

HW Programming

wk12 :

Arduino Coding IV.

LED & Digital input



Basic HW coding using Arduino and EV3 (RP3)

HIT, INJE University

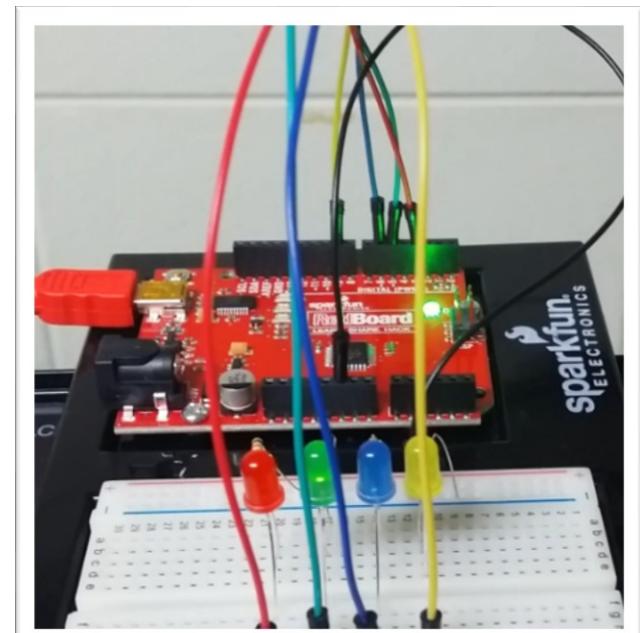
1st semester, 2017

Email : yish@inje.ac.kr

Weekly schedule of HP– 1st semester, 2017



- **wk01 :**
- **wk02 :**
- **wk03 :**
- **wk04 :**
- **wk05 :**
- **wk06 : Introduction to class and enrollment in cyber class, and installing SW**
- **wk07 : Basic HW : Arduino I. – circuit**
- **wk08 : Mid-term exam. → Practice of various circuits**
- **wk09 : Basic HW : Arduino II. – coding by Arduino IDE**
- **wk10 : Arduino Coding II. – Serial monitor**
- **wk11 : Arduino Coding III. – LED**
- **wk12 : Arduino Coding IV. – LED & Digital input**
- **wk13 :**
- **wk14 :**
- **wk15 : Final exam.**





Arduino SW

<http://fritzing.org/home/>



The screenshot of the Fritzing website features a red header bar with the Fritzing logo and navigation links: Projects, Parts, Download, Learning, Services, Contribute, FORUM (highlighted in blue), and FAB. Below the header is a large image showing an Arduino Uno connected to a breadboard with various components like resistors and LEDs. A red banner at the bottom of this image reads "fritzing APP". To the right of the image, text says "Download the free Fritzing App and start building immediately!"

Fritzing is an open-source hardware initiative that makes electronics accessible as a creative material for anyone. We offer a software tool, a community website and services in the spirit of Processing and Arduino, fostering a creative ecosystem that allows users to **document** their prototypes, **share** them with others, **teach** electronics in a classroom, and layout and **manufacture** professional pcbs.

Download and Start
Download our **latest version 0.9.3b** released on June 2, 2016 and start right away.

Produce your own board
With **Fritzing Fab** you can easily and inexpensively turn your circuit into a real, custom-made PCB. Try it out now!

Participate
Fritzing can only act as a creative platform if many



Arduino

Starter Kit



Arduino Starter Kit





Arduino Starter Kit : contents

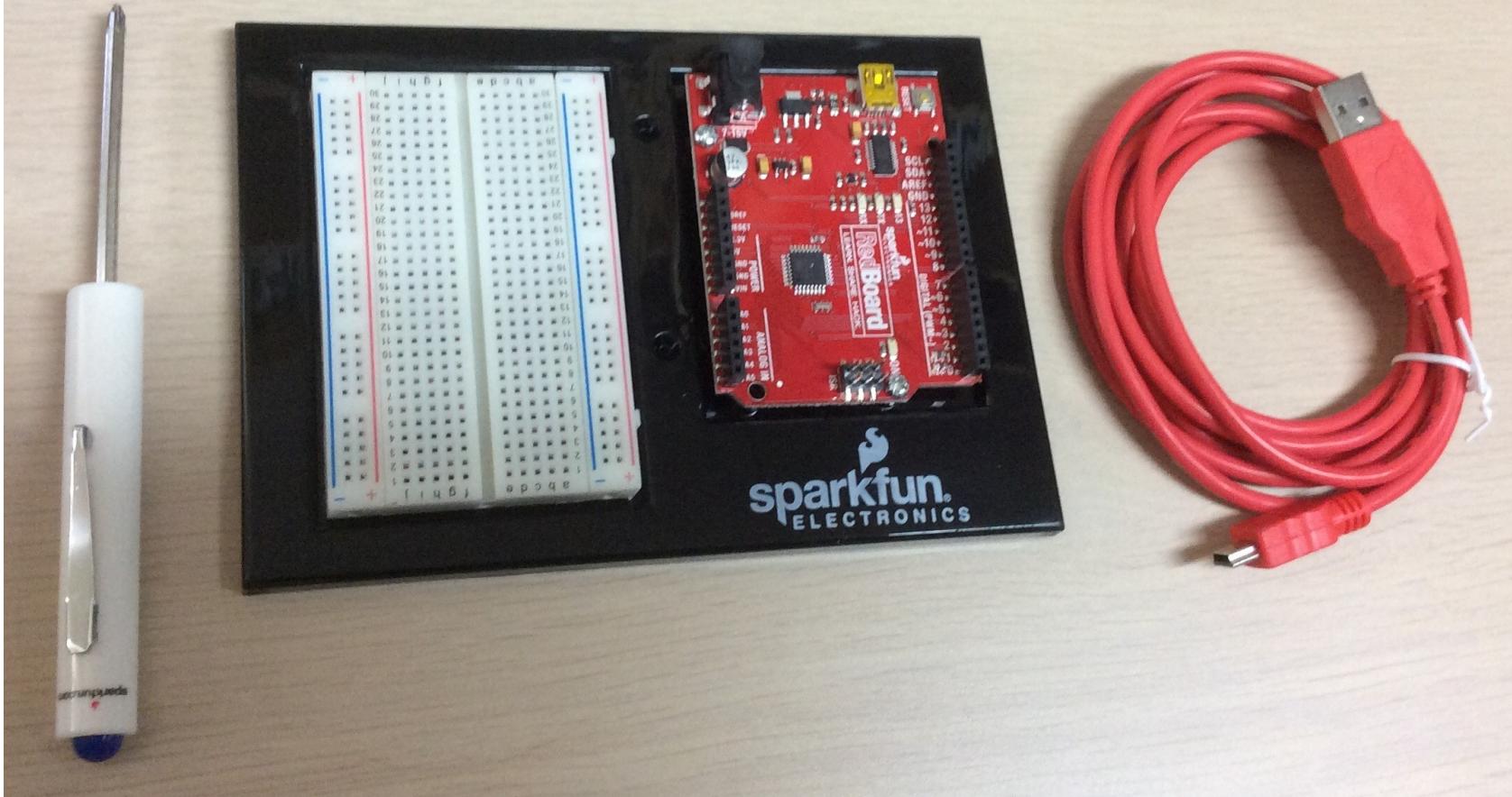
Arduino set

- **SparkFun 레드보드(아두이노 uno 호환)**
- 아두이노, 레드보드 고정판
- 한글판 가이드 북
- **400옴 브레드보드**
- 부품케이스
- **16x2 White on Black LCD**
- **74HC595** 쉬프트레지스터
- **2N2222** 트랜지스터
- **1N4148** 다이오드 (2)
- **기어달린 DC 모터**
- 서보모터
- **SPDT 5V 릴레이**
- **TMP36 온도센서**

- **플렉스 센서(Flex sensor)**
- **소프트 포텐시오미터(Softpot)**
- **SparkFun USB Cable**
- 점퍼케이블 (65)
- **Photocell**
- **RGB LED**
- **적색, 황색, 청색, 초록색 LED (5)**
- **10K 포텐시오미터 (가변저항, Trimpot)?**
- 피에조 부저
- **풀사이클 푸시버튼 (4)**
- **330, 10K 저항 (20)**

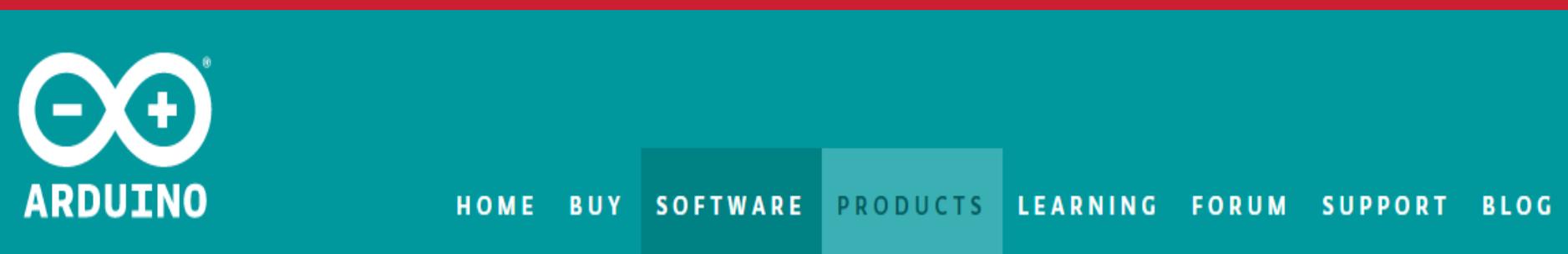


Arduino Starter Kit (Board setting)





1. Arduino SW: IDE



The screenshot shows the top navigation bar of the Arduino website. On the left is the Arduino logo. To its right are several menu items: HOME, BUY, SOFTWARE (which is highlighted in a darker shade), PRODUCTS, LEARNING, FORUM, SUPPORT, and BLOG. The background of the header is a teal color.

