

아날로그 1

아날로그 데이터입출력



부산대학교 공과대학 전기컴퓨터공학부
정보컴퓨터공학전공



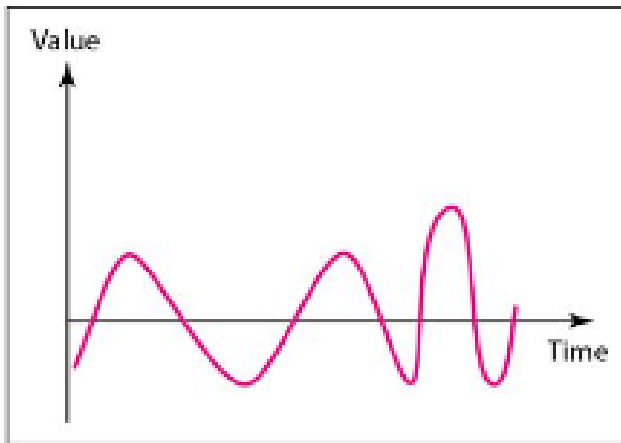
Analog vs. Digital

❖ Analog

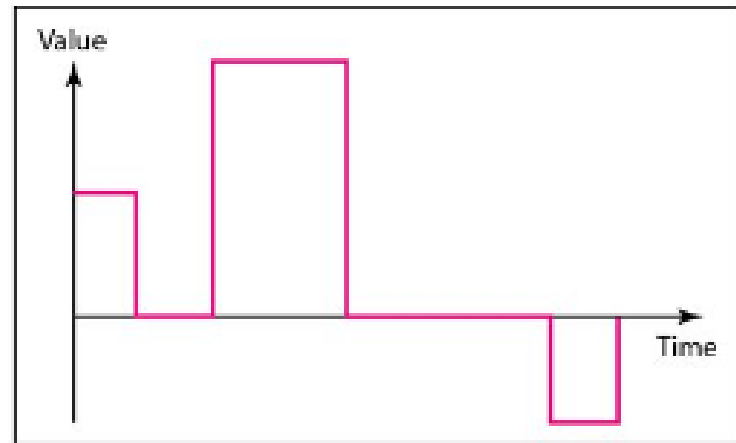
- 영어의 analogous(비슷한) 에서 유래함
- 수학적으로 정확하게 표현하면, 아날로그는 연속적 = continuous

❖ Digital

- 디지털의 어원은 손가락이라는 라틴어 digit에서 유래됨
- 디지털은 불연속적 = 이산적 = discrete



a. Analog signal



b. Digital signal

Analog vs. Digital

❖ 디지털의 장점

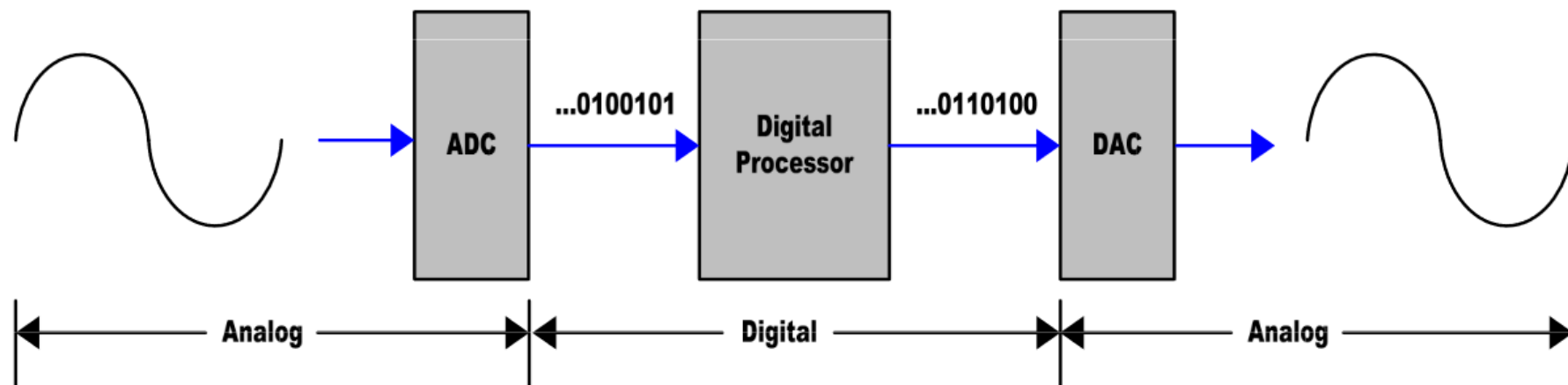
- 노이즈에 강하다
- 온도 특성이 좋다
- 데이터 처리가 편하다.

❖ 아날로그의 장점

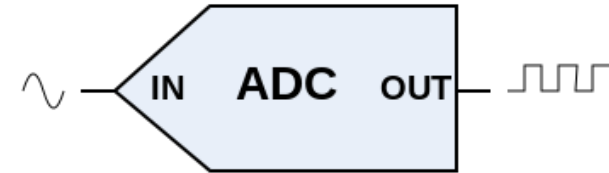
- 원본 데이터에 충실하다.

아날로그-디지털 변환 과정

- ❖ ADC (Analog to Digital Converter): 아날로그 데이터를 디지털 데이터로 변환
- ❖ DAC (Digital to Analog Converter): 디지털 데이터를 아날로그 데이터로 변환



ADC (Analog to Digital Converter)

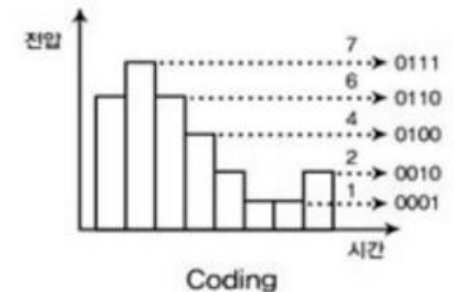
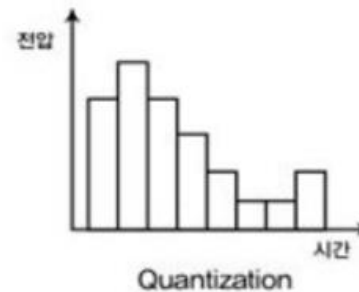
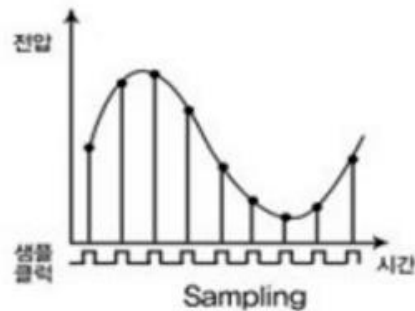
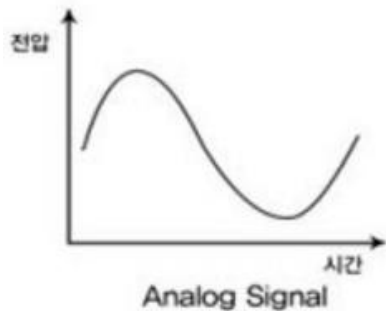


❖ ADC 1단계: 필터링

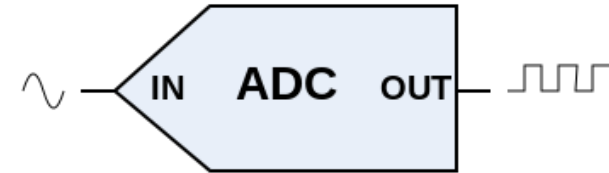
- 본래 신호를 정확히 "표본화(Sampling)"하기 위해 잡음 등의 신호를 차단하는 과정
- Ex) 음성 신호의 경우 원하는 대역폭에 대한 신호만 필터링한다.

