AVR - HW7

임베디드스쿨1기
Lv1과정
2020. 10. 30
김인겸

복습(Timer/Counter 관련 인터럽트)

8	0x000E	TIMER2 COMPA	Timer/Counter2 compare match A
9	0x0010	TIMER2 COMPB	Timer/Counter2 compare match B
10	0x0012	TIMER2 OVF	Timer/Counter2 overflow
11	0x0014	TIMER1 CAPT	Timer/Counter1 capture event
12	0x0016	TIMER1 COMPA	Timer/Counter1 compare match A
13	0x0018	TIMER1 COMPB	Timer/Counter1 compare match B
14	0x001A	TIMER1 OVF	Timer/Counter1 overflow
15	0x001C	TIMER0 COMPA	Timer/Counter0 compare match A
16	0x001E	TIMER0 COMPB	Timer/Counter0 compare match B
17	0x0020	TIMER0 OVF	Timer/Counter0 overflow

복습(CTC vs Fast PWM vs Phase Correct PWM)

Figure 14-5. CTC Mode, Timing Diagram

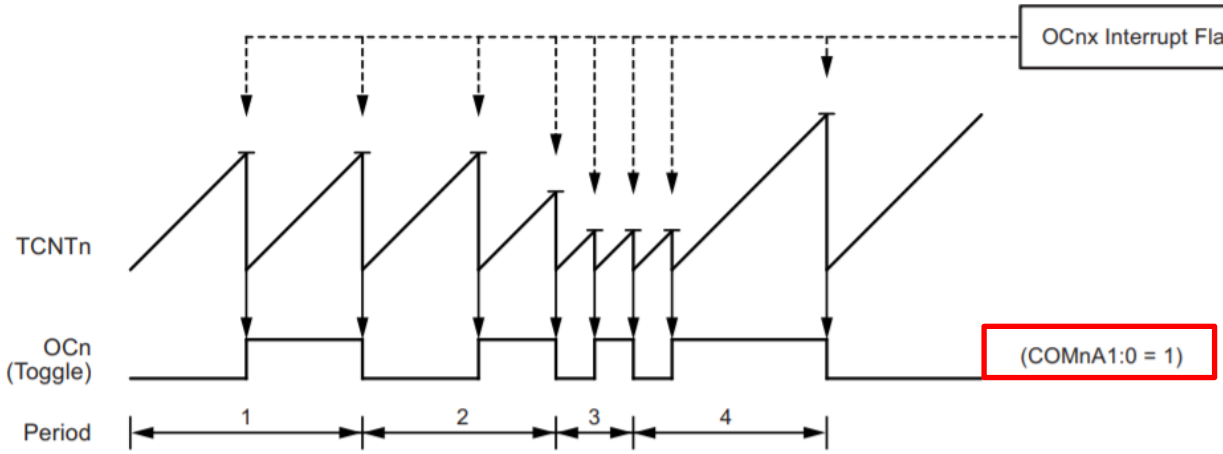


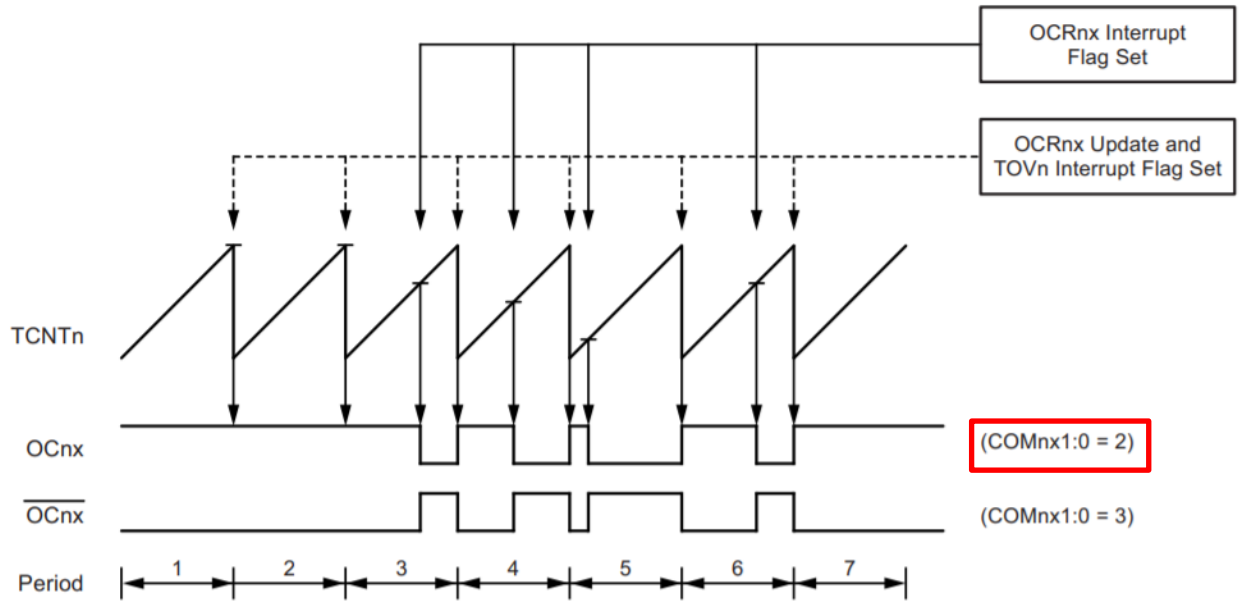
Table 14-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on compare match
1	0	Clear OC0A on compare match
1	1	Set OC0A on compare match

듀티는 50%로 일정.
OCR값을 조절하여 주기만 변경할 경우 사용함

복습(CTC vs Fast PWM vs Phase Correct PWM)

