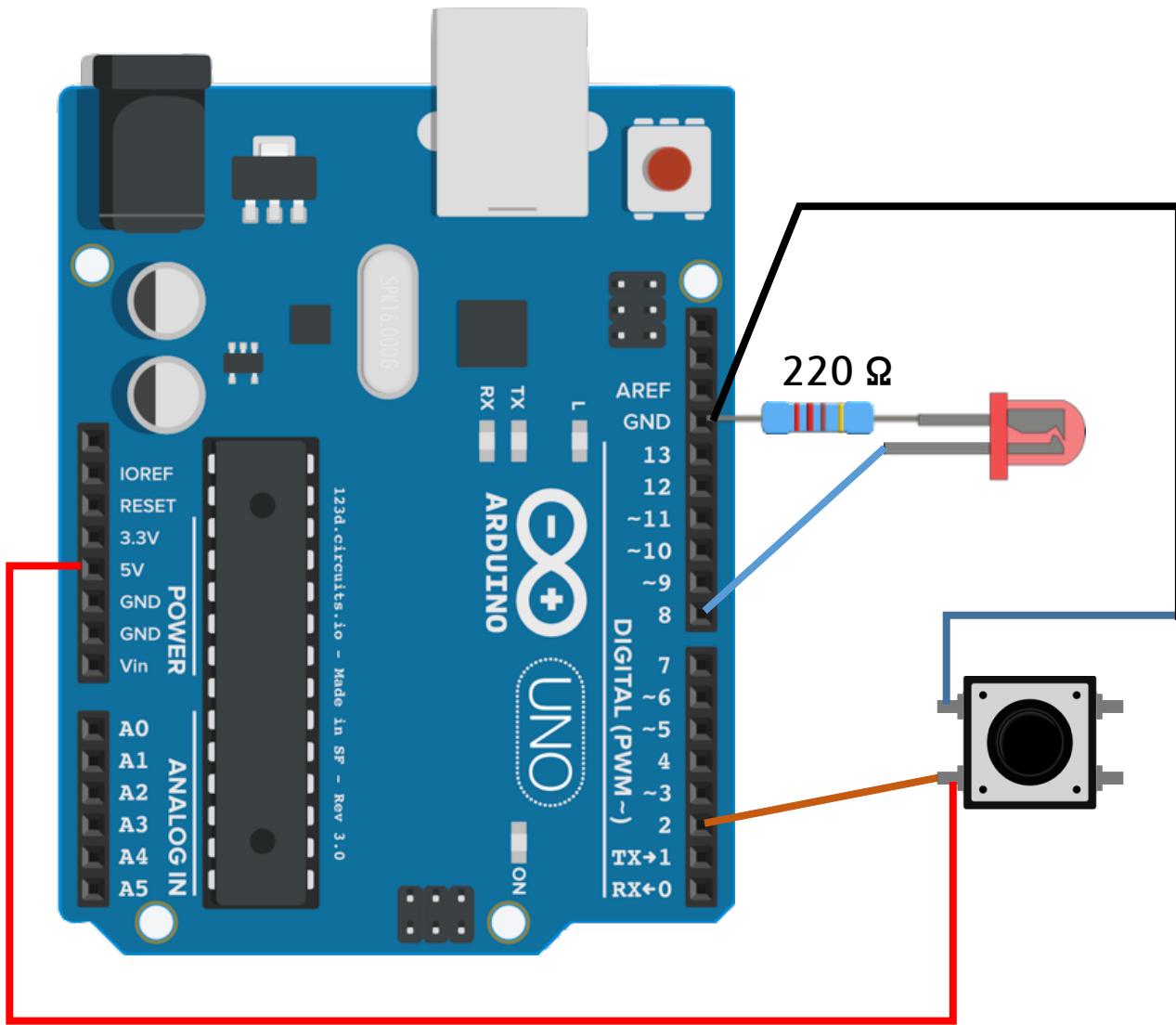
1 // Push button to turn ON/OFF LED
2
3 int LED = 8 ; // Global variable
4 int inputPin = 2 ; // Global variable
5
6 void setup() {
7   pinMode(LED, OUTPUT) ;
8   pinMode(inputPin, INPUT_PULLUP) ;
9   Serial.begin(9600) ;
10 }
11
12 void loop() {
13   int value = digitalRead(2) ; // Local variable
14
15   if( value == 1 ) {
16     digitalWrite(LED, 1) ;
17   }
18   else {
19     digitalWrite(LED, 0) ;
20   }
21   Serial.println(value) ;
22   delay(1000) ;
23 }
```



4) Internal Pull Up



PULL UP circuit



The screenshot shows the Arduino IDE interface with the following details:

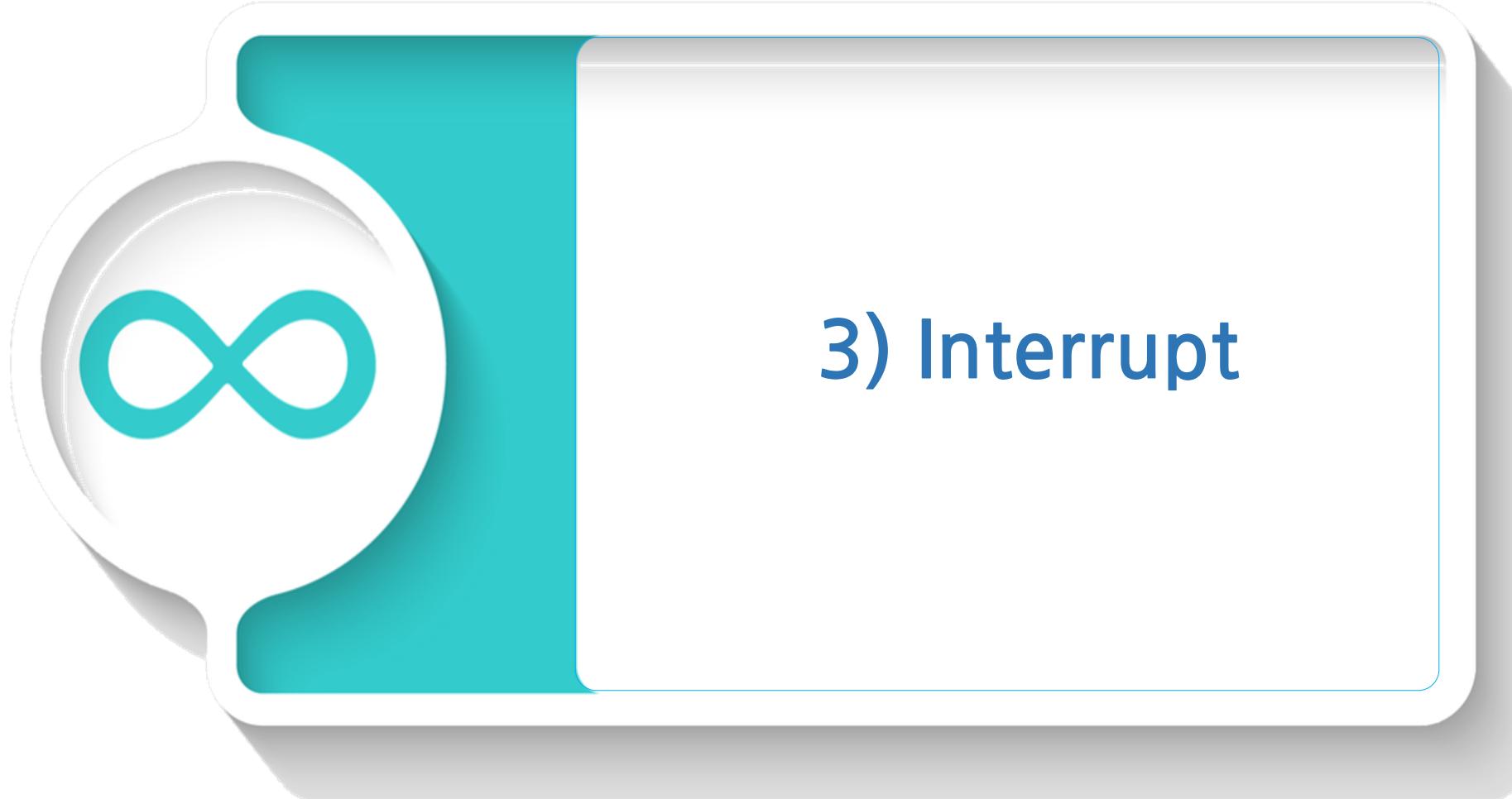
- Title Bar:** Push_button_PULLUP
- Tool Buttons:** Checkmark, Refresh, File, Open, Save, Print, Zoom.
- Code Area:** Displays the C++ code for the Push_button_PULLUP sketch.
- Code Content:**

```
1 // Push button to turn ON/OFF LED
2
3 int LED = 8 ; // Global variable
4 int inputPin = 2 ; // Global variable
5
6 void setup() {
7   pinMode(LED, OUTPUT) ;
8   pinMode(inputPin, INPUT_PULLUP) ;
9   Serial.begin(9600) ;
10 }
11
12 void loop() {
13   int value = digitalRead(2) ; // Local variable
14
15   if( value == 1 ) {
16     digitalWrite(LED, 1) ;
17   }
18   else {
19     digitalWrite(LED, 0) ;
20   }
21   Serial.println(value) ;
22   delay(1000) ;
23 }
```

The screenshot shows the Serial Monitor window with the following details:

- Output Content:** The monitor displays the character '1' repeated 15 times, indicating the state of the push button connected to pin 2.
