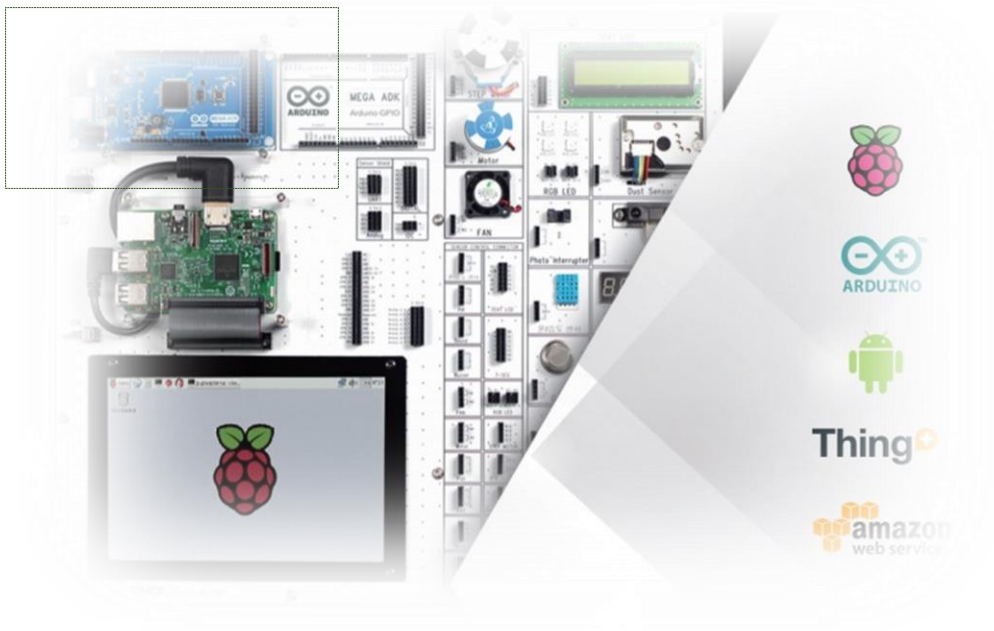
마이크로프로세서
(강의자료 #4)



교과목명 : 마이크로프로세서 (01)

담당교수 : 이 수 형

E-mail : soohyong@uu.ac.kr

교재명 : 스마트기기 개발을 위한 아두이노, 이수형. (LINC+ 사업단 배포)

수강생 안내

- 본 교과목은 기본적으로 “대면”수업입니다.
- ‘코로나바이러스감염증-19’ 등의 이유로 대면수업을 받지 못하는 학생들을 위해서 ‘비대면 실시간 수업’을 병행합니다. 기본적으로 대면 수업이므로 대면 수업에 참석한 학생들은 교실에서 출석을 체크하면 되며, 참석하지 못하고 ‘비대면 수업’을 듣는 학생들은 ‘실시간 온라인 강의’에 참석하면 출석이 반영됩니다.
- 수강생 여러분들은 강의 자료를 온라인 배포 및 게재 등의 행위를 할 시 저작권 문제가 발생할 수 있사오니 이에 유념하여 강의 자료를 공유하는 행위는 삼가 바랍니다.
- 교재는 “스마트기기 개발을 위한 아두이노”이며, 나누어준 키트는 각자 잘 관리하면서 실습실(대면) 및 집(비대면)에서 실험/실습을 수행할 수 있도록 진행합니다.

지난시간에는?

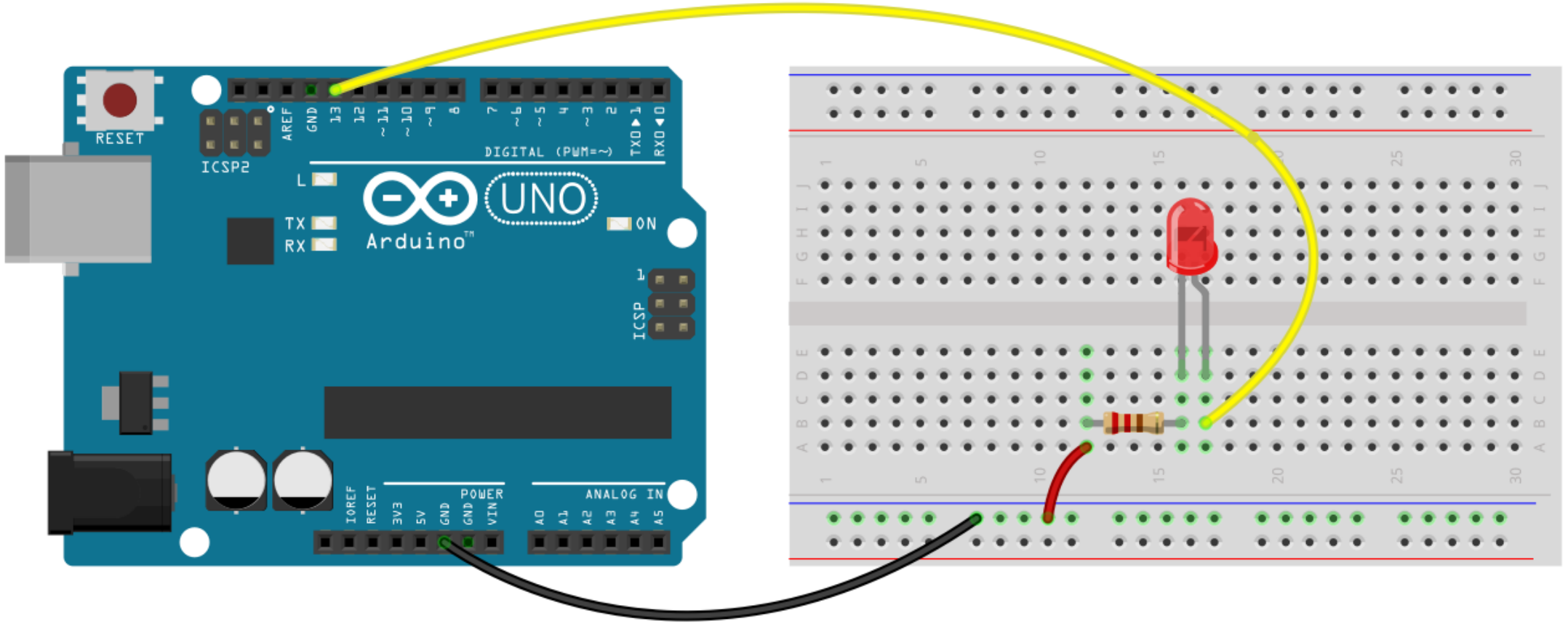
아두이노란

- 아두이노란?
- 아두이노의 종류
  - Uno, Mega, Leonardo, ...
- 아두이노 소프트웨어 개발
  - C/C++을 이용한 통합개발환경 (IDE)
  - setup() 함수 및 loop() 함수로 구성
  - 디지털 입/출력 설정 : pinMode(port, INPUT or OUTPUT)
  - 디지털 출력 : digitalWrite(port, HIGH or LOW)
  - 시간 지연 (기다리기) : delay(ms)

4.1 LED를 이용한 디지털 신호 출력

Review

- 회로 구성



- 스케치

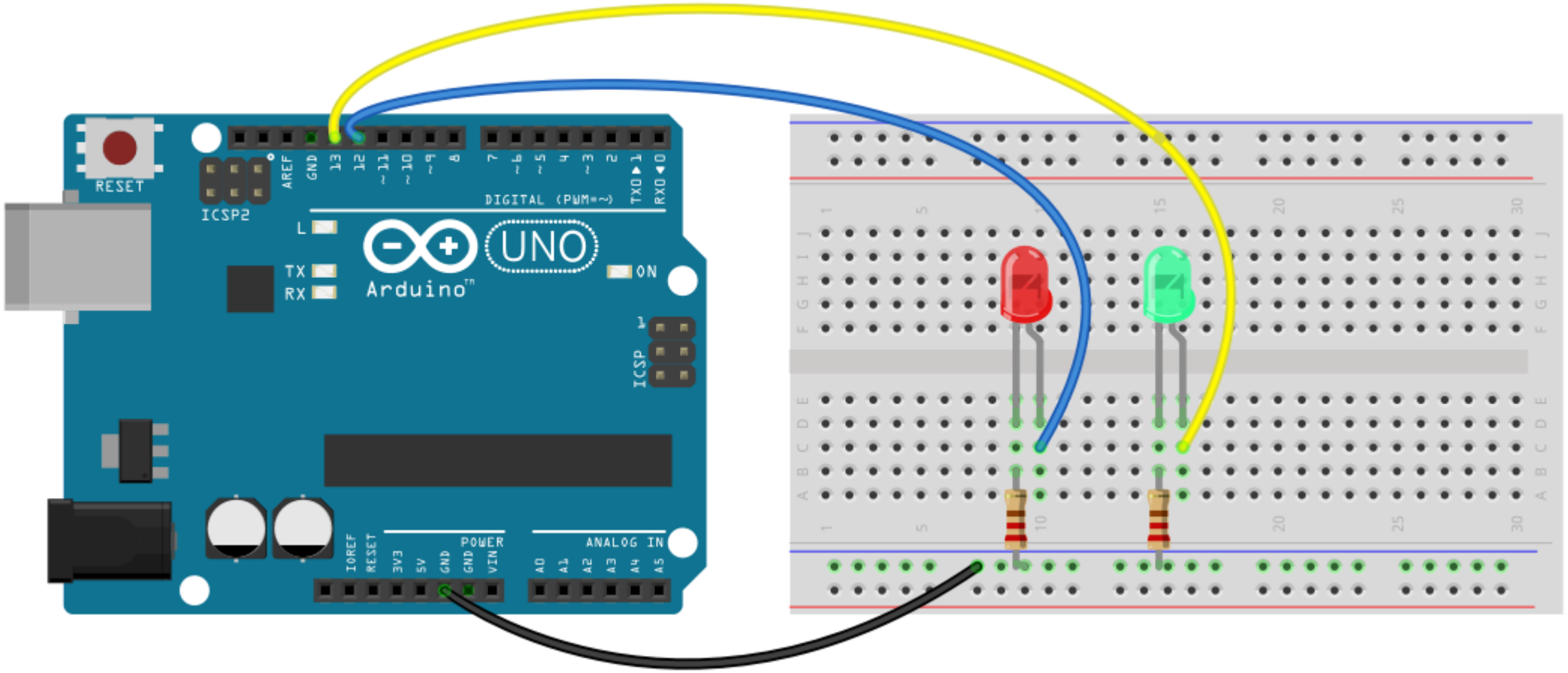
예제 4-1. 첫번째 스케치

```
int led = 13;

// 처음 시작시 한번만 수행하는 함수, 설정을 담당한다
void setup() {
    pinMode(led, OUTPUT);     // 13번 핀을 출력으로 설정한다
}

// 반복해서 호출되는 함수
void loop() {
    digitalWrite(led, HIGH);  // 13번 핀으로 5V 디지털 신호를 출력한다
    delay(1000);              // 1000 ms = 1초를 기다린다.
    digitalWrite(led, LOW);    // 13번 핀으로 0V 디지털 신호를 출력한다
    delay(1000);              // 1초를 기다린다.
}
```