Figure 14-6. Fast PWM Mode, Timing Diagram



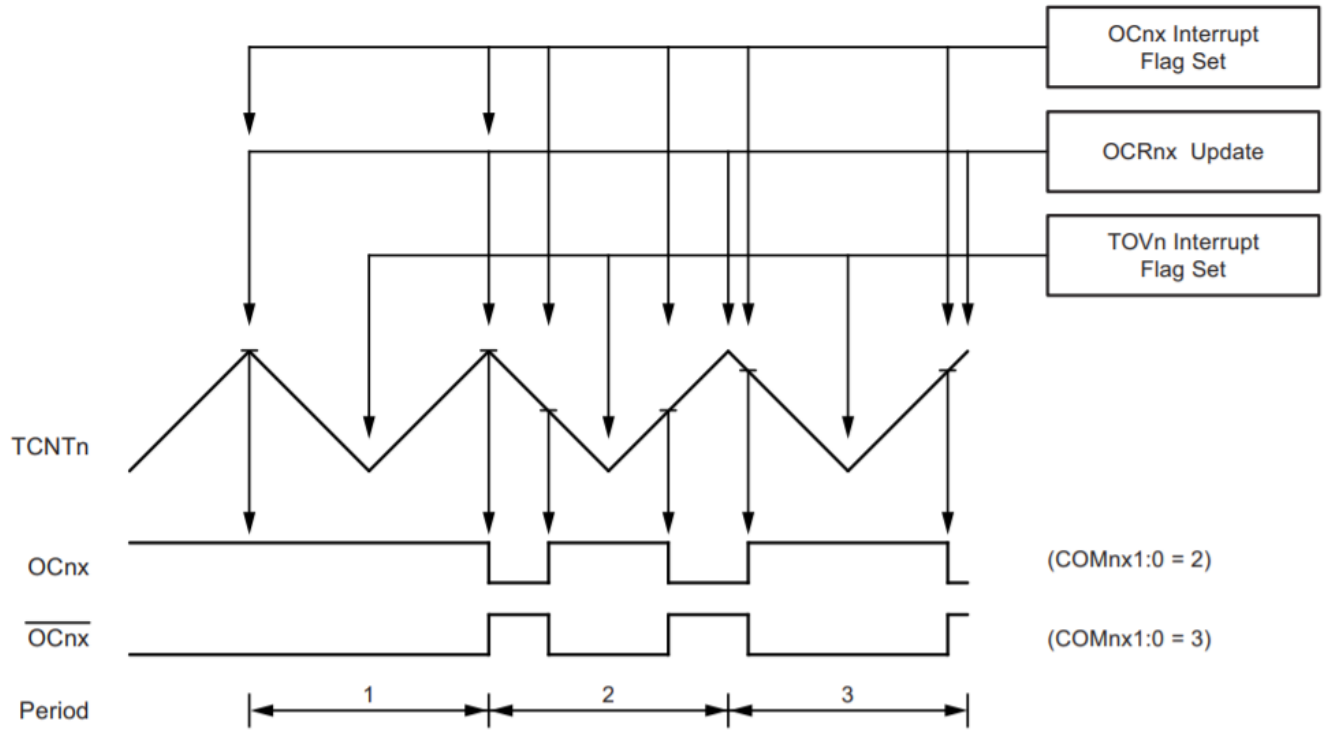
주기가 일정하고
듀티비를 조절할 때 사용함

Table 14-3. Compare Output Mode, Fast PWM Mode(1)

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal port operation, OC0A disconnected. WGM02 = 1: Toggle OC0A on compare match.
1	0	Clear OC0A on compare match, set OC0A at BOTTOM, (non-inverting mode).
1	1	Set OC0A on compare match, clear OC0A at BOTTOM, (inverting mode).

복습(CTC vs Fast PWM vs Phase Correct PWM)

Figure 14-7. Phase Correct PWM Mode, Timing Diagram



주기가 2배 -> 주파수 ½
대칭적인 특징이 있어서 모터제어에 많이 사용됨.

OCR값과 비교일치 시에 파형이 반전되지만
TCNT값이 0이 될 때 인터럽트를 사용할 수도 있음,

복습(CTC모드 - 노멀모드처럼 사용)

```
volatile unsigned int count = 0;

SIGNAL(TIMER0_COMPA_vect)
{
    count++;
}
```

