실험제목 : 타이머/카운터 II (Pulse Width Modulation, PWM)

실험목적

ATmega328PB에 내장된 타이머/카운터(Timer/Counter)의 PWM(Pulse Width Modulation) 기능에 대해 이해하고 이를 활용한 각종 직류 제어 장치의 구동 방법에 대해 알아본다.

실험 준비물

Microchip Studio 7

Atmega328PB Xplained Mini

LED

DC motor

실험에 필요한 예비지식

1. DC Motor

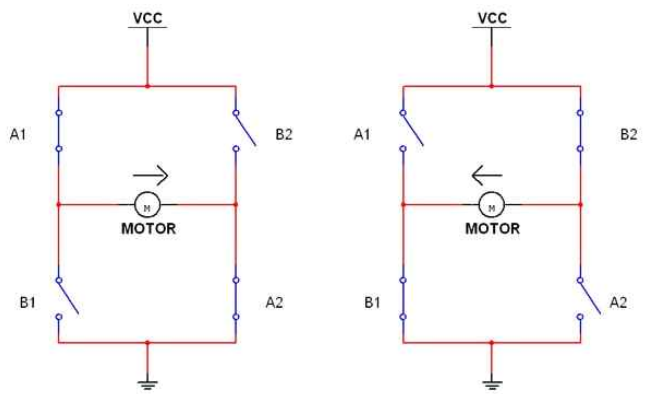
고정자로 영구자석을 사용하고, 회전자(전기자)로 코일을 사용하여 구성한 것으로, 전기자에 흐르는 전류의 방향을 전환함으로 자력의 반발, 흡인력으로 회전력을 생성시키는 모터

2. 모터 제어

1) 방향 제어

① 회로구성

모터의 정회전 역회전 방향은 Full Bridge회로를 구성하여 조절이 가능하다. Full Bridge회로는 그 모양이 알파벳 'H'와 비슷하다고 하여 H-Bridge라고도 불린다. 보통 트랜지스터를 사용하여 전류의 방향을 조절한다.



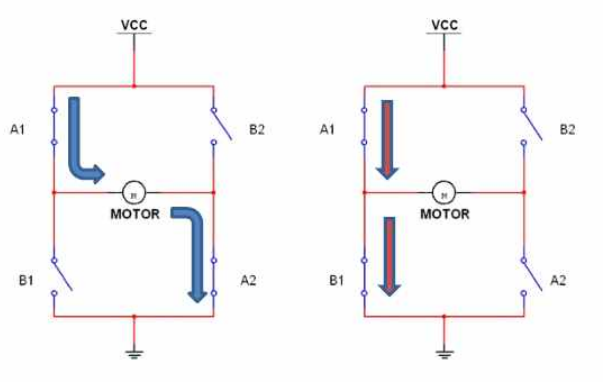
정회전 역회전

<그림 7-1> 모터의 방향 제어 방식

- 정회전 : A1과 A2의 스위치가 닫히고 B1과 B2의 스위치가 열리면 전류는 모터의 왼쪽에서 오른쪽으로 흐르게 되어 정회전 하게 됨
- 역회전 : A1과 A2의 스위치가 열리고 B1과 B2의 스위치가 닫히면 전류는 모터의 오른쪽에서 왼쪽으로 흐르게 되어 역회전 하게 됨

② Dead time

모터의 방향을 조절하기 위한 H-Bridge에는 트랜지스터가 사용되며, 모터의 회전 방향을 바꿀 때는 Dead time이라는 시간을 확보해 주어야 한다. Dead time이란 A1과 A2가 닫히는 모터의 정회전 동안의 시간과 B1과 B2가 닫히는 모터의 역회전 동안의 시간 사이를 뜻한다. 이 시간을 확보해 주어야 트랜지스터의 손상을 방지할 수 있다.