4.2 두개의 LED를 제어하기

Review

- 회로 구성



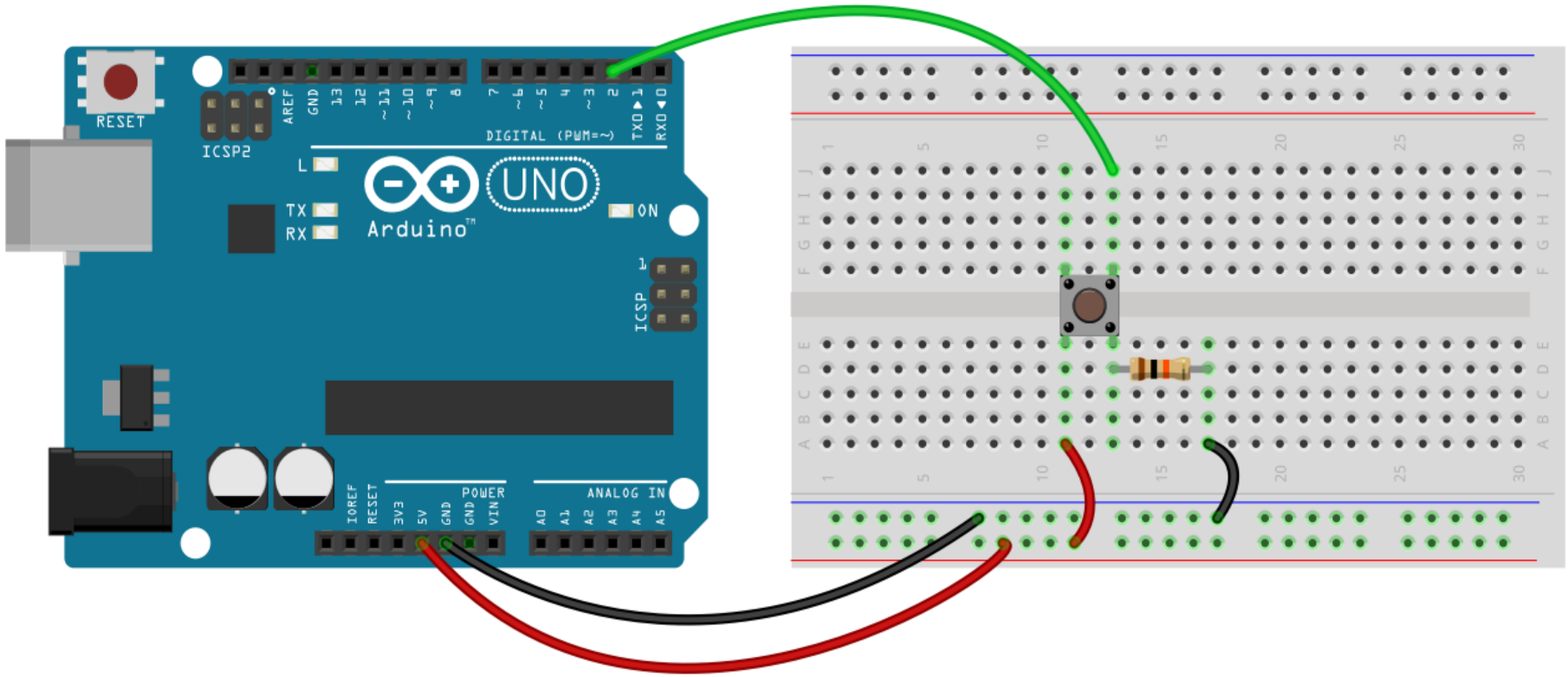
예제 4-2. 두 개의 LED 순차 점등

```
int ledA = 13;
int ledB = 12;

// 처음 시작시 한번만 수행하는 함수, 설정을 담당한다
void setup() {
    pinMode(ledA, OUTPUT);      // 13번 핀을 출력으로 설정한다
    pinMode(ledB, OUTPUT);      // 12번 핀을 출력으로 설정한다
}

// 반복해서 호출되는 함수
void loop() {
    // LED 1만 켜기
    digitalWrite(ledA, HIGH);  // LED 1 켜기
    digitalWrite(ledB, LOW);   // LED 2 끄기
    delay(1000);               // 1초를 기다린다.
    // LED 2만 켜기
    digitalWrite(ledA, LOW);   // LED 1 끄기
    digitalWrite(ledB, HIGH);  // LED 2 켜기
    delay(1000);               // 1초를 기다린다.
}
```

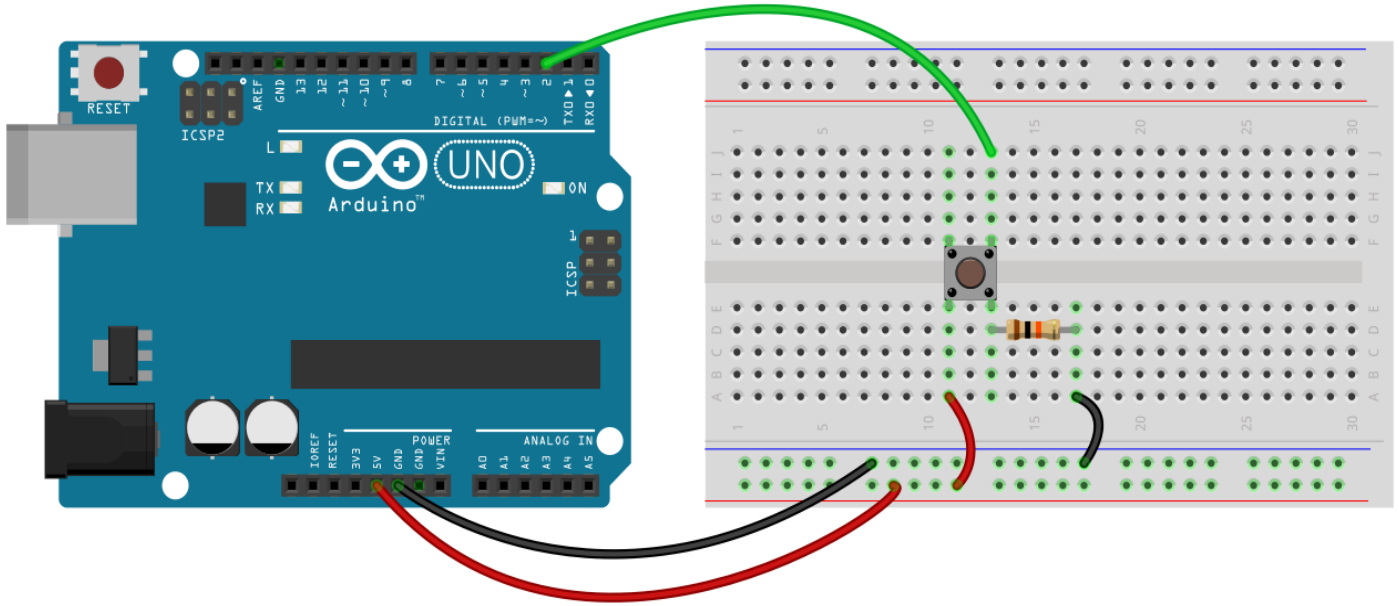
4.3 스위치를 이용한 디지털 입력

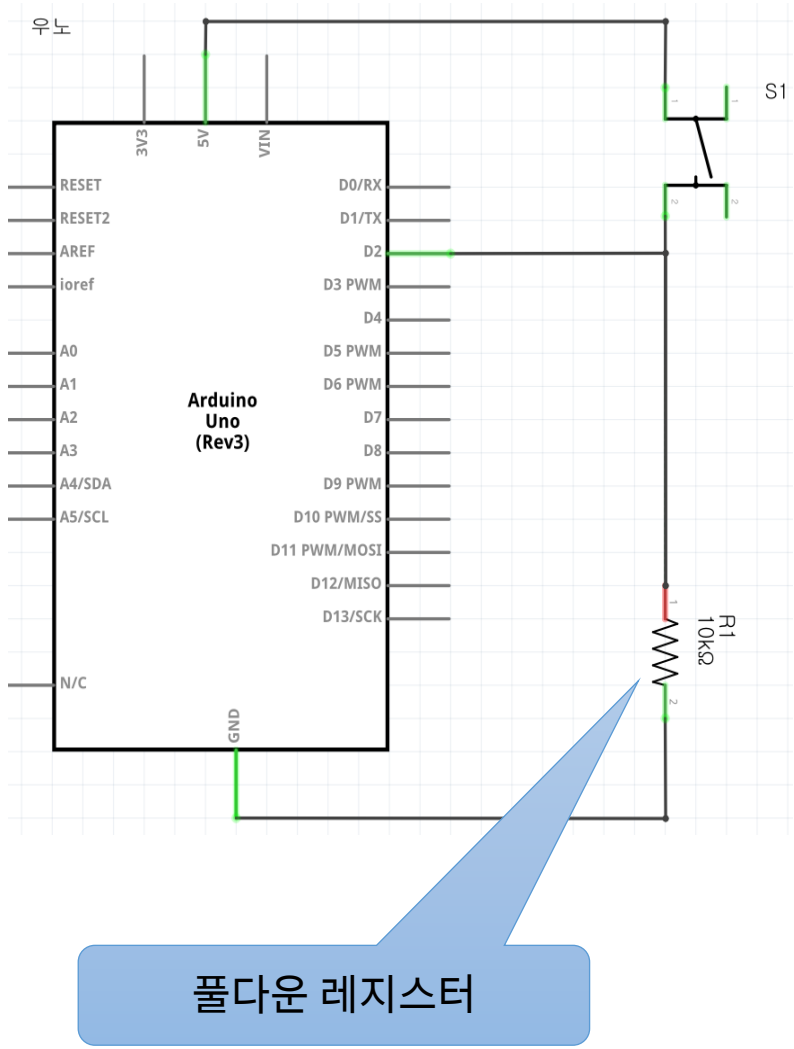
Review

- 택트 스위치를 이용한 입력



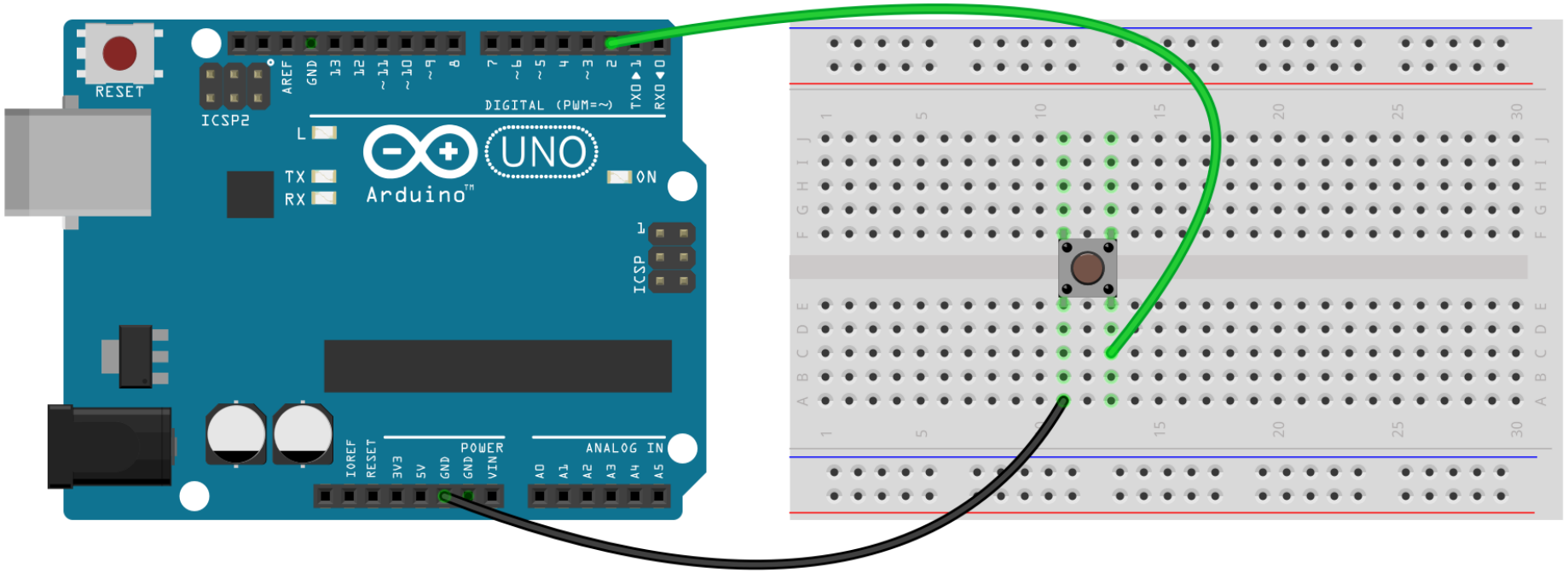
- 회로도
  - 저항의 역할 : 풀다운(pull-down) 저항
  - 스위치가 Off인 경우에 GND로 연결 → LOW
  - 스위치 On인 경우에는 5V 인가 → HIGH