1ms마다 발생하는 인터럽트

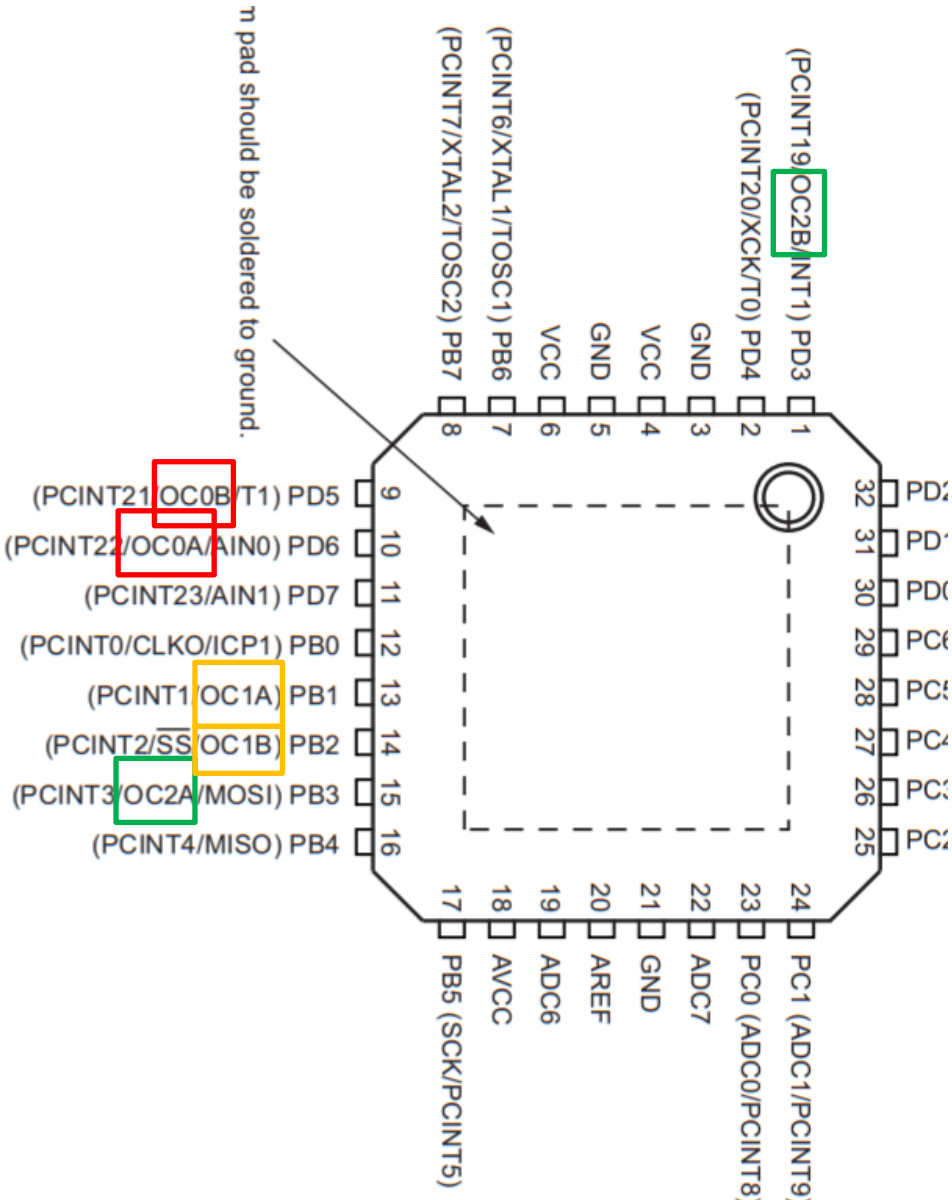
```
void OCM_timer_init(void)
{
    cbi(SREG,7);
    TCCR0A = 0;
    TCCR0B = 0;
    PORTB = 0x00;
    DDRB = 0xff;

    TCCR0A = (1<<WGM01);    //CTC모드 설정
    TCCR0B = (1 << CS01) | (1 << CS00);     //분주비 64  -> 16Mhz/64 = 250KHz  ,  주기=4us
    TIMSK0 = (1 << OCIE0A);     //인터럽트 활성화
    TCNT0 = 0;
    OCR0A = 249;     //CTC모드에서 OCR을 바꾸면 TOP값을 바꿀 수 있다.  데이터시트공식참고
    sbi(SREG, 7);
}
```