❖ ADC 2단계: 표본화 (Sampling)

- 아날로그 파형을 디지털 형태로 변환하기 위해 표본을 취하는 것을 의미
- 표본화율(Sampling Rate): 1초 동안에 취한 표본수(디지털화하는 횟수)를 말하며, 단위는 주파수와 같은 Hz를 사용

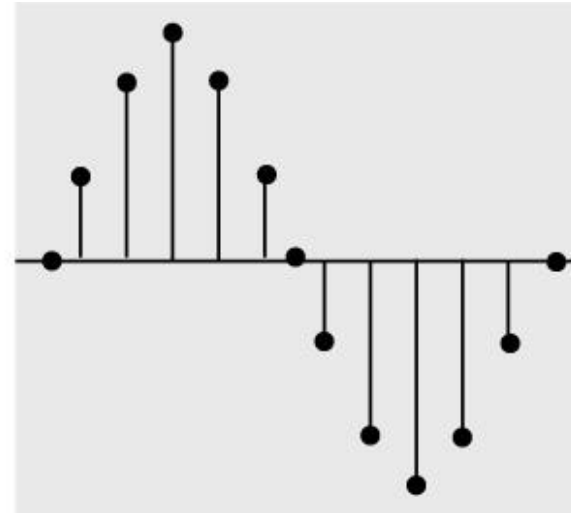
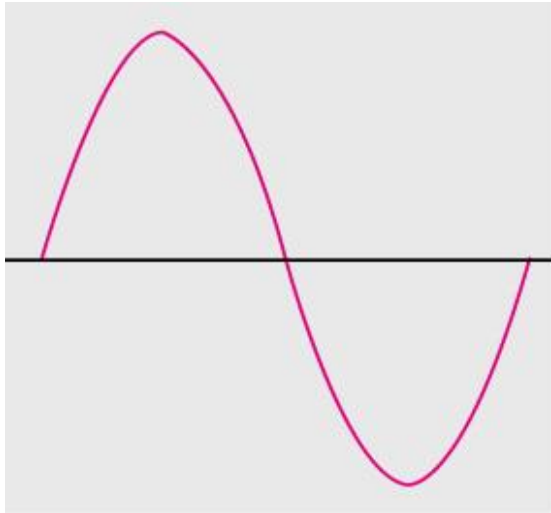


ADC (Analog to Digital Converter)



❖ ADC 2단계: 표본화 (Sampling)

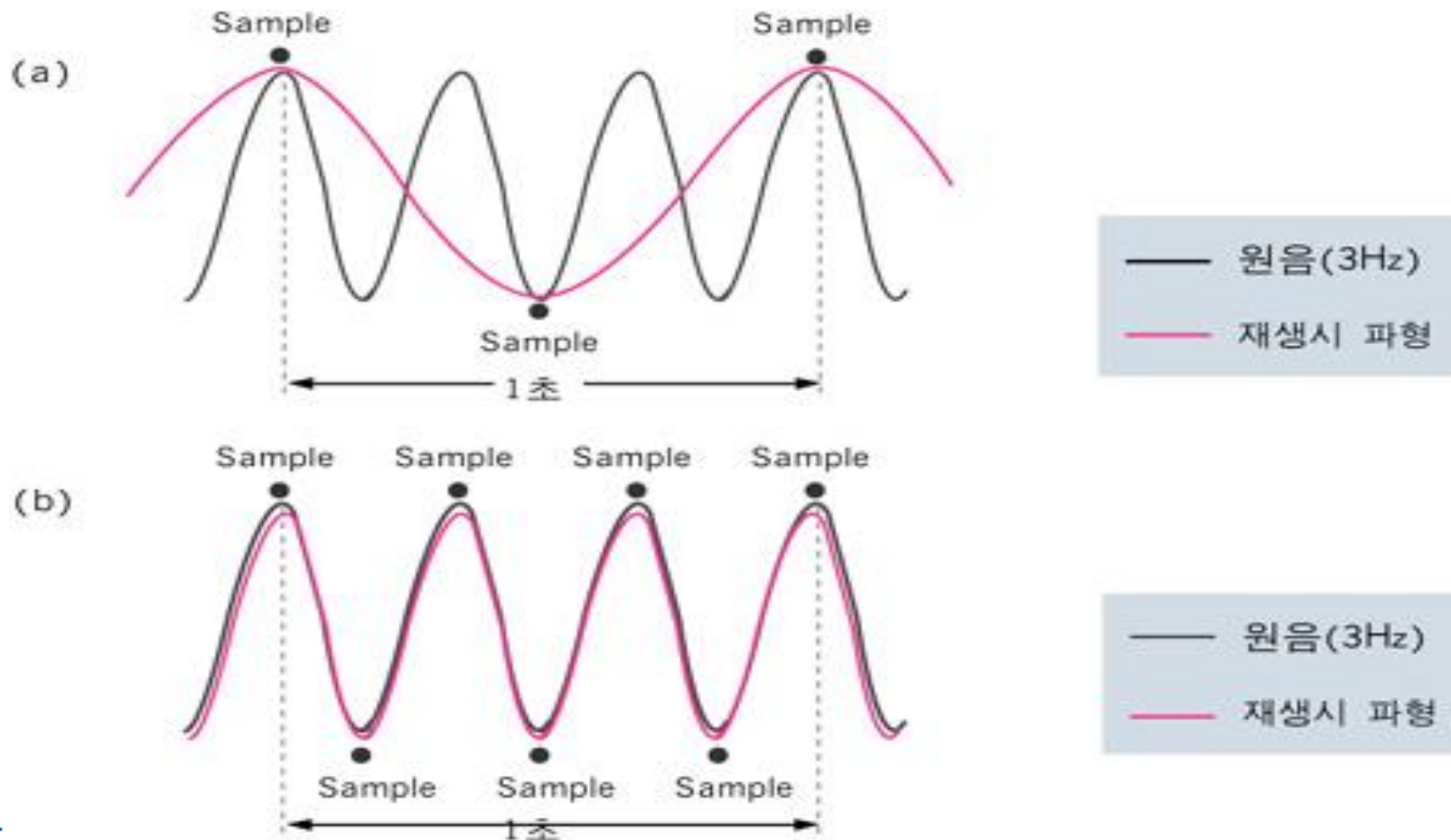
- 아날로그 파형을 디지털 형태로 변환하기 위해 표본을 취하는 것을 의미
- 표본화율(Sampling Rate): 1초 동안에 취한 표본수(디지털화하는 횟수)를 말하며, 단위는 주파수와 같은 Hz를 사용
 - 0.1초마다 샘플링을 수행하며, sampling rate는 10Hz



ADC (Analog to Digital Converter)

❖ ADC 2단계: 표본화 (Sampling)

- Sampling rate가 높아지면 본래 신호 특성을 더 잘 유지 할 수 있지만, 데이터양이 많아진다.



ADC (Analog to Digital Converter)



❖ ADC 2단계: 표본화 (Sampling)

- 나이퀴스트 정리(Nyquist theorem)
 - 최대 주파수 성분이 f_{max} 인 아날로그 신호는 적어도 $2f_{max}$ 이상의 sampling rate로 샘플링할 경우 원 신호를 완전히 복원할 수 있다.
- 나이퀴스트 주파수 (Nyquist Rate) = 최소 샘플링 주파수
 - 표본화 정리에 따라 원래의 정보를 재생할 수 있도록 신호가 갖는 최고 주파수(f_{max})의 두 배가 되는 표본화 주파수(f_s)를 말함
 - $f_s = 2f_{max}$