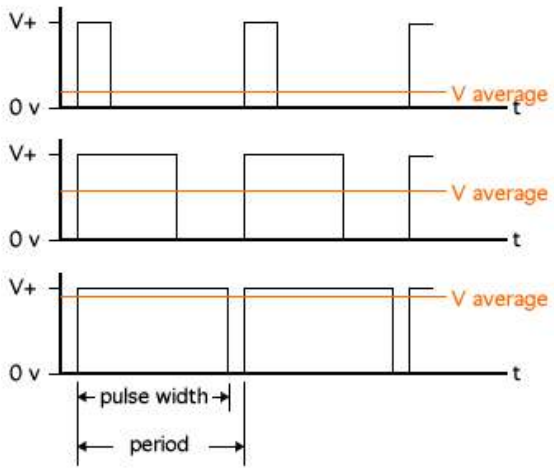
<그림 7-2> Dead time 미확보로 인한 회로 손상

2) 속도 제어

모터의 속도를 조절하는 방법에는 전류를 가변하여 조절하는 방식과 펄스의 폭을 가변하여 조절하는 방식이 있다. 처음 모터의 구동을 위해서는 많은 전류가 필요하기 때문에 보통은 펄스의 폭을 조절하는 방식으로 모터를 제어한다. 이 방식을 펄스폭변조방식(PWM)이라고 한다.

① PWM의 원리

PWM은 프로세서의 디지털 출력으로 아날로그 회로를 제어하는 강력한 기법이다. PWM은 모터 제어뿐만 아니라 계측과 통신에서 전력제어와 전력 변화에 이르기까지 광범위한 영역에서 사용되고 있다. PWM은 Pulse Width Modulation의 약자로 펄스폭변조라고 한다. 이는 ON과 OFF의 비율 즉, 펄스폭을 변화시켜서 제어하는 방법으로 평균전압을 조절할 수 있다. 조절된 평균전압으로 인해 모터의 속도를 조절할 수 있다. <그림 7-3>은 펄스폭에 따른 각각의 평균전압을 보여준다.