Internal pull-up register

- 아두이노 내부에 존재하는 풀업 저항
  - `pinMode(INPUT)` 대신 `pinMode(INPUT_PULLUP)` 사용
  - 외부에 별도의 풀업 저항이 필요 없음 ⇒ 회로가 단순해짐
  - 풀다운 → 풀업 : 스위치 On/Off시에 digitalRead()의 결과가 바뀜
    - ➢ On → LOW, Off → HIGH



예제 4-3. 디지털 입력 프로그램

```
// 디지털 입/출력 핀 2번을 스위치의 입력으로 설정한다.
int pushButton = 2;

void setup() {
  // 시리얼 통신을 위하여 초기화하는 과정, 9600보레이트(baud-rate)로 설정한다.
  Serial.begin(9600);
  // 버튼 스위치가 연결된 2번 핀을 입력 모드로 설정한다.
  pinMode(pushButton, INPUT);
}

void loop() {
  // 현재 버튼이 연결되어 있는 핀의 값을 읽어들인다.
  int buttonState = digitalRead(pushButton);
  // 입력받은 값을 시리얼 통신을 통해서 PC로 전송한다.
  Serial.println(buttonState);
    // 입력 상태를 안정화하기 위하여 1미리초 동안 대기한다
  delay(1);
}
```

UART 통신

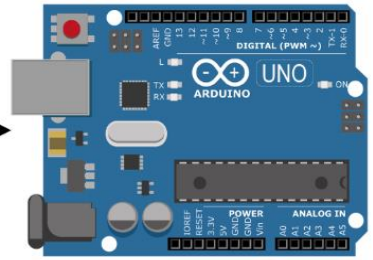
Review

- 시리얼 통신
  - 아두이노와 PC와의 통신, Serial 객체 사용
  - UART(Universal Asynchronous Receiver and Transmittor)
  - 아두이노 : H/W 방식(디지털 포트 0번과 1번), 추가 시리얼통신은 S/W 방식
- 방법
  - 초기화
    - `Serial.begin(9600);`
  - 전송 (PC ← 아두이노)
    - `Serial.print(전송값);`
    - `Serial.println(...);`
  - 전송 (PC → 아두이노)
    - `if(Serial.available()) { char data = Serial.read(); ... }`

아두이노 IDE & 시리얼 모니터

USB 연결

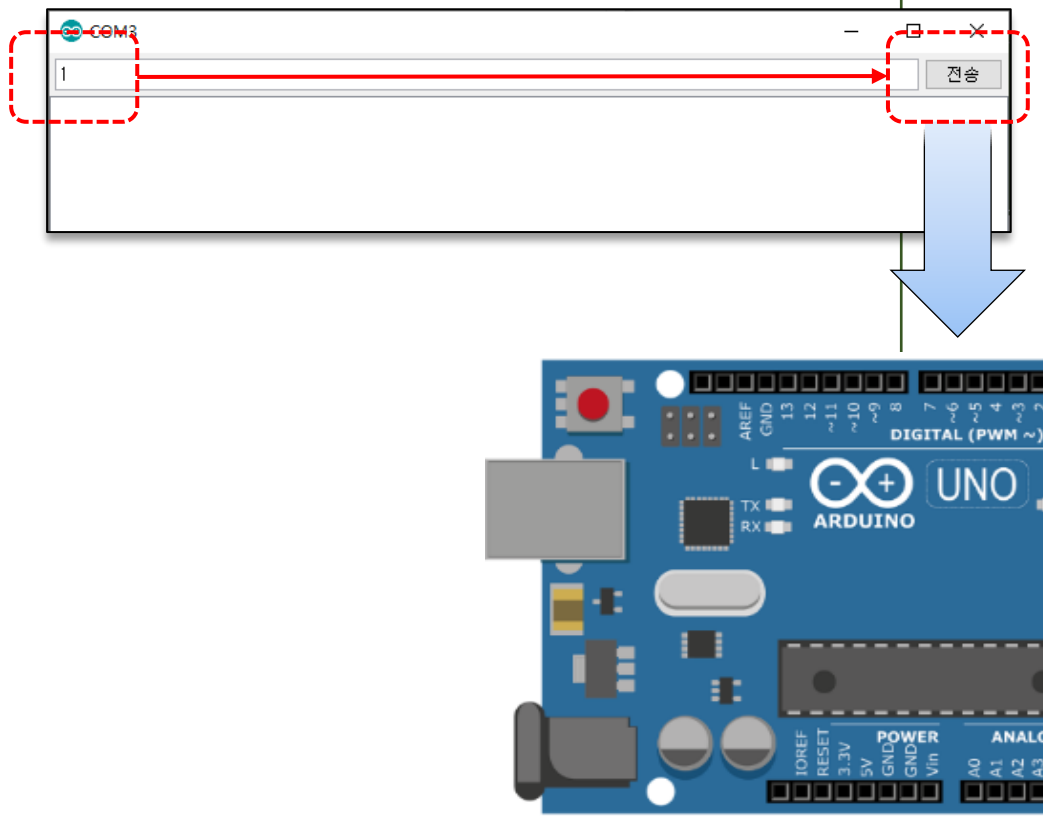
시리얼 통신



Serial 객체 사용

예제: 시리얼 입력을 통한 LED 제어

```
void setup() {
  // LED_BUILTIN : 내장된 LED 포트 번호 == 13
  Serial.begin(9600);
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  // 시리얼 통신 검사 후 데이터 읽기
  if(Serial.available()) {
    char ch = Serial.read();
    if(ch == '1')
      digitalWrite(LED_BUILTIN, HIGH);
    else if(ch == '2')
      digitalWrite(LED_BUILTIN, LOW);
  }
  delay(10);
}
```

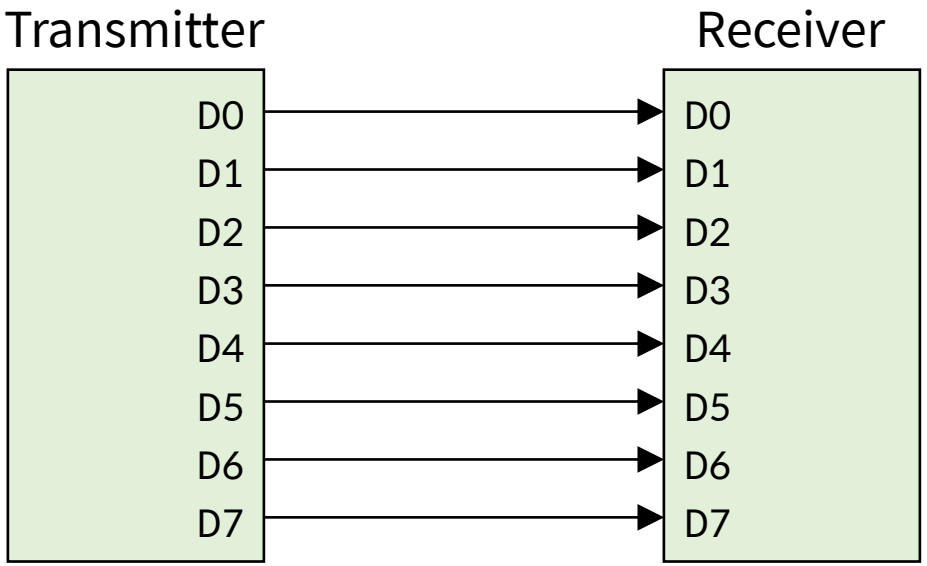


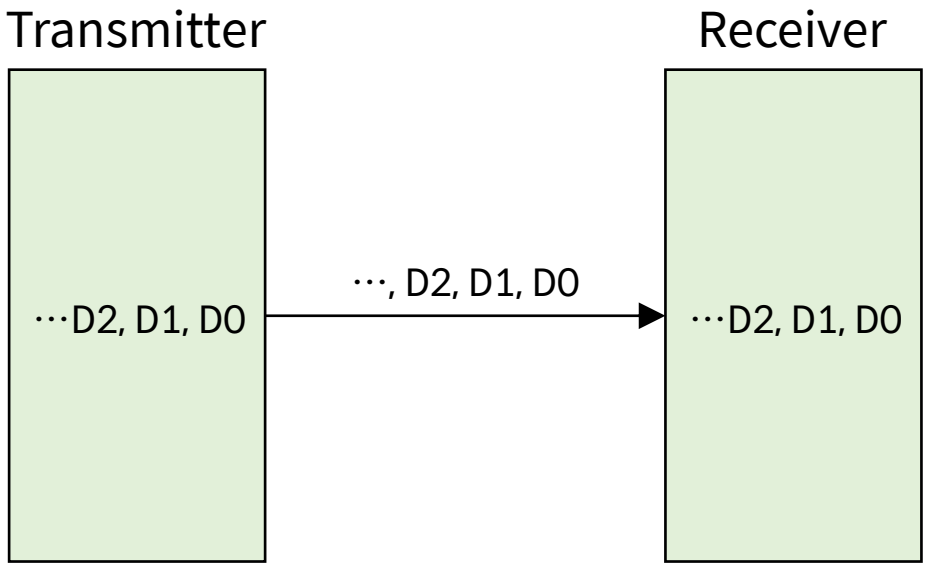
[참고] 디지털 통신

- 통신 (communication)
  - 송신자와 수신자가 상호 간에 데이터(정보)를 주고 받는 행위
  - 방식 : 아날로그 통신 (라디오, AM/FM), 디지털 통신
    AM : Amplitude Modulation, FM : Frequency Modulation
- 디지털 통신
  - 무선 디지털 통신
    - 송신 : 디지털 정보를 반송주파수(carrier)로 변조(modulation)한 후 무선으로 정보를 전송
    - 수신 : 수신된 신호를 복조(demodulation) 후 디지털 정보로 변환
    - 예 : 디지털 TV 방송, 무선인터넷(WiFi), 블루투스(Bluetooth)
  - 유선 디지털 통신
    - 2개 이상의 기기들이 서로 데이터를 주고 받음
    - 방식 : 병렬 통신, 직렬 통신