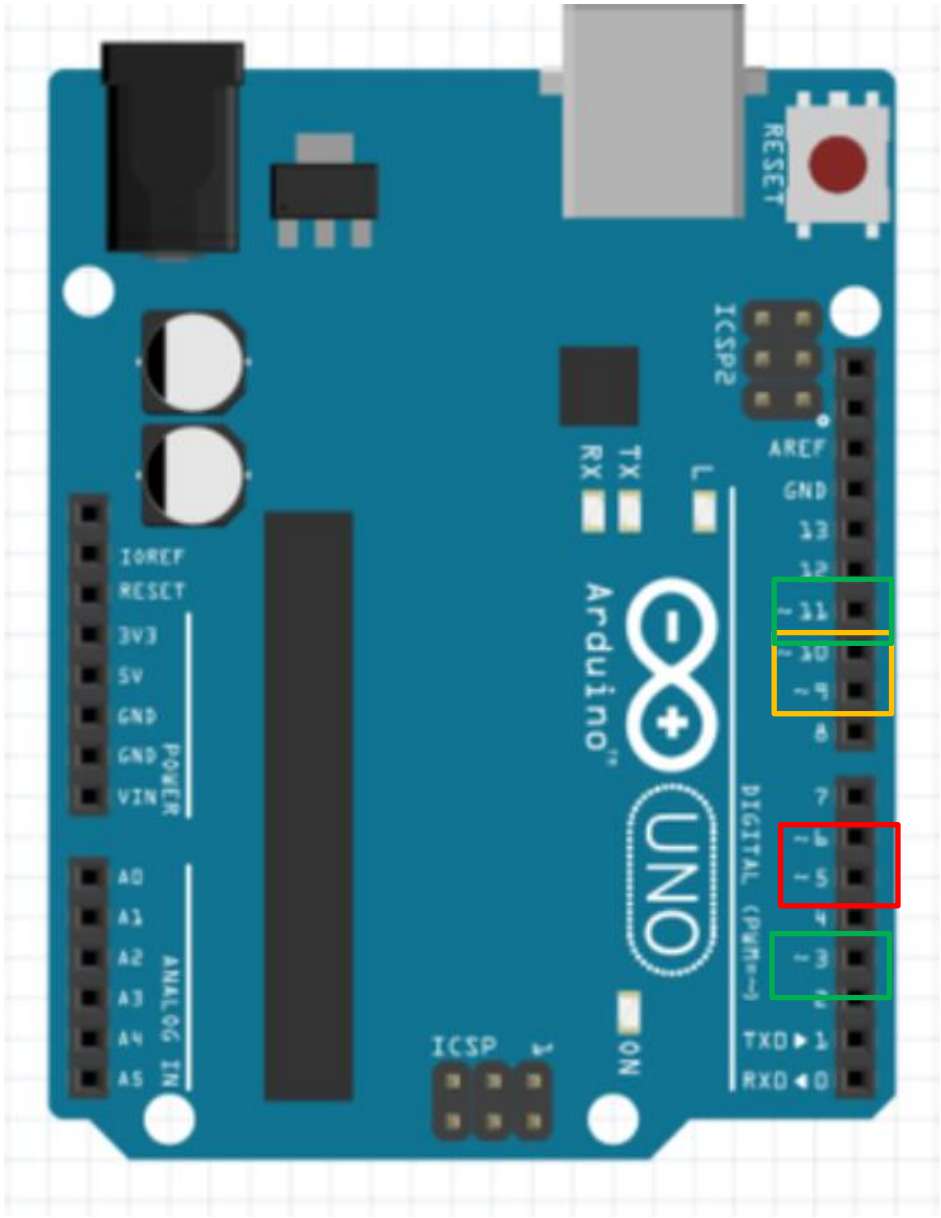
4us * (1+249) = 1ms

```
int main(void)
{
    OCM_timer_init();
    while (1)
    {
        if(count == 500)
        {
            PORTB = (~PORTB);
            count = 0;
        }
    }
```

1ms마다 발생하는
인터럽트가 500번 발생
= 500ms 마다 PORTB가
0.5초에 한 번씩 반전됨.

복습(CTC모드 - 파형생성)





복습(CTC모드 - 파형생성)

```
SIGNAL(TIMER0_COMPA_vect)
{

}

void CTC_timer_init(void)
{
    cbi(SREG,7);

    sbi(TCCR0A, WGM01); //CTC모드 설정
    sbi(TCCR0A, COM0A0);
    sbi(TCCR0B, CS02); //분주비
    sbi(TCCR0B, CS00); //1024
    sbi(TIMSK0, OCIE0A); //비교일치 인터럽트A 사용
    OCR0A = 255;
    DDRD = 0xff;
    PORTD = 0xff;
    sbi(SREG,7);
}

int main(void)
{
    CTC_timer_init();

    while(1)
    {

    }
}
```

Timer/Counter0의 OCR0A레지스터를
설정했으므로
PD6번 핀에서 파형이 출력됨.