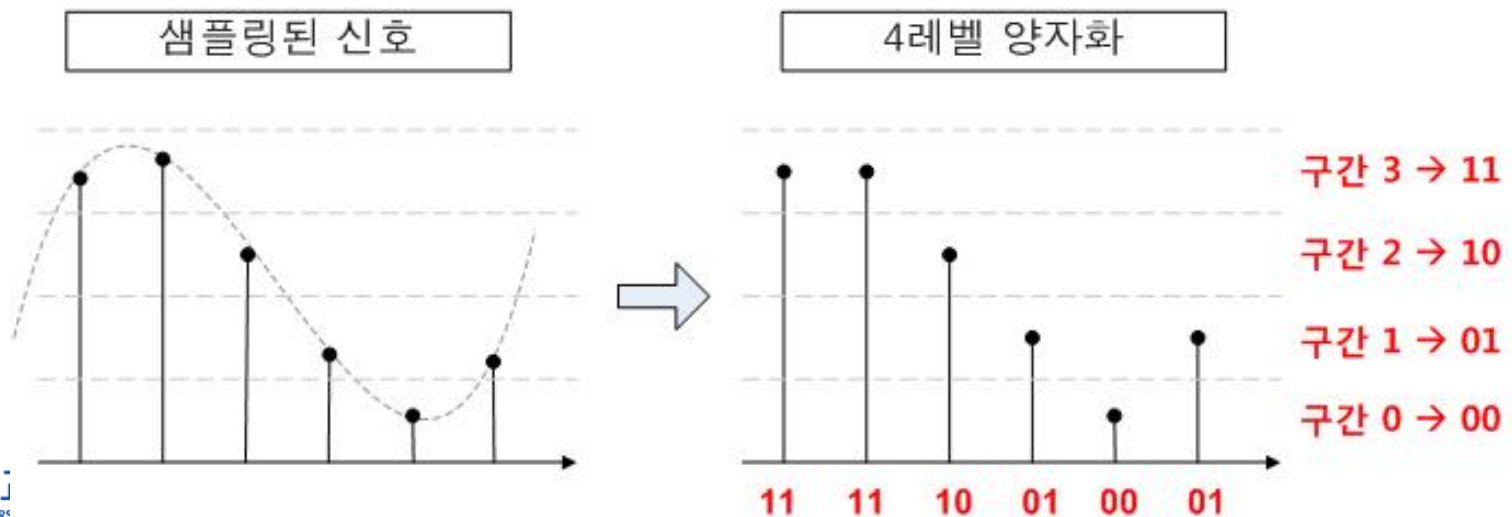
ADC (Analog to Digital Converter)

❖ ADC 3단계: 양자화 (Quantization)

- 표본화에서 얻어진 수치를 대표 값으로 n 개의 레벨로 분해하고, 샘플 값을 근사 시키는 과정
- 디지털 형태로 표현할 때 어느 정도의 정밀도를 가지고 표현할 것인지를 의미.
- 표본화된 각 점에서 값을 표현하기 위해 사용되는 비트 수

❖ ADC 4단계: 부호화 (Coding)

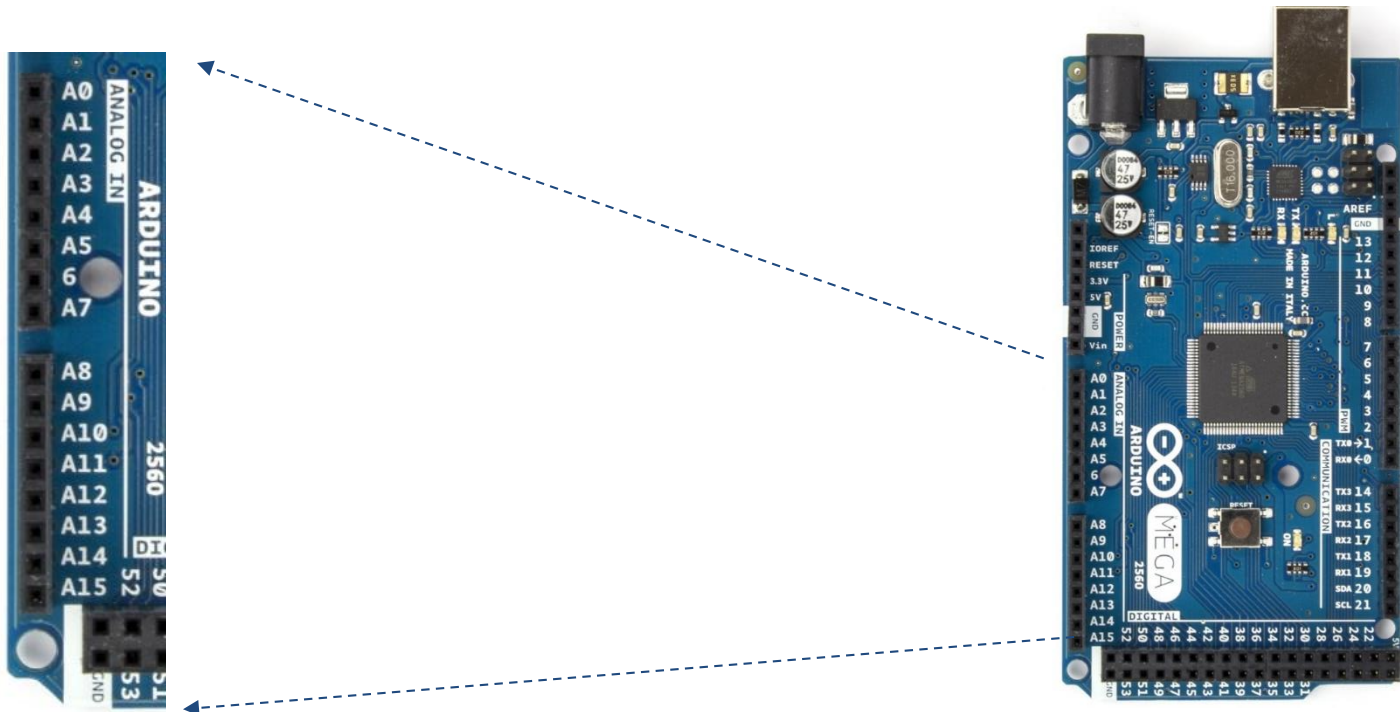
- 양자화된 값을 비트로 변환



아날로그 데이터 처리

❖ 아날로그 데이터 입력

- ATmega2560 마이크로컨트롤러에는 16채널의 10비트 해상도 아날로그-디지털 변환기 (ADC)가 포함되어 있음
- 각 채널에는 'A0'에서 'A15'까지의 상수가 할당
 - A0는 디지털 54번, A15는 디지털 69번과 동일함



아날로그 데이터 입력

❖ 16 채널의 ADC

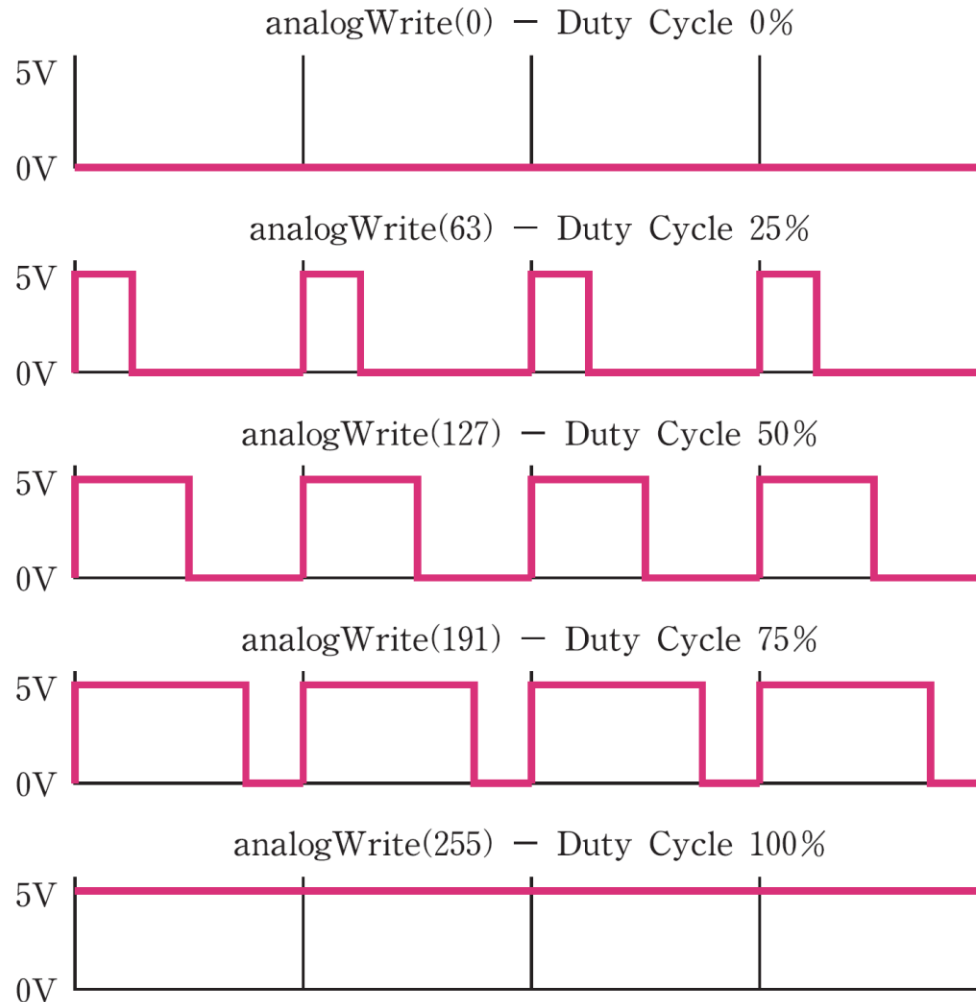
- 하나의 아날로그-디지털 변환기를 공유하므로 동시에 여러 채널 사용은 불가능

