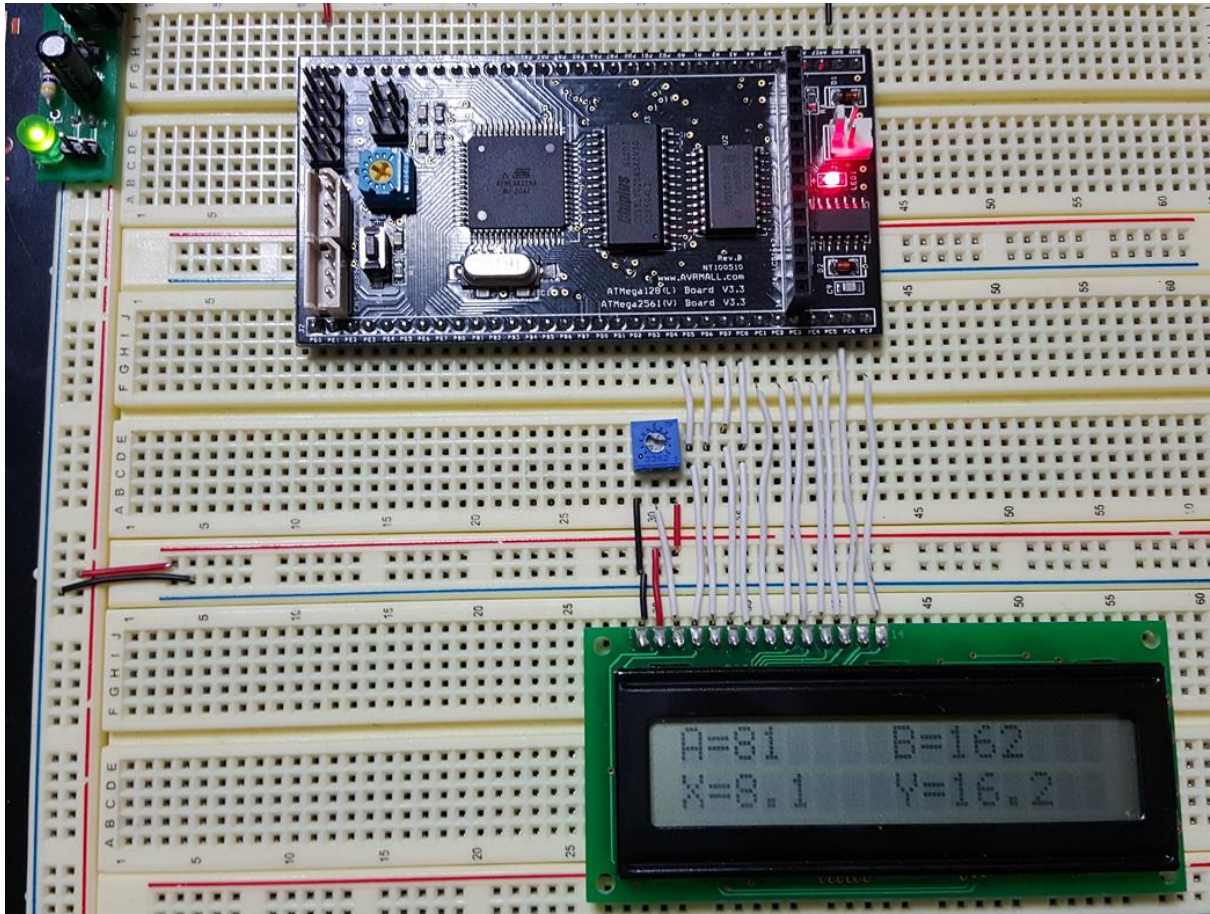


마이크로 컴퓨터

실험 4. PWM을 이용한 모터 속도 제어

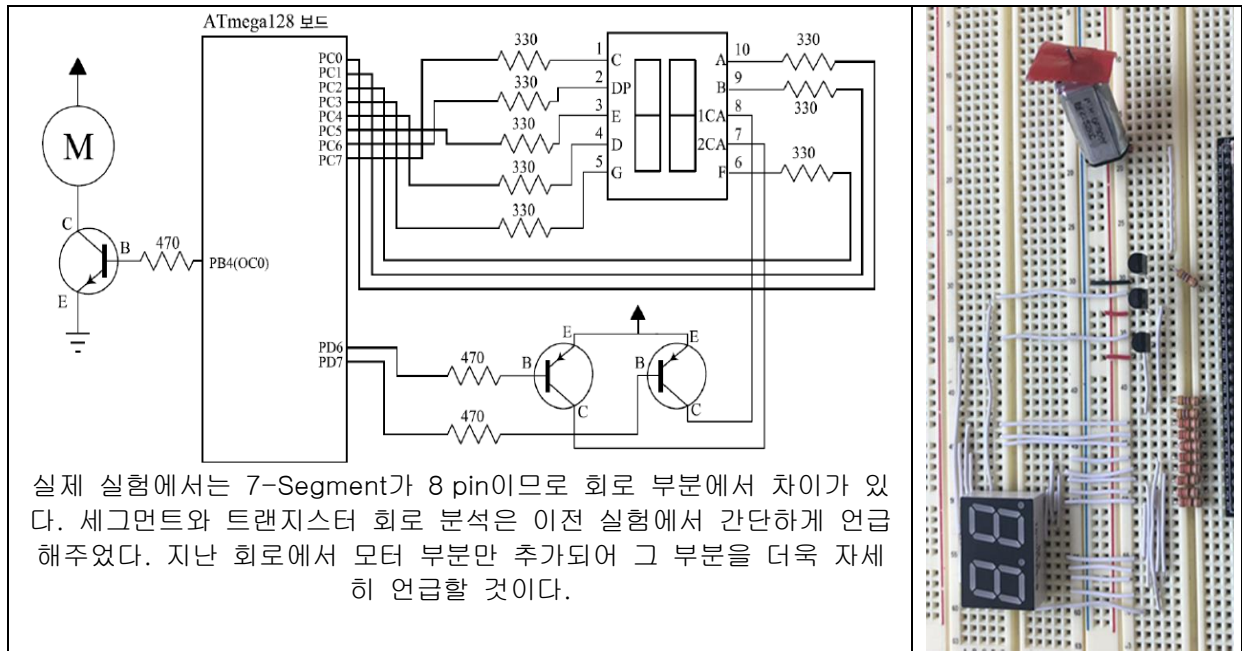


전자공학과 21611591 김 난 희

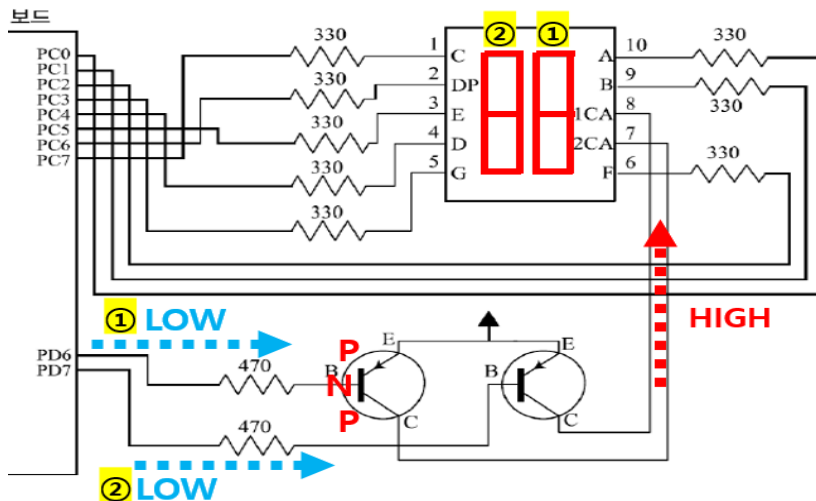
실험 목적

1. PWM 모드를 이해하고, PWM을 이용한 간단한 프로그램을 이해한다.
2. Duty의 이해와 모터에 미치는 영향을 이해한다.
3. OCR0 레지스터가 Duty와 관련됨을 이해한다.
4. COM00, COM01 레지스터를 이해한다.

