



# 아두이노를 통한 서보 모터 제어

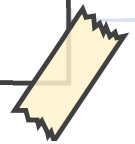


2022 Capstone Design

16100170 이동건



## 목차

1. Servo 와 Servo 모터
  2. 서보 모터의 제어
    - 1) 수동으로 구현
    - 2) Fast PWM 기능 사용
    - 3) 라이브러리 사용
- 

## 서보

- 서보 매커니즘(Servo Mechanism)
- 물체의 위치, 방위, 자세 등을 제어량으로 하고 목표치의 임의 변화에 추종하도록 구성된 제어계

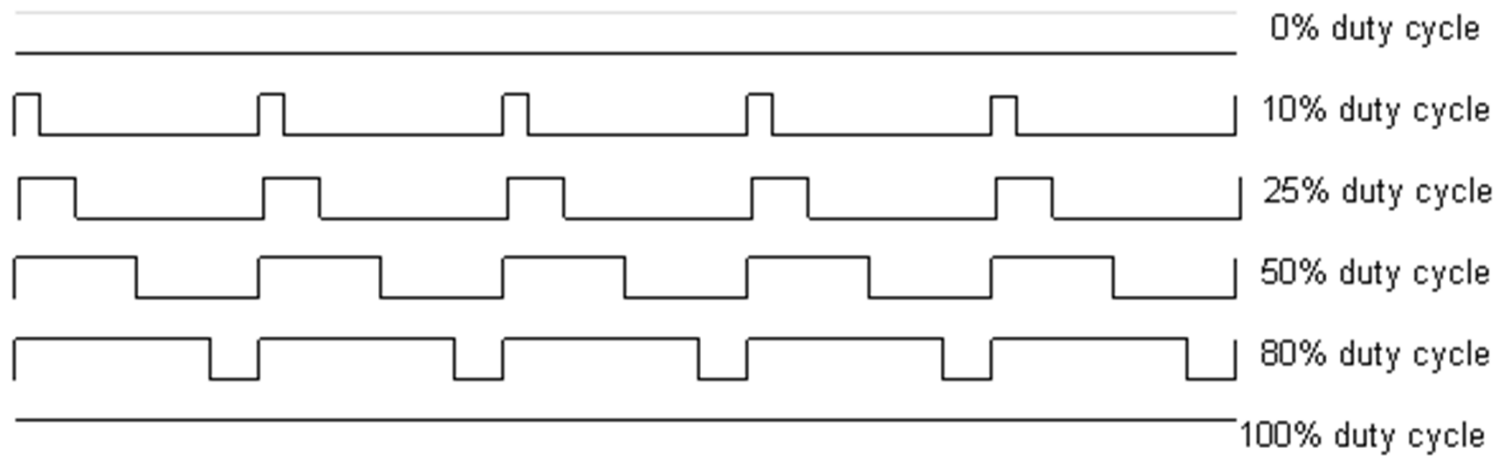
## 서보 모터

- 일반 모터와 다르게 빈번하게 변화하는 위치나 속도의 명령치에 대하여 신속하고 정확하게 추종할 수 있도록 설계된 모터
- 제어계측 회로를 통해 특정 위치, 속도를 측정하여 피드백을 통해 정확한 제어가 가능