❖ 10비트 해상도

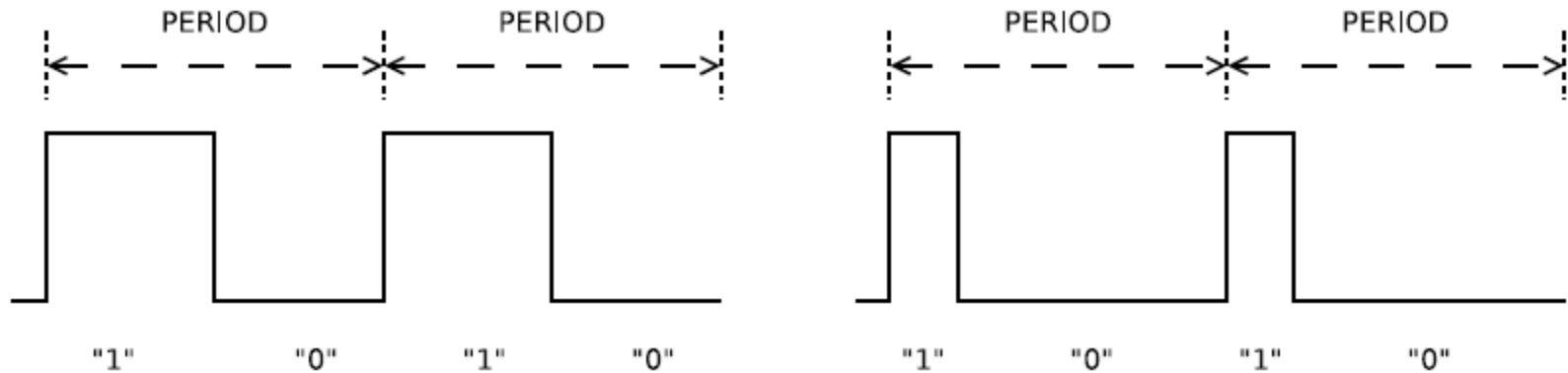
- 0에서 1023 사이의 정수값 반환
- 1023에 해당하는 기준 전압은 ATmega2560의 동작 전압인 5V가 일반적으로 사용됨
 - 기준 전압은 analogReference 함수로 변경 가능
- $5V / 1024 \cong 4.9mV$ 전압 차이 인식 가능

Pulse-Width Modulation (PWM)

❖ 아날로그 신호를 디지털화 하여 인코딩 하는 방법

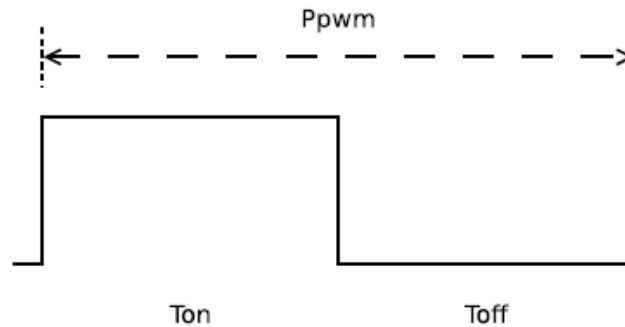


Pulse-Width Modulation (PWM)



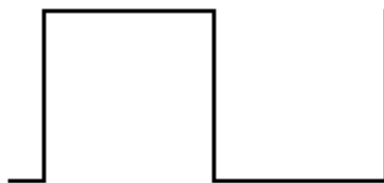
- ❖ **Pulse Width Modulation (PWM)** is a technique of modulation of a digital signal in order to obtain an analog value.
- ❖ It based on generating a square wave with a given frequency
- ❖ In the square wave, the “0” part and the “1” part have different duration
- ❖ The difference, in percentage, is called **duty cycle**

Pulse-Width Modulation (PWM)



- ❖ The **Duty Cycle** is defined as the percentage of T_{on} with respect to the total period P_{pwm} of the signal:

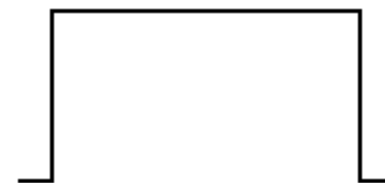
$$Duty\ Cycle = \frac{T_{on}}{P_{pwm}} \cdot 100$$



Duty Cycle = 50%



Duty Cycle = 10%



Duty Cycle = 90%

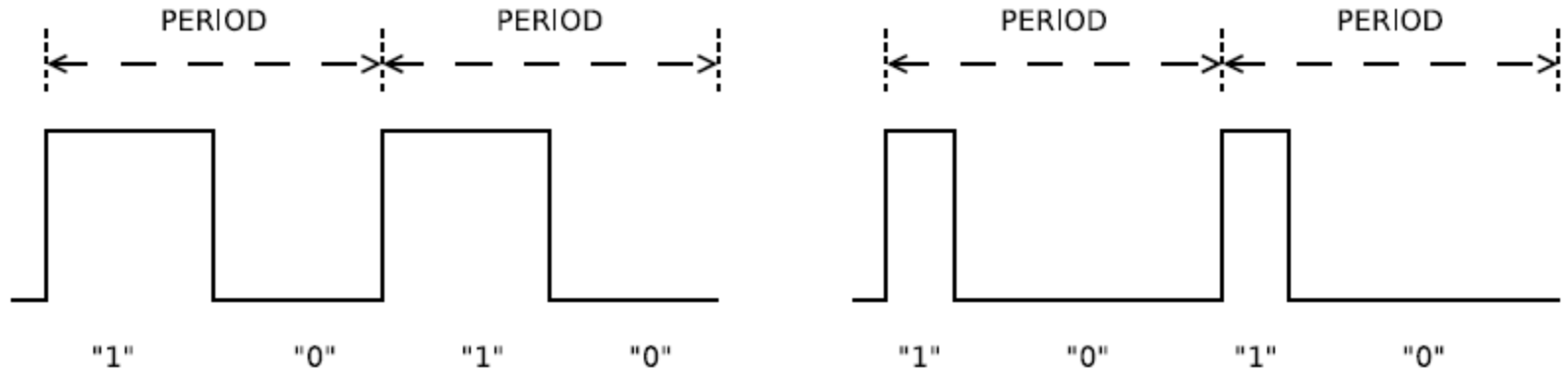


Duty Cycle = 0%



Duty Cycle = 100%

Pulse-Width Modulation (PWM)



❖ PWM has multiple utilisations:

1. To simply transfer an analog value over a digital line; devices receiving a PWM signal can interpret the “analog value” by measuring the duration of the “1” part with respect to the total frequency;
2. To modulate a typical on/off system; e.g. to change the intensity of a light generated by a lamp, a LED, etc.;
3. To drive power systems without affecting performances; e.g. To drive a DC motor.