1. 실험 회로

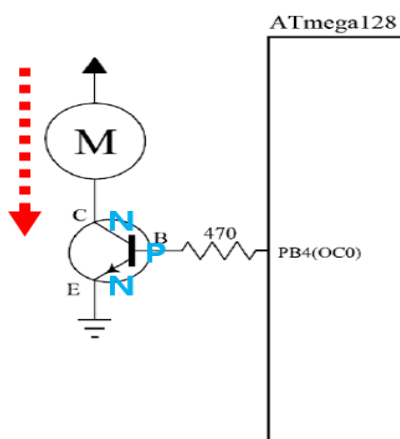


1-1. 회로 분석과 구동



왼쪽 그림은 세그먼트 구동 회로이다. PD6과 PD7을 빠른 속도로 번갈아 HIGH, LOW하여 세그먼트의 1의 자리를 켜지, 10의 자리를 켜지 선택한다. 이때 1의 자리와 10의 자리 세그먼트가 아주 빠른 속도로 번갈아 켜지기 때문에 둘다 켜져있는 것처럼 보인다. 트랜지스터는 외부 전원을 가져와 사용하여 Base에 LOW가 흐르면 통로를 열어준다. Emitter와 연결해 놓은 Vcc에서 전류가 통해 흘러 결국 세그먼트의 Select pin에 도달하게 된다. 1번 LOW 신호는

1의 자리 세그먼트를 구동하고, 2번 LOW 신호는 10의 자리 세그먼트를 구동한다.



다음의 왼쪽 그림은 모터 구동 회로이다.

사용한 트랜지스터는 NPN 트랜지스터로 모터에 흐르는 전류를 간접 구동한다. 간접 구동한다는 말은 즉, 외부 전원으로 모터에 파워를 공급한다는 말이다.

이로부터 모드가 Fast PWM인 경우, Phase correct인 경우로 나누어 분주비를 조절하여 실험을 해보았다.

2. 프로그램 분석

```
#include <avr/io.h>
#include <avr/interrupt.h>

#define DISP_INTERVAL 2
#define OCO PB4

volatile unsigned long timer0; // 오버플로마다 1씩 증가될 변수
volatile unsigned int number;
unsigned char led[] = {0x88, 0xBE, 0xC4, 0xA4, 0xB2, 0xA1, 0x83, 0xBC, 0x80, 0xB0};

// 타이머/카운터0 인터럽트 서비스 루틴
ISR(TIMERO_OVF_vect)
{
    timer0++; // 오버플로마다 1씩 증가

    // 두 자리 7-세그먼트 LED 디스플레이
    if( timer0 % DISP_INTERVAL == 0){
        PORTC = (timer0 % (DISP_INTERVAL<<1) == 0) ? led[(number % 100) / 10] : led[number%10];
        PORTD = (PORTD | OCO) & ~(1<<((timer0 % (DISP_INTERVAL<<1) == 0) ? PD7 : PD6));
    }
}

int main(void)
{
    DDRC = 0xFF;
    DDRD |= 1<<PD7 | 1<<PD6; // 두 자리 7-세그먼트 LED를 켜기위한 출력
    DDRB |= 1<<OCO; // OCO = PB4 출력

    TCCR0 = 1<<WGM01 | 1<<WGM00; // 고속 PWM 파형 발생모드
    TCCR0 |= 1<<CS02 | 1<<CS01 | 0<<CS00; // 프리스케일러 CS02:00 = (1,1,0) 256 분주
    TCCR0 |= 1<<COM01; // TCNT0 0xFF에서 1, 상승 도중 OCR0와 일치하면 0

    TIMSK |= 1<<TOIE0; // 타이머/카운터0 인터럽트 활성화

    timer0 = 0;
    sei();

    OCR0 = 100;
    number = OCR0*101/256; // OCR0 값과 number 값을 일치
    while(1); // 무한 loop

    return 0;
}
```

```
#include <avr/io.h>
#include <avr/interrupt.h>
```

```
#define DISP_INTERVAL 2
#define OCO PB4
```

```
volatile unsigned long timer0; // 오버플로마다 1씩 증가될
변수
```

```
volatile unsigned int number;
```

```
unsigned char led[] = {0x88, 0xBE, 0xC4, 0xA4, 0xB2, 0xA1,
0x83, 0xBC, 0x80, 0xB0};
```

위 부분에 대한 프로그램 분석은 이전 실험, 7-Segment LED 디스플레이에서 자세히 분석하였으므로 설명을 생략합니다.

위에서 WGM00과 WGM01비트를 1으로 설정해주었기 때문에 Waveform Generation Mode를 Fast PWM 모드로 설정해준 부분이다. 만약 Phase correct PWM 모드로 설정하여 실험할 시에는 WGM00과 WGM01비트를 순서대로 1, 0으로 설정해준다. 아래 그림을 참고하면 된다.

Table 52. Waveform Generation Mode Bit Description

Mode	WGM01 ⁽¹⁾ (CTC0)	WGM00 ⁽¹⁾ (PWM0)	Timer/Counter Mode of Operation	TOP	Update of OCR0 at	TOV0 Flag Set on
0	0	0	Normal	0xFF	Immediate	MAX
1	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR0	Immediate	MAX
3	1	1	Fast PWM	0xFF	TOP	MAX

$TCCR0 |= 1 \ll CS2 | 1 \ll CS01 | 0 \ll CS00;$ // 프리스케일
러 CS2:00 = (1,1,0) 256 분주 아래 노란색 부분으로 표시한 부분이 실험으로
변경해준 모든 조건이다. 순서대로 변경해주며 오실로스코프로 PWM 파형을 관찰하였다.

Table 56. Clock Select Bit Description

CS2	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	clk _{T0S} /(No prescaling)
0	1	0	clk _{T0S} /8 (From prescaler)
0	1	1	clk _{T0S} /32 (From prescaler)
1	0	0	clk _{T0S} /64 (From prescaler)
1	0	1	clk _{T0S} /128 (From prescaler)
1	1	0	clk _{T0S} /256 (From prescaler)
1	1	1	clk _{T0S} /1024 (From prescaler)

$TCCR0 |= 1 \ll COM01;$ // TCNT0 0xFF에서 1, 상승 도중
OCR0와 일치하면 0

Table 54. Compare Output Mode, Fast PWM Mode⁽¹⁾

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Reserved
1	0	Clear OC0 on compare match, set OC0 at TOP
1	1	Set OC0 on compare match, clear OC0 at TOP

$TIMSK |= 1 \ll TOIE0;$ // 타이머/카운터0 인터럽트 활성화

아래의 타이머/카운터0을 사용하므로, TIMSK에서 다음 주황색 형관펜으로 칠해진 부분만 보면 된다.

OC 관련해서는 사용하지 않기 때문에 제외하고 본다. TIMSK의 TOIE0(Timer/counter0 Overflow Interrupt Enable)가 1로 SET되면 Overflow 인터럽트를 Enable하는 것이다.

아래의 TIFR(Timer/Counter Interrupt Flag Register)의 TOV0가 1로 SET되면, 인터럽트를 걸 수 있다.

Timer/Counter Interrupt Mask Register – TIMSK

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Timer/Counter Interrupt Flag Register – TIFR

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

```
timer0 = 0; //타이머 오버플로마다 갱신되는 변수
```

처음 시작 전에 0으로 초기화 해놓는다

```
sei(); //Global interrupt Enable bit
```

The AVR status Register – SREG – is defined as:

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

위에서 인터럽트 허용 비트를 Enable 해주고 다음 sei()를 해주면, 전체 인터럽트를 허용하여 최종적으로 인터럽트를 사용 가능하게 한다. sei()는 Set Interrupt Flag로 위의 SREG의 최상위 비트, I비트를 1로 SET하여 전체 인터럽트를 허용 여부를 판단한다.

```
OCR0 = 100; //OCR0 값을 설정해주어 아래 number 변수에서 7-Segment에 디스플레이할 숫자를 정해주었다.
```

```
number = OCR0*101/256; // OCR0 값과 number 값을 일치
```