- Bottom Controls:**

 - 자동 스크롤
 - Both NL & CR
 - 9600 보드레이트



3) Interrupt

Polling



Interrupt



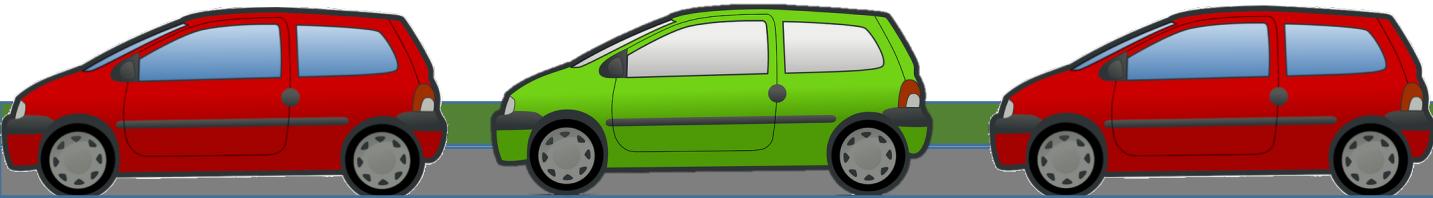
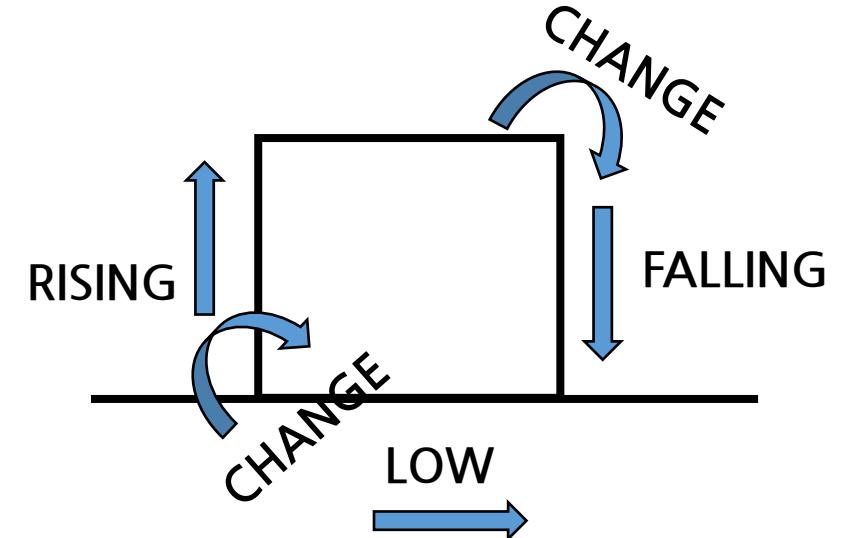
2) Interrupt

우선 통행

attachInterrupt(pin#, function name, condition)

pin# = attach interrupt pin number

condition: LOW, RISING, FALLING, CHANGE



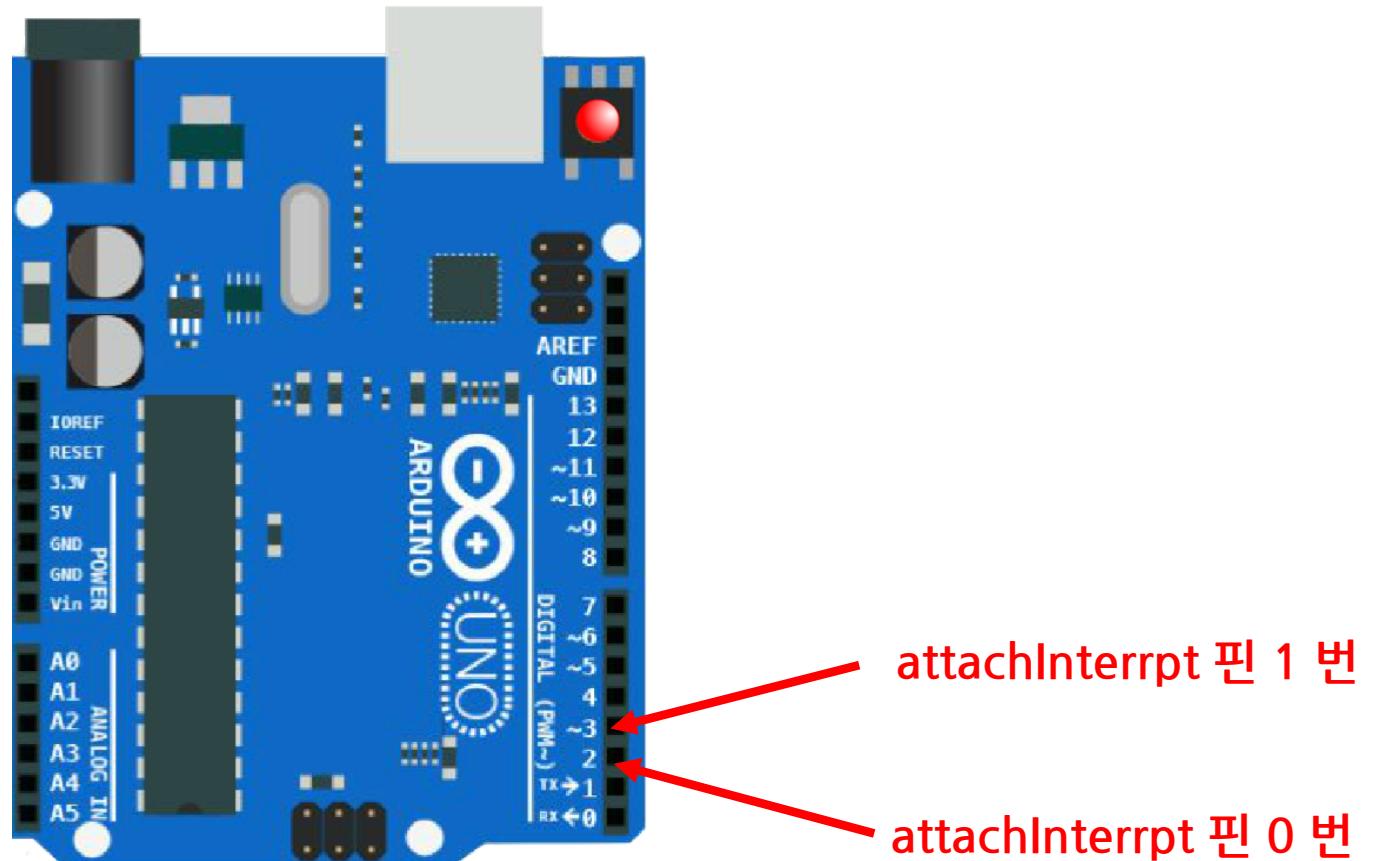
2) Interrupt

아두이노 우노 보드 경우 `attachInterrupt` 0 번과 1번은 디지털 핀 D2 와 D3 이다.

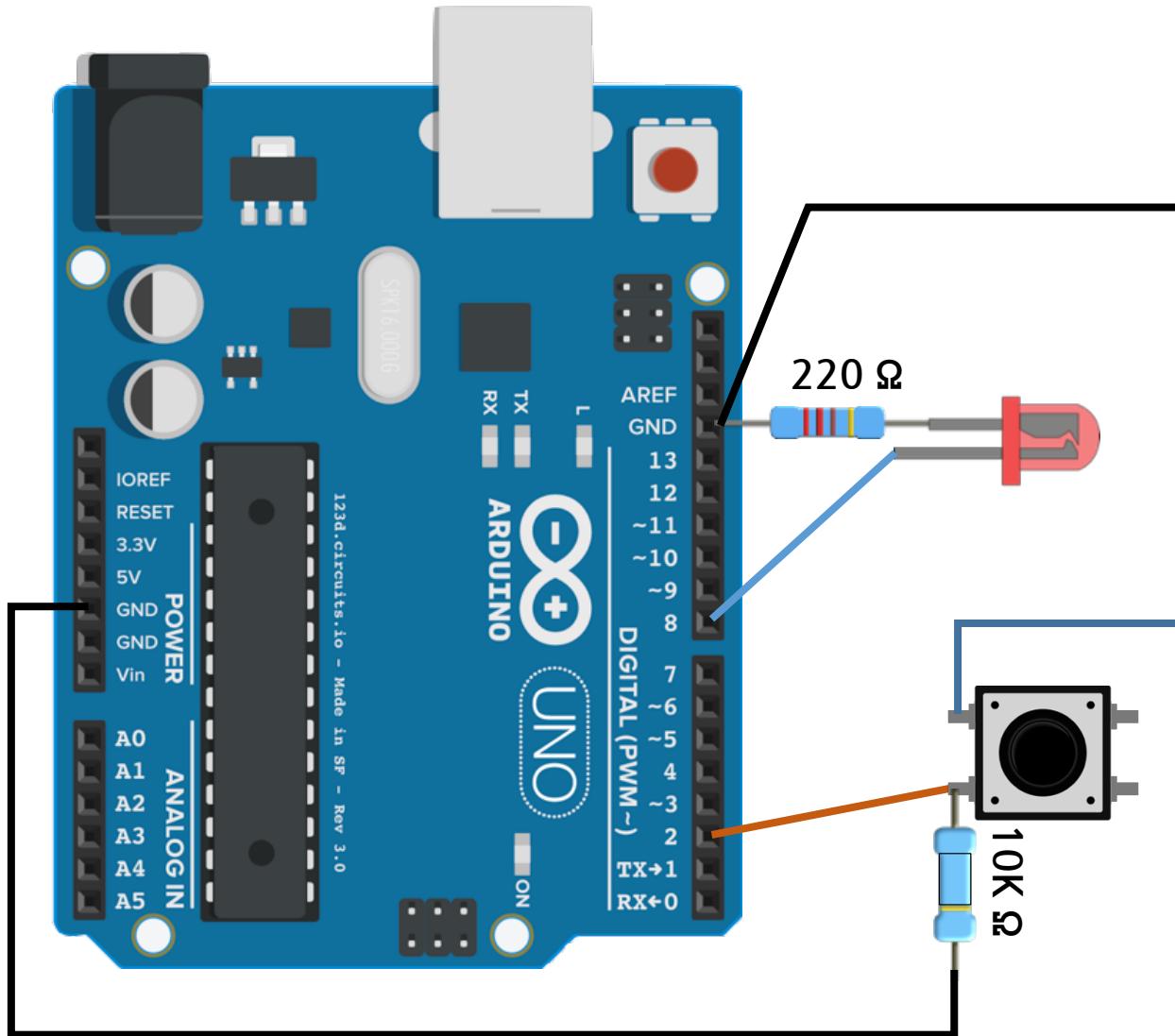
프로젝트 :

13번 핀에 연결한 LED 는 켜져 있는 상태를 유지한다.