디지털 통신 방식

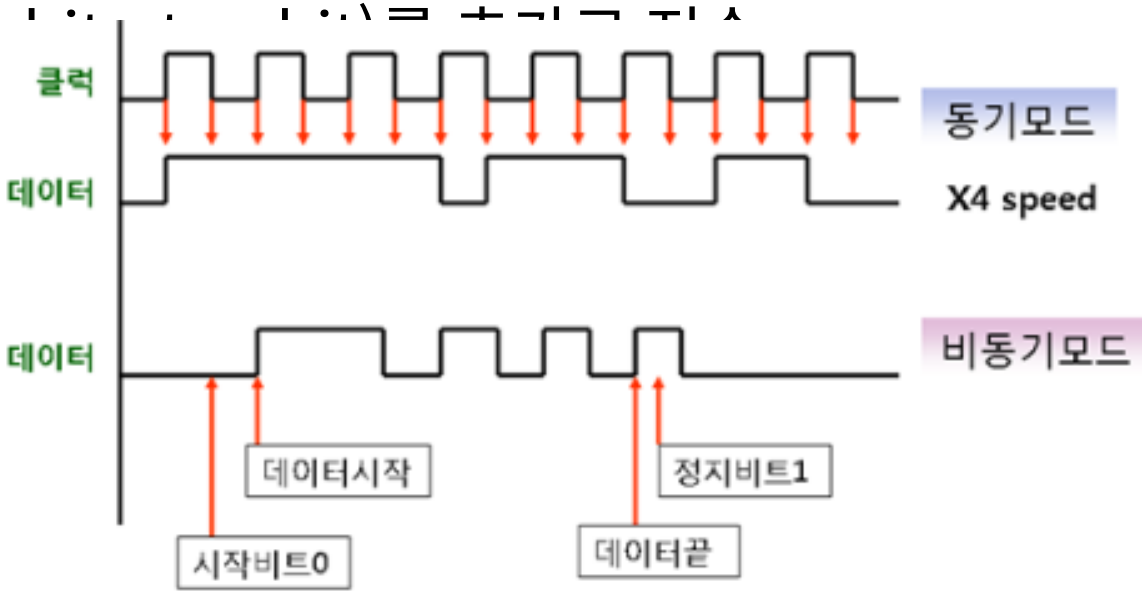
- 병렬 통신 vs. 직렬 통신
  - 병렬 통신 : 전송하고자 하는 모든 데이터(예: 1 바이트)를 선으로 연결하여 전송
    - ➢ 고속 전송 → 최근 : 초고속(수 GHz 이상)의 전송은 동기화가 어려움 (컴퓨터 내부만 사용)
  - 직렬 통신 : 데이터를 한 비트씩 순차적으로 전송
    - ➢ 하나의 선을 통하기 때문에 간단하게 구현
    - ➢ 저속 → 최근 : 프로세서의 처리속도 발전 및 통신 기술의 발달로 고속화가 가능해짐





직렬 통신의 방법

- 동기(Synchronous) 통신
  - 송신기와 수신기의 동기를 맞춰주는 Clock 신호를 추가로 연결하여 사용함
- 비동기(Asynchronous) 통신
  - 동기 신호 없이 통신 → 데이터의 동기를 맞춰주는 추가적인 방법을 사용해야 함
    ⇒ 데이터의 중간에 동기 신호 (star
  - 미리 통신 속도를 알고 있어야 함
- 대표적인 통신 방식
  - UART, I2C, SPI

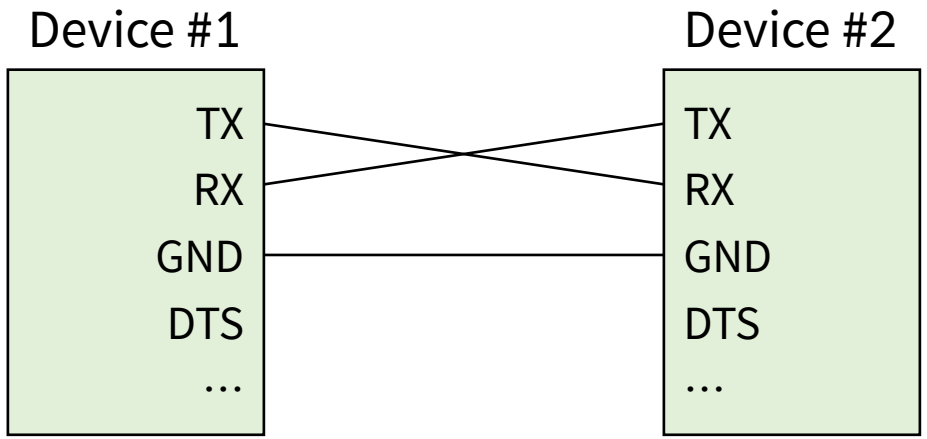


$11101010_{(2)}$ 데이터를 직렬전송시 비동기와 동기식의 비교도

그림 출처 : https://m.blog.naver.com/scw0531/220619256523

UART

- UART (Universal Asynchronous Receiver/Transmitter)
  - 미리 정해놓은 통신 속도(baud)에 맞추어서 송신하고자 하는 데이터를 직렬화하여 전송하고 수신한 데이터를 병렬화하여 데이터를 복원하는 방법
  - TX(Transmitter), RX(Receiver) 라인(line)을 사용하여 통신
  - 경우에 따라 부가적인 제어선(DTS/DTR 등)를 사용하기도 함
  - 종류 : RS-232, RS-422, RS-485



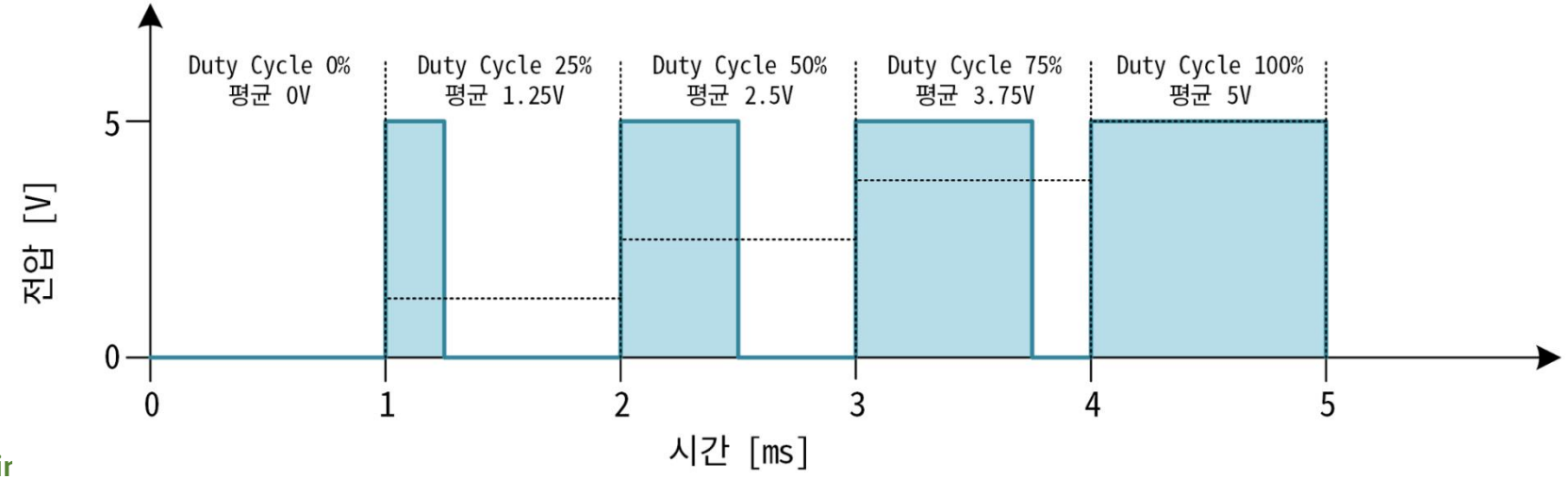
아두이노, NodeMCU 등에서는
RS-232 통신을 사용
(※ 이전 자료 참고)

참고 : http://blog.daum.net/shbaek1009/6766113

5.1 아날로그 출력

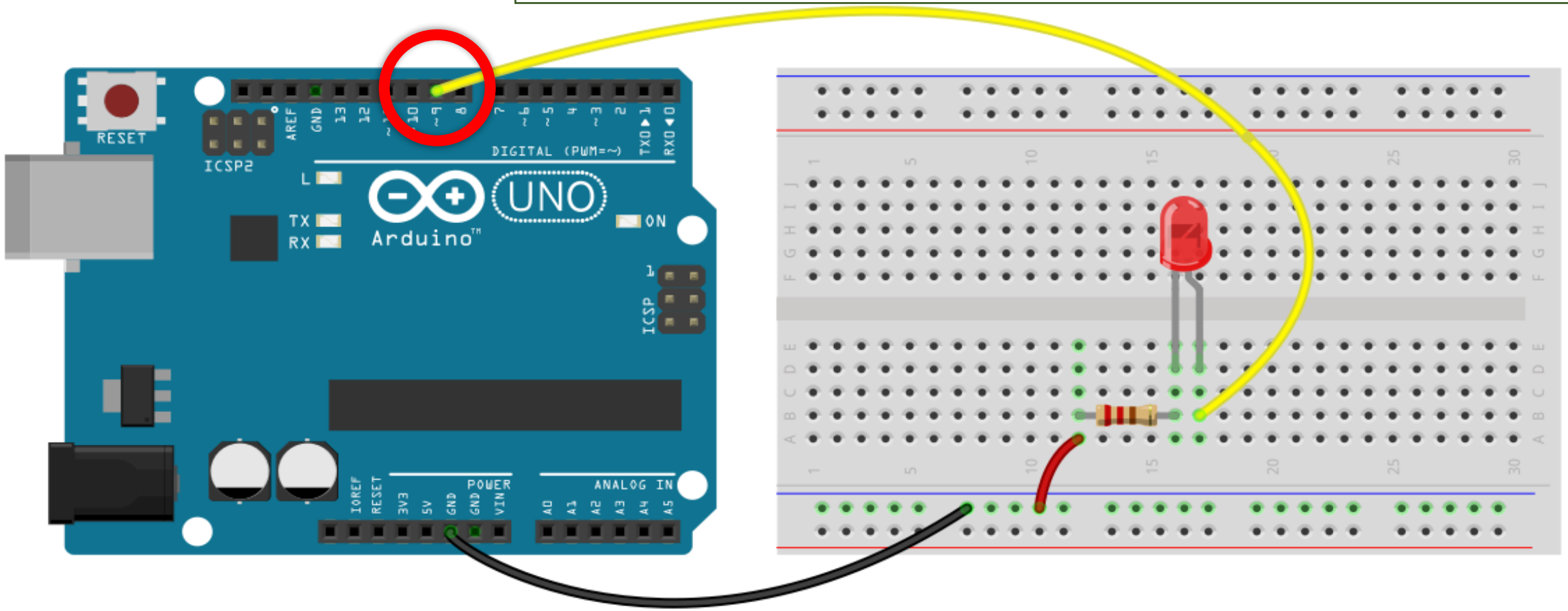
Review

- 아두이노 우노의 입/출력 기능
  - 14개의 디지털 입/출력 기능, 6개의 아날로그 입력 기능
  - 아날로그 출력 기능 → PWM(Pulse Width Modulation) 출력으로 해결
  - PWM(Pulse Width Modulation)
    - ➢ 주기적인 사각형 펄스의 5V, 0V의 발생 시간을 조절 하면서 평균전압을 제어
    - ➢ 5V, 0V의 발생 시간의 비 : 듀티 사이클 (duty cycle)
    - ➢ 아두이노 : 6개의 디지털 출력핀이 가능 (~기호로 표시)
    - ➢ analogWrite(9, 127); → 9번 핀으로 50%듀티 사이클 출력