Table 14-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on compare match
1	0	Clear OC0A on compare match
1	1	Set OC0A on compare match

복습(Fast-PWM모드)

```
SIGNAL(TIMER0_COMPA_vect)
{
    OCR0A += 1;

    if(OCR0A == 255)
    {
        OCR0A = 0;
    }
}

void Fast_PWM_mode(void)
{
    sbi(TCCR0A, WGM01);//Fast PWM모드 설정
    sbi(TCCR0A, WGM00);

    sbi(TCCR0A, COM0A1);

    sbi(TCCR0B, CS02);//분주비1024
    sbi(TCCR0B, CS00);

    sbi(TIMSK0, OCIE0A);//비교일치인터럽트 사용

    DDRD = 0xff;
    OCR0A = 0;

    sbi(SREG, 7);

}

int main(void)
{
    Fast_PWM_mode();
    while(1)
    {
```

Table 14-3. Compare Output Mode, Fast PWM Mode(1)

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal port operation, OC0A disconnected. WGM02 = 1: Toggle OC0A on compare match.
1	0	Clear OC0A on compare match, set OC0A at BOTTOM, (non-inverting mode).
1	1	Set OC0A on compare match, clear OC0A at BOTTOM, (inverting mode).

듀티비가 0부터 100인 펄스파를 계속해서 생성하는 코드.
이때 주기는 일정함.

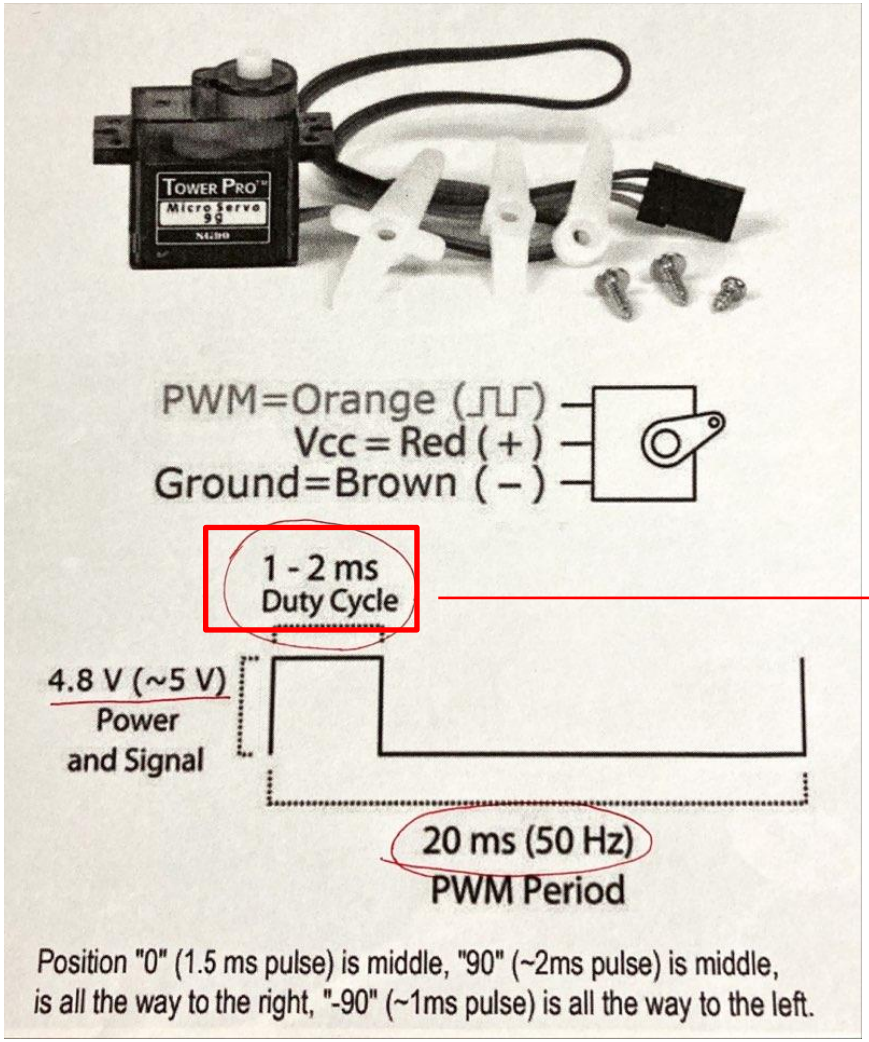
복습(Phase Correct PWM모드)

```
SIGNAL(TIMER0_COMPA_vect)
{
    OCR0A += 1;

    if(OCR0A == 255)
    {
        OCR0A = 0;
    }
}

void Phase_Correct_PWM_mode(void)
{
    cbi(SREG,7);

    sbi(TCCR0A, WGM02);//Phase Correct PWM모드
    sbi(TCCR0A, WGM00);

    sbi(TCCR0A, COM0A1);

    sbi(TCCR0B, CS02);//분주비 1024
    sbi(TCCR0B, CS00);

    sbi(TIMSK0, OCIE0A);//비교일치 인터럽트

    DDRD = 0xff;
    OCR0A = 0;
    sbi(SREG, 7);

}

int main(void)
{
    Phase_Correct_PWM_mode();

    while(1)
    {

    }
}
```