<https://www.arduino.cc/>

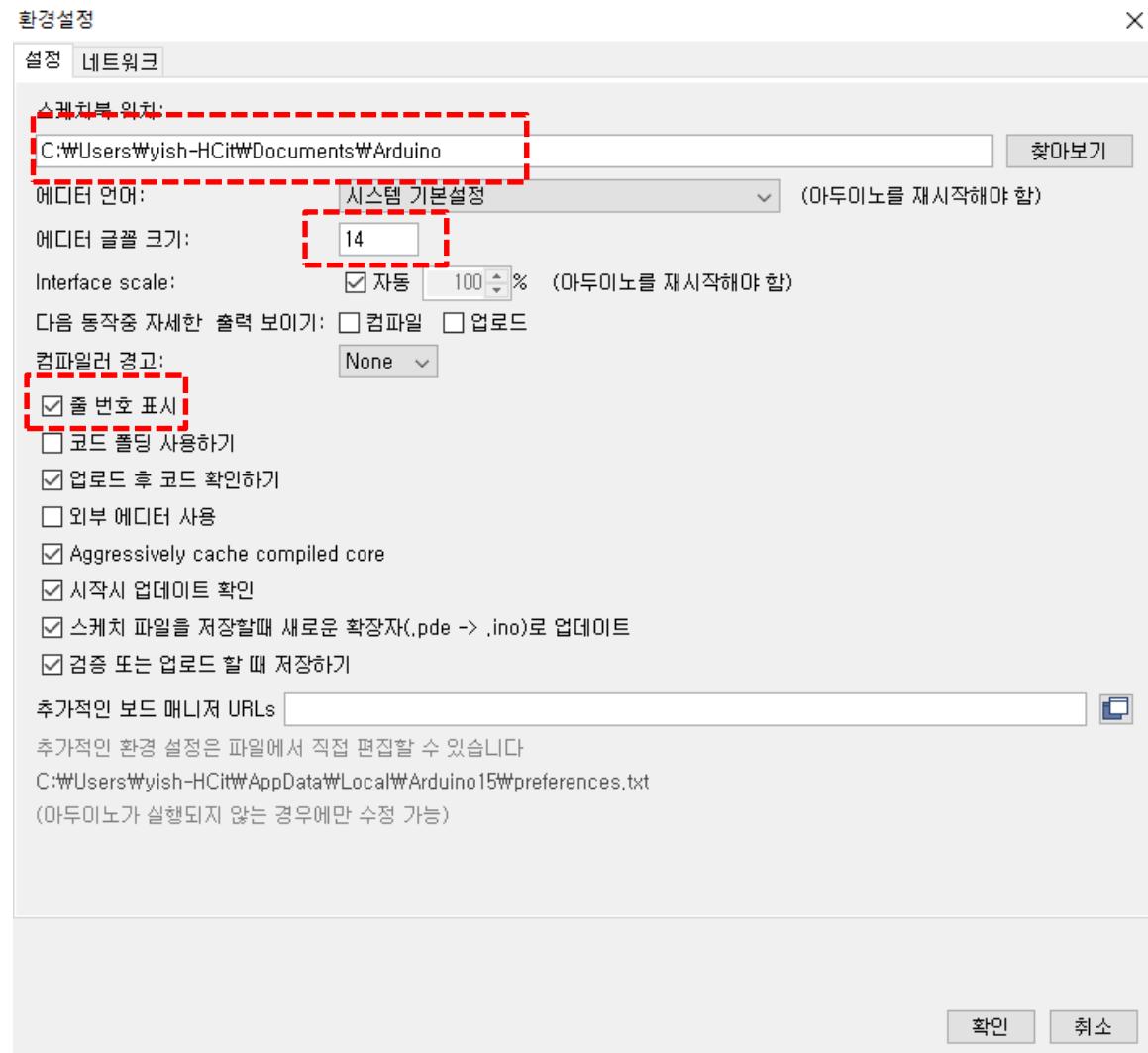


1.5 Arduino IDE

sketch_may01a | 아두이노 1.8.2

파일 편집 스케치 툴 도움말

- 새 파일 Ctrl+N
- 열기... Ctrl+O
- 최근 파일 열기 >
- 스케치북 >
- 예제 >
- 닫기 Ctrl+W
- 저장 Ctrl+S
- 다른 이름으로 저장... Ctrl+Shift+S
- 페이지 설정 Ctrl+Shift+P
- 인쇄 Ctrl+P
- 환경설정 Ctrl+Comma
- 종료 Ctrl+Q



Arduino coding IV.



1. **Arduino SW : IDE**
2. **Blink a LED**
3. **Serial monitor**
4. **LED Control**
5. **Digital input: Switch**
6. **Sensors & Motor**



4. LED

LED (Light Emitting Diode)

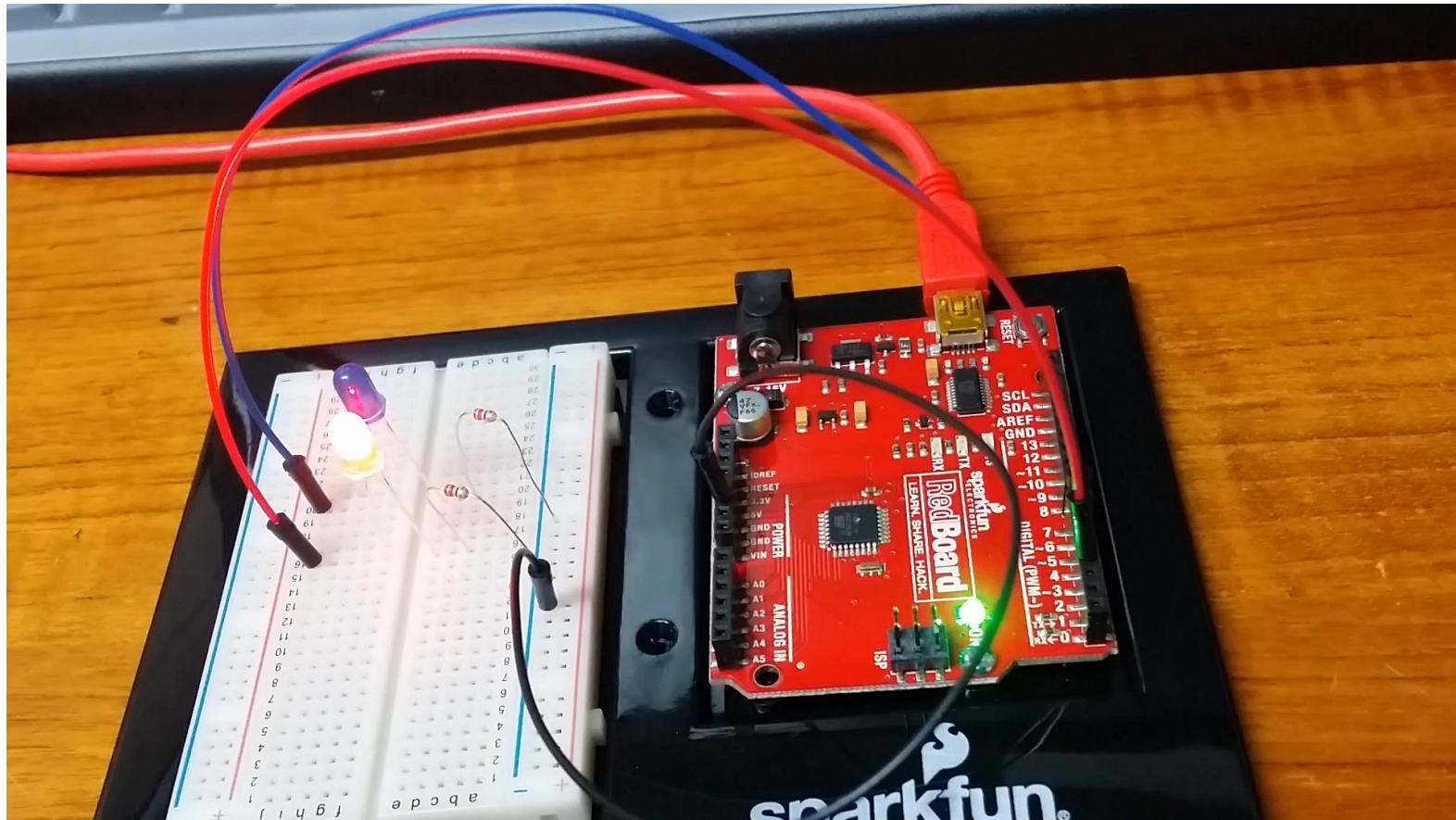
- ✓ 전기 신호를 빛으로 출력하는 반도체 소자
- ✓ 고효율, 반영구적 수명
- ✓ 가정용 실내등, 산업용 특수등, 자동차용 전조등 및 실내등에 사용