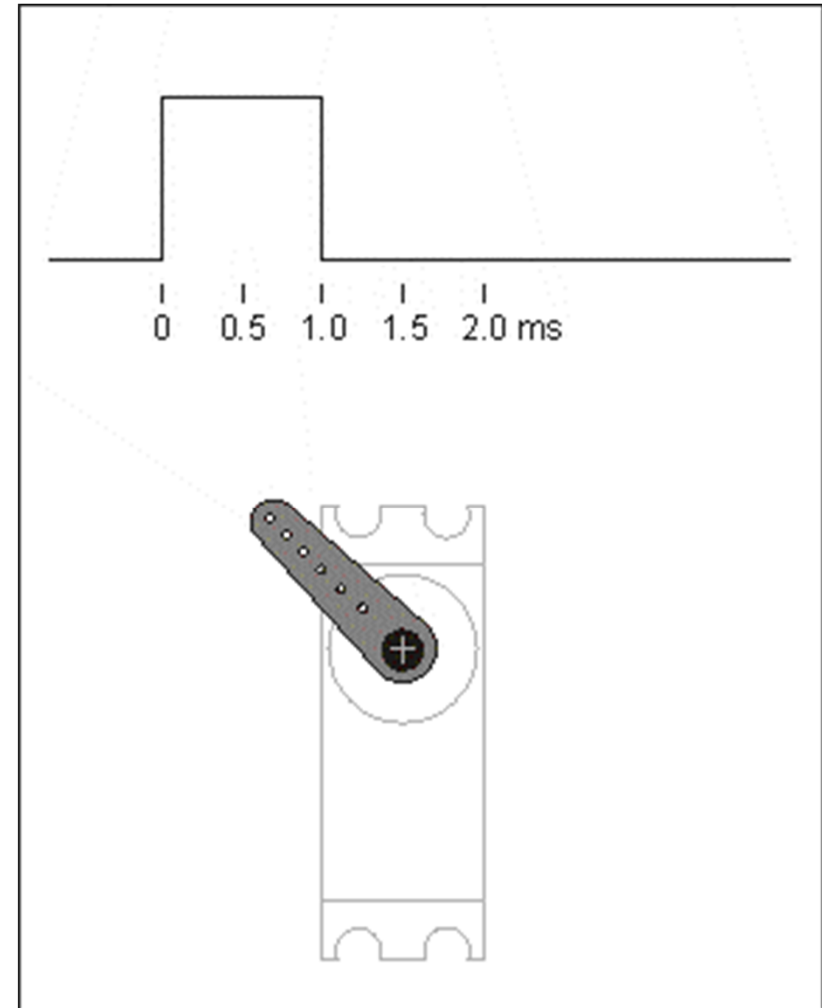
# PWM

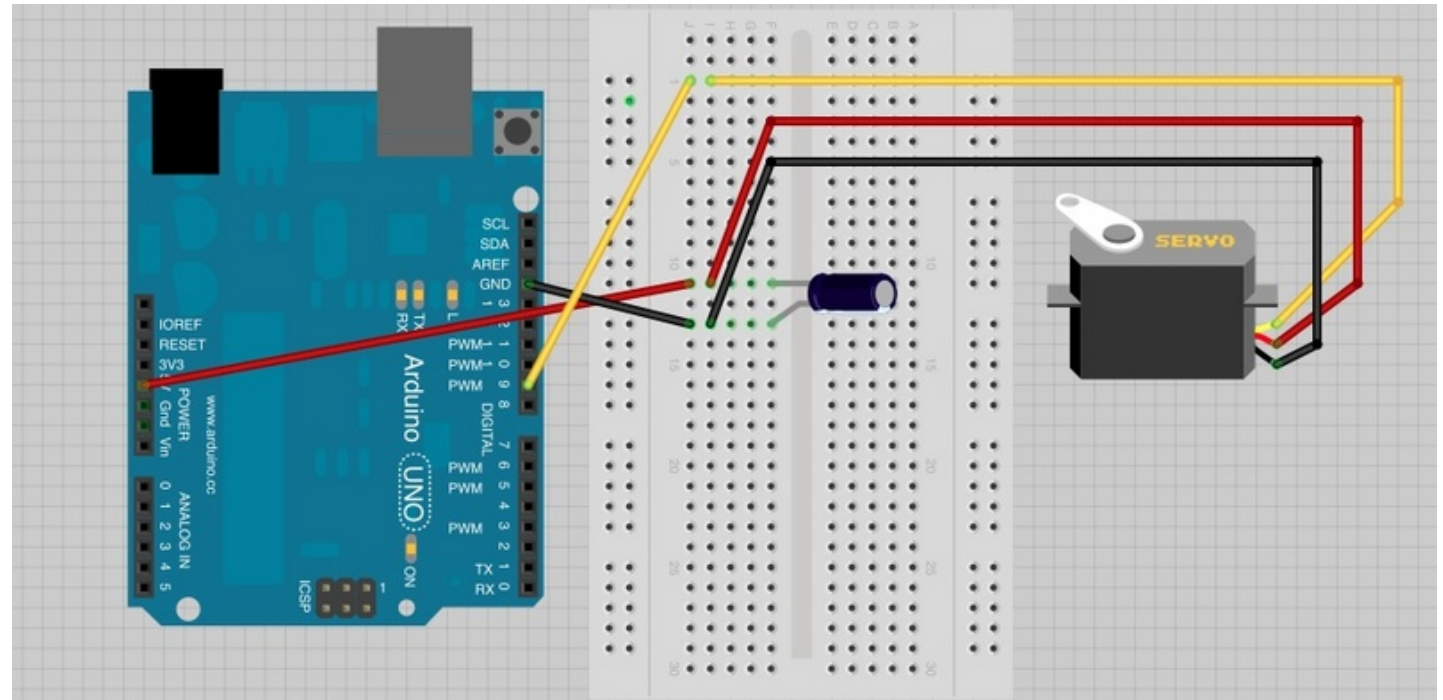
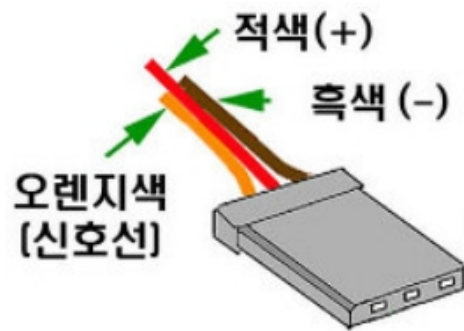
- PWM(Pulse Width Modulation, 펄스 폭 변조)
- 출력되는 전압값을 일정한 비율(duty) 동안 High를 유지하고, 나머지는 low로 출력
- PWM을 통해 0V 와 5V 사이의 아날로그 값을 모사할 수도 있으며 제어 및 통신에 사용