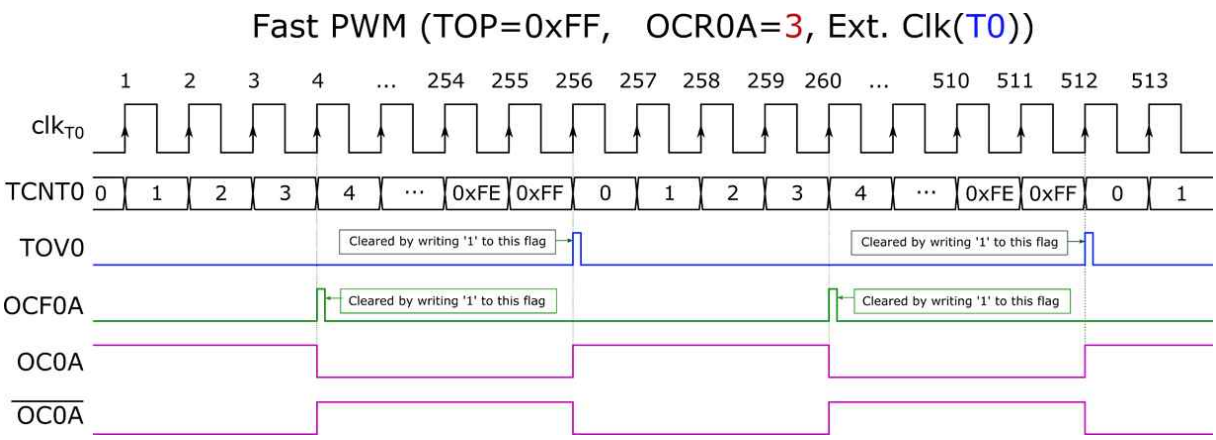
<그림 7-3> 펄스폭에 따른 평균전압의 변화

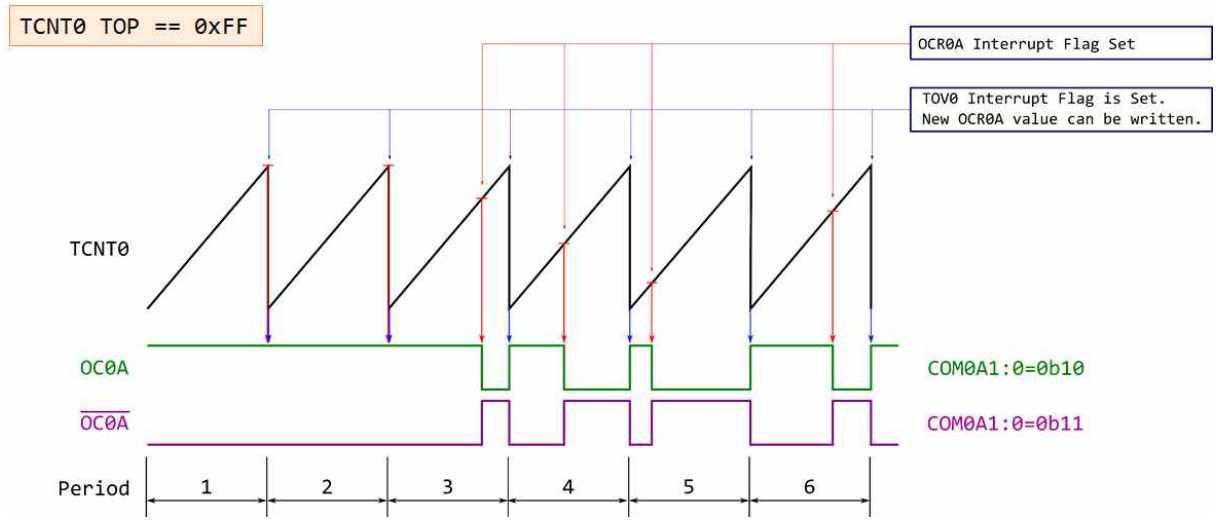
3. Timer/Counter에 의한 PWM 신호 생성

ATmega328PB에는 5개의 타이머/카운터가 내장되어 있으며 각각의 타이머/카운터는 PWM 신호를 생성시키는 기능을 가지고 있다.

1) Timer/Counter0에 의한 PWM

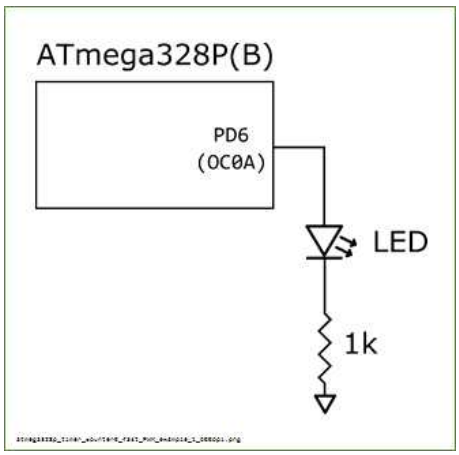
- WGM[02:00]=0b011 or WGM[02:00]=0b111
- The counter(TCNT0) counts from BOTTOM to TOP, then restarts from BOTTOM.
  - ✓ BOTTOM is always 0.
  - ✓ TOP is
    - 0xFF when WGM[02:00]=0b011.
    - OCR0A when WGM[02:00]=0b111.
- The Timer/Counter Overflow Flag (TOV0) is set each time the counter reaches TOP.
  - ✓ If the interrupt is enabled, the interrupt handler routine(TIMER0_OVF_vect) can be used for updating the compare value.
- Two types of PWM signals can be available
  - ✓ Invert: COM0A[1:0]=0b11
  - ✓ Non-invert: COM0A[1:0]=0b10


Fast PWM (TOP=0xFF, OCR0A=3, Ext. Clk(T0))
1 2 3 4 ... 254 255 256 257 258 259 260 ... 510 511 512 513
clkT0
TCNT0
0 1 2 3 4 ⋯ 0xFE 0xFF 0 1 2 3 4 ⋯ 0xFE 0xFF 0 1
TOV0
Cleared by writing '1' to this flag
Cleared by writing '1' to this flag
OCF0A
Cleared by writing '1' to this flag
Cleared by writing '1' to this flag
OC0A
OC0A