4.1 LED control

4.1 LED 교차 점멸





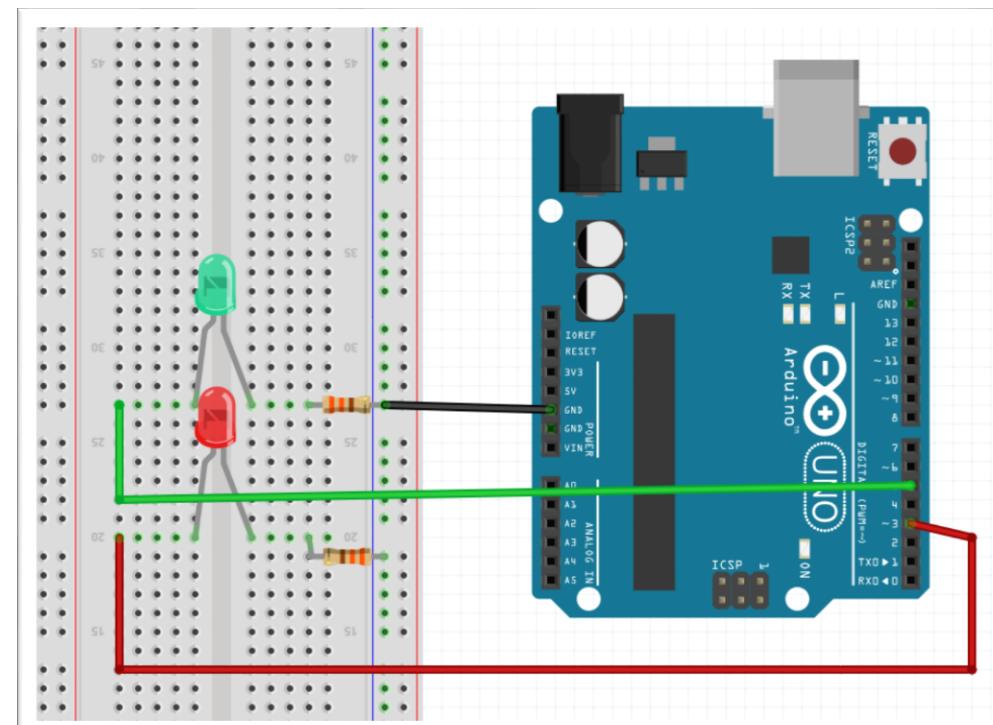
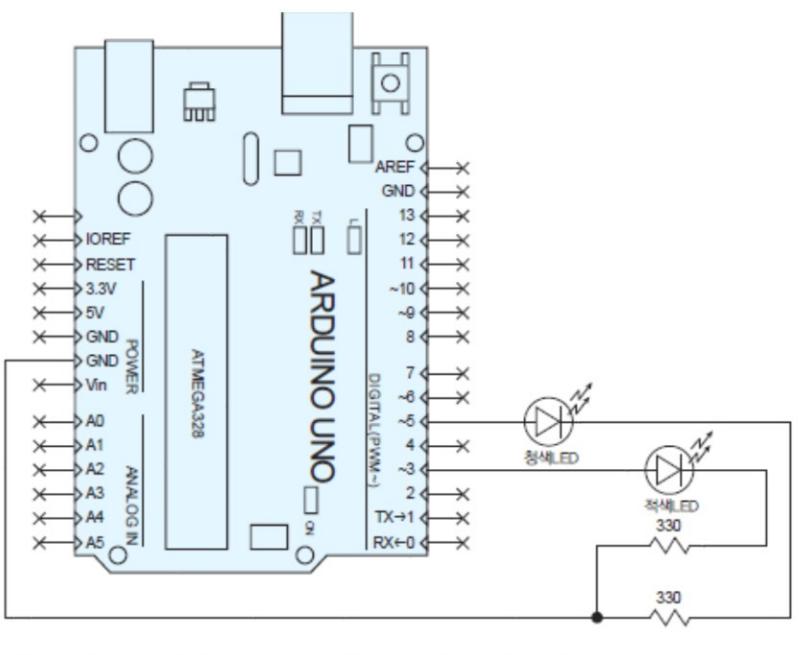
4.1 LED control – 교차 점멸

EX 4.1

LED 교차 점멸 (1/3)

실습목표 두 개의 LED를 0.1초 간격으로 교차하여 점멸시키자.

Hardware





4.1 LED control – 교차 점멸

EX 4.1

LED 교차 점멸 (2/3)

Commands

- **pinMode**(핀번호, 설정)

핀의 입출력 모드를 설정한다. ‘핀번호’에는 설정하고자 하는 핀의 번호와 ‘설정’에는 입력으로 사용하기 위해선 ‘INPUT’, 출력으로 사용하기 위해선 ‘OUTPUT’, 입력이며 풀업 사용시 ‘INPUT_PULLUP’을 설정한다.

- **digitalWrite**(핀번호, 값)

핀에 디지털 출력 (High or Low) 을 한다. ‘핀번호’에는 출력하고자 하는 핀의 번호를, ‘값’에는 ‘HIGH’ 혹은 ‘LOW’ 를 설정하여 High 혹은 Low 출력을 한다.

Sketch 구성

1. LED의 핀 번호를 설정한다.
2. `setup()`에서는 LED 출력으로 사용할 핀을 출력핀으로 설정한다.
3. `loop()`에서는 하나의 LED를 켜 후 일정시간이 지난 후에 소등하고, 다른 LED를 켠다.



4.1 LED control – 교차 점멸

EX 4.1

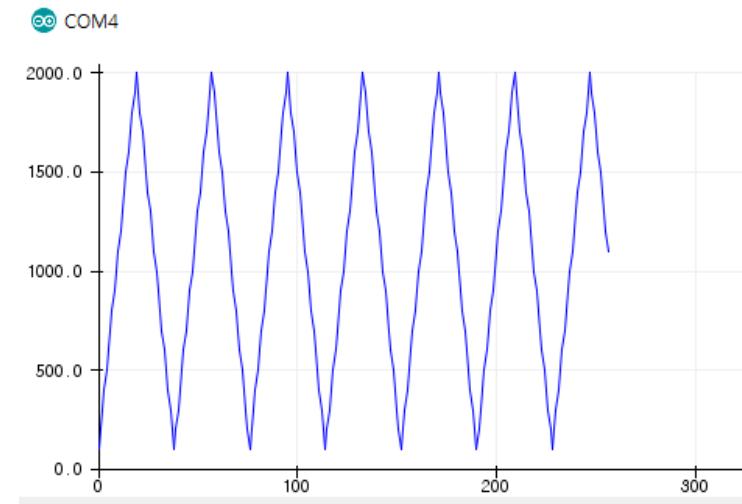
LED 교차 점멸 (3/3)

실습 결과 LED A와 B가 0.1초 단위로 교차하며 점멸한다.

응용 문제 점멸 주기가 0.1초부터 2초로 0.1초 단위로 증가하였다가 다시 반대로 2초부터 0.1초까지 감소하는 동작을 반복하는 스케치를 작성해 보자.
(hint: delay 명령어의 괄호 안의 숫자를 증감시킨다.)

delay = 1600 msec
delay = 1700 msec
delay = 1800 msec
delay = 1900 msec
delay = 2000 msec
delay = 1900 msec
delay = 1800 msec
delay = 1700 msec
delay = 1600 msec

delay = 500 msec
delay = 400 msec
delay = 300 msec
delay = 200 msec
delay = 100 msec
delay = 200 msec
delay = 300 msec
delay = 400 msec
delay = 500 msec



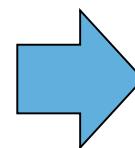


4.1 LED control – 교차 점멸 (code)

ex_4_1

```

1 /*
2 예제 4.1
3 LED 점멸
4 */
5
6 const int ledA = 3;
7 const int ledB = 5;
8
9 void setup()
10 {
11   pinMode(ledA, OUTPUT);
12   pinMode(ledB, OUTPUT);
13 }
14
15 void loop()
16 {
17   digitalWrite(ledA,HIGH);
18   digitalWrite(ledB,LOW);
19   delay(100);
20   digitalWrite(ledA,LOW);
21   digitalWrite(ledB,HIGH);
22   delay(100);
23 }
```



완성된 스케치 code를
HPnn_2led.ino
 로 저장해서 제출.

```

6 const int ledA = 3;
7 const int ledB = 5;
8
9 int number = 1;
10 boolean flag = true;

12 void setup()
13 {
14   Serial.begin(9600);
15   pinMode(ledA, OUTPUT);
16   pinMode(ledB, OUTPUT);
17 }
```

```

19 void loop()
20 {
21   digitalWrite(ledA, HIGH);
22   digitalWrite(ledB, LOW);
23   delay(100 * number);
24   digitalWrite(ledA, LOW);
25   digitalWrite(ledB, HIGH);
26   Serial.print("delay = ");
27   Serial.print(100 * number);
28   Serial.println(" msec");
29   delay(100 * number);

30
31   if (flag) {
32     number++;
33   } else {
34     number--;
35   }

36
37
38
39
40
41
42
43 }
```

Fill in your code!



4.2 LED control – 밝기 조절

밝기 조절 : 디밍 (Dimming)

- ✓ LED에 입력되는 전력은 **PWM (Pulse Width Modulation)**을 이용하여 조절.
- ✓ PWM : 고속의 스위칭으로 High와 Low 신호의 비율을 조절하여
LED의 밝기, 모터의 회전 등을 조절하는 방법
- ✓ Arduino에서는 **analogWrite()** 명령어로 구현
- ✓ Arduino UNO의 경우 **3, 5, 6, 9, 10, 11** 번 핀이 PWM을 지원한다.



4.2 LED control – 밝기 조절

PWM (Pulse Width Modulation)

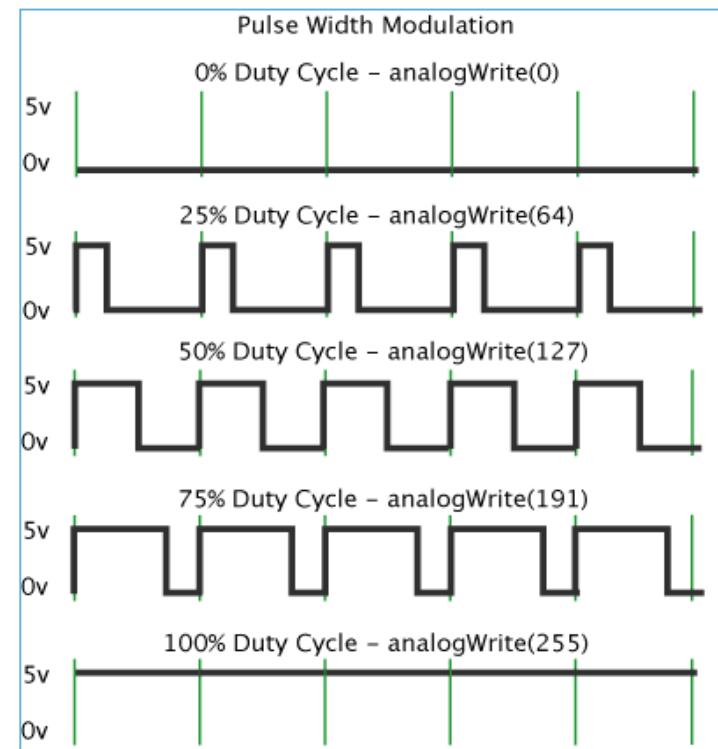
Using [analogWrite\(pin, pwm_value\)](#) function in fading an LED off and on.