## 서보 모터 제어

- PWM 신호를 입력해서 서보모터 제어
- 입력된 신호의 지속시간(Pulse Width)으로 위치를 제어
- Pulse Width 와 대응 각도는 각 서보 모터의 데이터 시트를 통해 확인

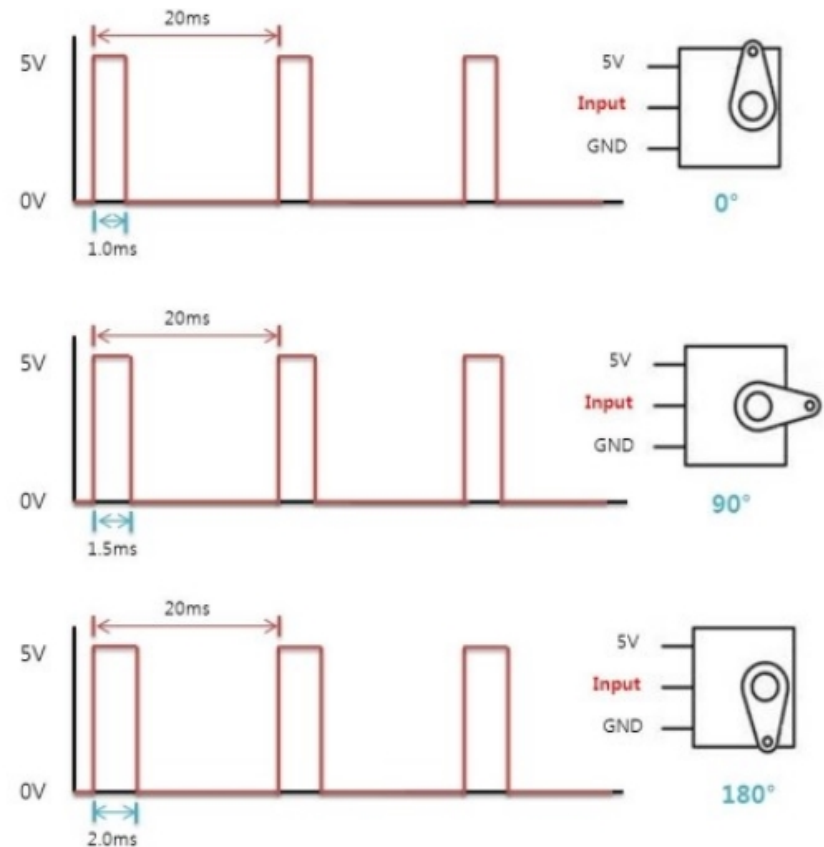
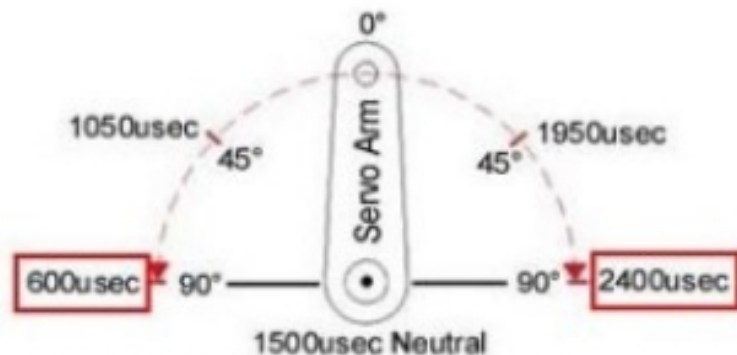