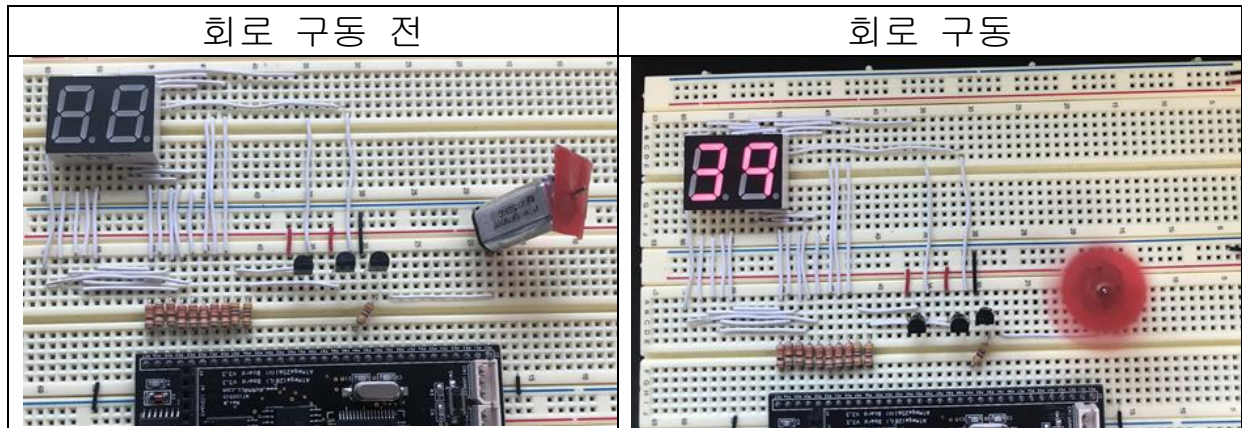
```
while(1); // 무한 loop
```

무한 루프를 통해 아무 것도 안하는 것을 수행하고 있다. 어셈블리어로 말하면 NOP를 수행하고 있다. 이 while문을 반복 수행하면서 인터럽트가 걸리면 ISR(인터럽트 서비스 루틴)을 수행하는 형태로 되어있다.

```
return 0; // main함수가 int형이기 때문에 0을 반환하면서 종료
```

```
}
```

3.실험 결과 및 분석




- Fast PWM

Fast PWM인 경우 주기 계산은 다음과 같이 한다.
 TCNT0이 최소 0부터 최대 FF까지 증가 후 다시 0으로 돌아와 반복하는 형태이다. 0~FF(255)는 총 256번의 클럭 펄스가 사용된다.
 즉 고속 PWM 주기 식은 다음과 같다.

$$\text{Fast PWM 주기} = \frac{256 \times \text{프리스케일}}{f_{clk_I/O}}$$

분주 비 변경으로부터 오실로스코프 결과 사진에서 다음의 주기가 변화한 것을 관찰할 수 있다.
 실험에서 COM01과 COM00을 1과 0으로 설정하였고, TCNT0로부터 총 256번의 클럭 펄스가 사용되므로, Duty는 다음의 식으로 계산되었다. 실험에서 OCR0는 100으로 설정했으므로, 0~100까지이므로 +1을 더해서 계산한다.

$$\text{Duty \%} = \frac{\text{OCR0} + 1}{256} \times 100(\%) = \frac{100 + 1}{256} \times 100(\%) = 39.45(\%)$$

오실로스코프 사진	분주비 및 주기 계산
	<p>사용한 분주비 : 1 (No prescaling) 관찰한 주기 : 16.00us</p> <p>이론식 :</p> $\text{Fast PWM} = \frac{256 \times 1}{f_{clk_I/O}}$ <p>= 16us</p> <p>→ 동일</p>



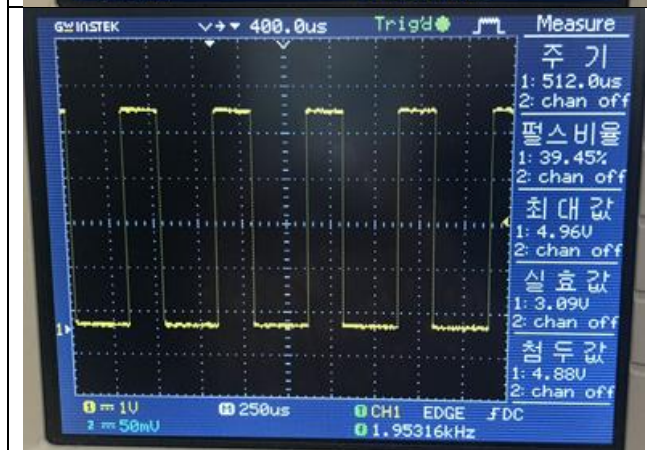
사용한 분주비 : 8(From prescaler)
관찰한 주기 : 128.0us

이론식 :

$$Fast\ PWM = \frac{256 \times 8}{f_{clk_I/O}}$$

=128us

→ 동일



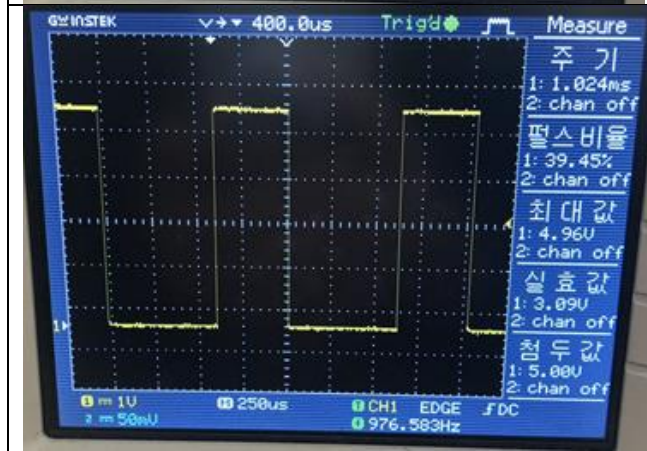
사용한 분주비 : 32(From prescaler)
관찰한 주기 : 512.0us

이론식 :

$$Fast\ PWM = \frac{256 \times 32}{f_{clk_I/O}}$$

=512us

→ 동일



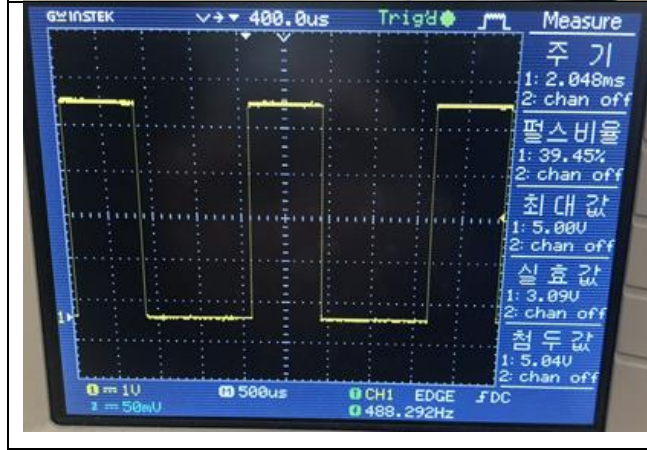
사용한 분주비 : 64(From prescaler)
관찰한 주기 : 1.024ms = 1024us

이론식 :

$$Fast\ PWM = \frac{256 \times 64}{f_{clk_I/O}}$$

=1024us

→ 동일



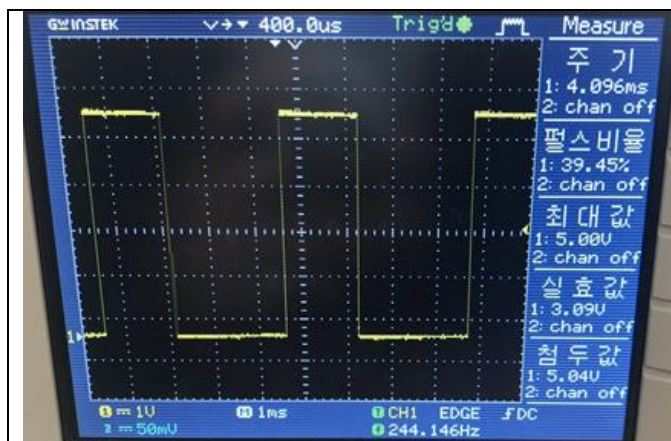
사용한 분주비 : 128(From prescaler)
관찰한 주기 : 2.048ms = 2048us