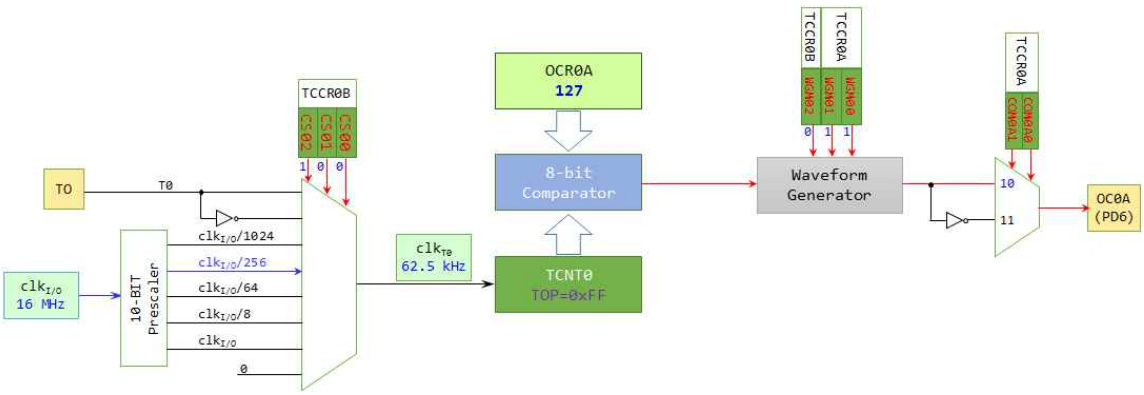

TCNT0 TOP == 0xFF
OCR0A Interrupt Flag Set
TOV0 Interrupt Flag is Set.
New OCR0A value can be written.
TCNT0
OC0A
COM0A1:0=0b10
OC0A
COM0A1:0=0b11
Period
1 2 3 4 5 6

실험 내용

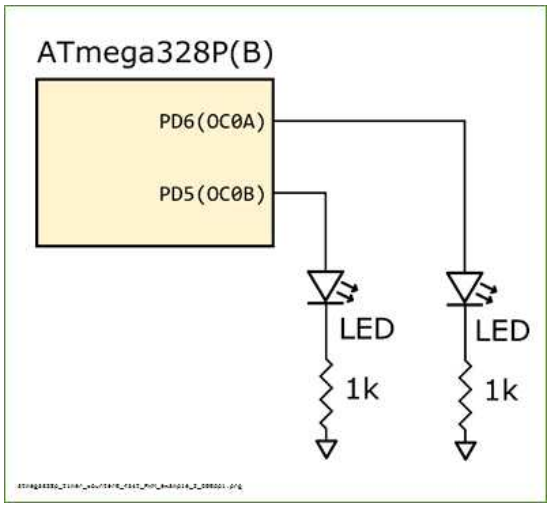
1. (고정된 duty cycle을 가지는 PWM 신호 발생) 아래의 조건에 맞도록 Timer/Counter0를 이용하여 PD6(OC0A) 핀에 연결된 LED의 밝기를 변화시킬 수 있는 PWM 신호를 얻는 프로그램을 작성하고 그 결과를 oscilloscope를 통해 확인하시오.
   (1) 시스템 클럭 주파수: 16 MHz
   (2) PWM 신호의 duty cycle: 50%로 고정
   (3) PWM 주파수 (LED refresh rate): 244.14Hz
   (4) 8-bit Fast PWM 기능 사용