AnalogWrite uses [pulse width modulation \(PWM\)](#), turning a digital pin on and off very quickly with different ratio between on and off, to create a fading effect.

A call to [analogWrite\(\)](#) is on a scale of **0 - 255**, such that analogWrite(255) requests a 100% duty cycle (always on), and analogWrite(127) is a 50% duty cycle (on half the time)

PWM frequency = 500 Hz

<https://www.arduino.cc/en/Tutorial/PWM>





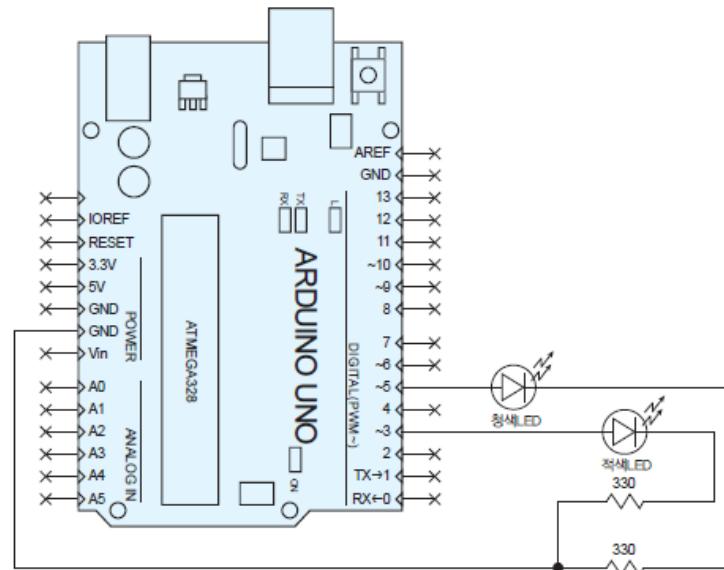
4.2 LED control – 밝기 조절

EX 4.2

LED 밝기 조절 (1/2)

- 실습목표
1. 두 개의 LED의 밝기를 조절하자.
 2. 각각의 LED가 교차하며 밝아졌다 어두워 졌다를 반복하도록 하자.

- Hardware
1. 청색과 적색 LED의 Anode핀을 Arduino의 3번 5번 핀에 연결한다.
 2. Cathode핀에 330Ω 저항을 연결하여 저항의 반대쪽은 Arduino의 GND에 연결한다.
 3. LED가 연결된 핀에 HIGH신호가 출력될 때 LED가 점등된다.





4.2 LED control – 밝기 조절

EX 4.2

LED 밝기 조절 (2/2)

Commands • `analogWrite(핀번호, 값)`

정해진 핀에 아날로그 출력을 한다. ‘값’에는 0~255의 값을 넣는다.

Sketch 구성 1. LED의 핀 번호를 설정한다.

2. `setup()`에서는 LED 출력으로 사용할 핀을 출력핀으로 설정한다.

3. 밝기를 저장할 변수를 설정한다.

4. 하나의 LED가 밝아질 때 다른 LED는 어두워져야 하므로 이를 조절할 변수를 설정한다.

5. `loop()`에서는 밝기와 밝기 변수 증감을 위한 변수를 조절하여 두 개의 LED를 교차 점멸시키는 동작을 반복한다.

실습 결과 LED A와 B가 밝기가 변화하며 점멸한다.

응용 문제 1. 네개의 다른 색깔의 LED를 Arduino에 연결한다.

2. 네개의 LED가 순서대로 디밍하는 스케치를 작성해보자.

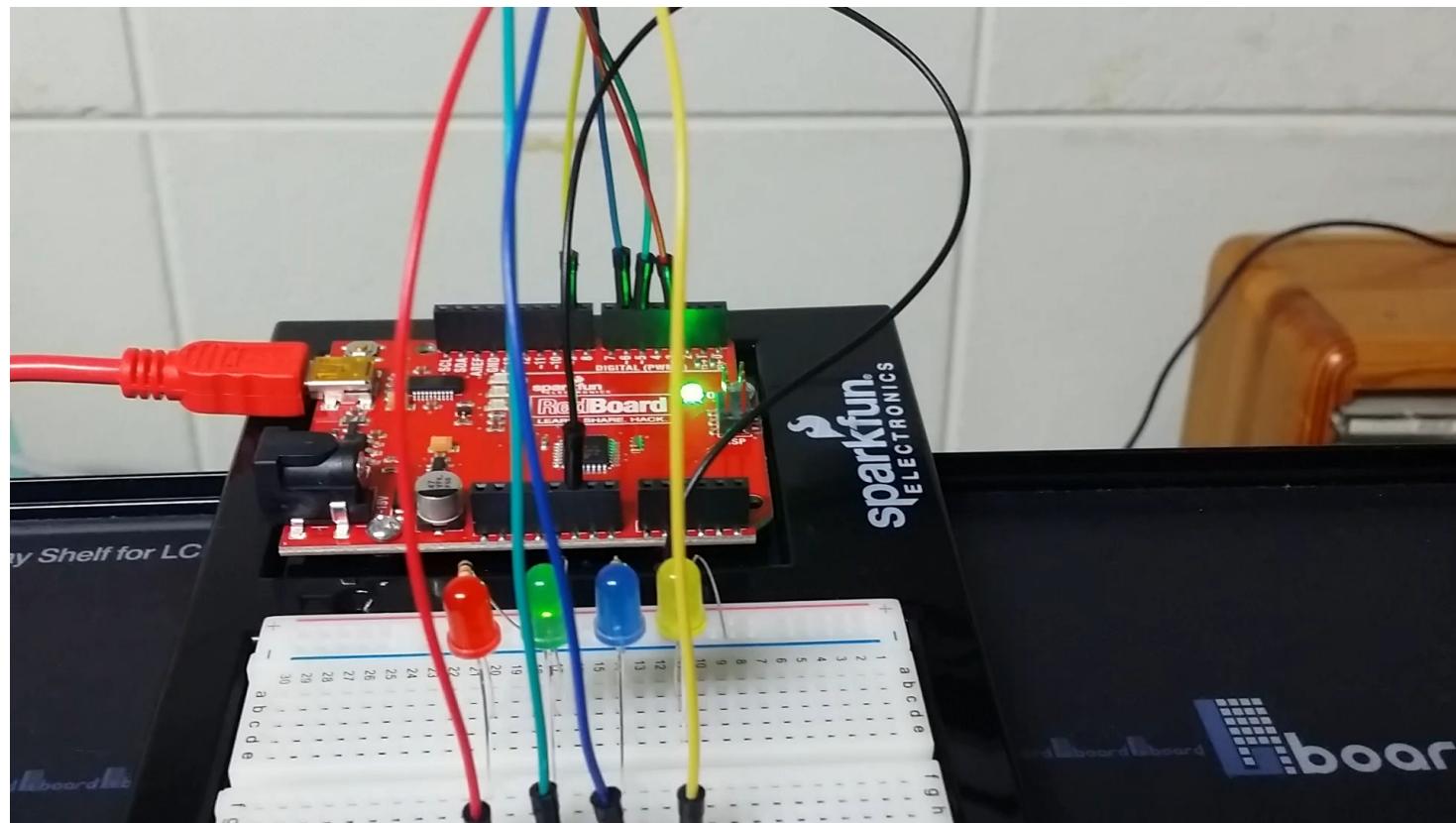


4.2 LED control – 밝기 조절 (code)

```
ex_4_2_start
1 /*
2 예제 4.2
3 LED 밝기 조절
4 */
5
6 const int ledA = 3; //LED A를 3번핀에 연결
7 const int ledB = 5; //LED B를 5번핀에 연결
8 int brightness = 0; //밝기를 조절하기 위한 변수
9 int increment = 1; //밝기 변수 증감을 위한 변수
10
11 void setup()
12 {
13 // analogWrite 핀에는 별도의 설정이 불필요하다.
14 }
15
16 void loop()
17 {
18 analogWrite(ledA,brightness); // LED A 밝기 조절
19 analogWrite(ledB,255-brightness); // LED B 밝기 조절
20
21 brightness = brightness + increment; // 밝기 조절
22 if((brightness >= 255) || (brightness <= 0)) increment = -increment; // 밝기 변수 증감 방향 변경
23 delay(10); // 0.01 초간 지연
24 }
```