이론식 :

$$Fast\ PWM = \frac{256 \times 128}{f_{clk_I/O}}$$

=2048us

→ 동일

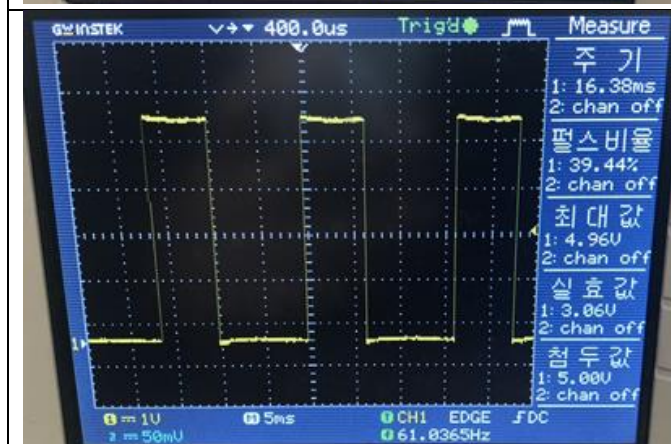


사용한 분주비 : 256(From prescaler)
관찰한 주기 : 4.096ms = 4096us

이론식 :

$$Fast\ PWM = \frac{256 \times 256}{f_{clk_{I/O}}} = 4096us$$

→ 동일



사용한 분주비: 1024(From prescaler)
관찰한 주기 : 16.38ms

이론식 :

$$Fast\ PWM = \frac{256 \times 1024}{f_{clk_{I/O}}} = 16384us$$

→ 동일

- Phase correct

Phase correct인 경우 주기 계산은 다음과 같이 한다.

TCNT0이 최소 0부터 최대 FF까지 증가 후 다시 0으로 감소해 반복하는 형태이다. 0~FF(255), FE(254)~1 까지 (256+254)번, 총 510번의 클럭 펄스가 사용된다. Fast PWM 주기의 약 2배정도를 가지게 된다. 즉, 고속 PWM 주기 식은 다음과 같다.

$$Phase\ correct\ PWM\ 주기 = \frac{510 \times 프리스케일}{f_{clk_{I/O}}}$$

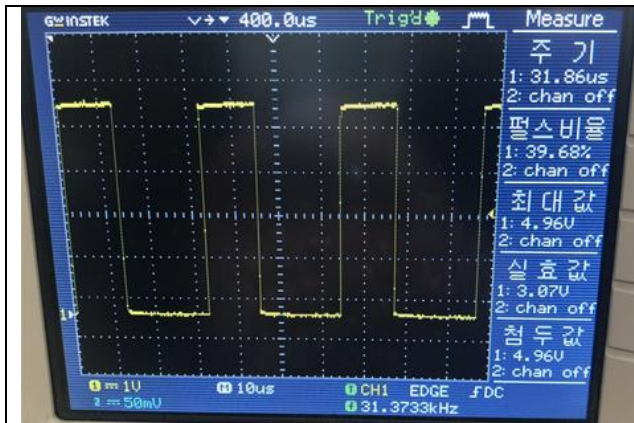
분주 비 변경으로부터 오실로스코프 결과 사진에서 다음의 주기가 변화한 것을 관찰할 수 있다. 실험에서 COM01과 COM00을 1과 0으로 설정하였고, TCNT0로부터 총 510번의 클럭 펄스가 사용되므로, Duty는 다음의 식으로 계산되었다. 실험에서 OCR0는 100으로 설정했다. TCNT0가 하강 중에 OCR0와 일치하면 1이 되고, 상승 중에 OCR0와 일치하면 0이 된다.

$$Duty\ \%/ = \frac{OCR0 \times 2}{510} \times 100(\%) = \frac{100 \times 2}{510} \times 100(\%) = 39.21(\%)$$

Fast PWM의 파형을 데칼코마니 형태로 한 것과 비슷하기 때문에 $\times 2$ 를 해주지만, 결국은 각각의 파형의 duty가 39%정도로 비슷하기 때문에 2배의 값도 39%와 비슷하게 나왔다는 식으로 생각하면 더욱 쉽게 이해할 수 있을 것이다. 더하여, 겹치는 부분이 존재하기 때문에 Fast PWM과는 완전히 같지 않고, 조금 작다고 생각하면 더욱 쉽다.

오실로스코프 사진

분주비 및 주기 계산



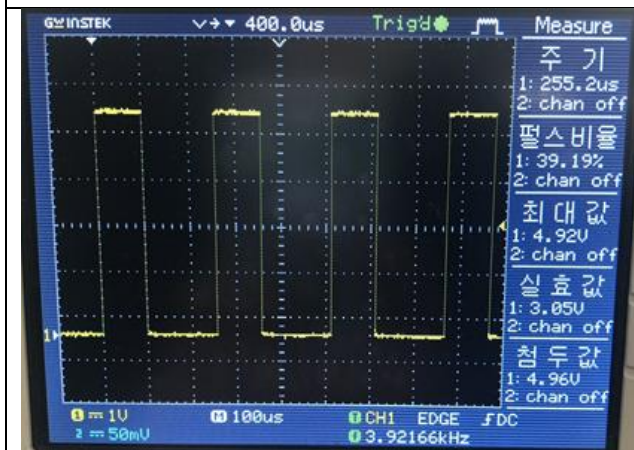
사용한 분주비 : 1(No prescaling)
관찰한 주기 : 31.86us

이론식 :

$$Phase\ correct\ PWM = \frac{510 \times 1}{f_{clk_I/O}}$$

=31.88us

→ 거의 동일



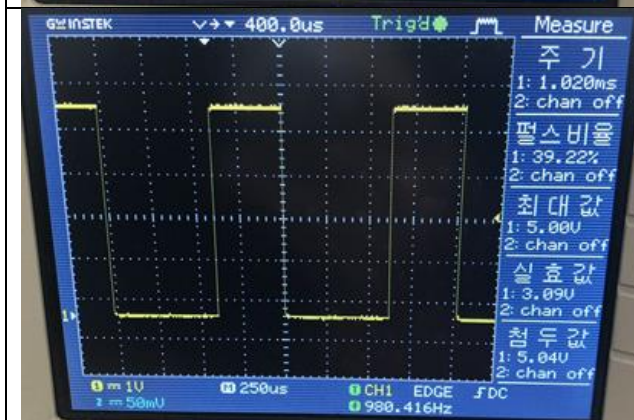
사용한 분주비 : 8(From prescaler)
관찰한 주기 : 255.2us

이론식 :

$$Phase\ correct\ PWM = \frac{510 \times 8}{f_{clk_I/O}}$$

=255us

→ 동일



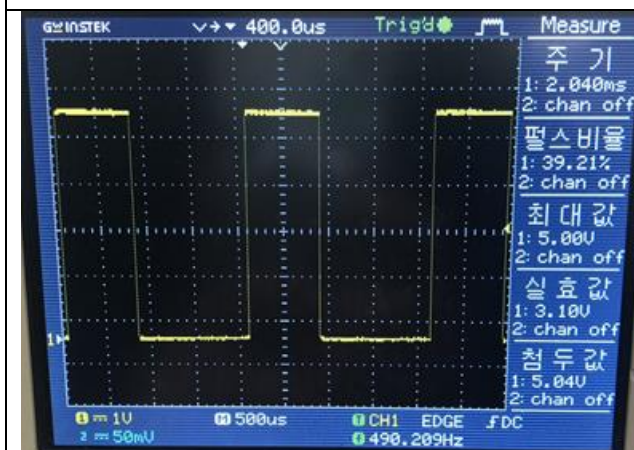
사용한 분주비 : 32(From prescaler)
관찰한 주기 : 1.020ms