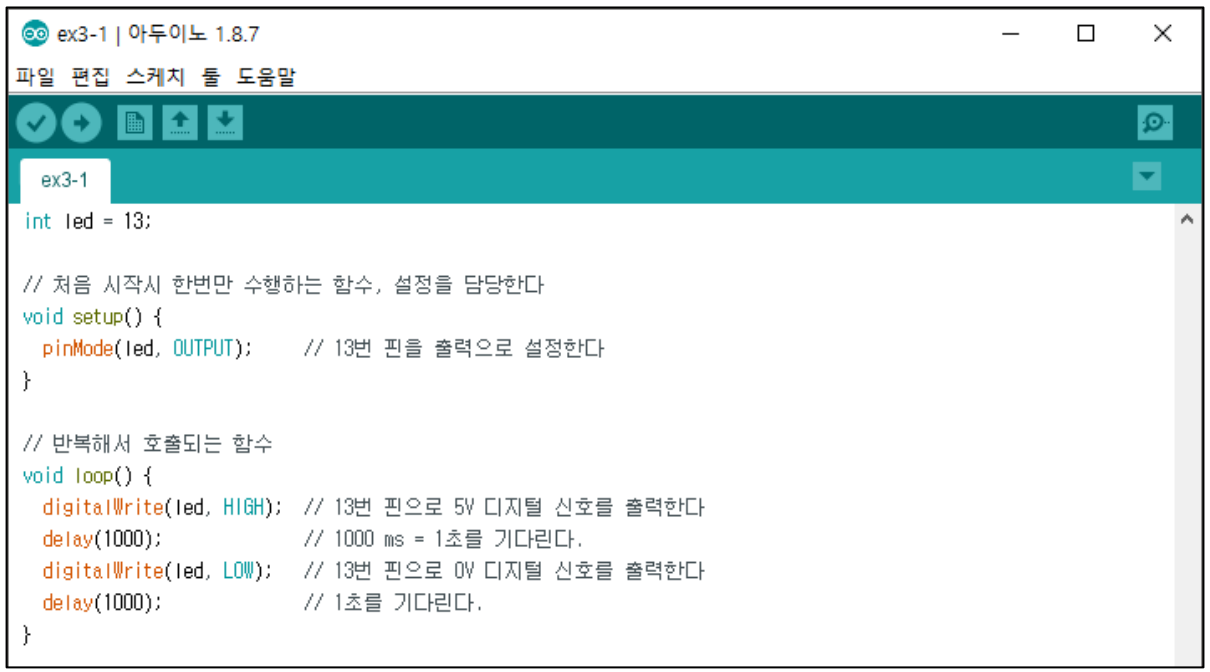


- 회로 구성

```
digitalWrite(9, HIGH;   →  5V digital signal
analogWrite(9, 0);      →  0% Duty Cycle Pulse (≈ 0V)
analogWrite(9, 128);    →  50% Duty Cycle Pulse (≈ 2.5V)
analogWrite(9, 255);    →  100% Duty Cycle Pulse (≈ 5V)
```

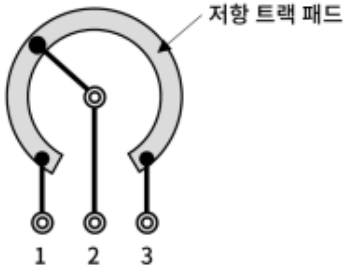


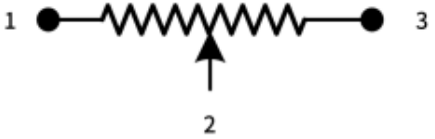
5. 아날로그 입출력의 기초 #2

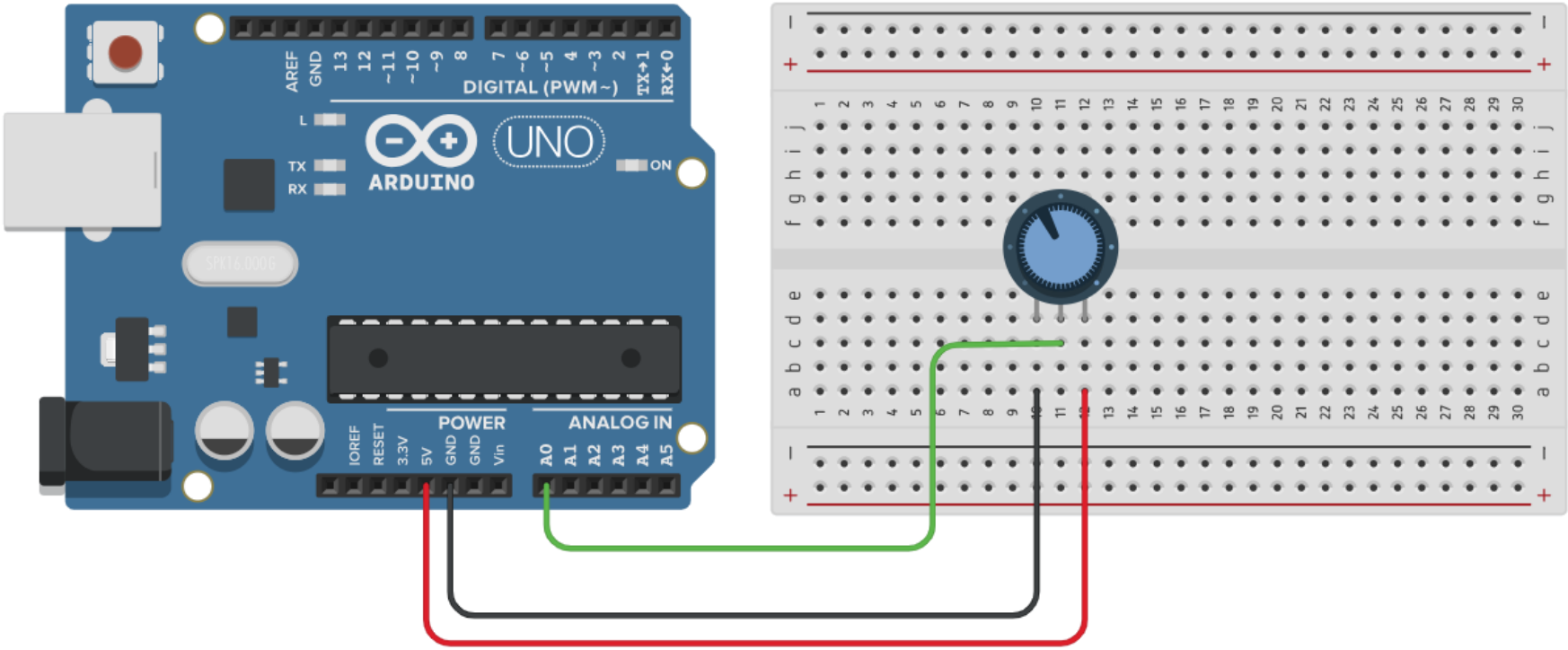


5.2 포텐쇼미터를 이용한 아날로그 입력

- 포텐쇼미터(potentiometer)
  - 가변저항
- 회로구성

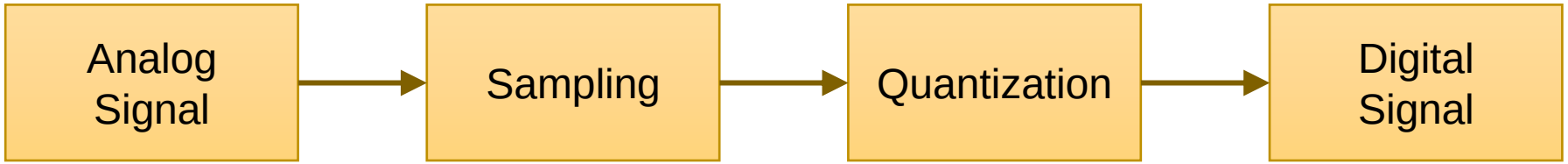






아날로그 신호 입력

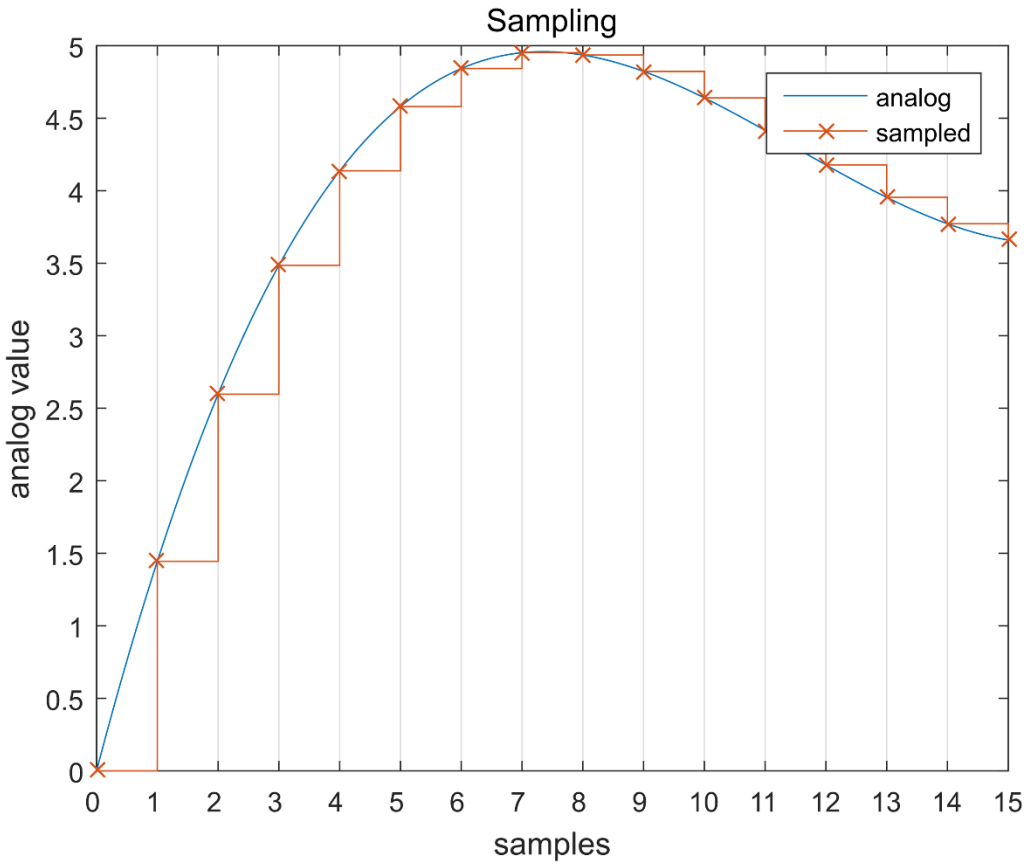
- 아날로그 신호 → 디지털 신호
  - Analog : 자연계의 물리적인 정보
  - Digital : 컴퓨터가 처리하는 정보