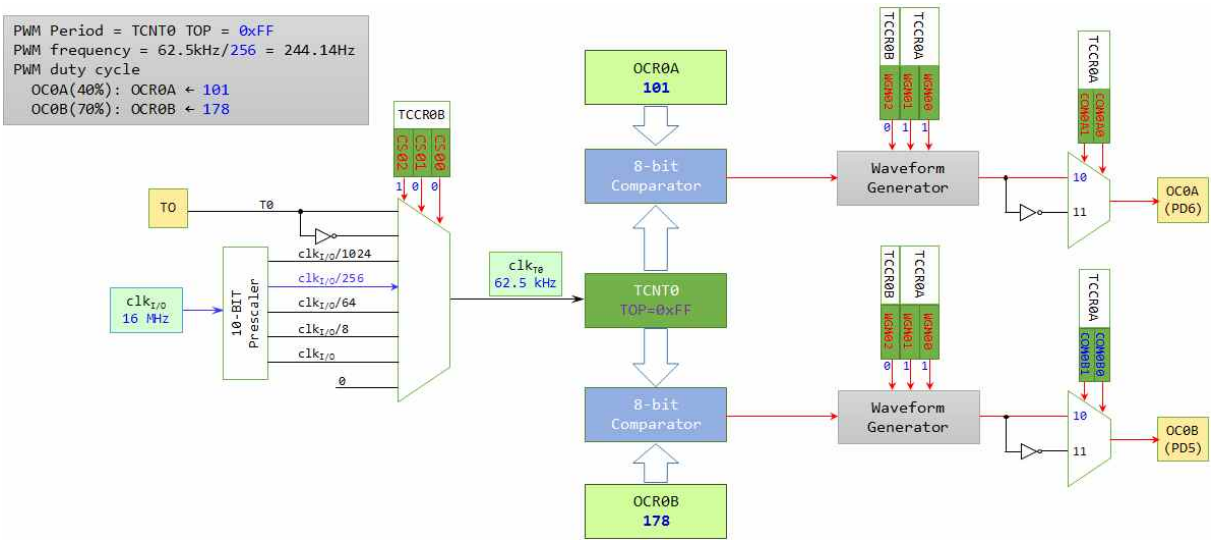



```
TCNT0 TOP = 0xFF
PWM frequency = 62.5kHz/256 = 244.14Hz
PWM duty cycle = 50% ➡ OCR0A ← 127
```

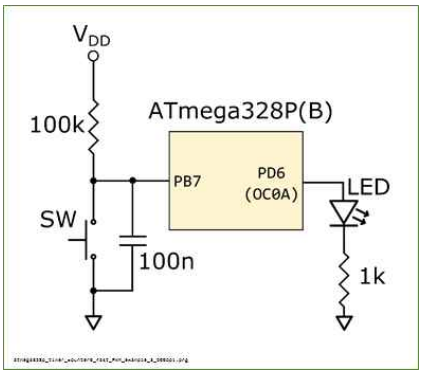


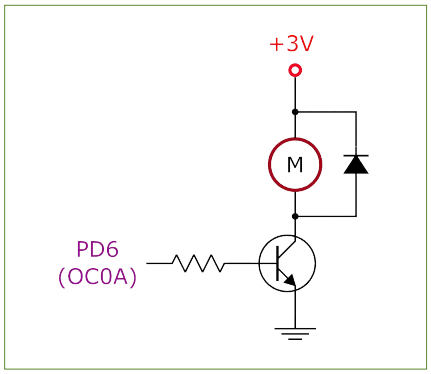
2. (고정된 duty cycle을 가지는 PWM 신호 발생) 아래의 조건에 맞도록 Timer/Counter0를 이용하여 PD6(OC0A) 핀과 PD5(OC0B) 핀에 각각 연결된 두 개의 LED의 밝기를 변화시킬 수 있는 PWM 신호를 얻는 프로그램을 작성하고 그 결과를 oscilloscope를 통해 확인하시오.
   (1) 시스템 클럭 주파수: 16 MHz
   (2) PWM 신호의 duty cycle: OC0A:40%, OC0B: 70%.
   (3) PWM 주파수 (LED refresh rate): 244.14Hz
   (4) 8-bit Fast PWM 기능 사용