이론식 :

$$Phase\ correct\ PWM = \frac{510 \times 32}{f_{clk_I/O}}$$

=1020us

→ 동일



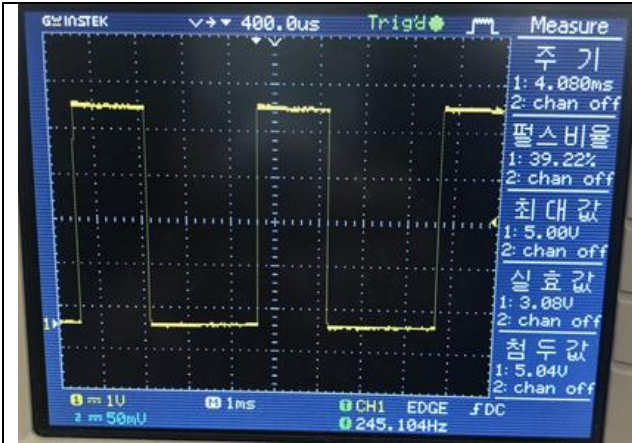
사용한 분주비 : 64(From prescaler)
관찰한 주기 : 2.040ms

이론식 :

$$Phase\ correct\ PWM = \frac{510 \times 64}{f_{clk_I/O}}$$

=2040us

→ 동일



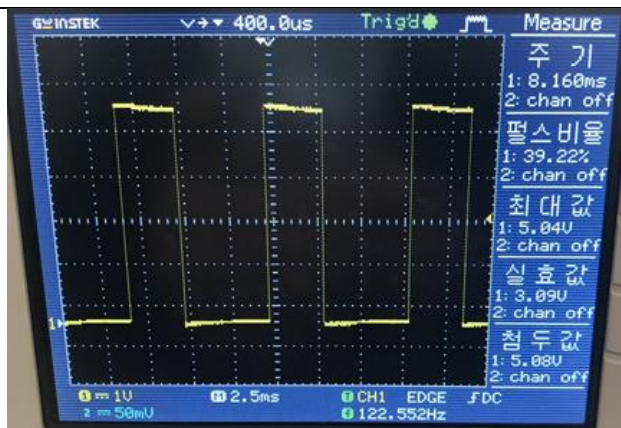
사용한 분주비 : 128(From prescaler)
관찰한 주기 : 4.080ms

이론식 :

$$\text{Phase correct PWM} = \frac{510 \times 128}{f_{clk_I/O}}$$

=4080us

→ 동일



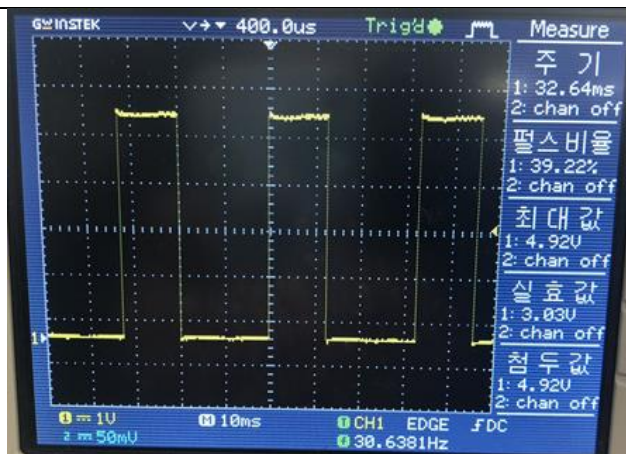
사용한 분주비 : 256(From prescaler)
관찰한 주기 : 8.160ms

이론식 :

$$\text{Phase correct PWM} = \frac{510 \times 256}{f_{clk_I/O}}$$

=8160us

→ 동일



사용한 분주비 : 1024(From prescaler)
관찰한 주기 : 32.64ms

이론식 :

$$\text{Phase correct PWM} = \frac{510 \times 1024}{f_{clk_I/O}}$$

=32640us

→ 동일

총 3번 정도 실험을 다시 하였는데, 문제는 주기가 제대로 나오지 않는 문제였다. 이는 오실로스코프를 변경하여 다시 실험해보는 것으로 간단히 해결하였다.

2. 추가 실험

(1) OCR0 변경하기

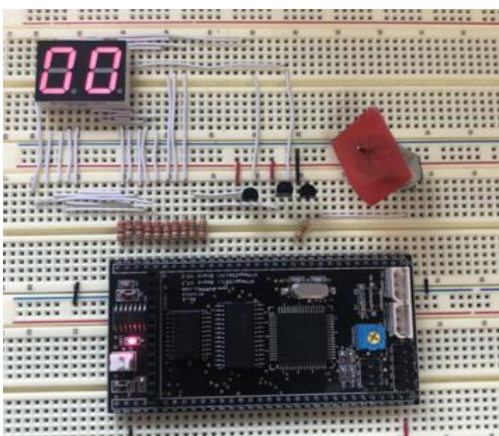
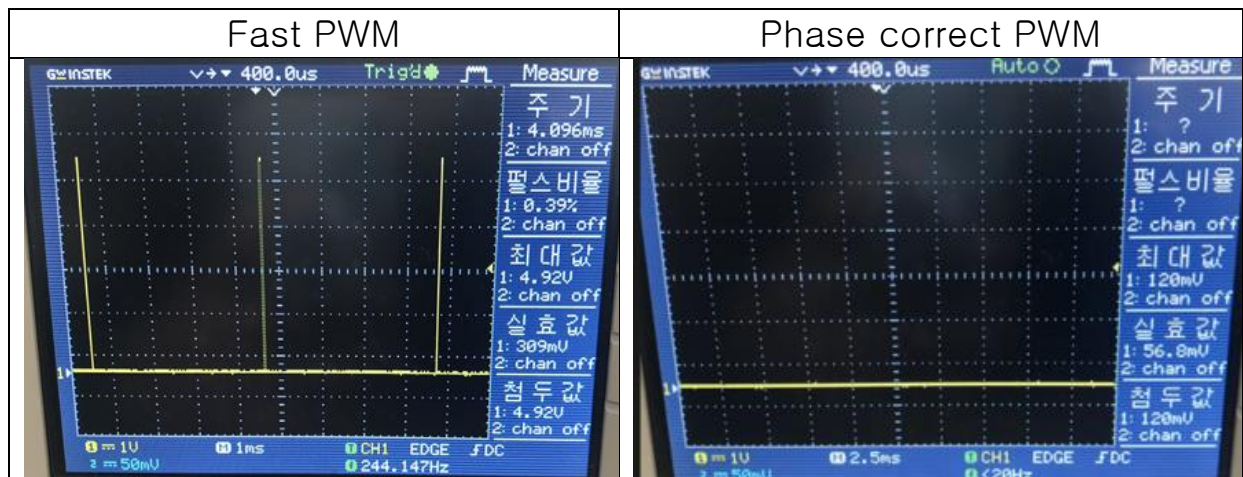
2-1. 실험 회로 → 본 실험과 회로가 동일합니다.

2-2. 프로그램 분석 및 결과

Phase correct PWM의 동영상 데모를 찍을 때 세그먼트의 Duty 계산은 Fast PWM으로 하고 찍었습니다. 모터 출력은 Phase Correct PWM으로 되어있습니다.

1. OCR0 = 0인 경우(DUTY 0%)

$256 \times 0 = 0 \rightarrow$ Duty가 0일 경우 OCR0도 0.



왼쪽 사진은 Fast PWM인 경우 회로 결과이다. OCR0를 0으로 설정해주었기 때문에, 0%의 Duty가 나온다. Fast PWM인 경우 Impulse의 형태로 파형이 관찰되었다. 이는 0.39%의 Duty비가 존재하기 때문이다. Phase correct PWM인 경우 주기

가 약 2배 정도로 더 크기 때문에 아예 파형이 관찰되지 않았다. Duty는 아래와 같이 계산되었다. Duty 이론식으로도 0이다.