- 샘플링 (sampling)
  - 샘플링 주기에 따른 값만 사용 (예: 1초당 44100번)
- 양자화 (quantization)
  - 물리 레벨을 주어진 단계로 나누어서 이진화

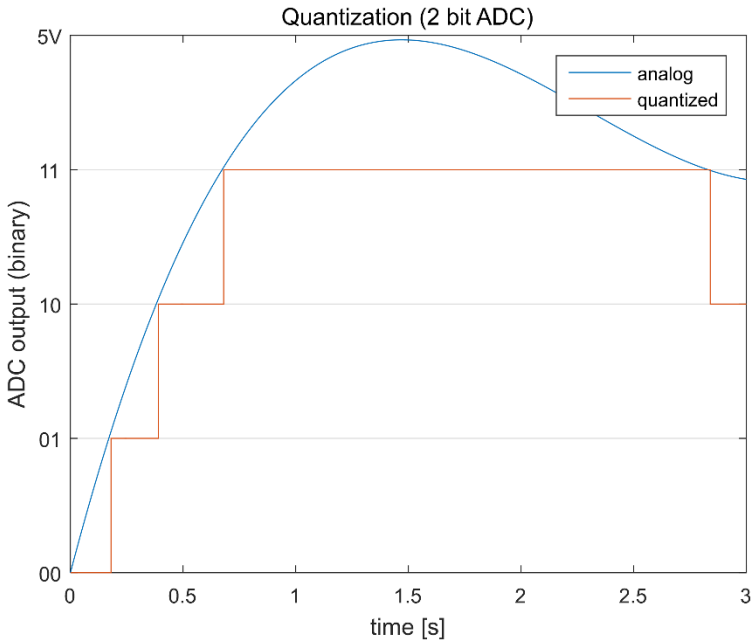
아날로그 신호 입력

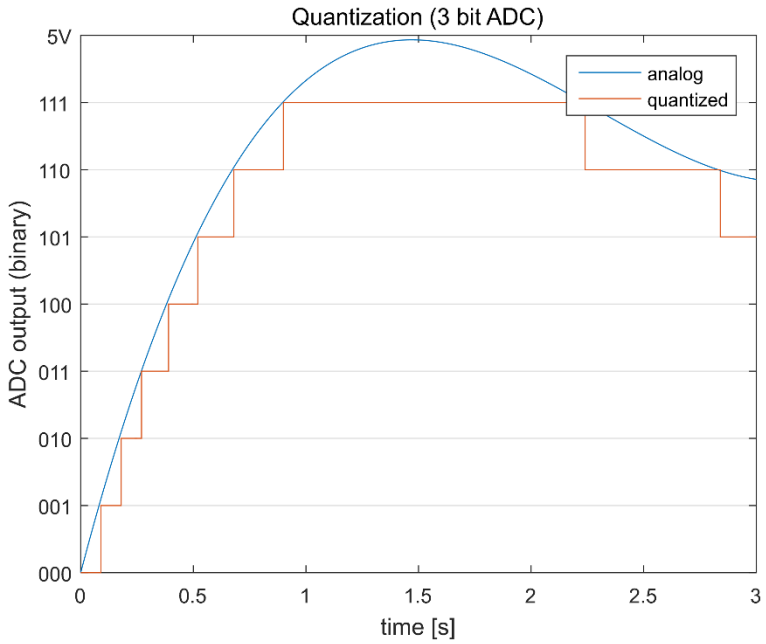
- 샘플링 (Sampling)
  - 일정 시간 (샘플링 주기 = 1 / 샘플링 주파수) 마다 값을 읽어들임 (중간값 무시)
  - 디지털 방식으로는 무한히 연속적인 신호를 저장할 수 없음
  - 샘플링 주파수 설정 : 신호의 정보를 포함할 수 있을 정도의 짧은 시간
    - ➢ 예] 오디오 샘플링 주파수 = 44.1kHz : 인간의 가청주파수를 고려

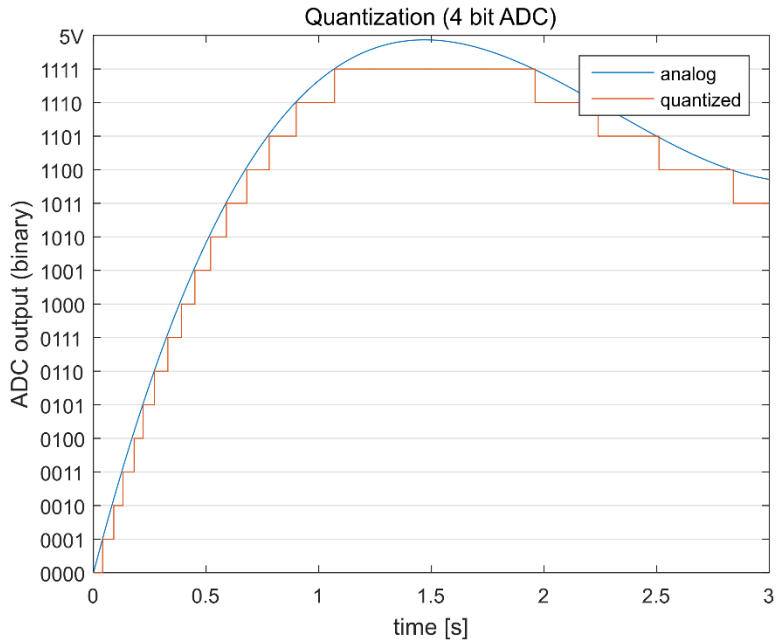


아날로그 신호 입력

- 양자화 (Quantization)
  - 연속적인 물리량의 값을 일정한 레벨 로 나누어서 변환하는 과정
  - ADC(Analog to Digital Converter) 사용
  - 내부회로의 비교기의 수에 따라 해상도가 결정됨 (3bit → 3번 비교 → 2^3레벨)
  - 아두이노 : 10bit ADC 사용 ⇒ $2^{10} = 1024$ 레벨

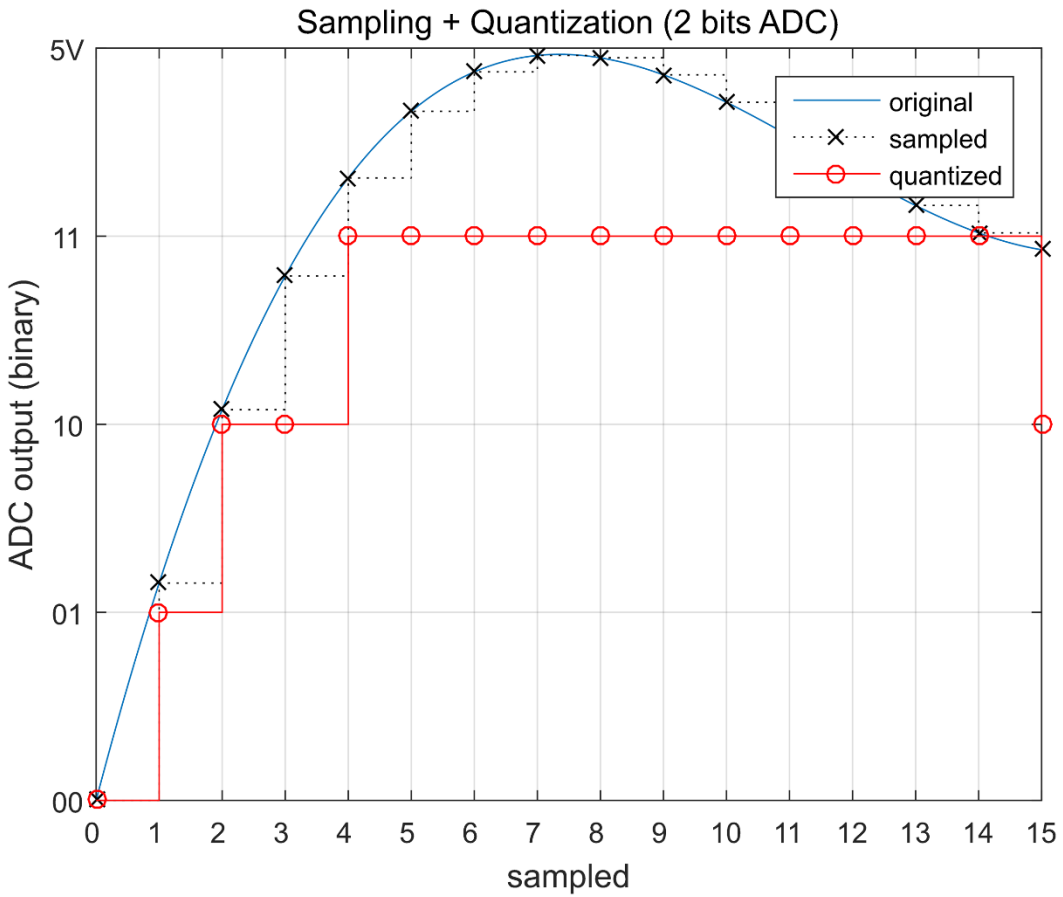


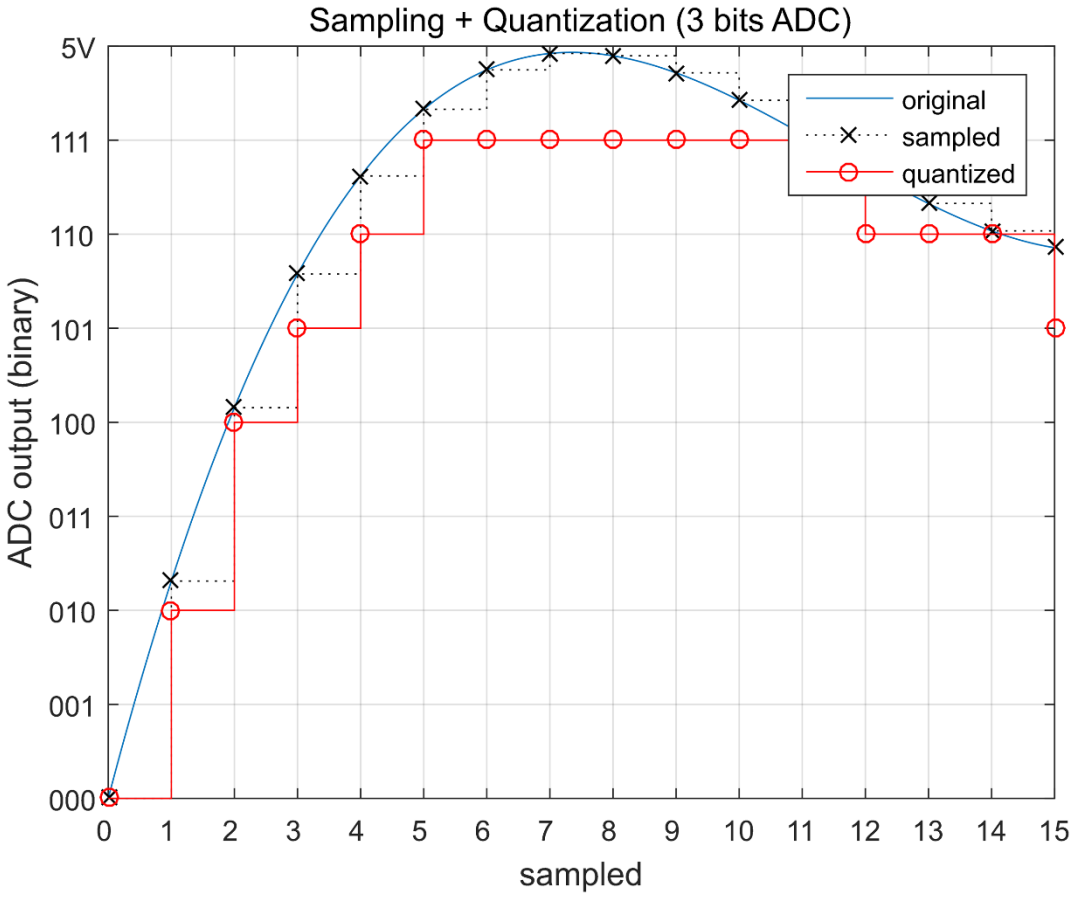




아날로그 신호 입력

- 아날로그 → 디지털 (샘플링 된 값의 양자화)