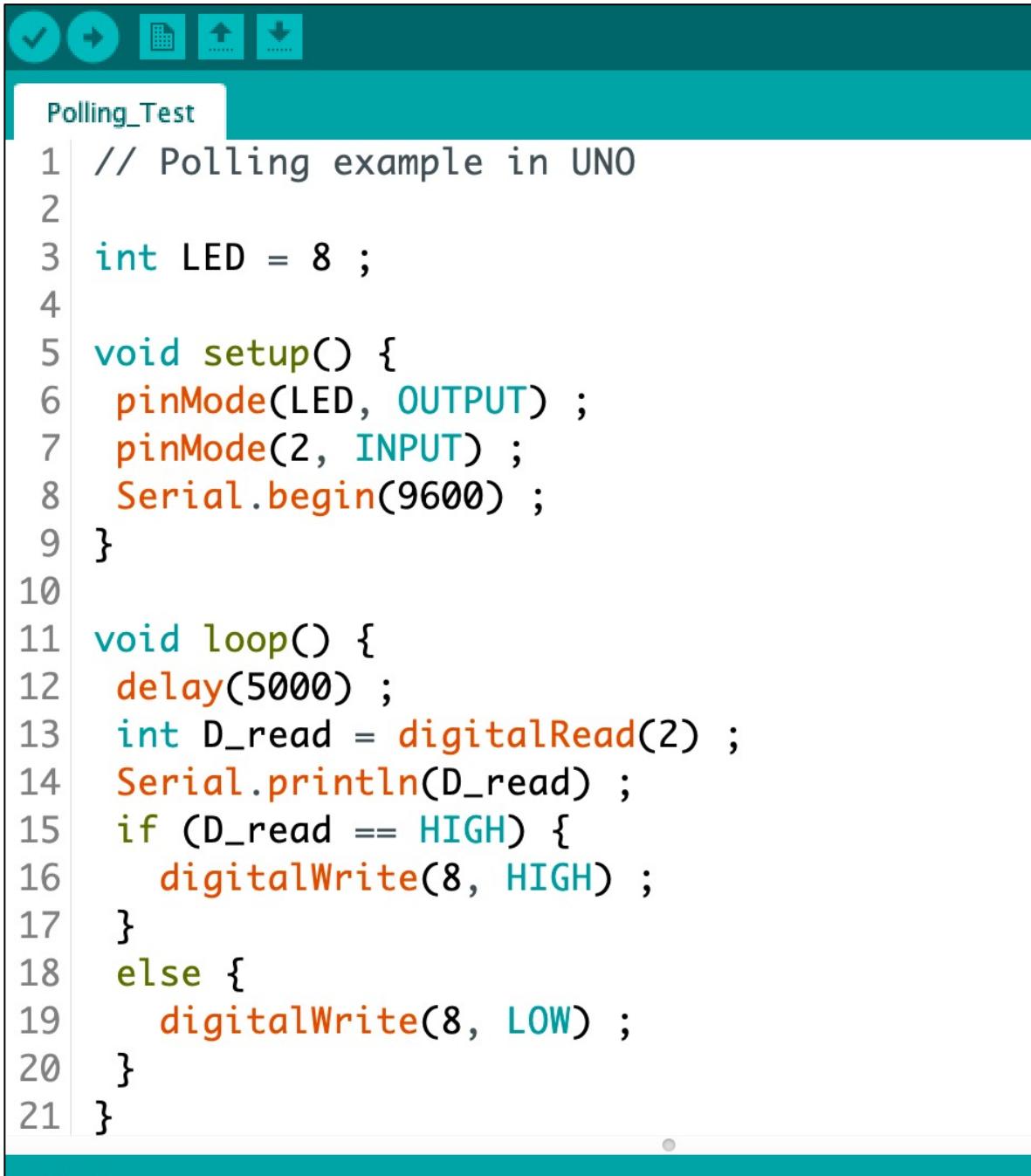
푸쉬 버튼을 `attachInterrupt` 0 번에
연결하고, 푸시 했을 때 8 번핀과 연결된
LED 가 켜지는 회로와 스케치 작성한다.



PULL DOWN circuit



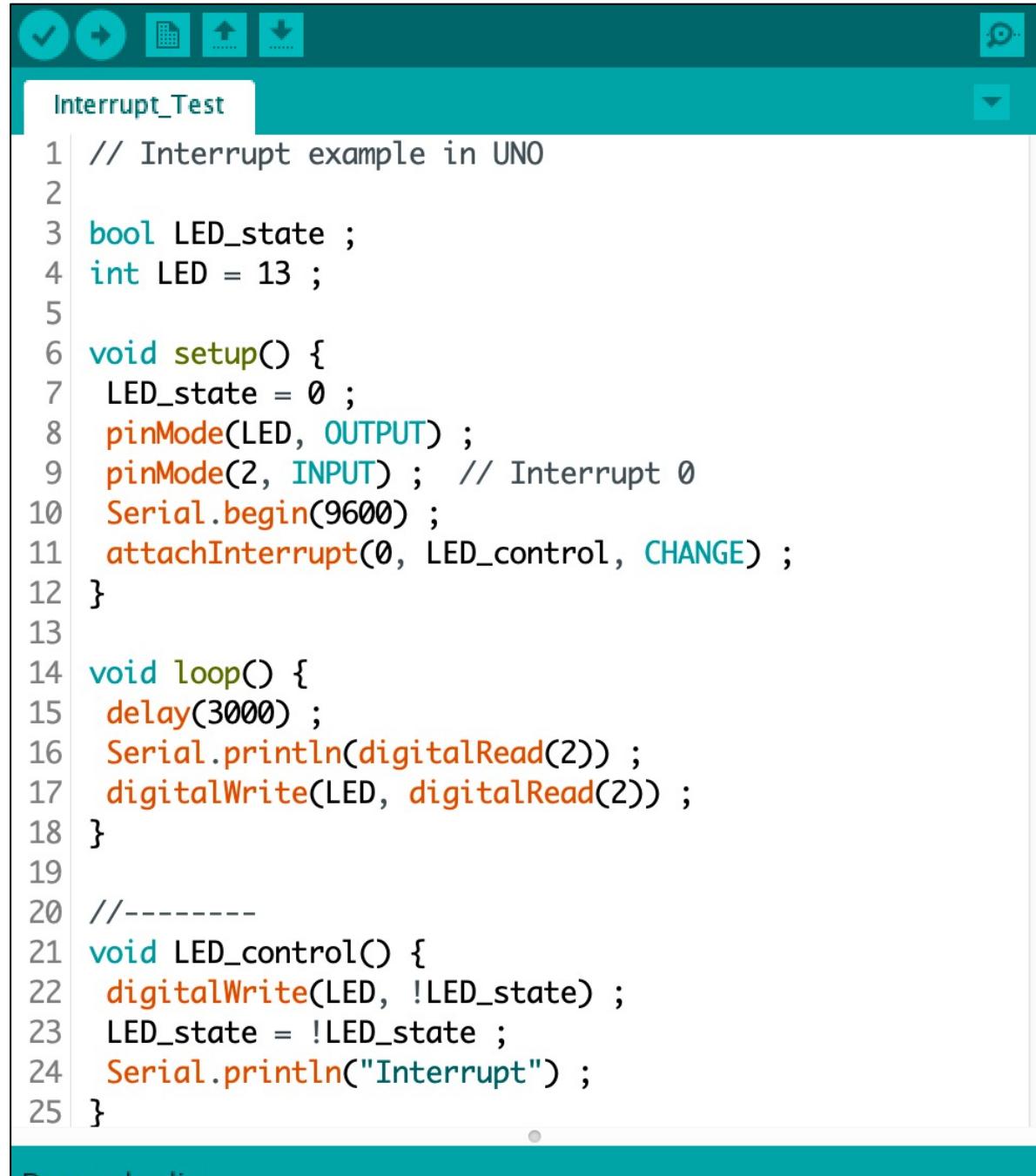
Polling



The screenshot shows the Arduino IDE interface with a sketch titled "Polling_Test". The code implements a polling-based control system for an LED connected to pin 8, controlled by a pushbutton connected to pin 2. The sketch includes setup and loop functions, with the loop function containing a digital read operation and an if-else conditional statement.