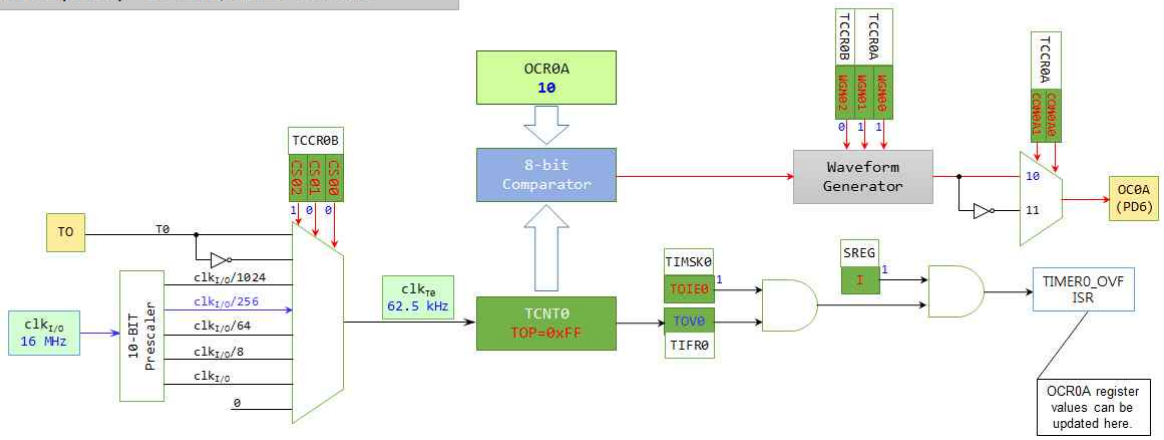


3. PB7에 연결된 스위치를 누를 때마다 Timer/Counter0의 duty cycle을 바꾸어 PD6(OC0A) 핀에 연결된 LED의 밝기를 변화시킬 수 있는 PWM 신호를 얻는 프로그램을 작성하고 그 결과를 oscilloscope를 통해 확인하시오.
   (1) 시스템 클럭 주파수: 16 MHz
   (2) PWM 신호의 duty cycle: 가변.
      (가) 초기 duty cycle은 10%로 시작
      (나) PB7에 연결된 스위치를 누를 때마다 duty cycle을 10%씩 증가
      (다) duty cycle이 100%가 넘으면 10%로 roll over.
   (3) PWM 주파수 (LED refresh rate): 244.14Hz
   (4) 8-bit Fast PWM 기능 사용
   (5) LED의 밝기 변화 실험이 끝나면 아래의 오른쪽 회로와 같이 LED와 저항을 직류 모터로 대치하여 모터의 속도가 바뀌는지를 확인.