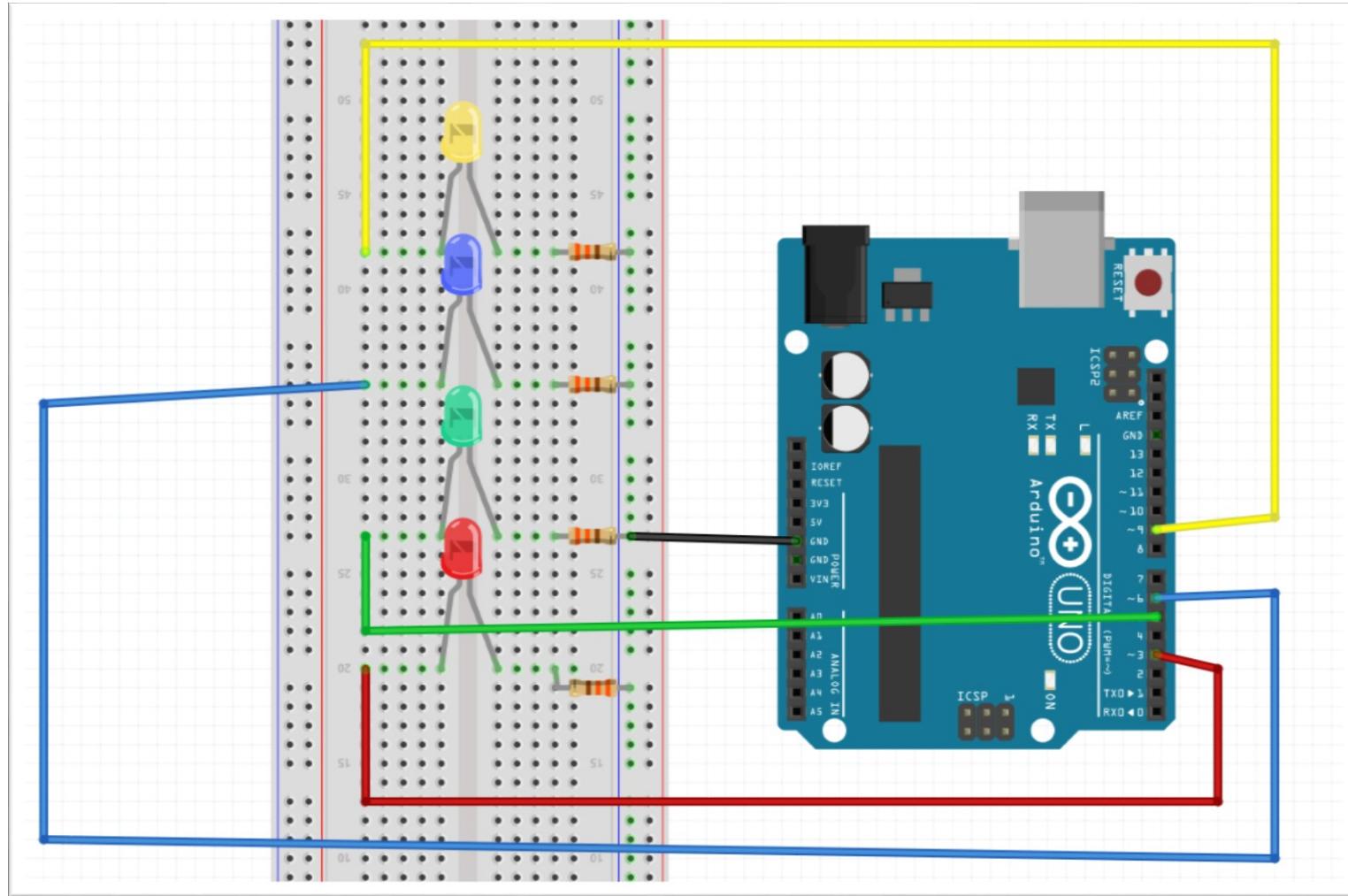
4.2 4 LED control – 밝기 조절

- DIY 6. 1. 네개의 다른 색깔의 LED를 Arduino에 연결한다.
2. 네개의 LED가 순서대로 디밍하는 스케치를 작성해보자.



4.2 4 LED control – 밝기 조절

DIY 6. 1. 네개의 다른 색깔의 LED를 Arduino에 연결한다. (pwm pin: 3,5,6,9)





4.2 4 LED control – 밝기 조절 (code-1)

```

1 /*
2 Dimming 4 leds
3 */
4
5 int ledR = 3; // LED connected to digital pin 3
6 int ledG = 5;
7 int ledB = 6;
8 int ledY = 9;
9
10 int dimTime = 20;
11
12 void setup() {
13 // nothing happens in setup
14 }

```

```

16 void loop() {
17 // fade in from min to max in increments of 5 points:
18 for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5) {
19 // sets the value (range from 0 to 255):
20 analogWrite(ledR, fadeValue);
21 // wait for 30 milliseconds to see the dimming effect
22 delay(dimTime);
23 }
24
25 // fade out from max to min in increments of 5 points:
26 for(int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5) {
27 // sets the value (range from 0 to 255):
28 analogWrite(ledR, fadeValue);
29 // wait for 30 milliseconds to see the dimming effect
30 delay(dimTime);
31 }
32
33 for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5) {
34 // sets the value (range from 0 to 255):
35 analogWrite(ledG, fadeValue);
36 // wait for 30 milliseconds to see the dimming effect
37 delay(dimTime);
38 }

```

각 led에 동일한 dimming code 적용



4.2 4 LED control – 밝기 조절 (code-2)

```

1 /*
2 Dimming 4 leds
3 */
4
5 int ledR = 3; // LED connected to digital pin 3
6 int ledG = 5;
7 int ledB = 6;
8 int ledY = 9;
9
10 int dimTime = 20;
11
12 void setup() {
13 // nothing happens in setup
14 }

```

완성된 스케치 code를
[HPnn_4led.ino](#)
 로 저장해서 제출.

```

16 void loop() {
17 // fade ledR
18 dimLed(ledR);
19 // fade ledG
20 dimLed(ledG);
21 // fade ledB
22 dimLed(ledB);
23 // fade ledY
24 dimLed(ledY);
25 }
26 void dimLed(int led) {
27 // fade in from min to max in increments of 5 points:
28 for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5) {
29 // sets the value (range from 0 to 255):
30 analogWrite(led, fadeValue);
31 // wait for 20 milliseconds to see the dimming effect
32 delay(dimTime);
33 }
34 // fade out from max to min in increments of 5 points:
35 for(int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5) {
36 // sets the value (range from 0 to 255):
37 analogWrite(led, fadeValue);
38 // wait for 20 milliseconds to see the dimming effect
39 delay(dimTime);
40 }
41 }

```

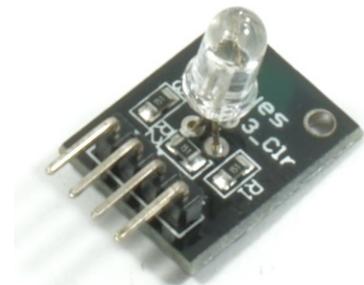
각 led에 동일한 dimming code 적용

dimLed(int led) 반복 사용

4.3 RGB LED control – 색상 조절

RGB LED

- ✓ 빛의 삼원색인 빨강(Red), 초록(Green), 파랑(Blue)빛을 조절하여 다양한 색을 표현하는 LED.
- ✓ 각각의 색이 0~255단계로 조절됨.
- ✓ 간판, 조명기구 등에 사용
- ✓ 모든 색이 출력될 때 백색 빛을 출력





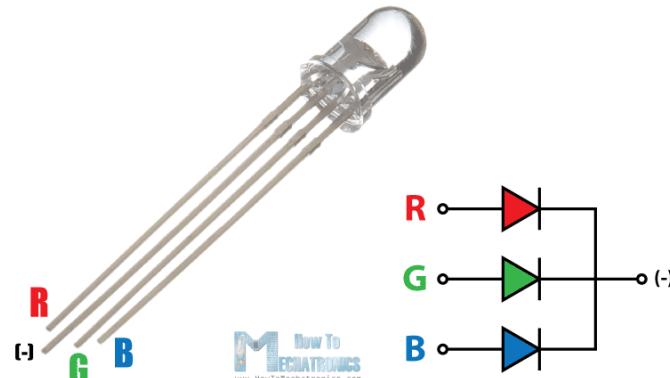
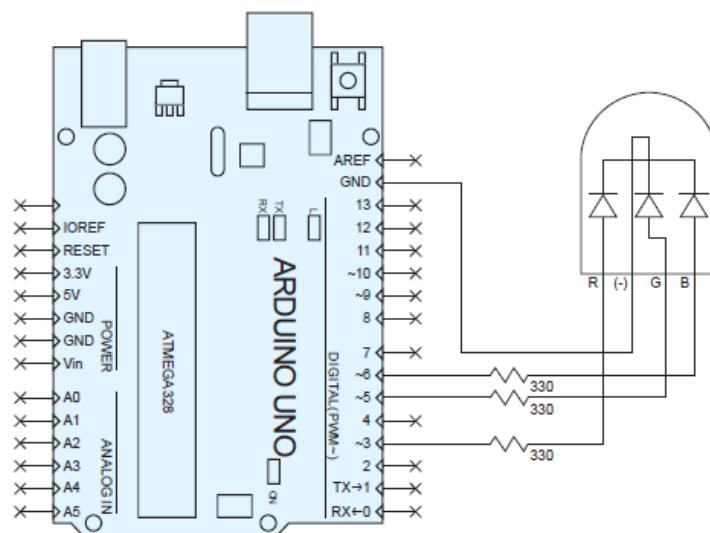
4.3 RGB LED control – 색상 조절

EX 4.3

RGB LED로 색상 표현하기 (1/2)

실습목표 RGB LED를 이용하여 다양한 색을 표현해 보자.