아날로그 데이터 처리

❖ 아날로그 데이터 출력

- ATmega2560에서 아날로그 데이터 출력은 불가능
- 펄스 폭 변조(PWM:Pulse Width Modulation) 신호를 통해 아날로그 데이터 출력과 유사한 효과를 얻을 수 있음
 - PWM 신호는 디지털 신호의 일종임
 - PWM 신호 출력 함수가 `analogWrite`이므로 흔히 아날로그 데이터 출력으로 불림
- 15개의 핀으로 PWM 신호 출력 가능
 - 디지털 2번 ~ 13번
 - 디지털 44번 ~ 46번
- 0과 255 사이의 값을 출력

아날로그 데이터 입출력 함수

`int analogRead(uint8_t pin)`

- 매개변수
 - pin : 핀 번호
- 반환값 : 0에서 1023 사이의 정수값

`void analogReference(uint8_t type)`

- 매개변수
 - type : DEFAULT, INTERNAL, INTERNAL1V1, INTERNAL2V56, EXTERNAL 중 한 가지
- 반환값 : 없음

`void analogWrite(uint8_t pin, int value)`

- 매개변수
 - pin : 핀 번호
 - value : 듀티 사이클(duty cycle). 0(항상 OFF)에서 255(항상 ON) 사이의 값
- 반환값 : 없음

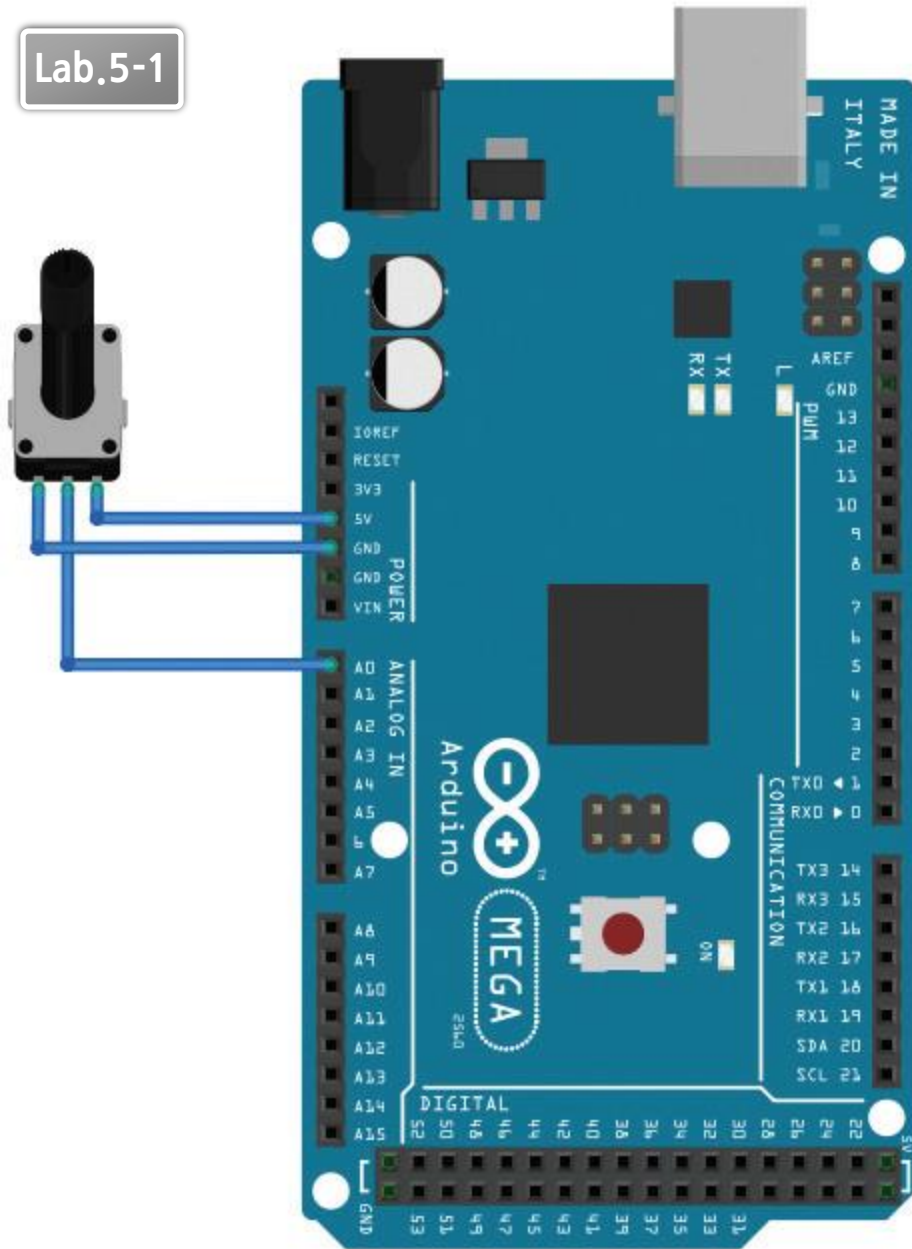
기준 전압 설정 옵션

| 옵션 | 설명 |
|--------------|---|
| DEFAULT | μ C의 동작 전압(5V)으로 설정 |
| INTERNAL | 내부 기준 전압(1.1V)으로 설정 ATmega2560에서는 사용 불가능 |
| INTERNAL1V1 | 내부 1.1V로 설정 ATmega2560에서만 사용 가능 |
| INTERNAL2V56 | 내부 2.56V로 설정 ATmega2560에서만 사용 가능 |
| EXTERNAL | AREF 핀에 인가된 0~5V 사이 전압으로 설정 |

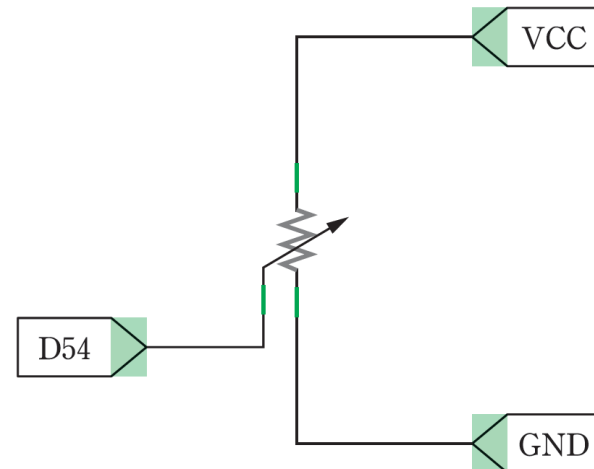
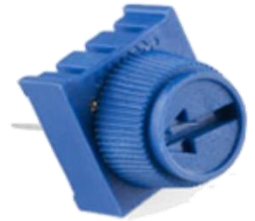
가변저항을 이용한 아날로그 입력처리