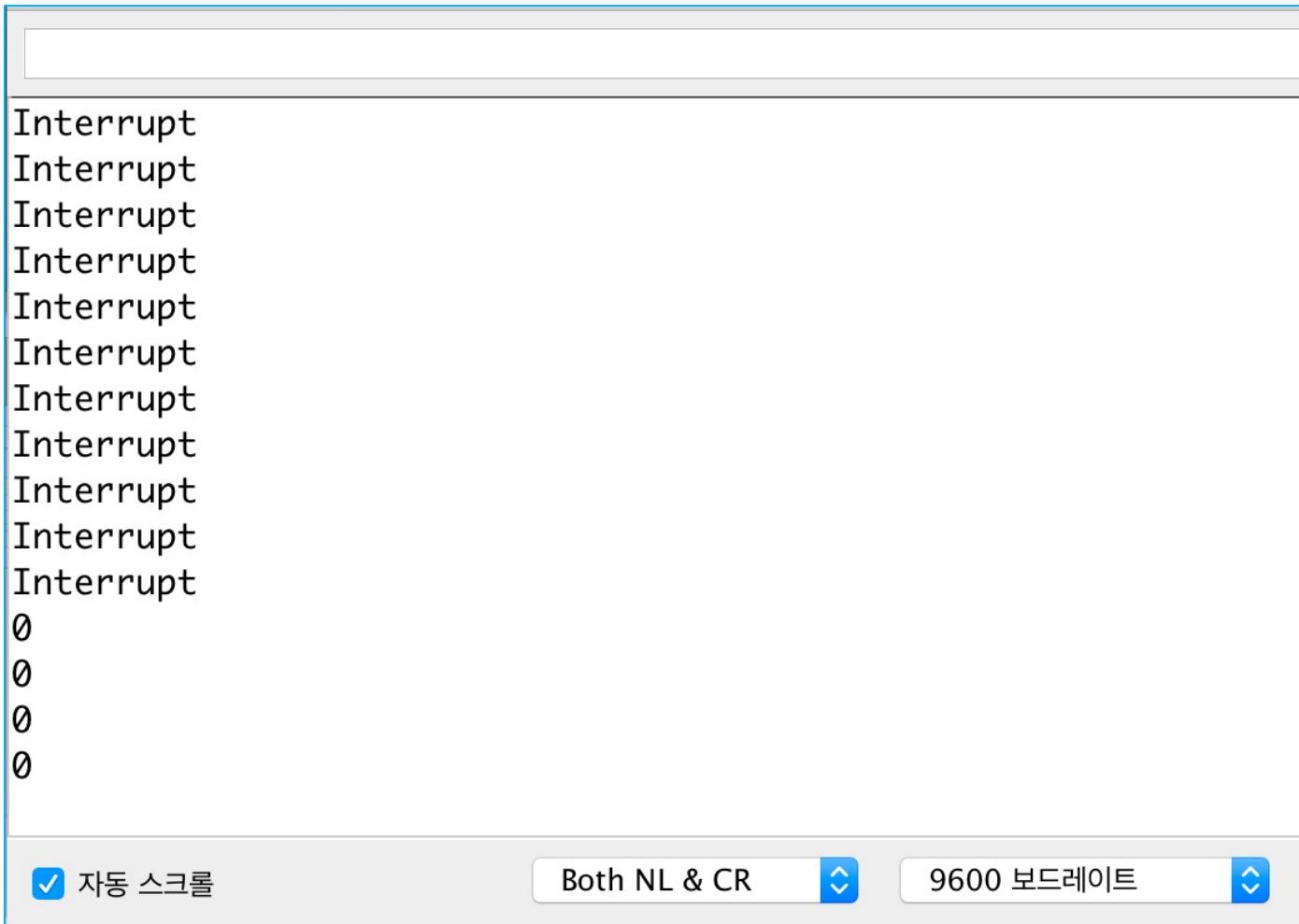
```
1 // Polling example in UNO
2
3 int LED = 8 ;
4
5 void setup() {
6     pinMode(LED, OUTPUT) ;
7     pinMode(2, INPUT) ;
8     Serial.begin(9600) ;
9 }
10
11 void loop() {
12     delay(5000) ;
13     int D_read = digitalRead(2) ;
14     Serial.println(D_read) ;
15     if (D_read == HIGH) {
16         digitalWrite(8, HIGH) ;
17     }
18     else {
19         digitalWrite(8, LOW) ;
20     }
21 }
```

Interrupt

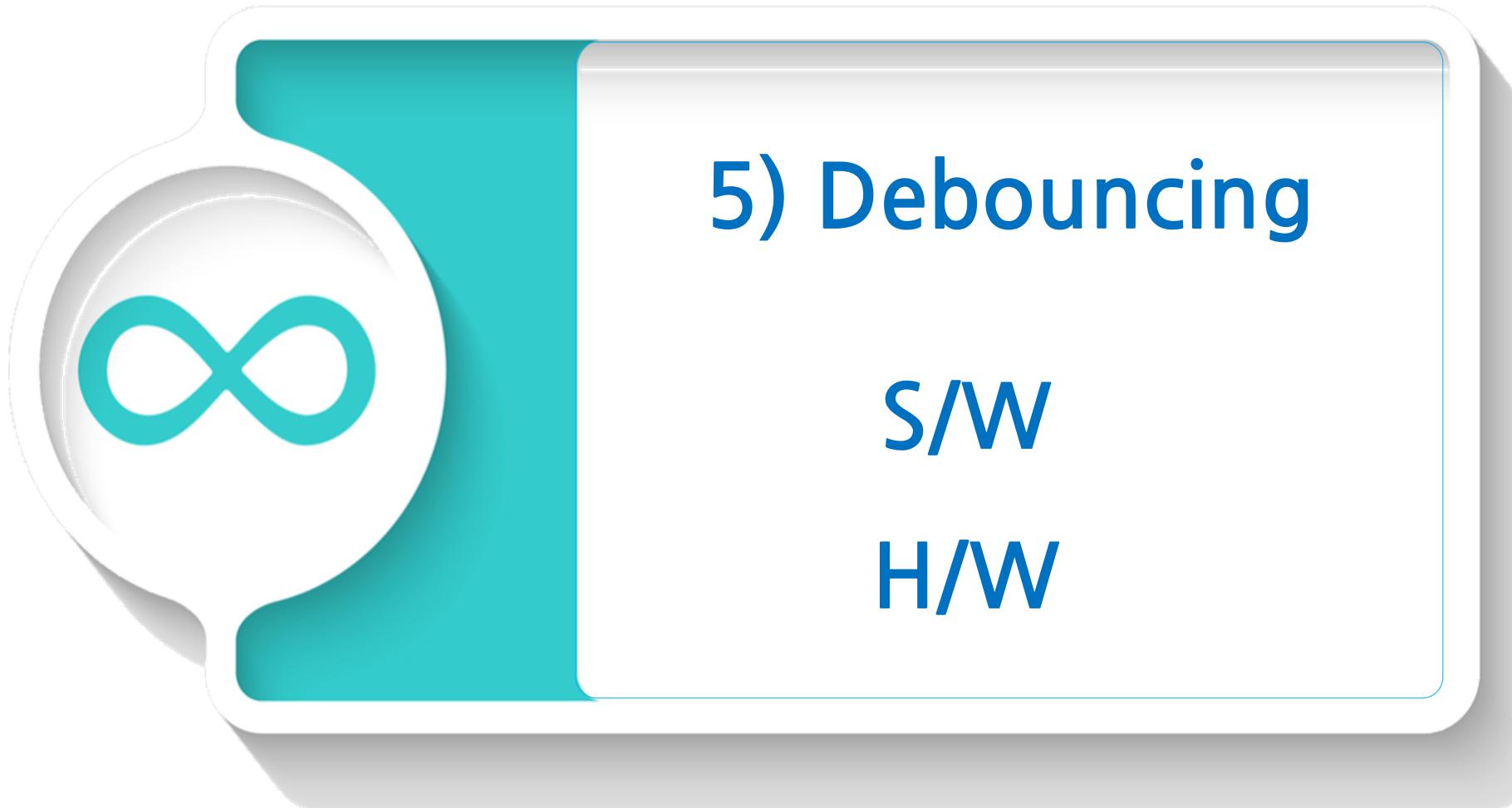


The screenshot shows the Arduino IDE interface with a sketch named "Interrupt_Test". The code implements a digital input interrupt on pin 2. When the state of pin 2 changes, it triggers a function called "LED_control". This function toggles the state of an LED connected to pin 13 and prints a message to the Serial monitor.