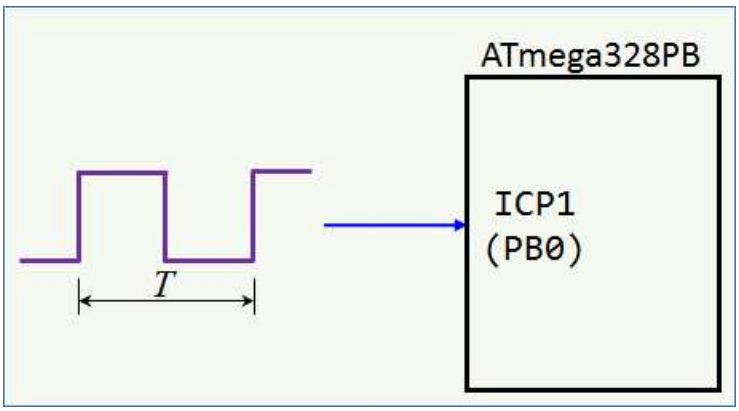


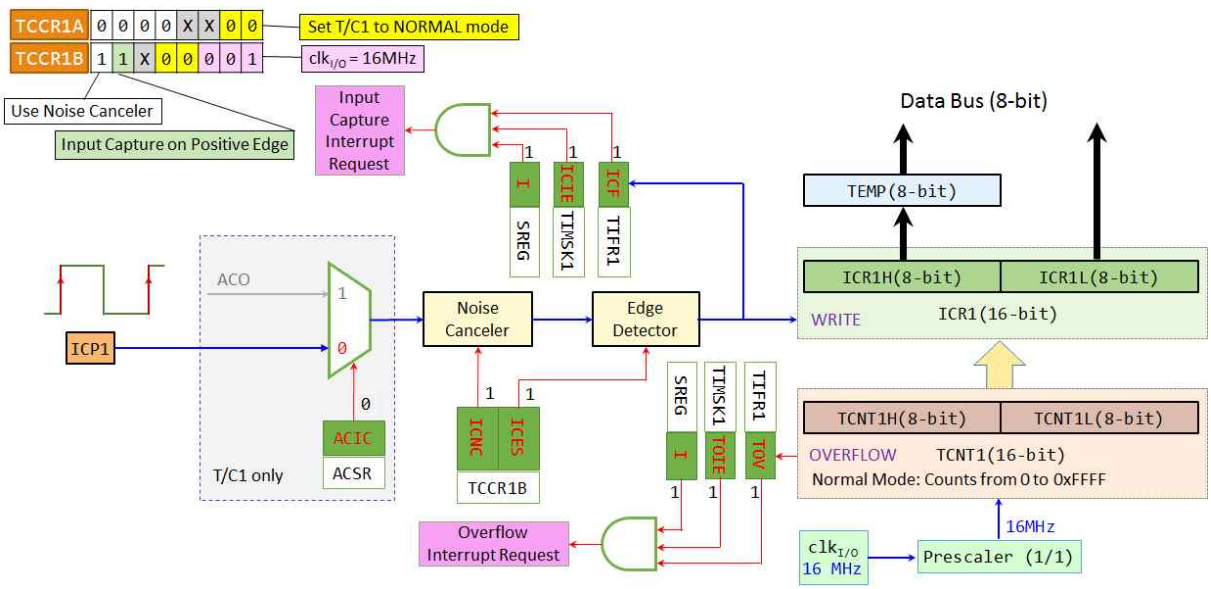


TCNT0 TOP = 0xFF
PWM frequency =62.5kHz/256=244.14Hz



4. ATmega328PB의 ICP1(PB0) 핀으로 입력되는 펄스 신호의 주기를 측정하여 USART0 port를 통해 컴퓨터의 터미널 창에 출력하는 프로그램을 작성하시오.
   (1) 시스템 클럭 주파수: 16 MHz
   (2) Timer/Counter1의 Input Capture 기능 사용
   (3) 주어지는 USART 관련 라이브러리를 프로젝트에 포함시켜 사용.
   (4) PC측에는 적절한 터미널 프로그램을 설치하여 사용. (예: TeraTerm)
   (5) 6주차 실험 내용 2의 방법을 사용하여 PB5로 1 Hz의 주파수를 가지는 구형파 신호를 출력한 후, 이 신호를 입력으로 사용한다.





usart0.c

```
#define F_CPU   16000000UL

#include <avr/io.h>
#include <stdio.h>

int uart0_putchar(char ch, FILE *stream);
int uart0_getchar(FILE *stream);

FILE uart0_dev = FDEV_SETUP_STREAM(uart0_putchar, uart0_getchar, _FDEV_SETUP_RW);

void uart0_init(uint32_t baudrate)
{
        UCSR0A = 0b00000000;    // U2X0=0: No double speed
        UCSR0B = 0b00011000;    // Enable Rx and Tx
        UCSR0C = 0b00000110;    // Async mode, No Parity,
                                                    // 1 Stop bit, 8 Data bits
        UBRR0  = F_CPU/(baudrate*16UL)-1;      // Baud Rate

        stdout = &uart0_dev;
        stdin = &uart0_dev;
}

int uart0_putchar(char ch, FILE *stream)
{
        while (!(UCSR0A & (1 << UDRE0)));      // Wait for empty transmit buffer
        UDR0 = ch;
        return 0;                                                   // sends the data
}

int uart0_getchar(FILE *stream)
{
        // Wait until data is received (RXC0 bit)
        while ((UCSR0A & (1 << 7)) == 0);

        return UDR0;    // Read received data
}
```