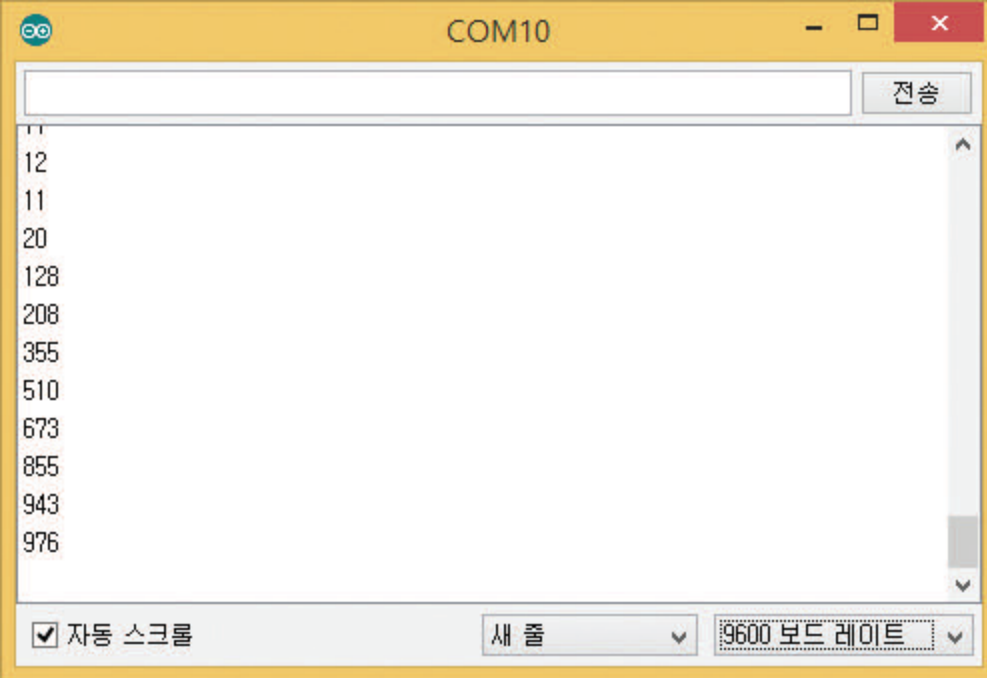
❖ 실험 순서

- ① 가변저항 연결 및 저항값 출력 (Sketch 7-1)
- ② LED4개 연결 및 가변 출력(Sketch 7-2)
- ③ 실행 결과 확인



값을 임의로 바꿀 수 있는 저항기

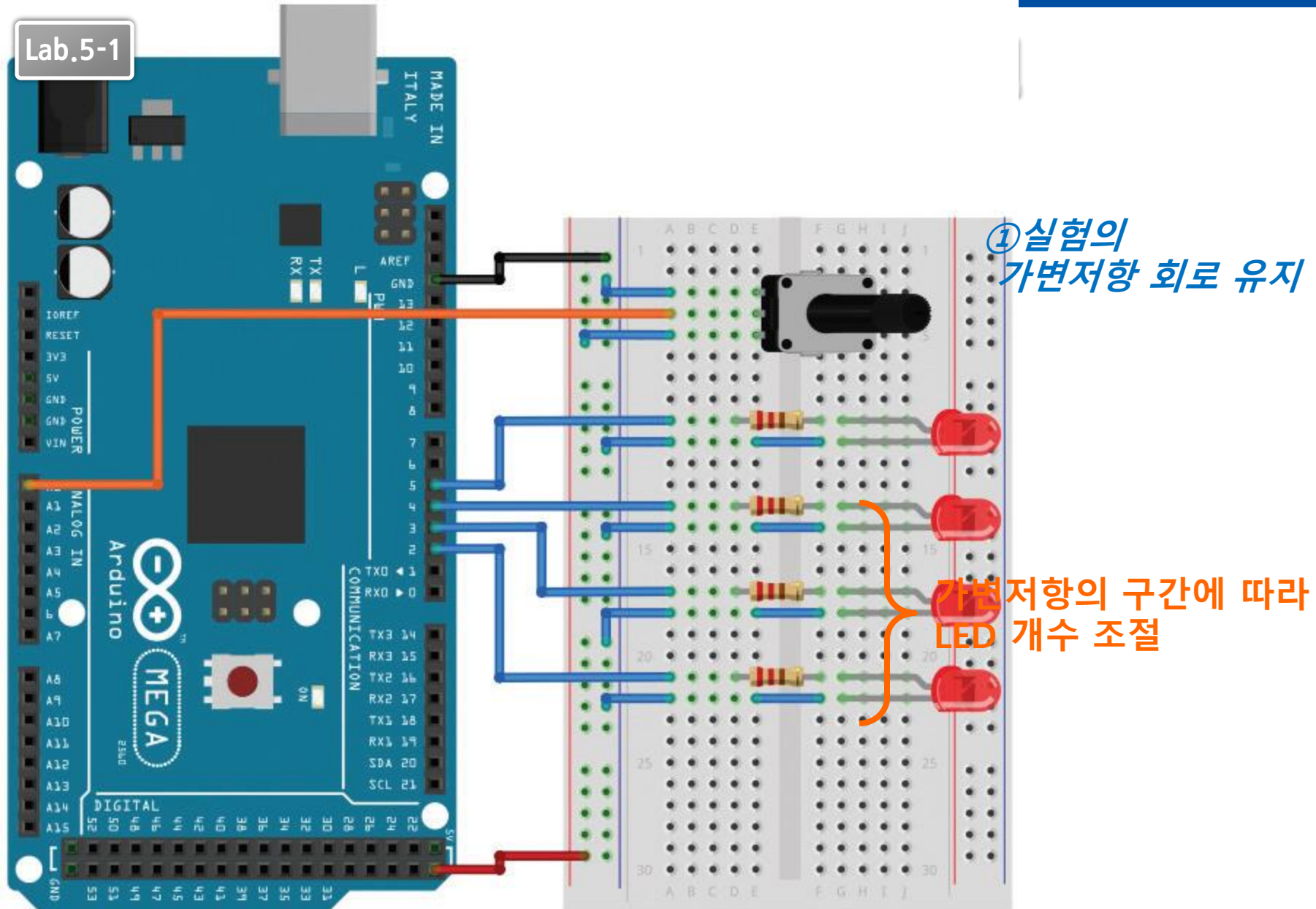




Sketch 7-1

결

// 가변저항 값을 읽어 출력



② 가변저항으로 LED 출력 개수 조절

Sketch 7-2

```
int vResistor = A0; // A0 핀에 가변저항 연결
int pins_LED[] = {2, 3, 4, 5}; // LED 연결 핀
```

```
void setup() {
  Serial.begin(9600); // 시리얼 통신 초기화
  pinMode(vResistor, INPUT);
  for (int i = 0; i < 4; i++) {
    pinMode(pins_LED[i], OUTPUT);
    digitalWrite(pins_LED[i], LOW);
  }
}
```

비트 이동에 의해 10비트 결과값을
2비트의 1~4 사이 값으로 변환

```
void loop() {
  int adc = analogRead(vResistor); // 가변저항 값 읽기
  int count_led = (adc >> 8) + 1; // LED 개수 결정
  for (int i = 0; i < 4; i++) { // LED 점멸
    if (i < count_led)
      digitalWrite(pins_LED[i], HIGH);
    else
      digitalWrite(pins_LED[i], LOW);
  }
  // ADC 값과 LED 개수 출력
  Serial.println(String("ADC : ") + adc + ", LED count : " + count_led);
  delay(1000); // 1초(1000ms) 대기
}
```

③ 결과확인

- ❖ 작성한 Sketch 편집창을 보여라
- ❖ Sketch 7-2의 실행을 보여라
- ❖ 조교의 물음에 답하고 채점 결과를 PLMS LAB 5에 직접 입력하라

PWM 아날로그 출력

❖ 실험 순서

① RGB LED 밝기 제어 (Sketch 7-3)

LED4개 연결 회로는 유지 할것

② 가변저항으로 LED 밝기 제어(1) (Sketch 7-4)

③ 가변저항으로 LED 밝기 제어(2) (Sketch 7-5)

④ 실행 결과 확인

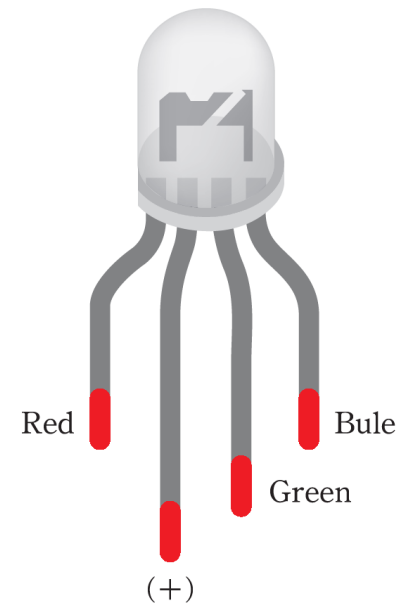
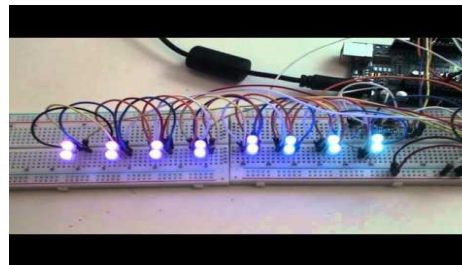
① RGB LED