```
1 // Interrupt example in UNO
2
3 bool LED_state ;
4 int LED = 13 ;
5
6 void setup() {
7     LED_state = 0 ;
8     pinMode(LED, OUTPUT) ;
9     pinMode(2, INPUT) ; // Interrupt 0
10    Serial.begin(9600) ;
11    attachInterrupt(0, LED_control, CHANGE) ;
12 }
13
14 void loop() {
15     delay(3000) ;
16     Serial.println(digitalRead(2)) ;
17     digitalWrite(LED, digitalRead(2)) ;
18 }
19
20 //-----
21 void LED_control() {
22     digitalWrite(LED, !LED_state) ;
23     LED_state = !LED_state ;
24     Serial.println("Interrupt") ;
25 }
```



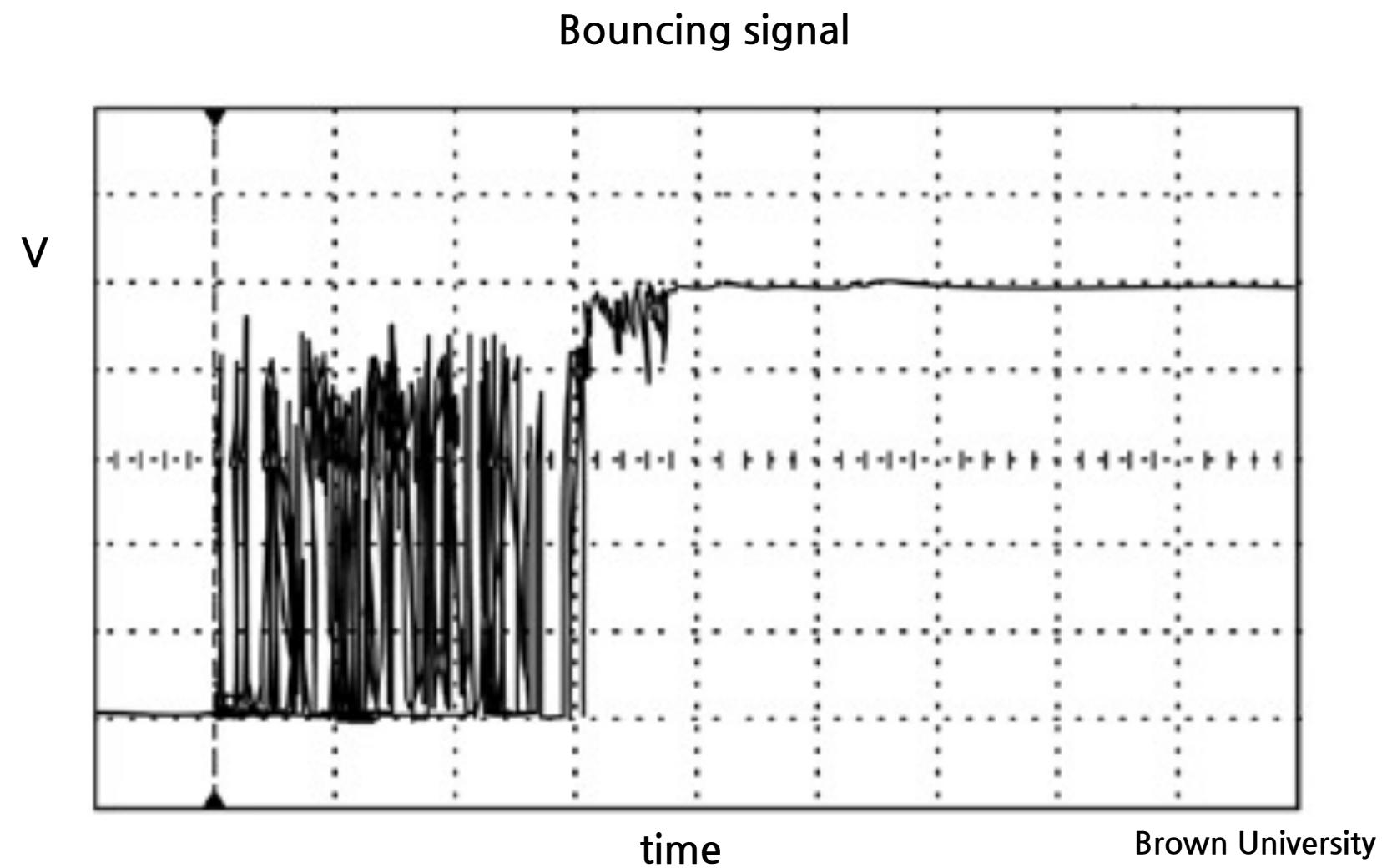
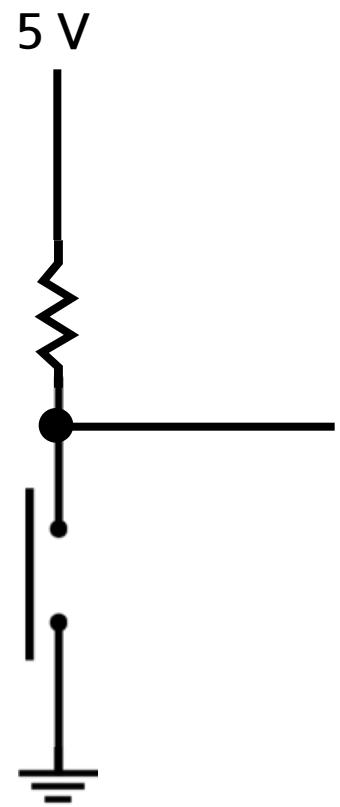
5) Debouncing



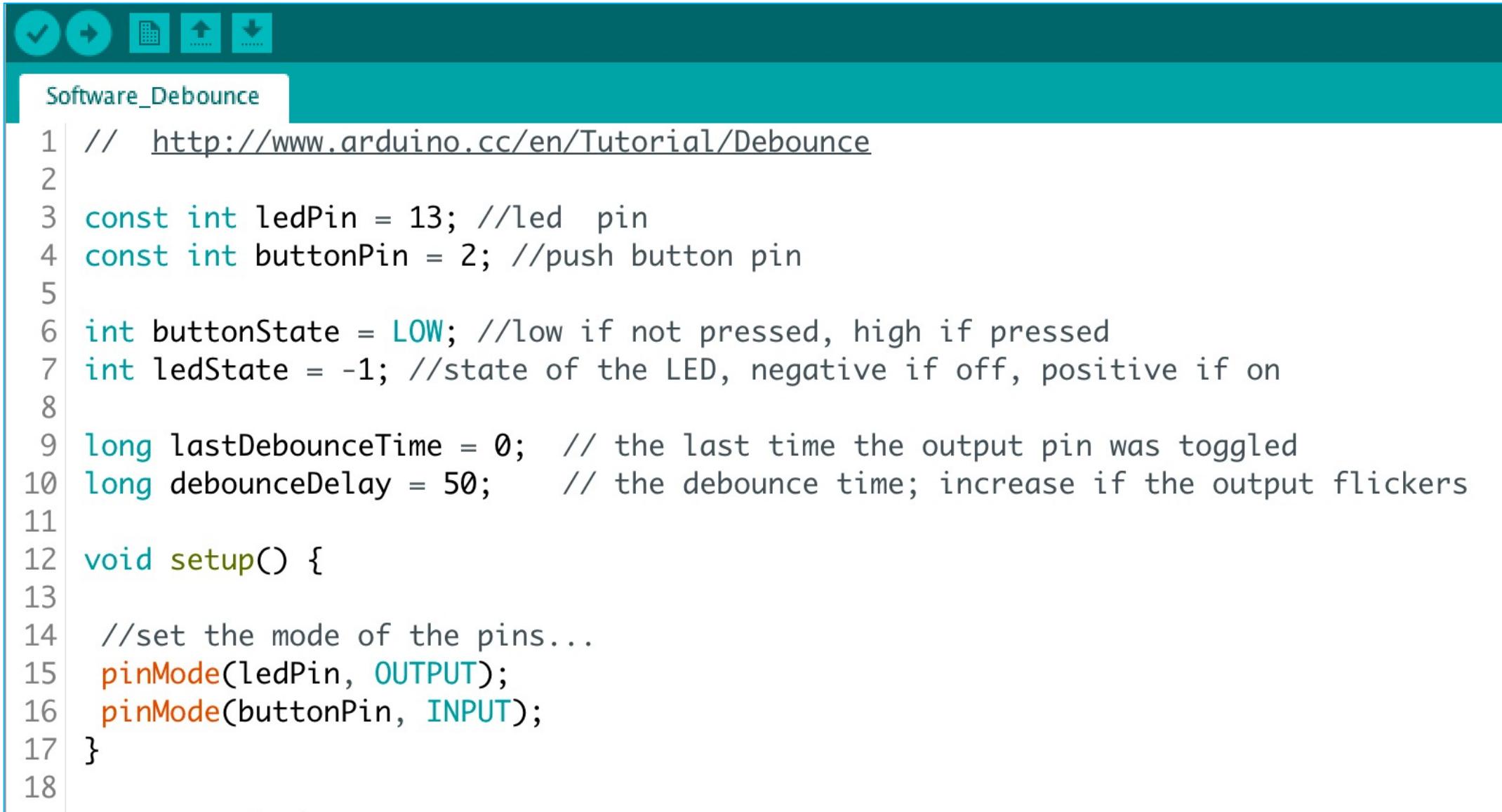
5) Debouncing

S/W

H/W



Software Debounce



The screenshot shows the Arduino IDE interface with the title bar "Software_Debounce". The code editor contains a C++ program for a microcontroller. The code defines variables for LED and button pins, initializes them, and sets up adebounce mechanism using a timer and state tracking.

```
1 // http://www.arduino.cc/en/Tutorial/Debounce
2
3 const int ledPin = 13; //led pin
4 const int buttonPin = 2; //push button pin
5
6 int buttonState = LOW; //low if not pressed, high if pressed
7 int ledState = -1; //state of the LED, negative if off, positive if on
8
9 long lastDebounceTime = 0; // the last time the output pin was toggled
10 long debounceDelay = 50; // the debounce time; increase if the output flickers
11
12 void setup() {
13
14 //set the mode of the pins...