듀티비가 0부터 100인 펄스파를 계속해서
생성하는 코드.
이때 주기는 일정함.

복습(Servo Motor)

모터제어의 원리 : 파형의 듀티비를 조절 -> 평균전압을 조절 -> 모터에 주는 전압을 조절하여 모터를 제어함



Servo Motor는 각도를 제어하는 모터이며 RC Servo Motor는 20ms(50Hz)에서 동작한다.

값	1ms	1.5ms	2ms
듀티비	5%	7.5%	10%
회전각도	-90도	0	+90도

복습(Servo Motor), ICR값이 39,999인 이유

```
void SG_Fast_PWM_mode(void)
{
    cbi(SREG,7);
    //16비트 타이머카운터1 사용
    sbi(TCCR1A, WGM11); //Fast PWM모드
    sbi(TCCR1B, WGM12);
    sbi(TCCR1B, WGM13);

    sbi(TCCR1A, COM1A1);
    sbi(TCCR1A, COM1B1);


    sbi(TCCR1B, CS11);//분주비 8

    ICR1 = 39999;

    DDRB = (1 << PORTB1);

    sbi(SREG,7);
}
```

타이머 카운터 클럭 : 16MHz/8 = 2MHz
타이머 카운터 주기 : 1/2MHz = 0.5us

16비트 타이머 카운터의 범위 : 0~ 2^16 – 1 (65535)

ICR1 값이 39999이므로 타이머는 0~39999까지 계수.

타이머/카운터가 0~39999까지 계수하는데 걸리는 시간
= 0.5us * 40,000 = 20ms

이때 20ms는 서보모터가 동작시키기 위해 맞춰줘야 되는 주기이다

따라서 16MHz, 8비트 분주비에서 ICR1값이 39999인 이유는 서보모터의 동작 주기인 20ms를 맞춰주기 위함이다

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP

복습(Servo Motor), ICR값이 39,999인 이유

```
int main(void)
{
    SG_Fast_PWM_mode();

    while(1)
    {
        OCR1A = 2000;
        _delay_ms(1000);

        OCR1A = 3000;
        _delay_ms(1000);

        OCR1A = 4000;
        _delay_ms(1000);

        OCR1A = 3000;
        _delay_ms(1000);

    }
}
```

(2,000/40,000) * 100 = 5% -> -90도

(3,000/40,000) * 100 = 7.5% -> 0도

(4000/40,000) * 100 = 10% -> +90도