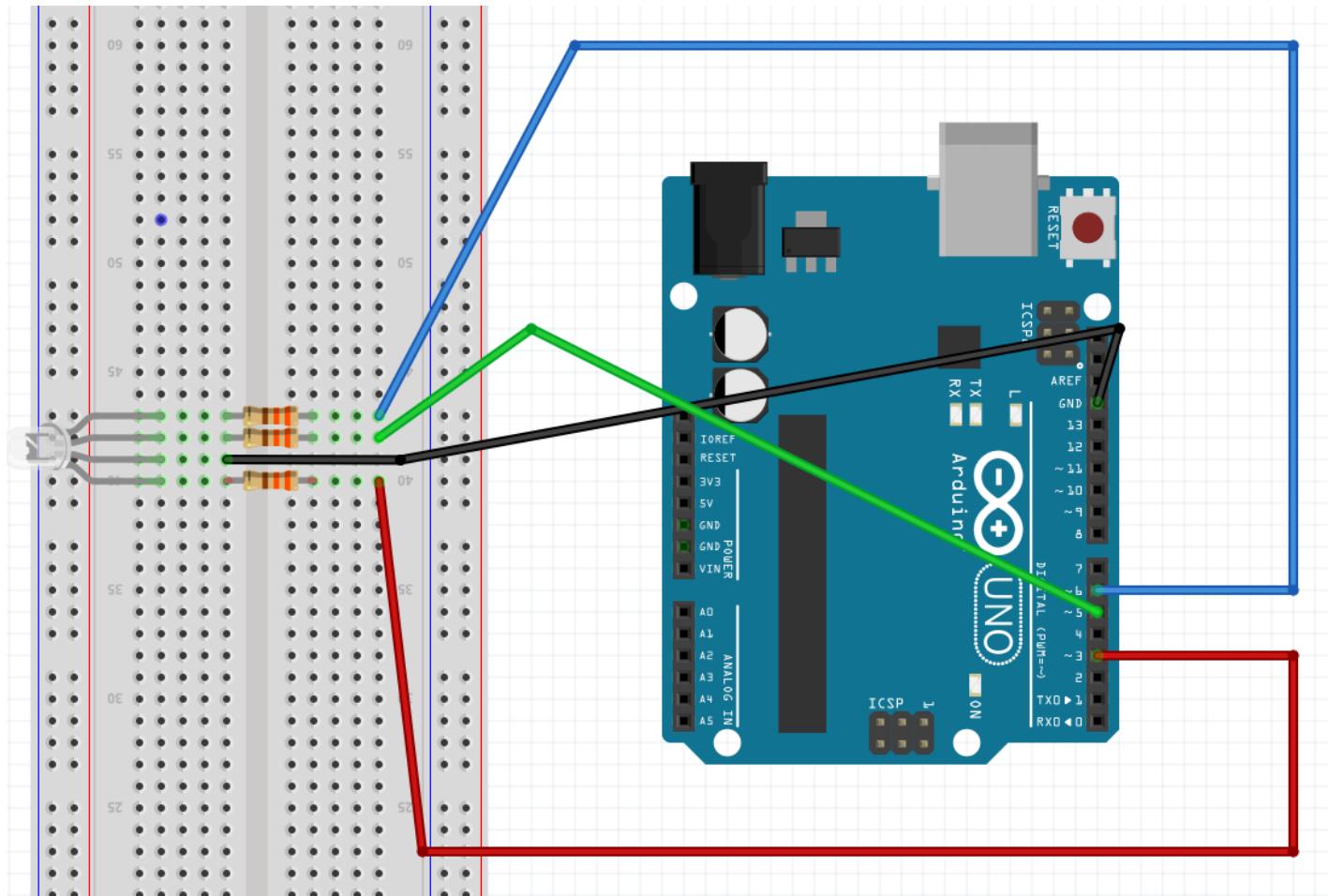
- Hardware**
1. RGB LED는 Red, Green, Blue의 세 개의 Anode 핀과 공통으로 연결된 캐소드핀으로 구성되어 있다.
 2. RGB LED 단독으로 연결하려면 각 Anode 핀에 **330Ω**의 저항을 연결해야 한다.
 3. 저항이 내장된 RGB LED 모듈을 사용한다면 별도의 저항이 필요 없다.
 4. Red, Green, Blue의 세 개의 Anode 핀을 Arduino의 3, 5, 6 번핀에 연결한다.



<http://howtomechatronics.com/tutorials/arduino/how-to-use-a-rgb-led-with-arduino/>



4.3 RGB LED control - 색상 조절





4.3 RGB LED control – 색상 조절

EX 4.3

RGB LED로 색상 표현하기 (2/2)

Commands

- `analogWrite(핀번호, 값)`

정해진 핀에 아날로그 출력을 한다. '값'에는 0~255의 값을 넣는다.

- `delay(지연시간)`

지연시간에는 잠시 동작을 지연시키기 위한 값을 넣는다. 1/1000초 단위로 넣는다.

즉 1초를 지연시키기 위해선 1000의 값을 입력시킨다.

- `for(변수=시작 값 ; 조건 ; 변수의 증분){ }`

변수의 시작 값부터 조건이 만족하는 경우 '{ }' 내의 명령을 수행한다. '변수의 증분'에서는 1회 명령이 수행될 때마다 변수를 증가 혹은 감소시킨다.

Sketch 구성

1. LED의 핀번호를 설정한다.
2. `setup()`에서는 LED 출력으로 사용할 핀을 출력 핀으로 설정한다.
3. `ledOutput(int Red, int Green, int Blue)`라는 함수를 만든다. 적색, 녹색, 청색 LED의 빛의 세기를 조합하여 원하는 색을 출력하는 함수이다.
4. 적색, 녹색, 청색 LED의 세기를 조절하면서 LED에 빛을 출력시킨다.

실습 결과 DIY

LED의 색상이 변화를 조사한다.

[http://www.rapidtables.com/
web/color/RGB_Color.htm](http://www.rapidtables.com/web/color/RGB_Color.htm)

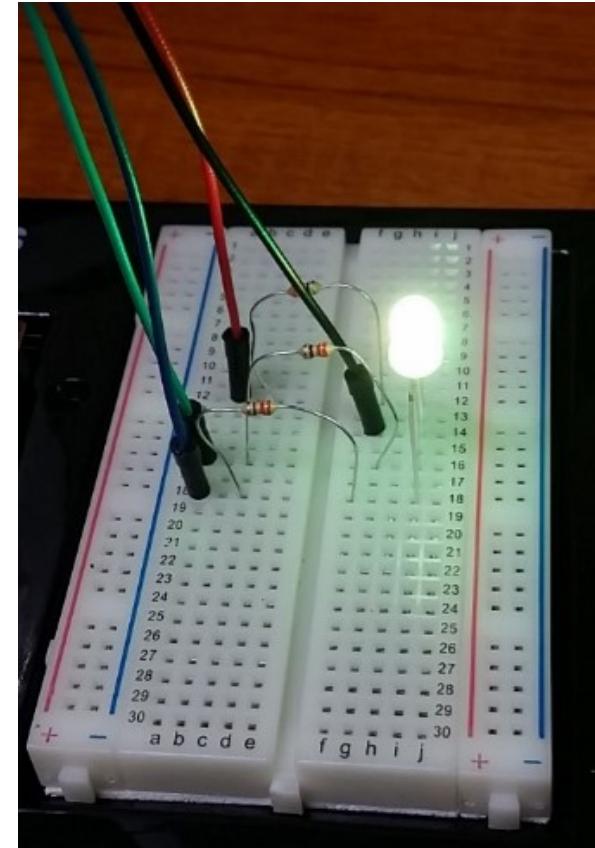
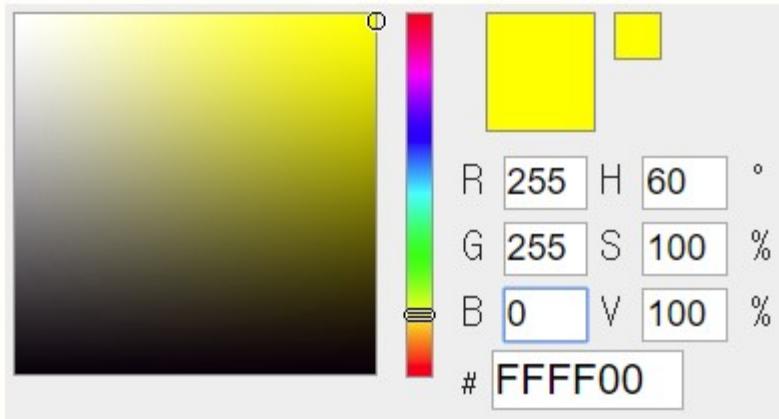




4.3 RGB LED control – 색상 조절

DIY 7. 1. RGB LED의 색이 노란색일 때 사진을 촬영하시오.

RGB color picker



HPnn_RGB.png 로 저장



5. Digital input



5. 디지털 신호 입력

디지털 신호

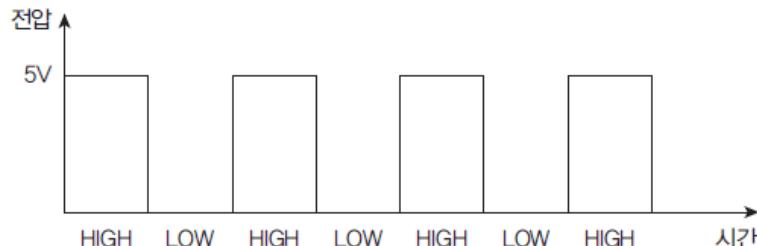


그림 5.1 HIGH LOW 신호가 반복되는 디지털신호

풀업 / 풀다운

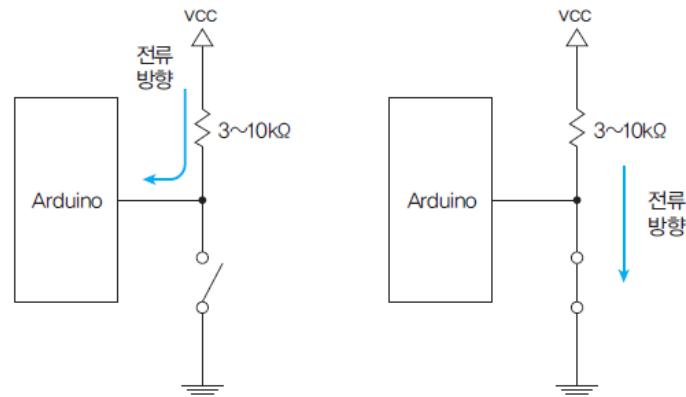


그림 5.2 풀업저항과 스위치 동작. Arduino는 스위치가 오프상태인 (a)에서는 HIGH 신호로, 스위치가 온상태인 (b)에서는 LOW 신호로 인식한다.