- 일반적으로 5V 전원선, GND 선, 신호선으로 구성됨
- 컴퓨터와 USB를 통해 연결된 아두이노 전원에도 동작 가능
- 콘덴서는 생략 가능(전원 보충, 노이즈 제거 등 효과)

# 1. PWM 을 수동으로 구현

- 아두이노 언어의 DigitalWrite, Delay를 사용
- 보통 제어 주파수는 주기 당 20ms (50Hz)
- 20ms의 주기 안에서 High 신호의 폭을 조절하는 함수를 만들어 제어
- 데이터 시트를 통해 Pulse Width를 확인



# 1. PWM 을 수동으로 구현

- PWM 신호를 내보낼 핀과 각도 입력
- 데이터 시트를 따라 주파수와 max, min 펄스 폭을 입력
- High와 Low의 입력 시간을 계산하여 해당 pin 에 입력

```
int servoPin = 9;
int freq = 20000;    // 20 milliseconds (50Hz)
int minPulse = 600;  // 600 microseconds
int maxPulse = 2400; // 2400 microseconds

void Motor_operation(int pin, int degree) {

    float hTime = 0;
    float lTime = 0;

    hTime = (minPulse + ((maxPulse - minPulse) / 180.0 * degree));
    lTime = freq - hTime;

    digitalWrite(pin, HIGH);
    delayMicroseconds(hTime);
    digitalWrite(pin, LOW);
    delayMicroseconds(lTime);

}
```



## 1. PWM 을 수동으로 구현



- 서보 라이브러리 없이 함수만으로 서보 모터를 제어 가능
- 지정한 각도를 유지하기 위해서는 반복해서 함수를 호출해야 함
- 반복해서 함수를 호출해야 하므로 불안정

## 2. ATmega Fast PWM 기능 사용

- ATmega328P의 Timer register를 직접 조작
- 아트가 칩의 데이터 시트 참조

Timer	Timer output	Arduino output	Chip pin	Pin name
0	OC0A	6	12	PD6
	OC0B	5	11	PD5
1	OC1A	9	15	PB1
	OC1B	10	16	PB2
2	OC2A	11	17	PB3
	OC2B	3	5	PD3

## 2. ATmega Fast PWM 기능 사용

### Timer0 (8bit)

- delay(), millis(), micro()등의 타이머 함수에 사용
- 레지스터 변경 시 타이머 기능에 오류가 발생할 수 있음
- 256개의 클럭을 세고 오버 플로우