15 pinMode(ledPin, OUTPUT);
16 pinMode(buttonPin, INPUT);
17 }
18
```

```
19 void loop() {  
20     buttonState = digitalRead(buttonPin);  
21  
22     //filter out any noise by setting a time buffer  
23     if ( (millis() - lastDebounceTime) > debounceDelay) {  
24  
25         //if the button has been pressed, lets toggle the LED from "off to on" or "on to off"  
26         if ( (buttonState == HIGH) && (ledState < 0) ) {  
27  
28             digitalWrite(ledPin, HIGH); //turn LED on  
29             ledState = -ledState; //now the LED is on, we need to change the state  
30             lastDebounceTime = millis(); //set the current time  
31         }  
32         else if ( (buttonState == HIGH) && (ledState > 0) ) {  
33  
34             digitalWrite(ledPin, LOW); //turn LED off  
35             ledState = -ledState; //now the LED is off, we need to change the state  
36             lastDebounceTime = millis(); //set the current time  
37         }  
38     }  
39 } //== End of loop =====
```

6) Rotary Encoder



프로젝트 : **Rotary Encoder**

목적 : Encoding 이해, 위치 및 길이 측정 방법

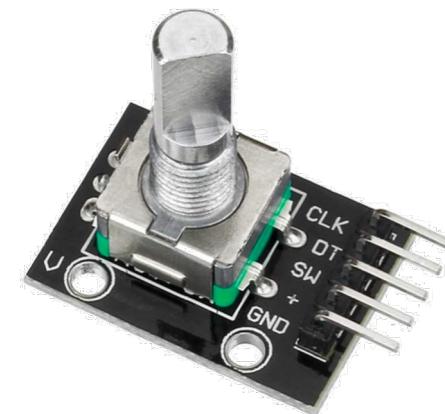
준비물 :

아두이노 우노 1 개

브래드보드 1 개

KY040 Encoder 1 개

점퍼케이블



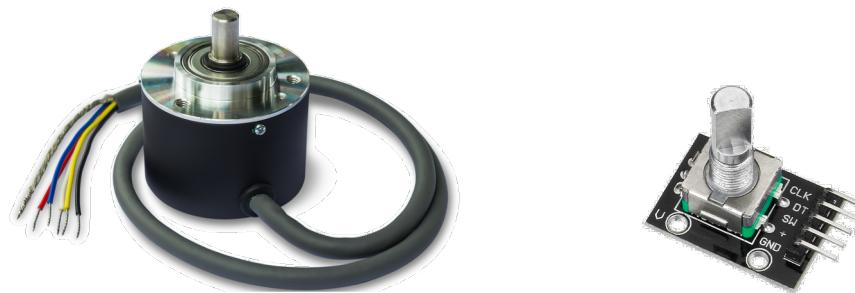
6) Rotary Encoder

인코더 : 포지션 및 회전수 측정

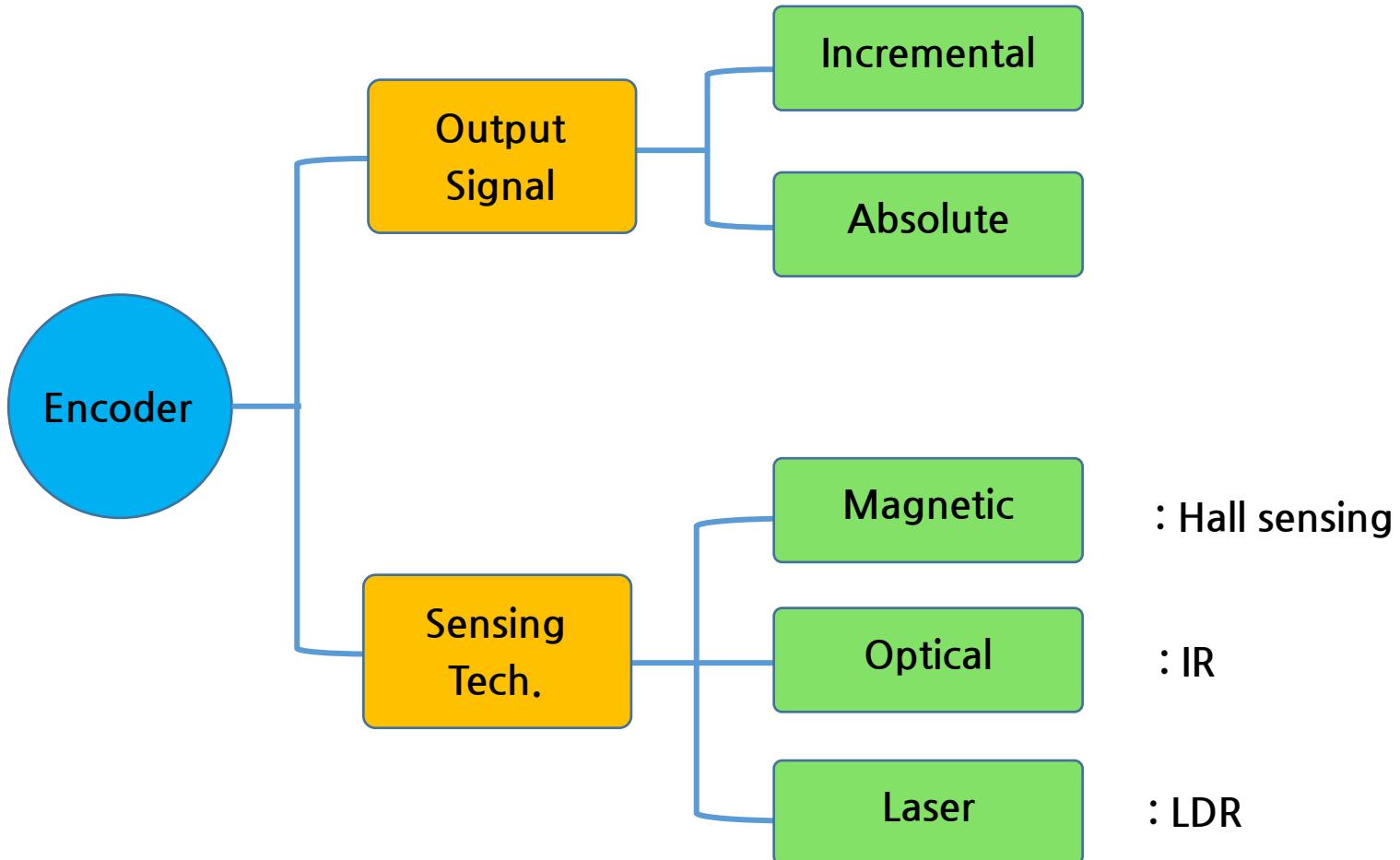
저항 사용 방법 : 저항 크기 변화로 위치와 회전 방향

Optic 사용 방법 : 펄스 변화로 위치와 회전 방향

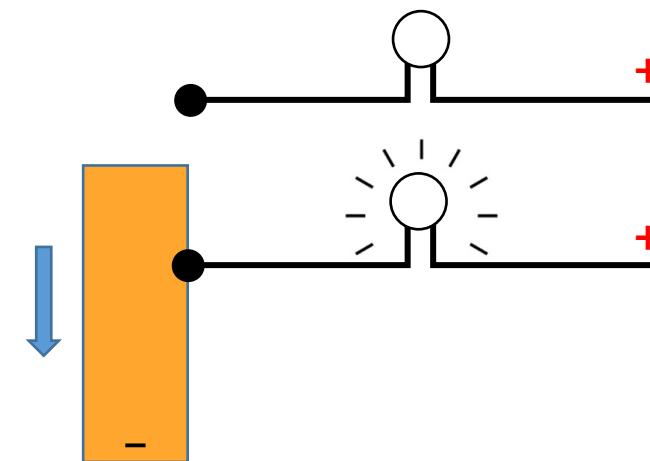
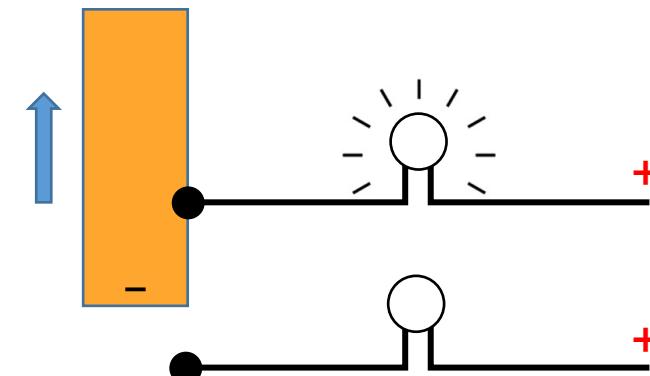
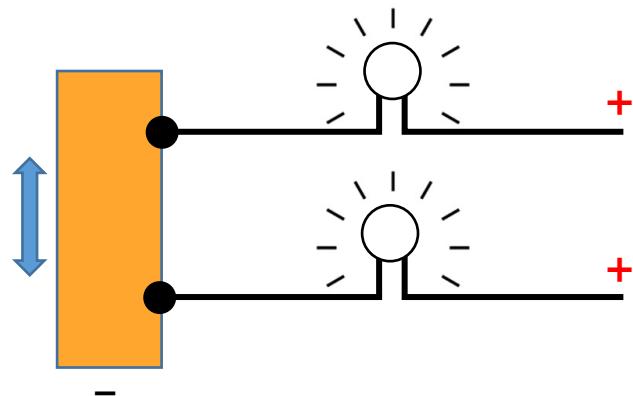
Incremental vs absolute encoder



6) Rotary Encoder

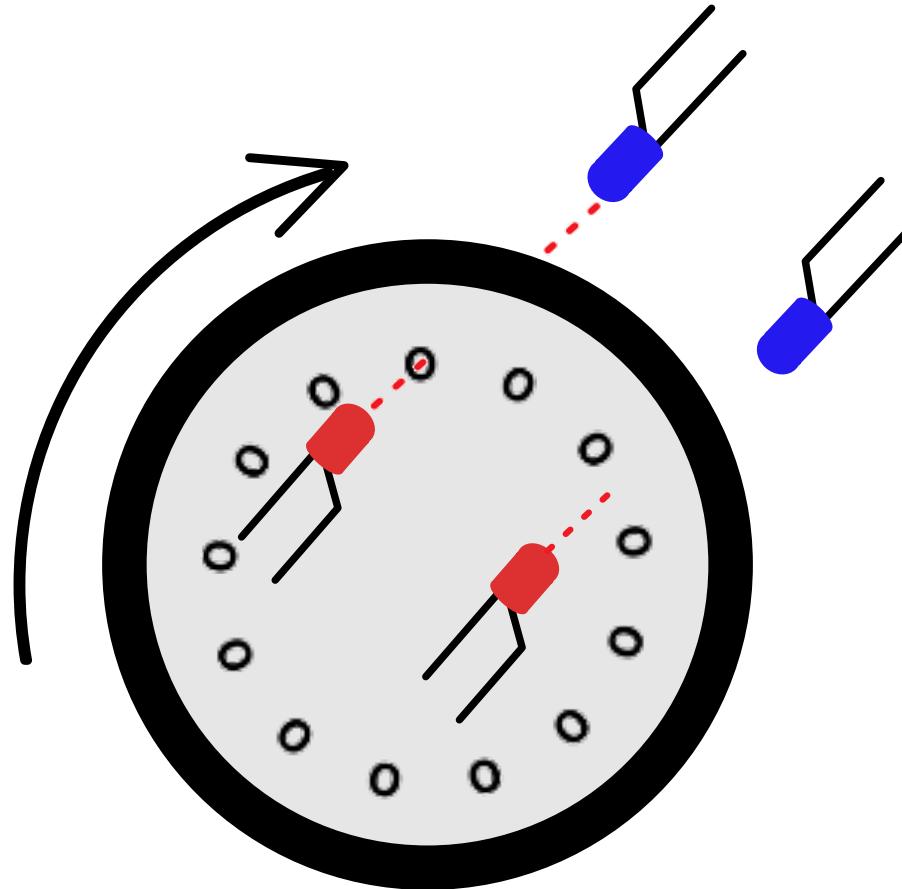


인코더로 방향 확인

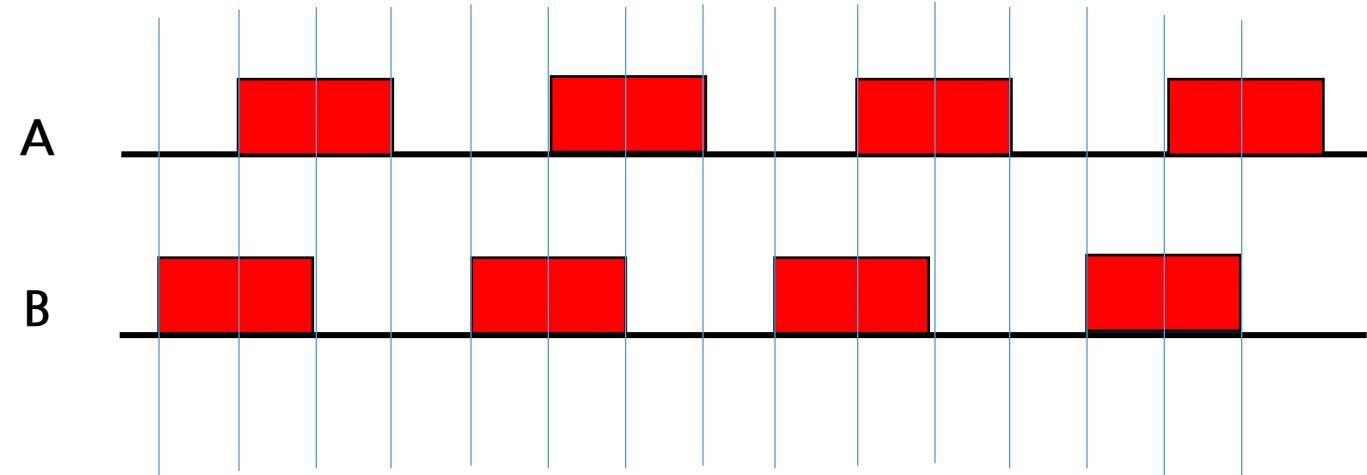
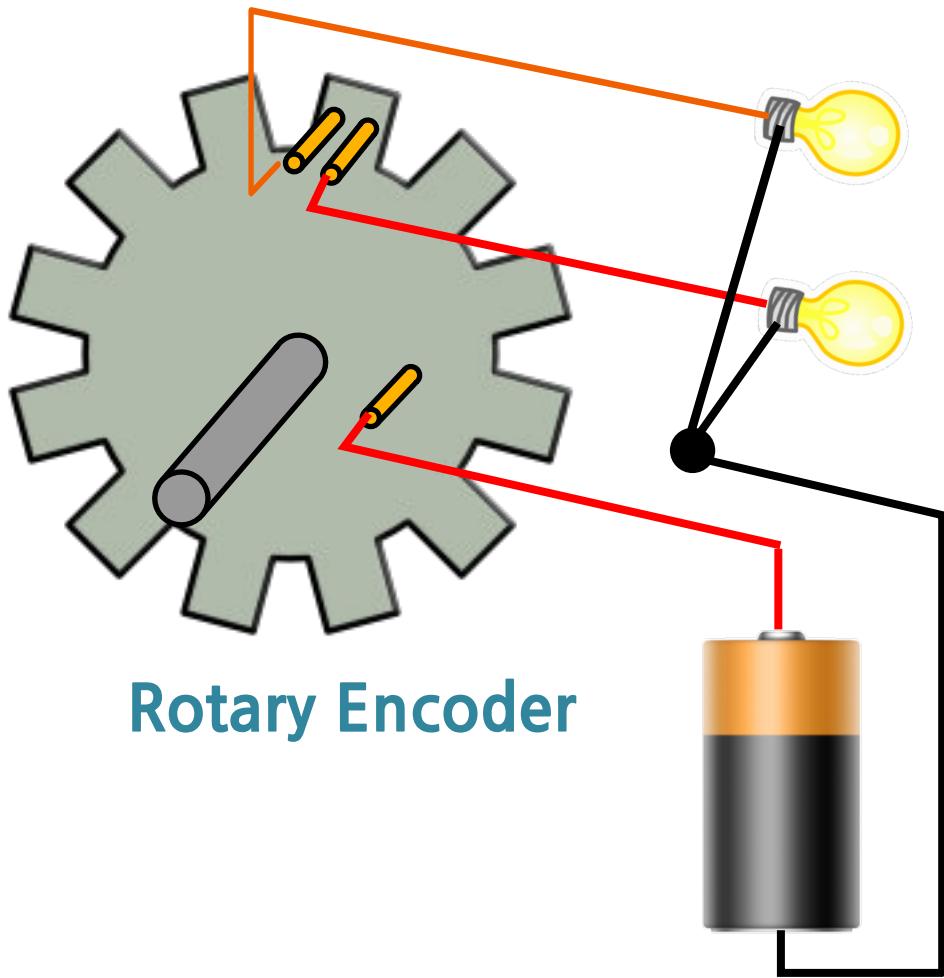


6) Rotary Encoder

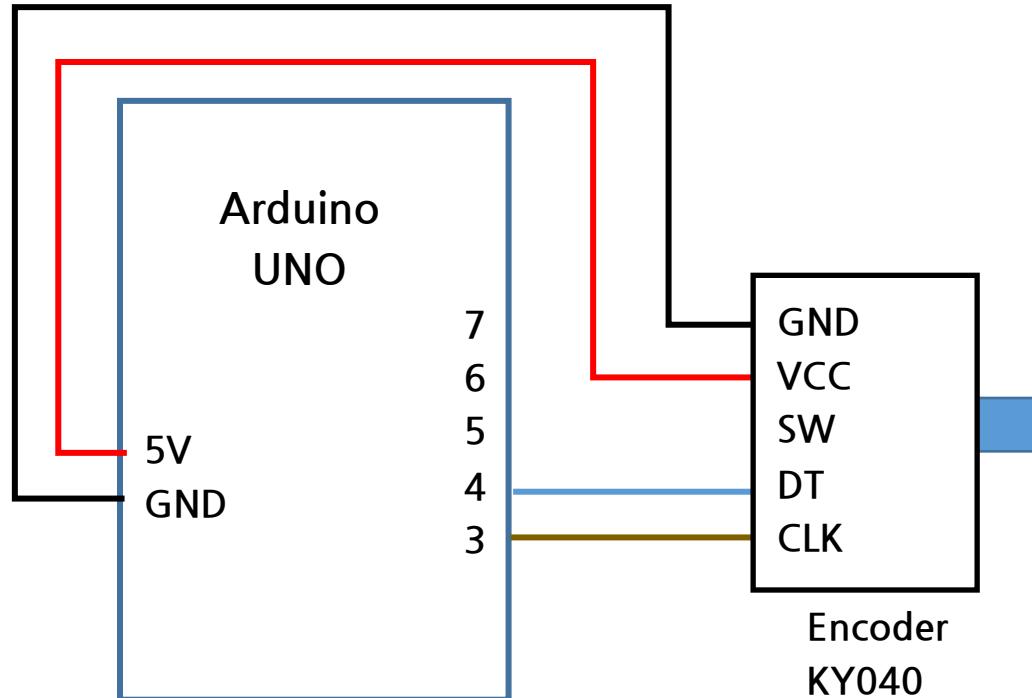
Rotary Encoder
Optical type



6) Rotary Encoder



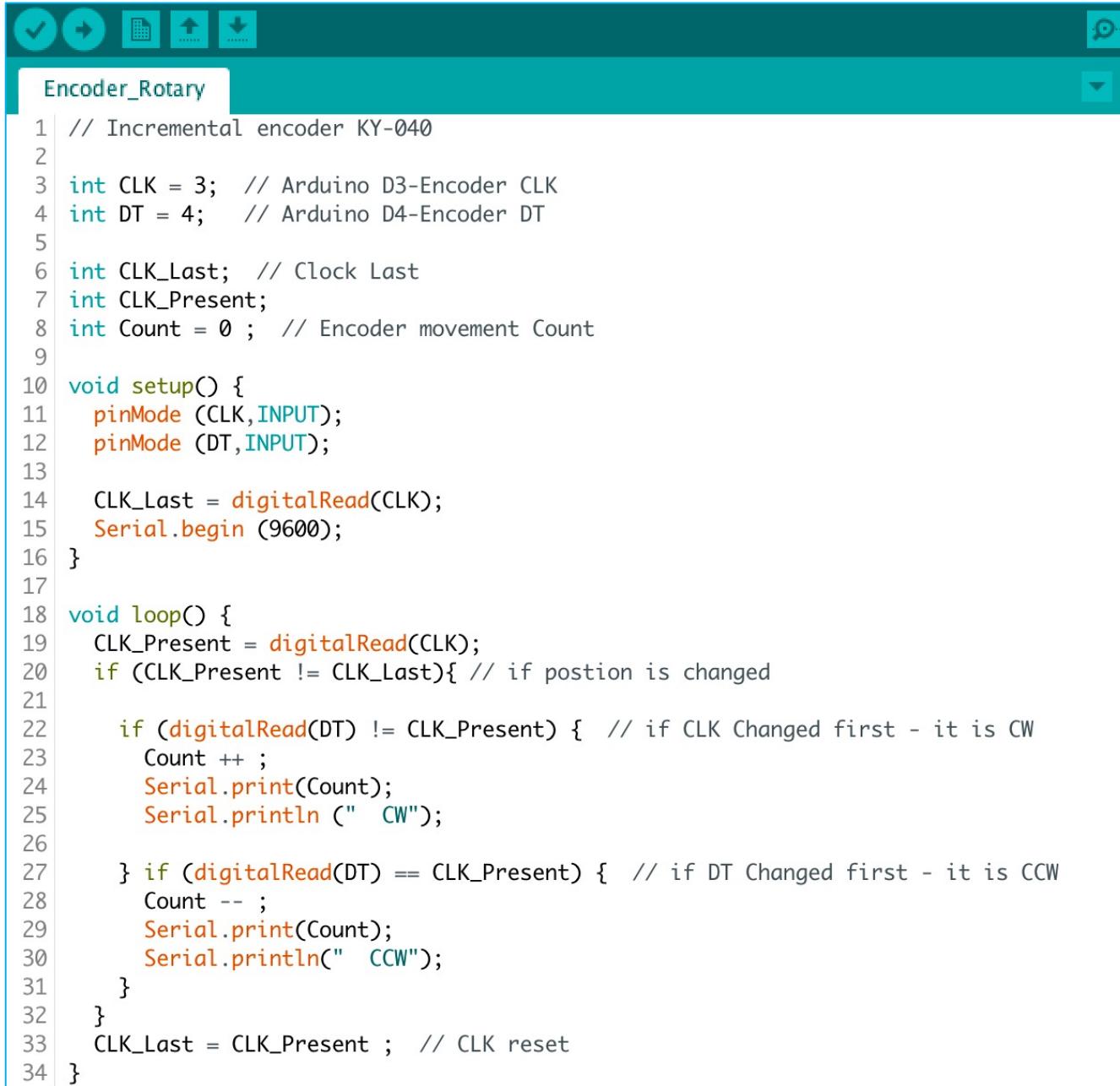
6) Rotary Encoder



Rotary Encoder

6) Rotary Encoder

Rotary Encoder