- ✓ 0과 1 혹은 High Low 두 가지 값으로 표현되는 신호
- ✓ 잡음에 강하고 데이터의 저장 및 처리가 용이

- ✓ 디지털 신호 입력핀에 아무것도 연결되지 않았을 때의 상태를 High 혹은 Low 신호로 만들어 안정시킴



5.1 디지털 신호 입력 - Switch

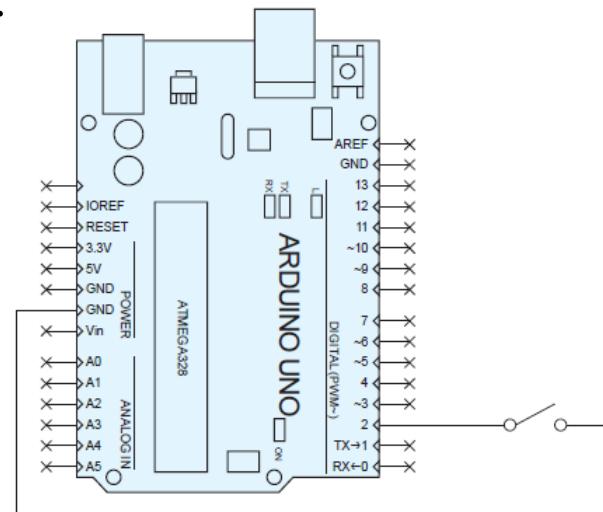
EX 5.1

스위치 입력 (1/3)

실습목표 스위치를 이용하여 디지털 신호를 입력 받아 스위치가 눌러졌을 때 LED를 점등시키자.

Hardware

1. 디지털 입출력핀인 2번핀으로 스위치 입력을 받는다.
2. 스위치의 한쪽을 2번핀에 연결하고 다른 한쪽을 GND에 연결한다.
3. 스위치 입력이 없을 때 2번핀의 상태는 HIGH신호인지 LOW신호인지 알 수가 없다.
그러므로 반드시 핀의 출력 설정 때 'INPUT_PULLUP' 명령어를 사용하여 풀업시켜
줘야 한다.
4. 풀업을 해줬다면 스위치 입력이 없을 때 2번핀의 상태는 HIGH신호이고 스위치 입
력이 있을 때 2번핀의 상태는 LOW신호이다.



5.1 디지털 신호 입력 - Switch

EX 5.1

스위치 입력 (2/3)

Commands

- `pinMode(핀번호, 설정)`

핀의 입출력 모드를 설정한다. ‘핀번호’에는 설정하고자 하는 핀의 번호와 ‘설정’에는 입력으로 사용하기 위해선 ‘INPUT’, 출력으로 사용하기 위해선 ‘OUTPUT’, **입력이며 풀업 사용시 ‘INPUT_PULLUP’을 설정한다.**

- `digitalRead(핀번호)`

‘핀번호’에 해당하는 핀의 디지털 입력값을 읽는다. LOW 이면 0, HIGH이면 1의 값을 갖는다.

- `digitalWrite(핀번호, 값)`

핀에 디지털 출력 (High or Low)을 한다. ‘핀번호’에는 출력하고자 하는 핀의 번호를, ‘값’에는 ‘HIGH’ 혹은 ‘LOW’를 설정하여 High 혹은 Low 출력을 한다.

Sketch 구성

1. 2번 핀을 입력으로 설정하고 LED가 내장된 13번 핀을 출력으로 설정한다.
2. 2번 핀은 INPUT_PULLUP으로서 평상시에는 HIGH 상태로 유지되다가 스위치 입력이 있을 때 LOW로 변화한다.
3. 2번핀에 LOW 신호가 입력될 때 LED를 점등(HIGH)시킨다.



5.1 디지털 신호 입력 - Switch

EX 5.1

스위치 입력 (3/3)

실습 결과 스위치를 누르면 LED가 점등되고, 손을 떼면 LED가 소등된다.

응용 문제 스위치 입력이 감지될 때마다 LED가 점등되어있다면 소등시키고, 소등되어 있다면 점등시키도록 `loop()` 부분을 아래와 같이 수정해 보자.

```
void loop(){
    int swInput = digitalRead(inputPin); // 스위치 입력을 받는다
    int ledOutput = digitalRead(ledPin); // LED의 출력 상태를 확인한다

    if (swInput == LOW){
        if (ledOutput) digitalWrite(ledPin, LOW); // LED가 점등되어 있으면 소등
        else digitalWrite(ledPin, HIGH); // LED가 소등되어 있으면 점등
    }
}
```

이 때 어떠한 현상이 발생하는가? → 측정 결과를 검토하시오.



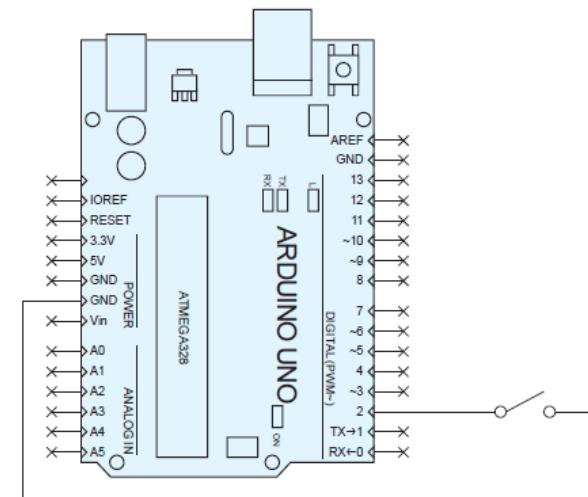
5.2 디지털 신호 입력 - 안정된 Switch

EX 5.2

안정적인 스위치 입력 (1/3)

- 실습목표**
1. 스위치를 이용하여 디지털 신호 입력을 받아 LED를 점멸시킨다.
 2. 스위치가 입력된 횟수를 시리얼 통신으로 전송해 보자.

- Hardware**
1. 디지털 입출력핀인 2번핀으로 스위치 입력을 받는다.
 2. 스위치의 한쪽을 2번핀에 연결하고 다른 한쪽을 GND에 연결한다.
 3. 스위치 입력이 없을 때 2번핀의 상태는 HIGH 신호인지 LOW신호인지 알 수가 없다. 그러므로 반드시 핀의 입출력 설정 때 'INPUT_PULLUP' 명령어를 사용하여 풀업시켜줘야 한다.
 4. 풀업을 해줬다면 스위치 입력이 없을 때 2번핀의 상태는 HIGH신호이고 스위치 입력이 있을 때 2번핀의 상태는 LOW신호이다.





5.2 디지털 신호 입력 – 안정된 Switch

EX 5.2

안정적인 스위치 입력 (2/3)

Sketch 구성

1. LED의 핀 번호를 설정한다.
2. `setup()`에서는 LED 출력으로 사용할 핀을 출력핀으로 설정한다.
3. 스위치 누른 횟수를 저장할 변수를 설정한다. (`count`)
4. 시리얼모니터로 스위치 누른 상태를 점검한다.

실습 결과

1. 스위치를 누르면 LED가 점등되고, 손을 떼면 LED가 소등된다.
2. 스위치를 누르는 동안에 시리얼 모니터로 카운트 값이 증가하는 것을 볼 수 있다.

실행 시 어떤 문제가 발생하는가? → 오 작동을 해결하는 방법을 찾아보시오.



5.2 디지털 신호 입력 – 안정된 Switch

EX 5.2

안정적인 스위치 입력 (3/3)

응용 문제 스위치를 누르고 있는 동안에는 'if (swInput == LOW){ }' 내의 명령어는 수없이 반복한다. 실제 디지털 입력을 받을 때에는 단 **한번의 명령만 실행되어야** 한다. 이를 위해서 이 예제의 스위치가 눌렸을 때 처리되는 부분을 다음과 같이 수정해 보자.

```
// 스위치가 눌렸을 때
if (swInput == LOW){
    delay(100);
    if (ledOutput) digitalWrite(ledPin, LOW); // LED가 점등되어 있으면 소등
    else digitalWrite(ledPin, HIGH); // LED가 소등되어 있으면 점등
    ++count;
}
// 스위치 입력 횟수를 시리얼 통신으로 전송한다.
Serial.println(count);
```

실행 시 어떤 문제가 발생하는가? → 해결 방법을 찾아 보시오.



5.2 디지털 신호 입력 – 안정된 Switch

DIY 8. 현재의 LED 상태 (LOW or HIGH)와 스위치 누른 횟수를 아래와 같이 출력하는 code를 만들어서 [HPnn_Switch.ino](#)로 저장하시오.

```
COM6
LED: 0, count: 1
LED: 1, count: 2
LED: 0, count: 3
LED: 1, count: 4
LED: 0, count: 5
LED: 1, count: 6
LED: 0, count: 7
LED: 1, count: 8
LED: 0, count: 9
LED: 1, count: 10
```

```
COM6
LED: LOW, count: 1
LED: HIGH, count: 2
LED: LOW, count: 3
LED: HIGH, count: 4
LED: LOW, count: 5
LED: HIGH, count: 6
LED: LOW, count: 7
LED: HIGH, count: 8
LED: LOW, count: 9
LED: HIGH, count: 10
```

```
void loop(){
    int swInput = digitalRead(inputPin); // 스위치 입력을 받는다
    int ledOutput = digitalRead(ledPin); // LED의 출력 상태를 확인한다

    if (swInput == LOW){
        delay(400);
        if (ledOutput) {
            // 스위치 입력 횟수를 시리얼 통신으로 전송한다.
            Serial.print(", count: ");
            Serial.println(count);
        }
        ++count;
    }
}
```

insert your code here!

[HPnn_Switch.ino](#)로 저장



[Practice]

- ◆ [wk12]
 - Arduino coding IV
 - Complete your codes
 - Upload file name : HPnn_Rpt07.zip

wk12 : Practice-07 : HPnn_Rpt07.zip

◆ [Target of this week]

- Complete your 7th project
- Save your outcome and compress all files.