- Fast PWM

$$\text{Duty \%} = \frac{\text{OCR0} + 1}{256} \times 100(\%) = \frac{0 + 1}{256} \times 100(\%) = 0.39(\%)$$

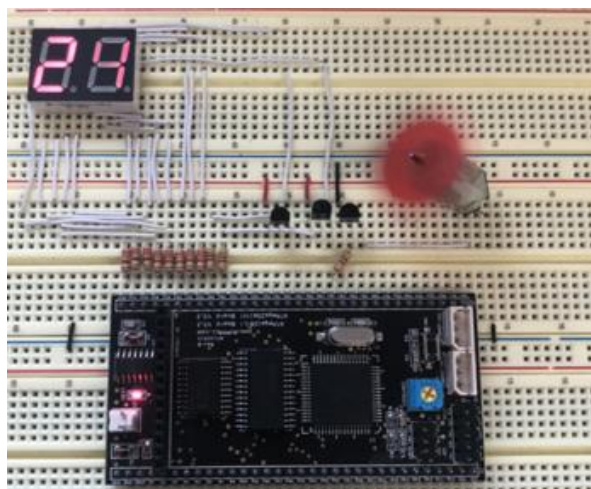
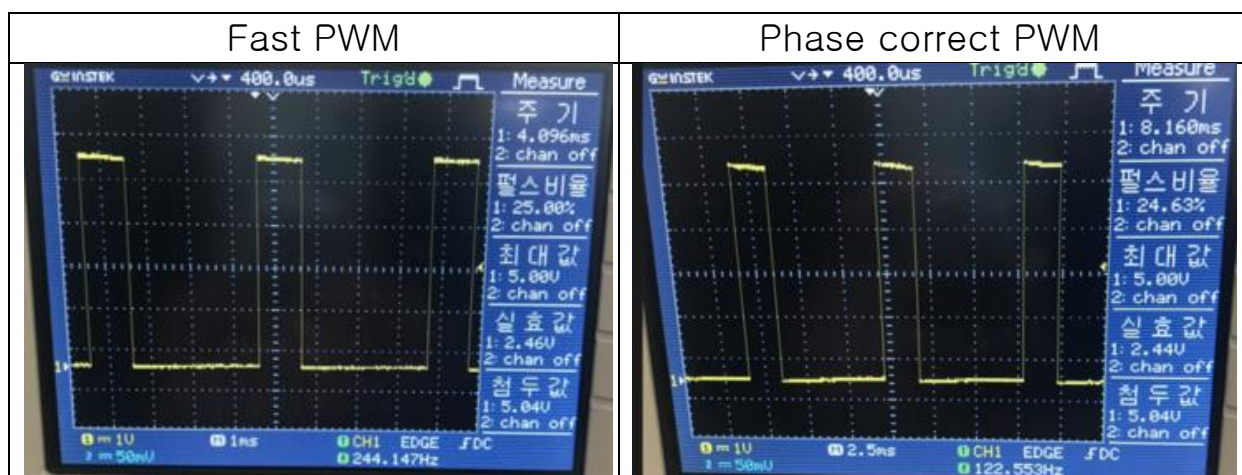
- Phase correct PWM

$$\text{Duty \%} = \frac{\text{OCR0} \times 2}{510} \times 100(\%) = \frac{0 \times 2}{510} \times 100(\%) = 0(\%)$$

2. OCR0 = 63인 경우(DUTY 25%)

$256 \times 0.25 = 64 \rightarrow$ Duty가 25%일 경우 OCR0는 63.

0~255이므로 256 짝수로 계산하여, 64에서 1을 빼주는 형식으로 계산, 결국 255에서 계산하여 반올림하는 효과.



왼쪽 사진은 Fast PWM인 경우 회로 결과이다. OCR0를 63으로 설정해주었기 때문에, 24%의 후반으로 Duty가 나온다. 소수점이 생략되어서 표시되어 24로 표시되었다. 오실로스코프는 기계적으로 더 정확히 계산되어 25%로 잘 나왔다. Phase correct PWM인 경우도 이론식으로부터 결과가 잘 나온 것을 관찰할 수 있다. 이전보다 Duty가 증가했기

때문에 모터도 돌기 시작했음을 관찰할 수 있었다.

- Fast PWM

$$Duty\ \% = \frac{OCR0 + 1}{256} \times 100(\%) = \frac{63 + 1}{256} \times 100(\%) = 25(\%)$$

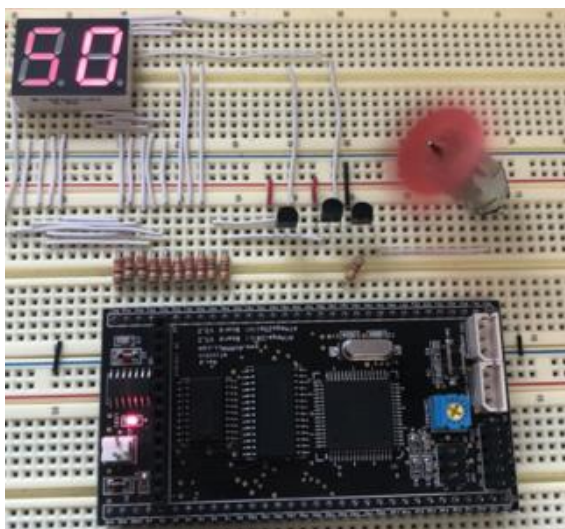
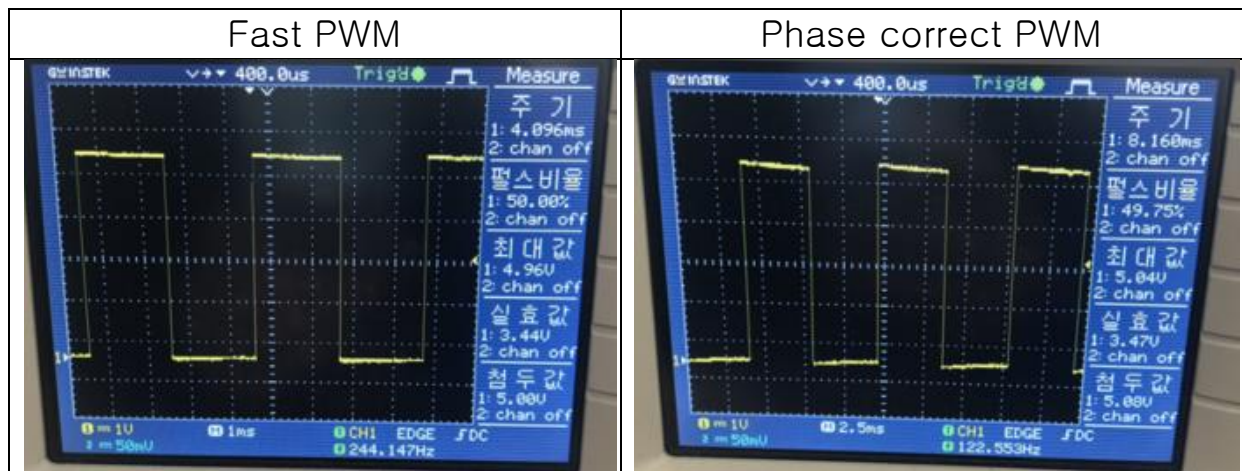
- Phase correct PWM

$$Duty\ \% = \frac{OCR0 \times 2}{510} \times 100(\%) = \frac{63 \times 2}{510} \times 100(\%) = 24.7(\%)$$

3. OCR0 = 127인 경우(DUTY 50%)

$256 \times 0.5 = 128 \rightarrow$ Duty가 50%일 경우 OCR0는 127.

0~255이므로 256 짝수로 계산하여, 128에서 1을 빼주는 형식으로 계산, 결국 255에서 계산하여 반올림하는 효과.



왼쪽 사진은 Fast PWM인 경우 회로 결과이다. OCR0를 127로 설정해주었기 때문에, 50%의 Duty가 나온다. 오실로스코프는 기계적으로 더 정확히 계산되어 50%로 잘 나왔다. Phase correct PWM인 경우도 이론식으로부터 결과가 잘 나온 것을 관찰할 수 있다. 이전보다 Duty가 증가했기 때문에 모터가 더 빨리 돌고 있음을 관찰할 수 있었다.