❖ RGB LED

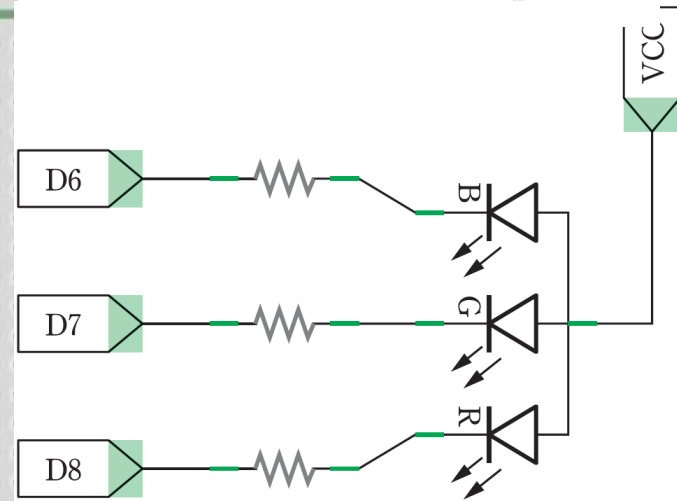
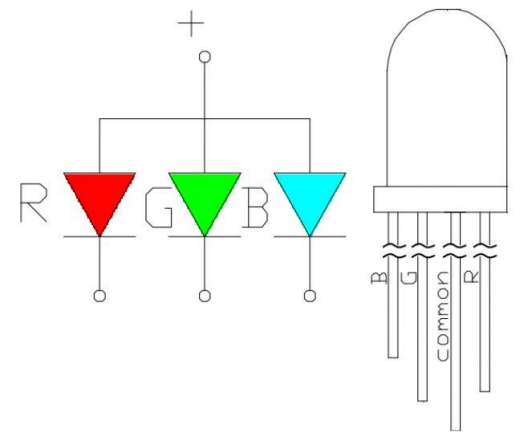
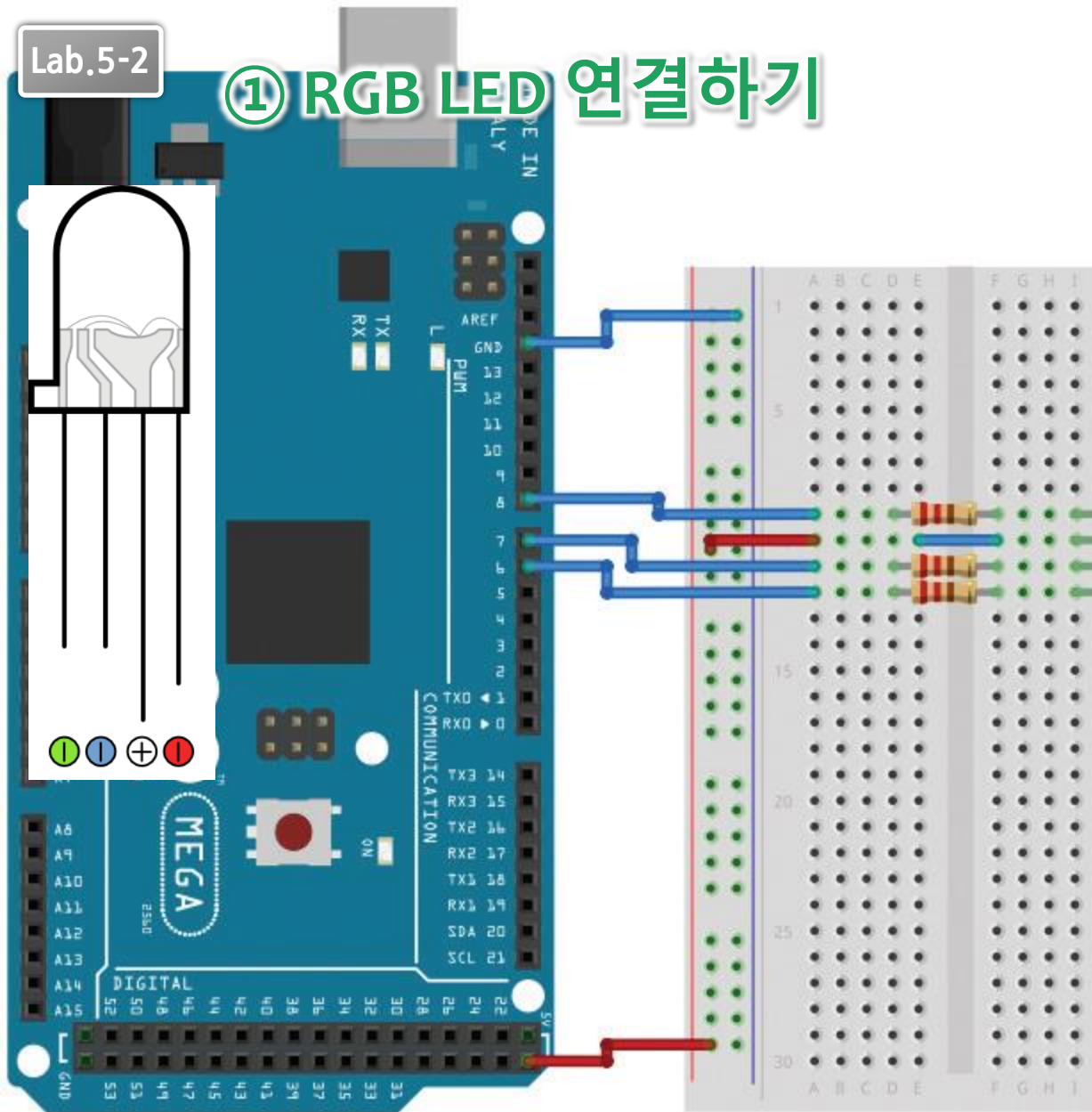
- 공통 핀과 3개의 R, G, B 제어핀으로 구성
- 공통 양극 방식 (Common anode)의 경우 제어핀에 GND를 출력하면 켜짐 (cf. Common cathode)

❖ 밝기 제어: 공통 양극 방식

- 최대 밝기 : `analogWrite(0)`
- 최소 밝기 : `analogWrite(255)`



① RGB LED 연결하기



```
// RGB LED 연결 핀
```

```
int RGB_LED[] = {6, 7, 8};
```

```
void setup() {
```

```
    for (int i = 0; i < 3; i++) {  
        pinMode(RGB_LED[i], OUTPUT);  
    }
```

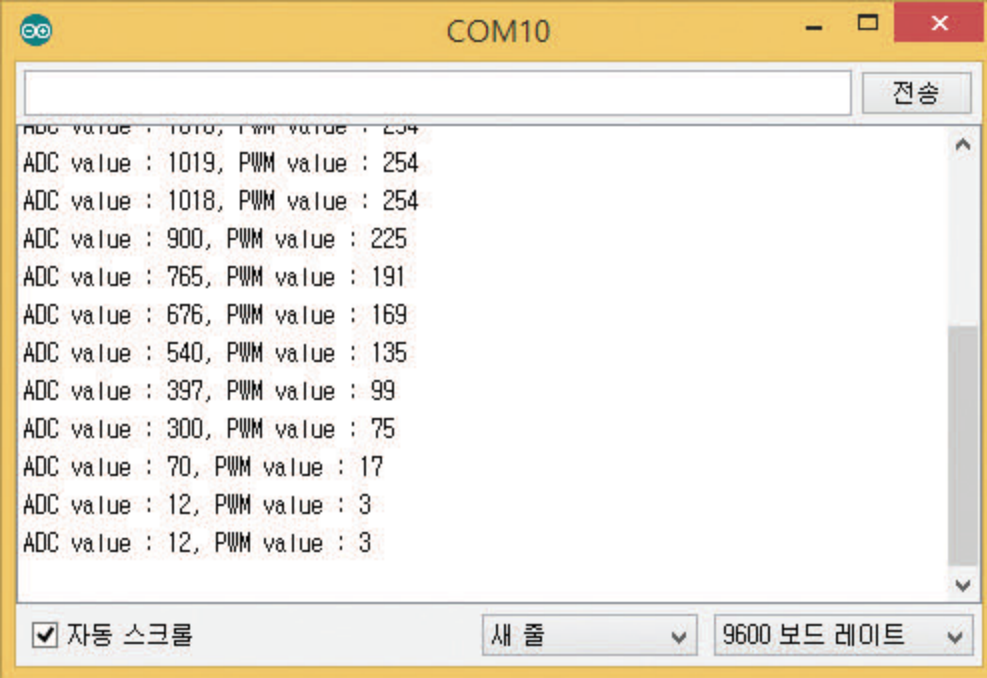
```
}
```

```
void loop() {
```

```
    // Blue 색상 조절. Green, Red는 끄  
    digitalWrite(RGB_LED[1], HIGH);  
    digitalWrite(RGB_LED[2], HIGH);  
    for (int i = 255; i >= 0; i--) {  
        analogWrite(RGB_LED[0], i);  
        delay(10);  
    }
```

```
// Green 색상 조절. Blue, Red는 끄  
digitalWrite(RGB_LED[0], HIGH);  
digitalWrite(RGB_LED[2], HIGH);  
for (int i = 255; i >= 0; i--) {  
    analogWrite(RGB_LED[1], i);  
    delay(10);  
}
```

```
// Red 색상 조절. Green, Blue는 끄  
digitalWrite(RGB_LED[0], HIGH);  
digitalWrite(RGB_LED[1], HIGH);  
for (int i = 255; i >= 0; i--) {  
    analogWrite(RGB_LED[2], i);  
    delay(10);  
}
```