### Timer1 (16 bit)

- Servo 라이브러리 등에 관여
- 65536 개의 클럭을 세고 오버 플로우
- 해당 핀은 9번(OC1A),10번(OC1B) 핀



## 2. ATmega Fast PWM 기능 사용

Timer1(16bit)를 사용한 정밀한 제어 가능

서보 모터를 사용하는 상황을 가정

- 서보 모터 2개를 동시 제어
- Fast PWM 제어를 사용
- 분주비는 8



## 2. ATmega Fast PWM 기능 사용

```
//pin 설정. 서보모터 쪽 출력(ouput) 설정  
DDRB = 0b00000110;
```

```
//PWM 설정. 8 분주, top :ICR1 Fast PWM mode  
TCCR1A = (1<<COM1A1) | (1<<COM1B1) | (1<<WGM11);  
TCCR1B = (1<<WGM13) | (1<<WGM12) | (1<<CS11);  
ICR1 = 39999; // 20 ms
```

```
OCR1A = ocr1a; // 1200~4800  
OCR1B = ocr1b; // 1200~4800
```

주파수 50Hz (20ms)

0도 600μs (20 : 0.6 = 39999 : X) X= 1200

180도 2400μs (20 : 2.4 = 39999 : X) X= 4800

- 출력 설정 \*pinMode(pin, OUTPUT)

- Timer1의 레지스터 설정  $f_{OCnXPWM} = \frac{f_{clk\_I/O}}{N \cdot (1 + TOP)}$   
- 클럭과 주파수, 분주비를 통해 ICR1(TOP)을 결정

- OCR1A, OCR1B 출력 핀으로 서보 모터 제어



## 2. ATmega Fast PWM 기능 사용

데이터 시트를 활용하여 사용하고 싶은 타이머 레지스터를 설정

Table 15-5. Waveform Generation Mode Bit Description<sup>(1)</sup>

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, phase correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, phase correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, phase correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, phase and frequency correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, phase and frequency correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, phase correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, phase correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	–	–	–
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Table 15-6. Clock Select Bit Description

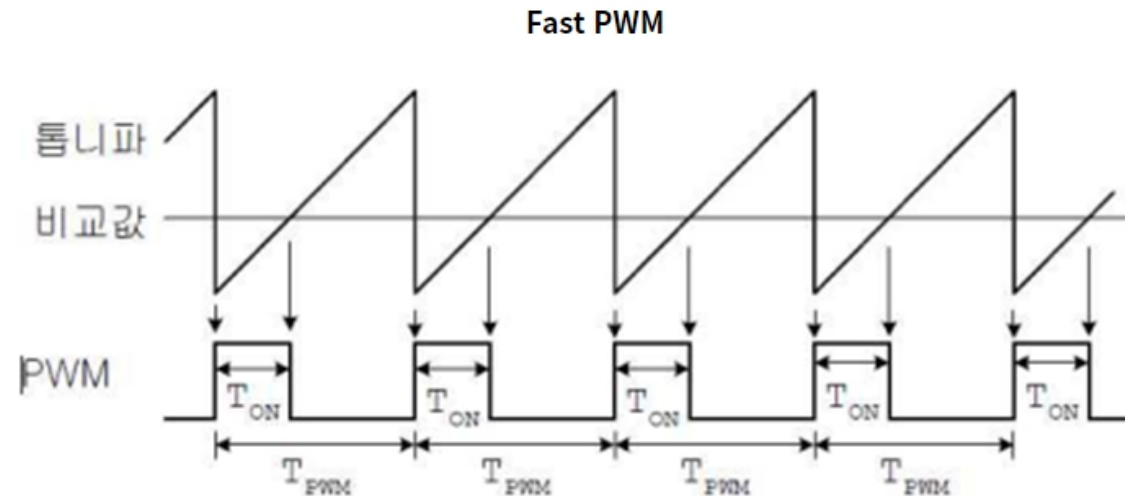
CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}/1$ (no prescaling)
0	1	0	$clk_{I/O}/8$ (from prescaler)
0	1	1	$clk_{I/O}/64$ (from prescaler)
1	0	0	$clk_{I/O}/256$ (from prescaler)
1	0	1	$clk_{I/O}/1024$ (from prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

ICR1 Fast PWM 모드 설정

분주 비 8 설정

## 2. ATmega Fast PWM 기능 사용

- 비교값이 ICR1 값이 되어 OCR1A, OCR1B 가 반복해서 PWM 신호를 내보냄
- OCR1A, OCR1B 값을 변경하여 서보 모터의 각도를 제어