- Fast PWM

$$Duty\ \% = \frac{OCR0 + 1}{256} \times 100(\%) = \frac{127 + 1}{256} \times 100(\%) = 50(\%)$$

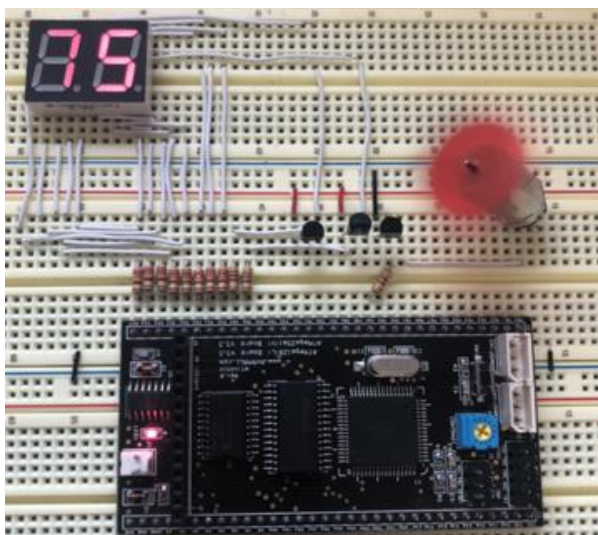
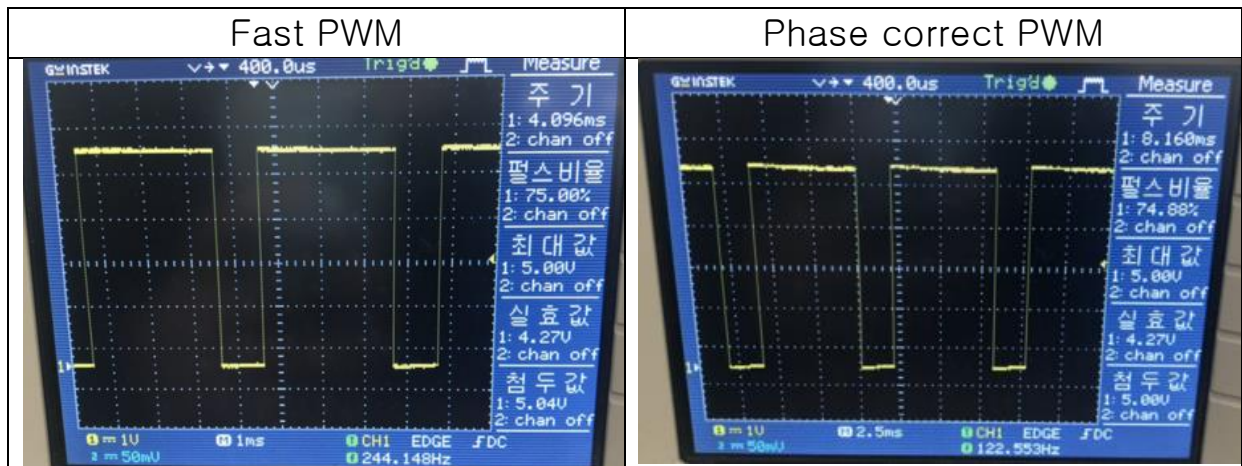
- Phase correct PWM

$$Duty \text{ \%} = \frac{OCR0 \times 2}{510} \times 100(\%) = \frac{127 \times 2}{510} \times 100(\%) = 49.8 (\%)$$

4. OCR0 = 191인 경우(DUTY 75%)

256 × 0.75 = 192 → Duty가 75%일 경우 OCR0는 191.

0~255이므로 256 짝수로 계산하여, 192에서 1을 빼주는 형식으로 계산, 결국 255에서 계산하여 반올림하는 효과.



왼쪽 사진은 Fast PWM인 경우 회로 결과이다. OCR0를 191로 설정해 주었기 때문에, 75%의 Duty가 나온다. 오실로스코프는 기계적으로 더 정확히 계산되어 75%로 잘 나왔다. Phase correct PWM인 경우도 이론식으로부터 결과가 잘 나온 것을 관찰할 수 있다. 이전보다 Duty가 증가했기 때문에 모터가 더 빨리 돌고 있음을 관찰할 수 있었다.

- Fast PWM

$$Duty \text{ \%} = \frac{OCR0 + 1}{256} \times 100(\%) = \frac{191 + 1}{256} \times 100(\%) = 75 (\%)$$

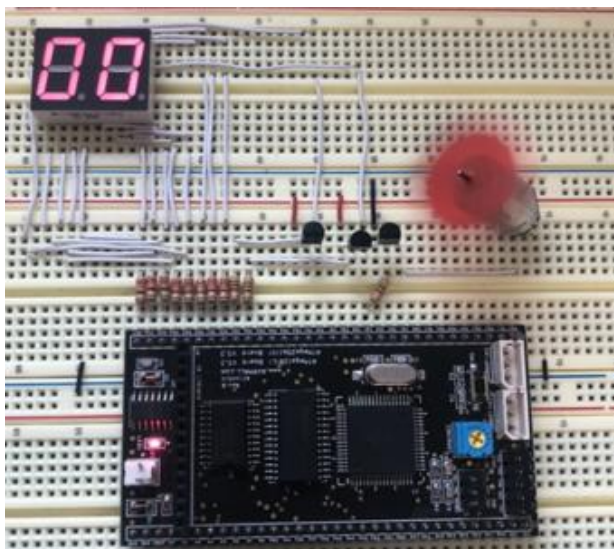
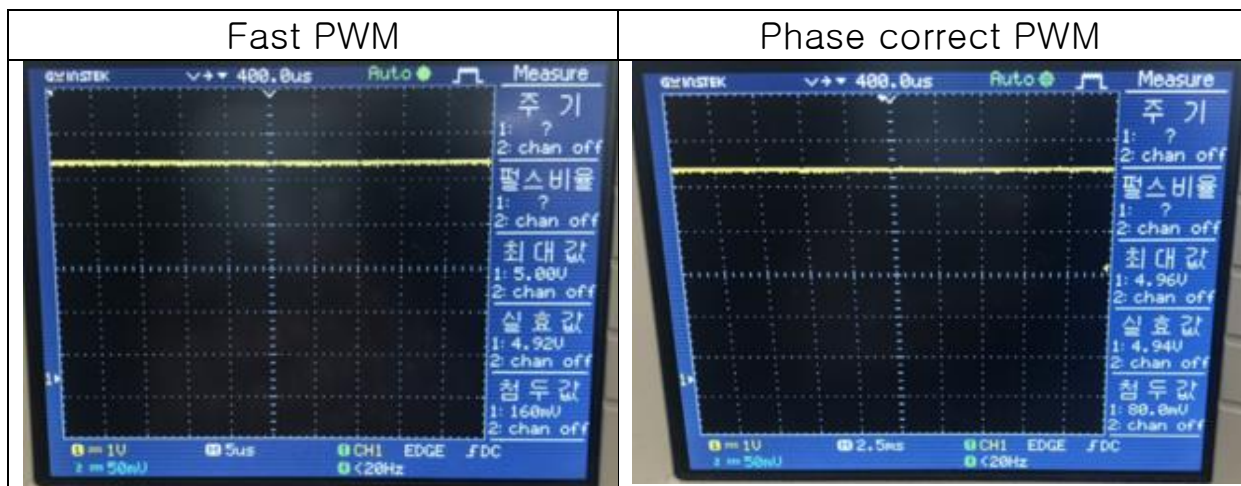
- Phase correct PWM

$$Duty \% = \frac{OCR0 \times 2}{510} \times 100(\%) = \frac{191 \times 2}{510} \times 100(\%) = 74.9(\%)$$

5. OCR0 = 255인 경우(DUTY 100%)

$256 \times 1 = 256 \rightarrow$ Duty가 100%일 경우 OCR0는 255.

0~255이므로 256 짝수로 계산하여, 256에서 1을 빼주는 형식으로 계산, 결국 255에서 계산하여 반올림하는 효과.



왼쪽 사진은 Fast PWM인 경우 회로 결과이다. OCR0를 255로 설정해주었기 때문에, 100%의 Duty가 나온다. 오실로스코프에서는 100%이므로 계속 High인 상태로 관찰되었다.

Phase correct PWM인 경우도 마찬가지이다. 이전보다 Duty가 증가했기 때문에 모터가 더 빨리 돌며, 최대 속도로 돈다.

- Fast PWM

$$Duty \% = \frac{OCR0 + 1}{256} \times 100(\%) = \frac{255 + 1}{256} \times 100(\%) = 100(\%)$$

- Phase correct PWM

$$Duty\ \% = \frac{OCR0 \times 2}{510} \times 100(\%) = \frac{255 \times 2}{510} \times 100(\%) = 100(\%)$$

2. 추가 실험

(1) COM00과 COM01 변경하기

2-1. 실험 회로 → 본 실험과 회로가 동일합니다.