The screenshot shows the Arduino IDE interface with the sketch titled "Encoder_Rotary". The code is written in C++ and uses the Arduino API. It defines pins CLK and DT, initializes them as inputs, and sets up the serial port at 9600 bps. In the loop, it reads the current state of CLK and DT. If CLK has changed since the last read, it checks if DT has also changed. If DT is high, it increments the count (CW direction); if DT is low, it decrements the count (CCW direction). The count is then printed to the Serial monitor.

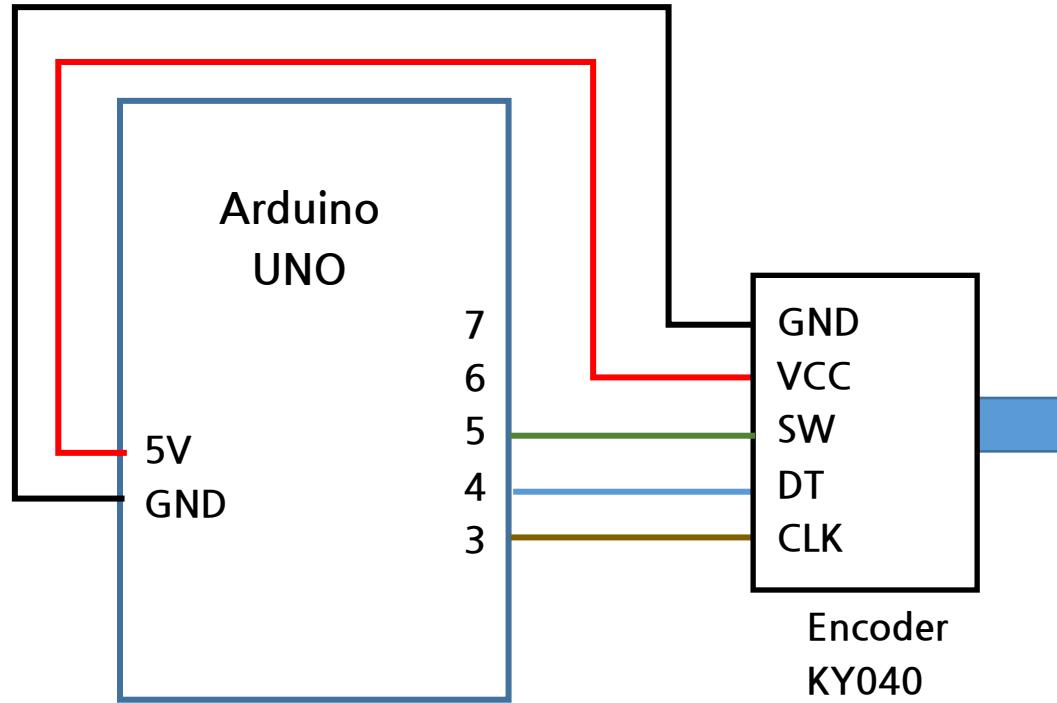
```
// Incremental encoder KY-040
int CLK = 3; // Arduino D3-Encoder CLK
int DT = 4; // Arduino D4-Encoder DT
int CLK_Last; // Clock Last
int CLK_Present;
int Count = 0 ; // Encoder movement Count
void setup() {
  pinMode (CLK,INPUT);
  pinMode (DT,INPUT);
  CLK_Last = digitalRead(CLK);
  Serial.begin (9600);
}
void loop() {
  CLK_Present = digitalRead(CLK);
  if (CLK_Present != CLK_Last){ // if position is changed
    if (digitalRead(DT) != CLK_Present) { // if CLK Changed first - it is CW
      Count ++ ;
      Serial.print(Count);
      Serial.println (" CW");
    }
    if (digitalRead(DT) == CLK_Present) { // if DT Changed first - it is CCW
      Count -- ;
      Serial.print(Count);
      Serial.println(" CCW");
    }
  }
  CLK_Last = CLK_Present ; // CLK reset
}
```

The screenshot shows a terminal window with the title bar reading "/dev/cu.usbmodem14111 (Arduino/Genuino Uno)". The main pane displays a series of 16 lines of text, each consisting of a number followed by "CW". The numbers are 2, 3, 4, 5, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15. At the bottom of the window, there are three controls: a checked checkbox labeled "Autoscroll", a dropdown menu set to "Both NL & CR", and another dropdown menu set to "9600 baud".

```
2 CW
3 CW
4 CW
5 CW
4 CCW
5 CW
6 CW
7 CW
8 CW
9 CW
10 CW
11 CW
12 CW
13 CW
14 CW
15 CW
```

6) Rotary Encoder

With Switch



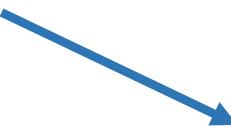
Rotary Encoder

6) Rotary Encoder