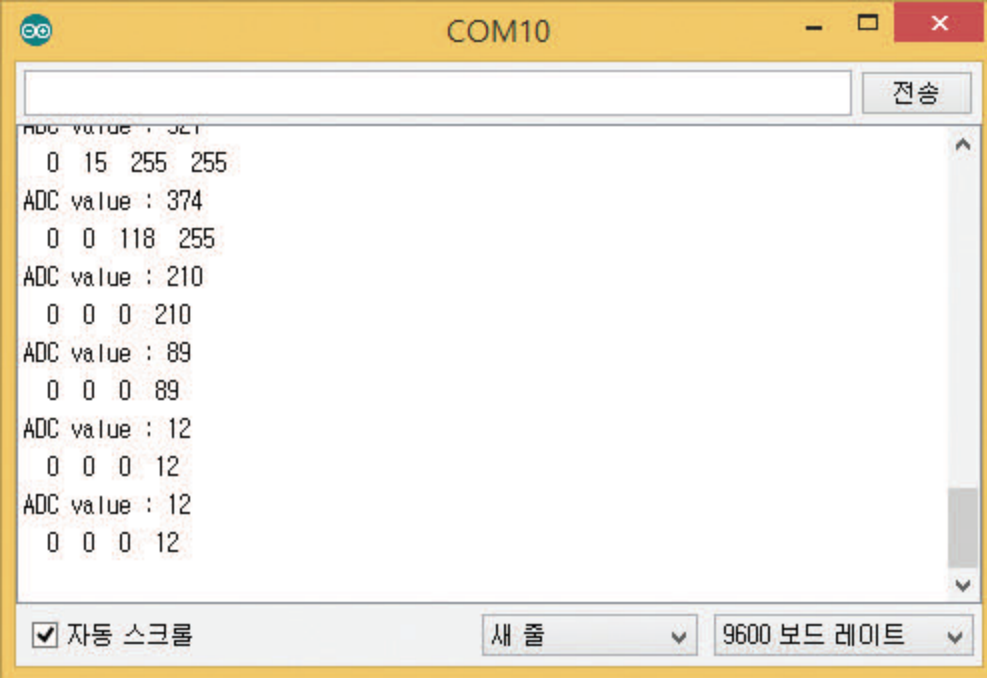


제어(1)

Sketch 7-4

르 설정

```
int ADC_value = analogRead(A0); // ADC 값  
int PWM_value = ADC_value >> 2; // PWM 값  
Serial.print(String("ADC value : ") + ADC_value);  
Serial.println(String(", PWM value : ") + PWM_value);  
for (int i = 0; i < 4; i++) { // LED 밝기 조절  
    analogWrite(pins_LED[i], PWM_value);  
}  
delay(1000);  
}
```

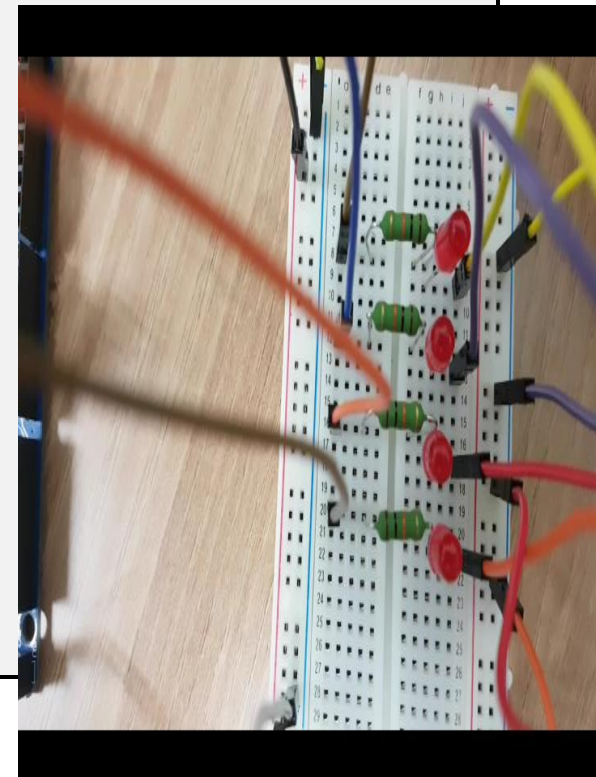


어(2)

Sketch 7-5

르 설정

```
int ADC_value = analogRead(A0); // ADC 값
int PWM_value[4] = {0, };
Serial.println(String("ADC value : ") + ADC_value);
for (int i = 3; i >= 0; i--) { // 4개 LED의 밝기로 나눔
    if (ADC_value >= 256 * i) {
        PWM_value[i] = ADC_value - 256 * i;
        ADC_value -= (PWM_value[i] + 1);
    }
    analogWrite(pins_LED[i], PWM_value[i]); // LED 밝기 조절
    Serial.print(" ");
    Serial.print(PWM_value[i]);
}
Serial.println();
delay(500);
}
```



④ 결과확인

- ❖ 작성한 Sketch 편집창을 보여라
- ❖ Sketch 7-3과 7-5의 실행을 보여라
- ❖ 조교의 물음에 답하고 채점 결과를 PLMS LAB 5에 직접 입력하라

❖ 아날로그 데이터 입력

- 10비트 해상도의 ADC를 통해 0~1023 사이의 양자화된 디지털 값 입력
- 16 채널
- 1023에 해당하는 기준 전압을 설정하여야 함

❖ 아날로그 데이터 출력

- ATmega2560에는 DAC가 없으므로 아날로그 데이터 출력은 불가능
- 디지털 신호의 일종인 펄스 폭 변조 신호를 통해 아날로그 데이터와 유사한 효과를 얻을 수 있음
- LED 밝기 제어, 모터 속도 제어 등에 PWM 신호가 사용

❖ 아래 동작을 하는 Sketch를 작성하라

- LED 4개가 차례로 점멸됨
 - 각각 디지털 2,3,4,5번 핀에 연결
- PWM을 이용하여, 0~100% 변화하면서 서서히 밝게함
 - 디지털 2번 핀에 연결된 LED는 밝기 0%에서 시작해 255까지 증가 후 다시 0으로 초기화

| 디지털 핀 | 시작 밝기 | 초기 analogWrite value |
|-------|-------|----------------------|
| 2 | 0% | 0 |
| 3 | 25% | 63 |
| 4 | 50% | 127 |
| 5 | 75% | 191 |

- 동작 동영상을 참고할 것

❖ 제출 방법

1. PLMS의 HW 5에 작성한 Sketch 코드를 입력하라
2. 또한 **실행결과를 촬영한 동영상 링크를 포함** 하여 제출하라