아두이노의 아날로그 입력

- 아날로그 센서 읽기 (예: A0)
  - 초기화 : `setup()`함수에서 `pinMode(A0, INPUT);`
  - 아날로그 값 읽기 : `int value = analogRead(A0);`
  - 입력 : 0 ~ 5V 전압 측정
  - 값의 범위 : 0 ~ 1023 (아두이노에서는 10 bit ADC를 사용)
    0000000000_2 (0_{10}) ~ 1111111111_2 (1023_{10})
  - 입력전압의 계산 : 입력전압 $= \frac{\text{아날로그입력값}}{1024} \times 5\ [\text{V}]$

- 스케치

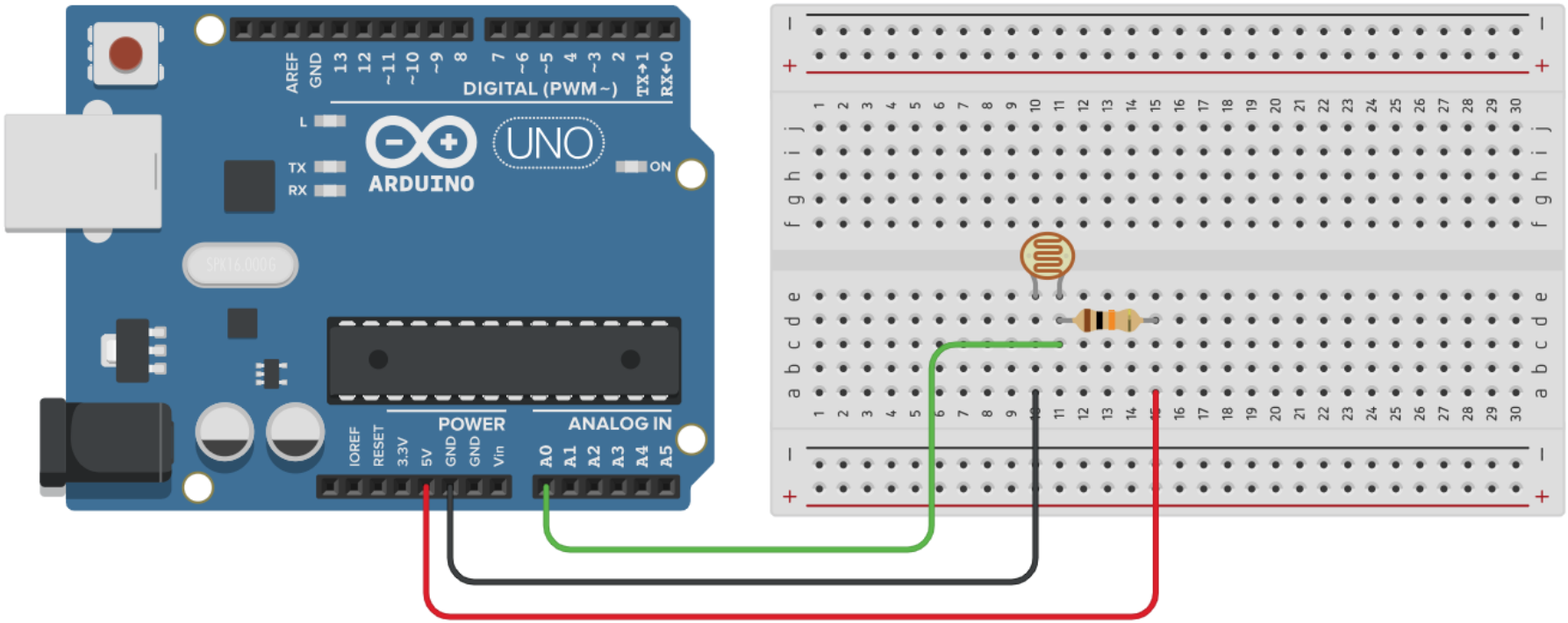
예제 5-2. 포텐쇼미터를 이용한 아날로그 입력

```
// 아날로그 입력 핀(A0~A5) 중에서 A0를 입력으로 사용
int sensor = A0;

void setup()
{
  // 시리얼 통신 초기화
  Serial.begin(9600);
  // 아날로그 센서 포트를 입력으로 사용
  pinMode(sensor, INPUT);
}

void loop()
{
  // 아날로그 센서를 통해서 값을 읽어들임
  int value = analogRead(sensor);
  // 시리얼 통신을 이용하여 PC로 전송
  Serial.println(value);
  // 100ms 대기
  delay(100);
}
```

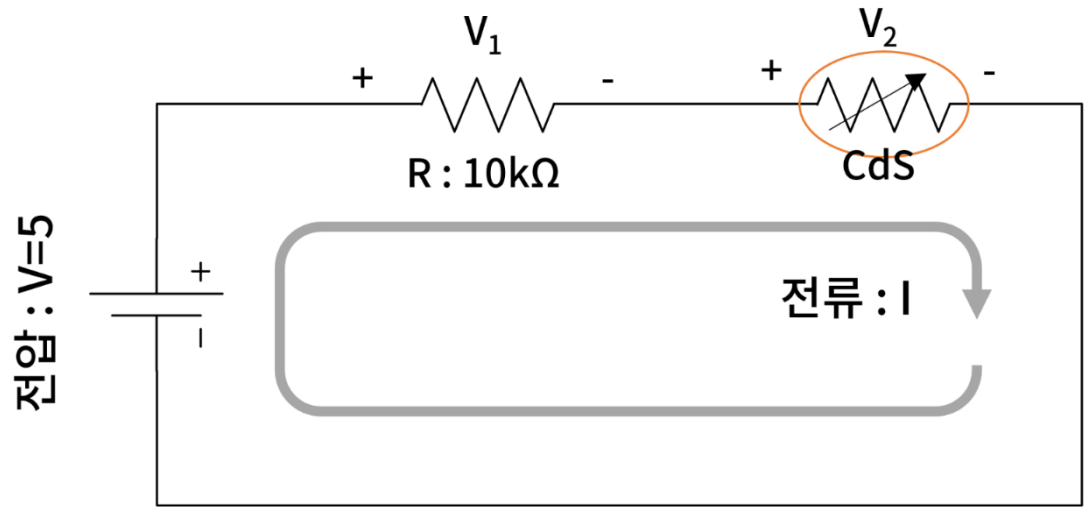
5.3 조도 센서

- 센서 : 조도, 온도, 유량, 음향, 가스 등의 물리적인 양을 측정하여 전기 성분으로 변환시켜주는 소자
- CdS (황화카드뮴) : 포토레지스터, 조도센서 → 주위의 밝기에 반응하여 소자의 저항값이 수십Ω~수백kΩ으로 변하는 소자

- 회로 구성 : 스케치는 동일하게 사용



- 회로도



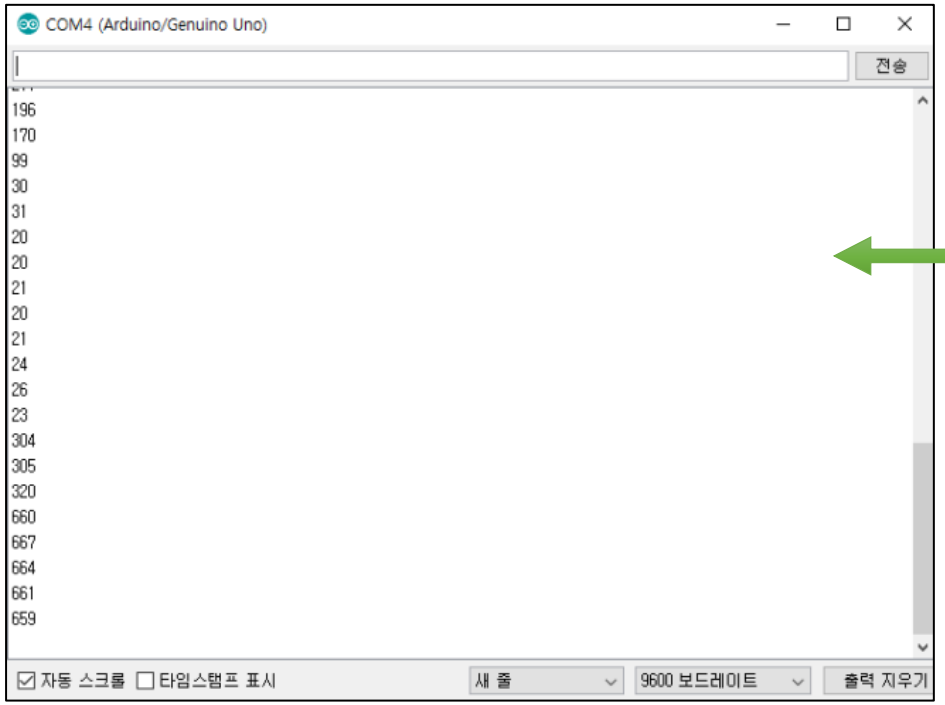
- KVL : $V_1 + V_2 = 5$, CdS의 저항 : $R_C \rightarrow V_1 : V_2 = R : R_C$
- $V_2 = \frac{R_C}{R} V_1 = \frac{R_C}{R}(5 - V_2) \Rightarrow V_2 = \frac{R_C}{R+R_C} \times 5\ [\mathrm{V}]$
- AO의 입력값 : $AO = \frac{R_C}{R+R_C} \times 1024$