With Switch

The screenshot shows a code editor window with the title "Encoder_Rotary_SW". The code is written in C++ for an Arduino microcontroller. It defines pins for CLK (3), DT (4), and SW (5). The setup() function initializes the pins as inputs and sets the switch pin to HIGH. The loop() function reads the CLK and DT pins. If the switch is pressed (LOW), it initializes the count to 0. It then checks if the CLK or DT pin has changed. If CLK has changed first, it increments the count and prints " CW". If DT has changed first, it decrements the count and prints " CCW". Finally, it updates the CLK_Last variable to the current value of CLK_Present.

```
1 // Incremental encoder KY-040, Written by SH Yang
2
3 int CLK = 3; // Arduino D3-Encoder CLK
4 int DT = 4; // Arduino D4-Encoder DT
5 int SW = 5 ; // Switch for reset Count
6
7 int CLK_Last; // Clock Last
8 int CLK_Present;
9 int Count = 0 ; // Encoder movement Count
10
11 void setup() {
12   pinMode (CLK,INPUT);
13   pinMode (DT,INPUT);
14   pinMode (SW,INPUT);
15   digitalWrite (SW,HIGH);
16
17   CLK_Last = digitalRead(CLK);
18   Serial.begin (9600);
19 }
20
21 void loop() {
22   CLK_Present = digitalRead(CLK);
23
24   int T_Count = digitalRead(SW) ;
25   if (T_Count==LOW) { Count=0 ; } // if switch is pressed Count=0
26
27   if (CLK_Present != CLK_Last){ // if position is changed
28
29     if (digitalRead(DT) != CLK_Present) { // if CLK Changed first - it is CW
30       Count ++ ;
31       Serial.print(Count);
32       Serial.println (" CW");
33
34     } if (digitalRead(DT) == CLK_Present) { // if DT Changed first - it is CCW
35       Count -- ;
36       Serial.print(Count);
37       Serial.println(" CCW");
38     }
39   }
40   CLK_Last = CLK_Present ; // CLK reset
41 }
```



```
/dev/cu.usbmodem14111 (Arduino/Genuino Uno)

1 CW
2 CW
3 CW
4 CW
5 CW
4 CCW
5 CW
6 CW
7 CW
1 CW
2 CW
3 CW
4 CW
3 CCW
4 CW

 Autoscroll Both NL & CR 9600 baud
```

6) Rotary Encoder

Rotary Encoder Optical type