2-2. 프로그램 분석 및 결과

이전 실험까지는 Fast PWM모드에서 COM00과 COM01을 다음과 같이 설정해주었다. TCNT0 0xFF에서 1, 상승 도중 OCR0와 일치하면 0으로 clear된다.

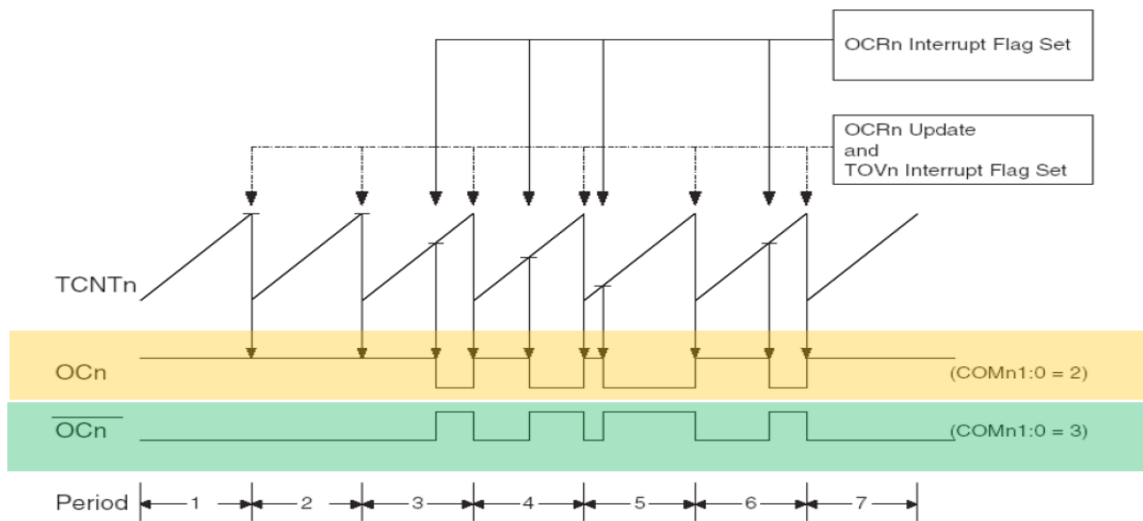
Table 54. Compare Output Mode, Fast PWM Mode⁽¹⁾

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Reserved
1	0	Clear OC0 on compare match, set OC0 at TOP
1	1	Set OC0 on compare match, clear OC0 at TOP

이번 실험에서는 COM00과 COM01을 다음과 같이 설정해주어 실험을 해볼 것이다. 둘 다 1, 1로 설정해주면 TCNT0 0xFF에서 0, OCR0와 일치하면 1로 SET된다.

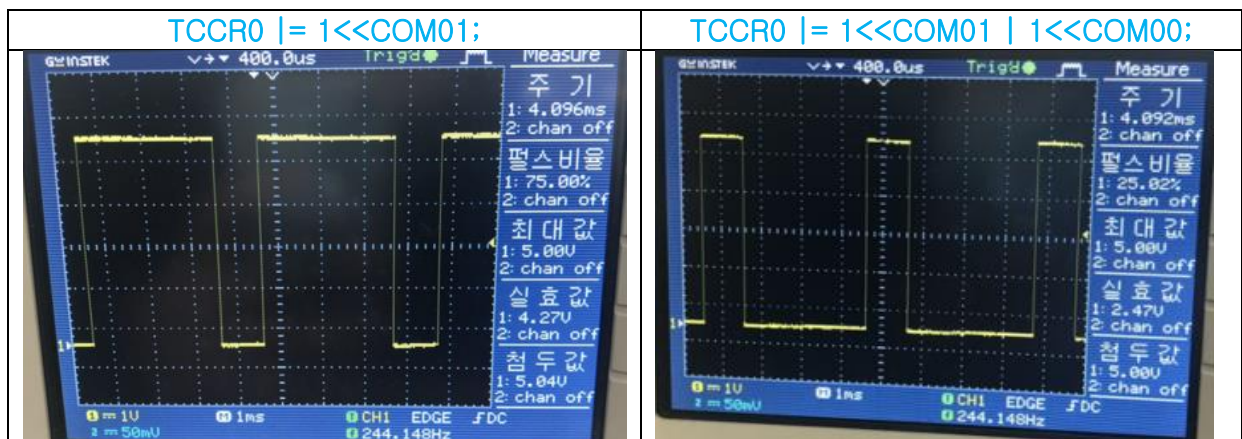
Table 54. Compare Output Mode, Fast PWM Mode⁽¹⁾

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Reserved
1	0	Clear OC0 on compare match, set OC0 at TOP
1	1	Set OC0 on compare match, clear OC0 at TOP



이전 설정과 반대로 Set되므로, 예상하는 실험 결과는 파형과, Duty가 반전하여 나올 것이다. 아래 실험은 Fast PWM일 경우, 256분주일 경우로 동일한 조건하에 실시되었다.

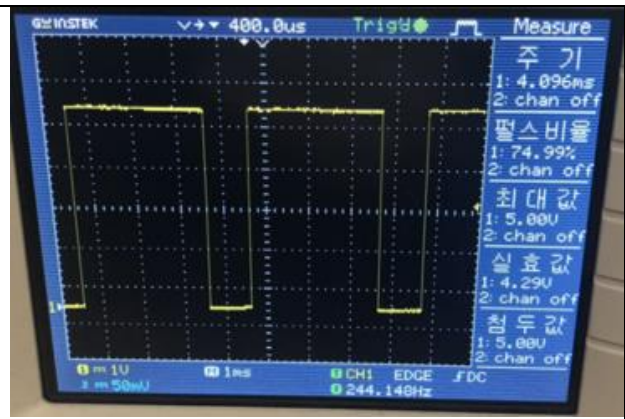
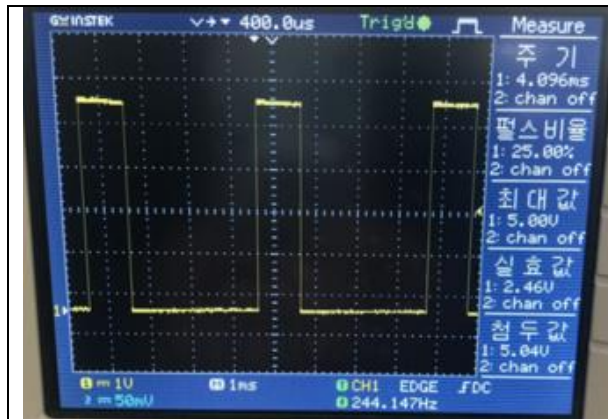
1. OCR0 가 191일 때 비교



왼쪽의 사진은 추가 실험(1)에서 나온 결과이다. 오른쪽 사진은 $1 \ll COM01 | 1 \ll COM00$ 로 설정을 해주었다. 실제 예상과 같게 Duty는 75%에서 반대로 25%로 관찰되었고, 주기는 같은 256분주이므로 앞 실험에서 계산한 바와 같이 알맞게 값이 측정되었다. 파형을 보아서도 넓은 폭의 파형부분이 좁은 폭의 파형 부분이 반대로 뒤집혀서 나온 결과를 볼 수 있다. 왼쪽 사진의 넓은 폭의 HIGH가 오른쪽 사진의 넓은 폭의 LOW로 나타났다.

2. OCR0 가 63일 때 비교

$TCCR0 = 1 \ll COM01;$	$TCCR0 = 1 \ll COM01 1 \ll COM00;$
-------------------------	---------------------------------------



왼쪽의 사진은 추가 실험(1)에서 나온 결과이다. 오른쪽 사진은 1<<COM01 | 1<<COM00 로 설정을 해주었다. 이 실험도 위의 실험 결과와 같이 예상대로 파형이 관찰되었다. Duty는 25%에서 반대로 75%로 관찰되었고, 주기는 같은 256분주이므로 앞 실험에서 계산한 바와 같이 알맞게 값이 측정되었다. 파형을 보아서도 넓은 폭의 파형부분이 혹은 좁은 폭의 파형 부분이 반대로 뒤집혀서 나온 결과를 볼 수 있다. 왼쪽 사진 넓은 폭의 LOW가 오른쪽 사진 넓은 폭의 HIGH로 나타났다.