main.c

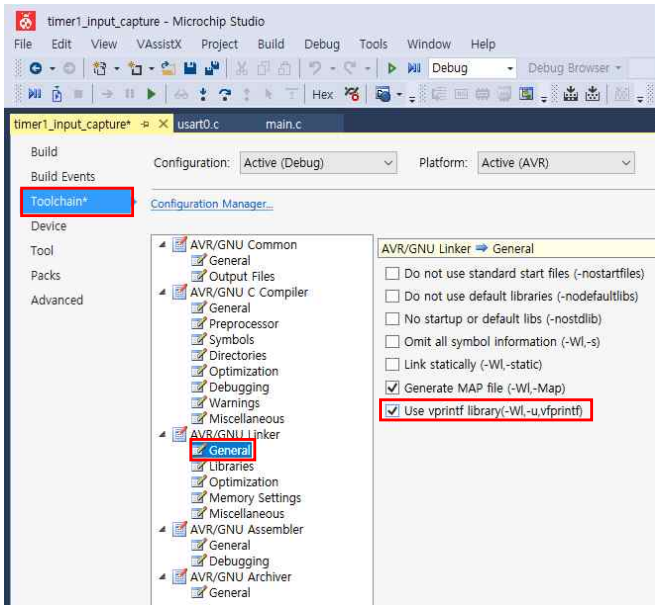
```
#include <avr/io.h>
#include <stdio.h>

void uart0_init(uint32_t baudrate);

int main(void)
{
        uart0_init(9600UL);
        ...
        printf("frequency=%.4f\n", frequency);
        ...
}
```

[setup for printing floating point number in Microchip Studio 7]

- In Project properties

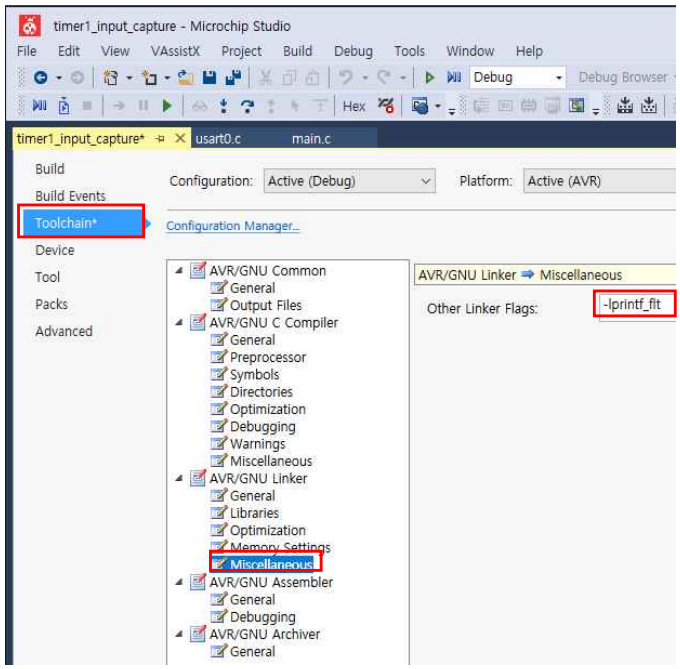
① Toolchain – AVR/GNU Linker – General

Check [Use vprintf library(...)] box.



② Toolchain – AVR/GNU Linker – Miscellaneous

Type -lprintf_flt in the Other Linker Flags



숙제

1. PB7에 연결된 스위치를 누를 때마다 Timer/Counter1의 duty cycle을 바꾸어 OC1A(PB1) 핀에 연결된 LED의 밝기를 변화시킬 수 있는 PWM 신호를 얻는 프로그램을 작성하시오.
   (1) 시스템 클럭 주파수: 16 MHz
   (2) PWM 신호의 duty cycle: 가변.
      (가) 최소값(10%)으로 시작
      (나) PB7에 연결된 스위치를 누를 때마다 duty cycle을 10%씩 증가
      (다) duty cycle이 100%가 넘으면 10%로 roll over.
   (3) PWM 주파수: **400Hz**
   (4) **16-bit Fast PWM** 기능 사용

2. PWM의 응용분야를 조사하여 제출하시오.