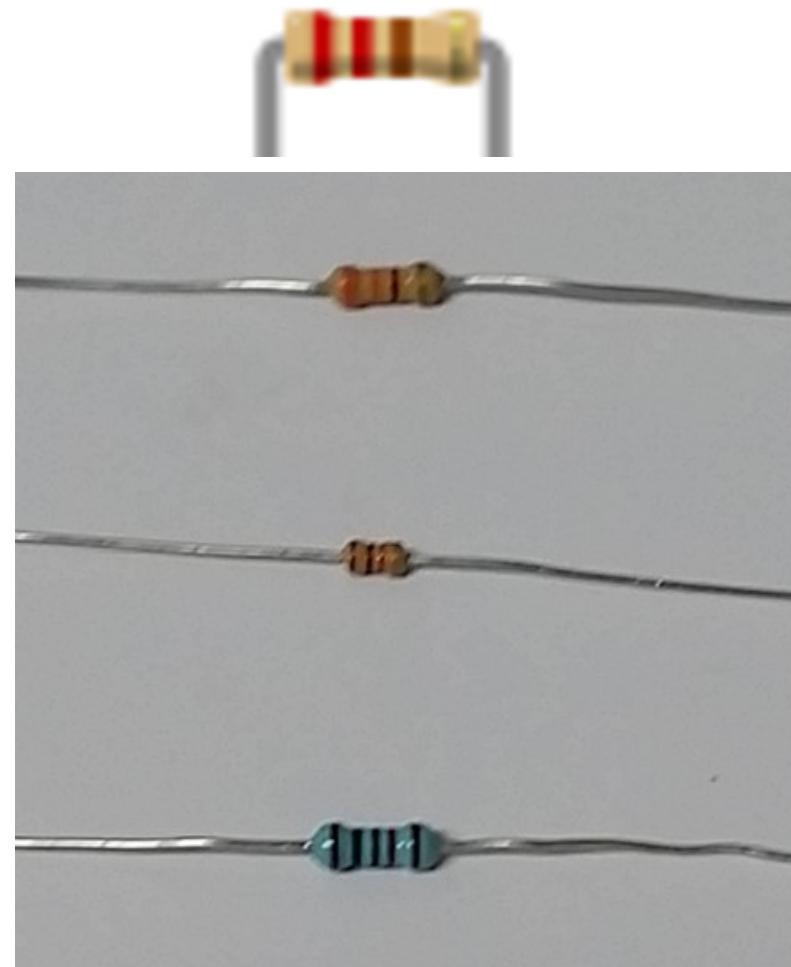
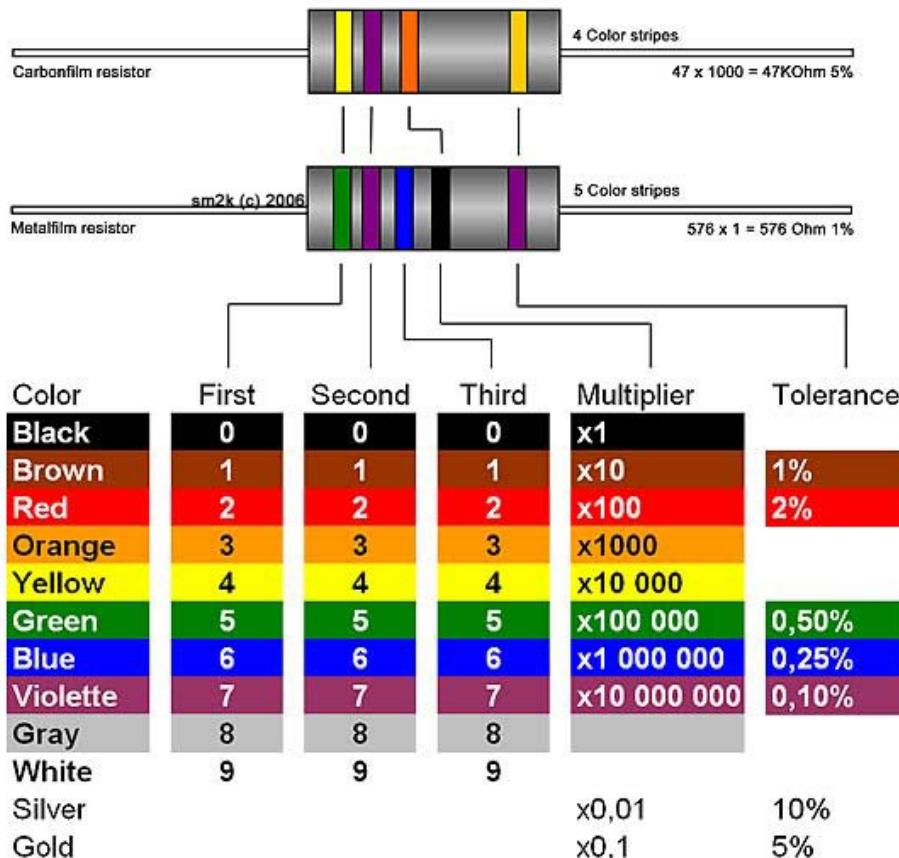
제출파일명 : **HPnn_Rpt07.zip**

- 압축할 파일들

- ① **HPnn_4led.ino**
- ② **HPnn_RGB.png**
- ③ **HPnn_Switch.ino**



[참고 : 저항 값 읽기]





1장 Arduino 소개 및 사용법

2장 시리얼 통신

3장 LCD 출력

4장 LED 출력 1

4장 LED 출력 2

4장 LED 출력 3

5장 디지털 신호 입력

중간평가

6장 아날로그 신호 입력 1

| 6장 아날로그 신호 입력 2

| 7장 모터 구동

| 8장 적외선 리모컨

| 9장 여러 가지 부품들

| 10장 프로젝트 1

| 10장 프로젝트 2

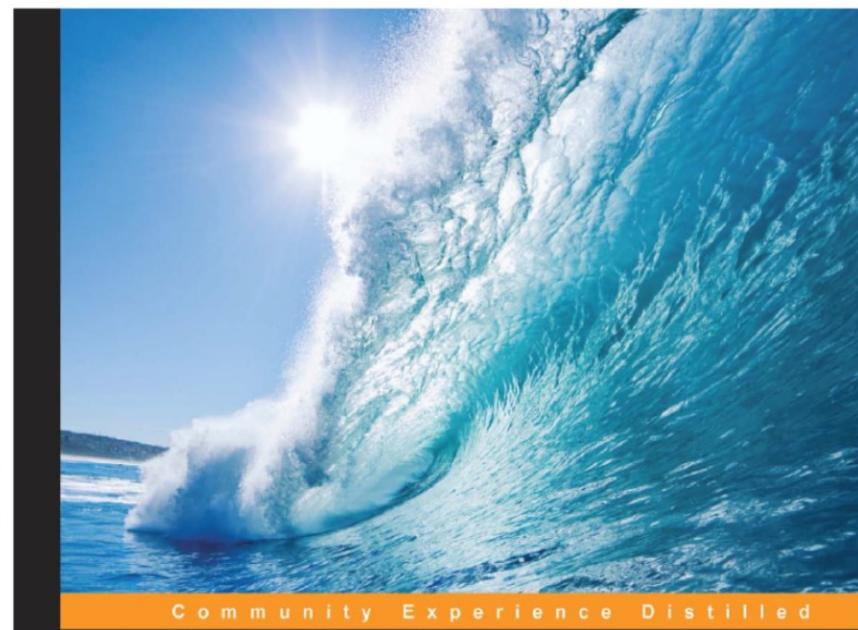
| 기말평가

생능출판사

2016



References

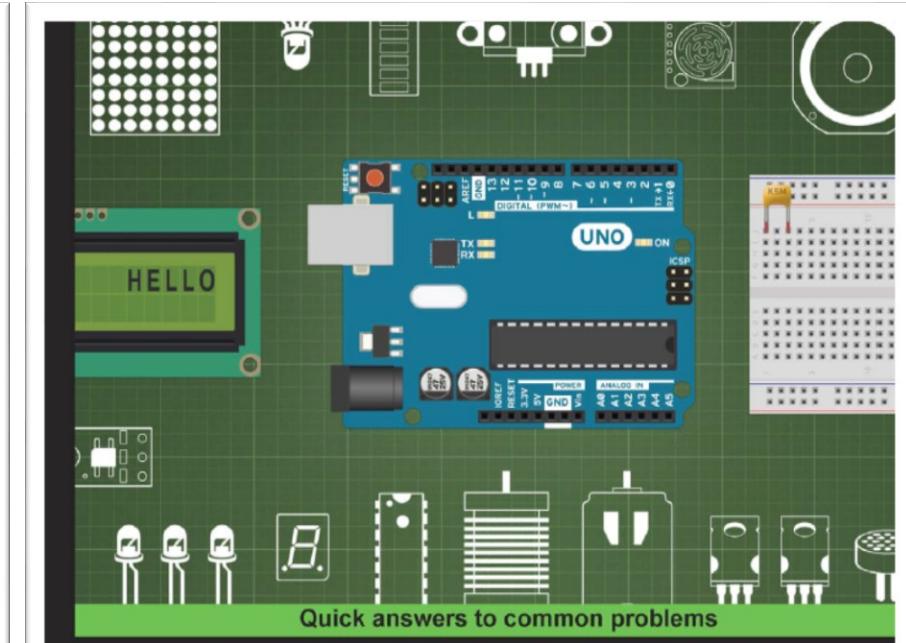


Arduino Essentials

Enter the world of Arduino and its peripherals and start creating interesting projects

Francis Perea

[PACKT]
PUBLISHING



Arduino Development Cookbook

Over 50 hands-on recipes to quickly build and understand Arduino projects, from the simplest to the most extraordinary

Cornel Amariei

[PACKT] open source*
PUBLISHING