### 3. 아두이노의 Servo 라이브러리 사용

아두이노의 Timer1(16bit)을 사용하는 Servo 라이브러리를 사용

- 직접 타이머 레지스터를 변경하는 방법보다 매우 쉬움
- 라이브러리 내의 함수를 사용하여 서보 모터의 출력 핀과 각도를 제어



### 3. 아두이노의 Servo 라이브러리 사용

```
#include<Servo.h>    //서보모터 라이브러리 추가
Servo myservo;        //myservo 라는 서보모터 변수 지정

void setup(){
  myservo.attach(9);  //PWM가용핀에 서보모터 연결
}

void loop(){
  myservo.write(180); //서보각 180도
  delay(1000);        //딜레이 1초
  myservo.write(10);  //서보각 0도
  delay(1000);        //딜레이 1초
}
```

#include<Servo.h> 라이브러리 추가

Servo.attach(pin)

-> 출력 핀을 지정

Servo.attach(pin, min, max)

-> 출력 핀, Pulse Width 의 가용 범위 지정

### 3. 아두이노의 Servo 라이브러리 사용

The methods are:

Servo - Class for manipulating servo motors connected to Arduino pins.

attach(pin ) - Attaches a servo motor to an i/o pin.

attach(pin, min, max ) - Attaches to a pin setting min and max values in microseconds  
default min is 544, max is 2400

write() - Sets the servo angle in degrees. (invalid angle that is valid as pulse in microseconds is treated as microseconds)

writeMicroseconds() - Sets the servo pulse width in microseconds

read() - Gets the last written servo pulse width as an angle between 0 and 180.

readMicroseconds() - Gets the last written servo pulse width in microseconds. (was read\_us() in first release)

attached() - Returns true if there is a servo attached.

detach() - Stops an attached servos from pulsing its i/o pin.

서보 라이브러리 안의 설명을 통해 상황에 맞는 여러 함수를 이용

### 3. 아두이노의 Servo 라이브러리 사용

`attach(pin )` - Attaches a servo motor to an i/o pin.

`attach(pin, min, max )` - Attaches to a pin setting min and max values in microseconds

default min is 544, max is 2400

Pin의 기본 값이 min 544, max 2400 microseconds로 지정되어 있으므로  
사용할 서보 모터의 사양에 따라 `attach(pin, min, max)`로 설정값을 변경

No.	Item	Specification
4-1	Control System	PWM(Pulse width modification)
4-2	Pulse width range	500~2500μsec
4-3	Neutral position	1500μsec
4-4	Running degree	180° or 270° (when 500~2500 μ sec)
4-5	Dead band width	2 μsec



### 3. 아두이노의 Servo 라이브러리 사용

`attach(pin )` - Attaches a servo motor to an i/o pin.

`attach(pin, min, max )` - Attaches to a pin setting min and max values in microseconds

default min is 544, max is 2400

Pin의 기본 값이 min 544, max 2400 microseconds로 지정되어 있으므로  
사용할 서보 모터의 사양에 따라 `attach(pin, min, max)`로 설정값을 변경

No.	Item	Specification
4-1	Control System	PWM(Pulse width modification)
4-2	Pulse width range	500~2500μsec
4-3	Neutral position	1500μsec
4-4	Running degree	180° or 270° (when 500~2500 μ sec)
4-5	Dead band width	2 μsec



### 3. 아두이노의 Servo 라이브러리 사용

```
#define MIN_PULSE_WIDTH    544    // the shortest pulse sent to a servo
#define MAX_PULSE_WIDTH    2400    // the longest pulse sent to a servo
#define DEFAULT_PULSE_WIDTH 1500    // default pulse width when servo is attached
#define REFRESH_INTERVAL    20000  // minimum time to refresh servos in microseconds

#define SERVOS_PER_TIMER    12     // the maximum number of servos controlled by one timer
```

제어 주파수를 변경해야 할 경우

```
#define REFRESH_INTERVAL    20000
```

50Hz, 20ms에서 원하는 시간으로 변경(333Hz의 경우 3ms, 3000)

#define SERVOS\_PER\_Timer 에서 숫자를 줄여가며 체크

감사합니다