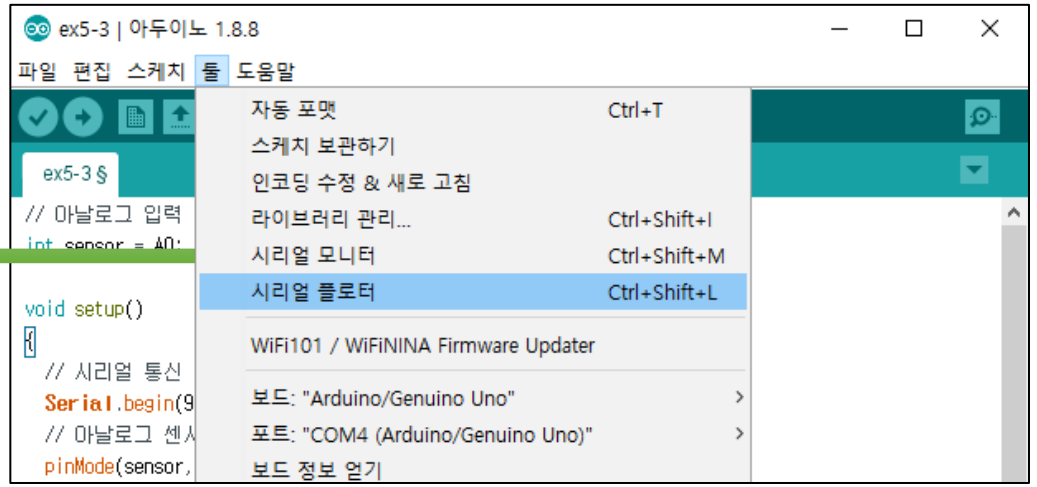
- 스케치

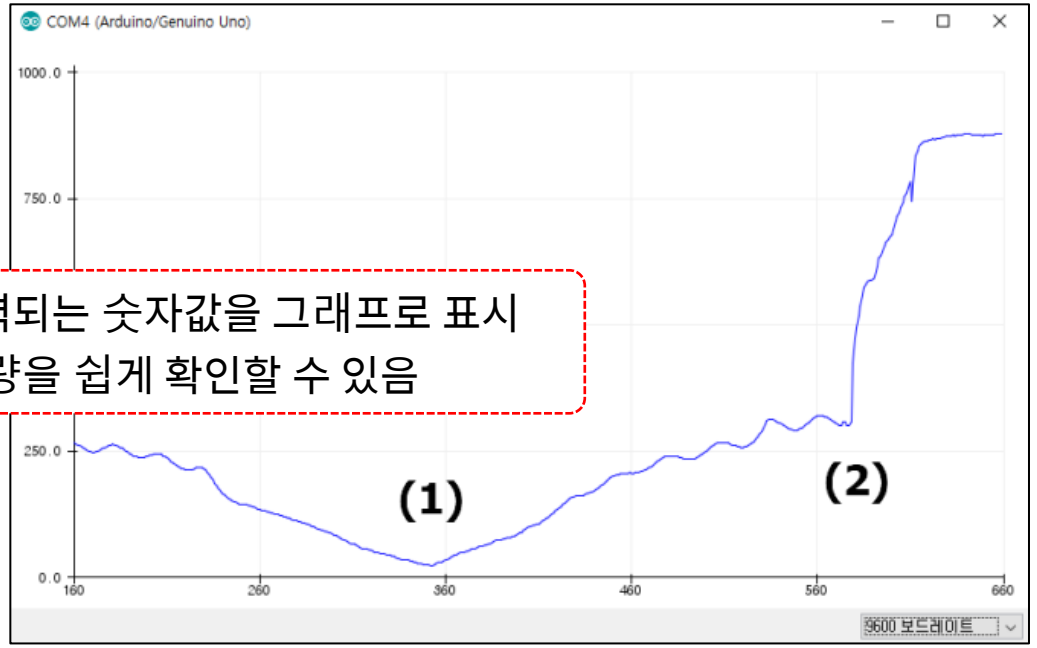
예제 5-3. 포텐쇼미터를 이용한 아날로그 입력 (예제 5-2와 동일)

```
// 아날로그 입력 핀(A0~A5) 중에서 A0를 입력으로 사용
int sensor = A0;

void setup()
{
  // 시리얼 통신 초기화
  Serial.begin(9600);
  // 아날로그 센서 포트를 입력으로 사용
  pinMode(sensor, INPUT);
}

void loop()
{
  // 아날로그 센서를 통해서 값을 읽어들임
  int value = analogRead(sensor);
  // 시리얼 통신을 이용하여 PC로 전송
  Serial.println(value);
  // 100ms 대기
  delay(100);
}
```

시리얼 모니터 & 시리얼 플로터







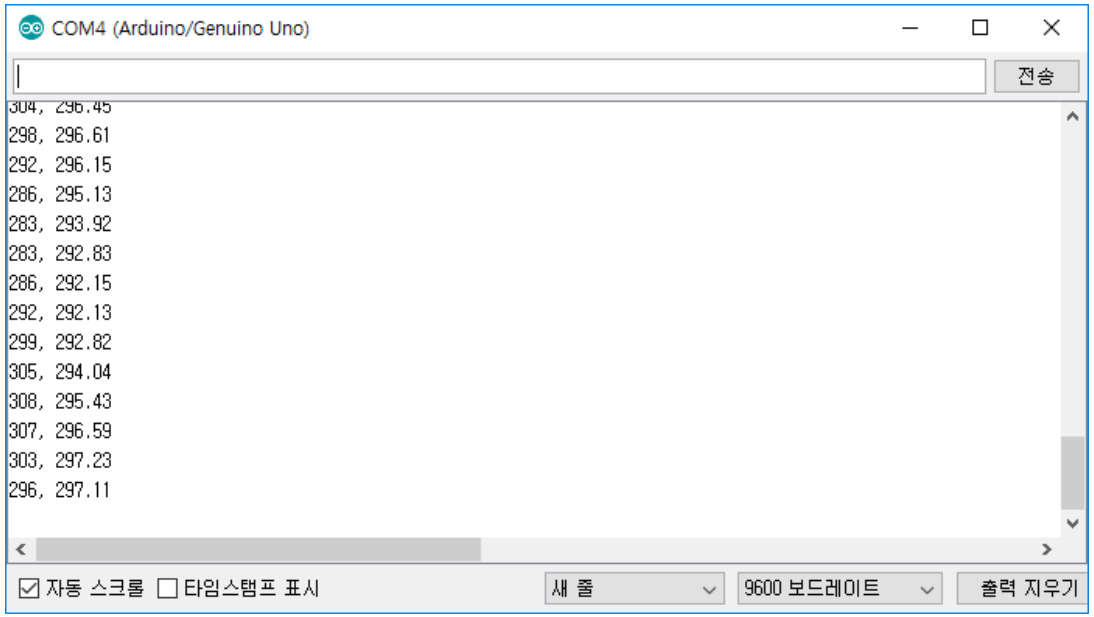
시리얼 플로터 : 시리얼 통신으로 입력되는 숫자값을 그래프로 표시
⇒ 시간에 따른 센서값의 변화량을 쉽게 확인할 수 있음

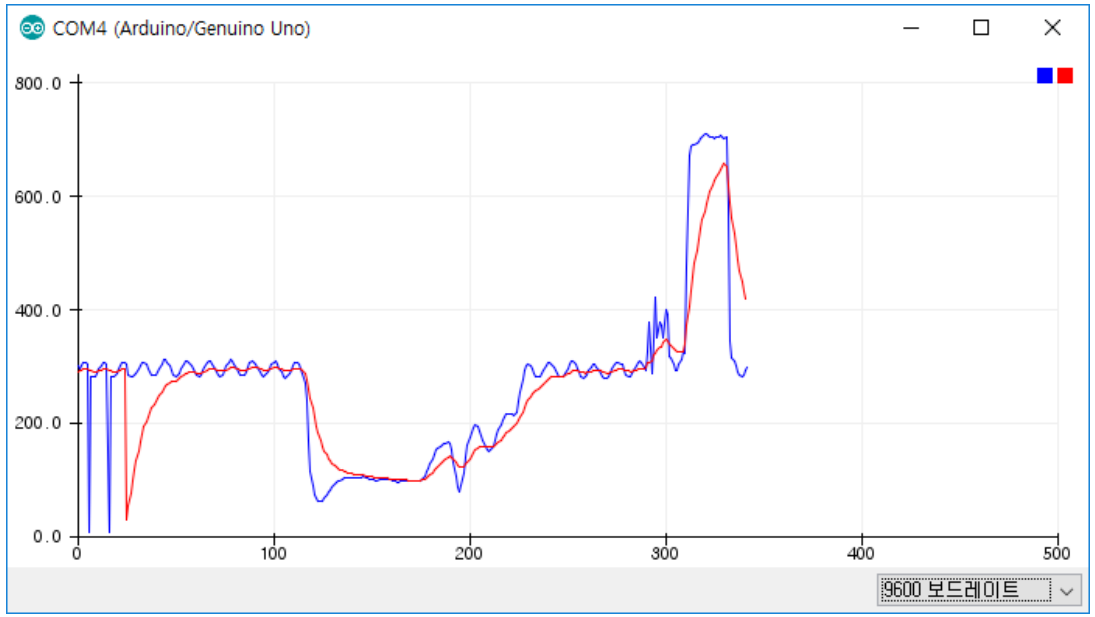
조도 센서의 응용

- 지수평균(exponential average) 구하기
  - 센서의 값이 외부 잡음 등의 영향으로 불안정할 때의 보상
  - 과거의 센서값과 현재의 센서값의 평균값 계산
  - 가중평균 : 각 값들의 가중치를 주어 평균 계산
    - 예] `value_ave = value_ave * (1.0 – 0.5) + value * 0.5;`
    - 현재값 가중치 : `0.5`

예제 5-4. 조도 센서값의 지수평균 구하기

```
int sensor = A0;       // 아날로그 입력 핀(A0~A5) 중에서 A0를 입력으로 사용
double value_ave = 0; // 센서값의 지수평균을 저장하는 변수

void setup()
{
  Serial.begin(9600); // 시리얼 통신 초기화
  pinMode(sensor, INPUT); // 아날로그 센서 포트를 입력으로 사용
}

void loop()
{
  int value = analogRead(sensor); // 아날로그 센서를 통해서 값을 읽어들임
  value_ave = value_ave * 0.9 + value * 0.1;   // 지수평균 구하기, 계수 : 0.1
  // 시리얼 통신을 이용하여 PC로 전송
  Serial.print(value); // 현재 측정값 전송, 줄바꿈 안함
  Serial.print(", ");  // 쉼표 및 공백
  Serial.println(value_ave); // 평균값 전송, 줄바꿈 하기
  // 100ms 대기
  delay(100);
}
```





시리얼통신으로 한 줄에 2개의 값을 전송하면
시리얼 플로터에서는 두가지 색으로 나타남

- 파란색 선 : 센서로부터 입력받은 값
- 빨간색 선 : 지수평균값

조도센서 응용

- 아두이노 보드에 조도 센서와 LED를 동시에 연결하고 밝기에 따라 LED를 점등시키는 스케치를 완성하라. 주위가 밝을 때에는 LED를 끄고 주위가 어두워지면 LED를 켜지도록 하라.
- 앞과 동일하게 조도 센서와 LED를 연결하고 밝기에 따라 LED를 제어하는 스케치를 완성하라. 단, 조도 센서의 밝기에 따라 PWM 출력을 이용하여 LED의 밝기를 제어하도록 한다.
- 조도 센서를 이용하여 현재 밝기를 나타내는 레벨미터를 만들어라. LED 5개를 사용하여 어두운 경우에는 모두 꺼진 상태로 있으면서, 밝기에 따라 켜지는 LED의 수를 조절하여 아주 밝은 상태에서는 모두 ON되도록 한다.