AVR – HW6

임베디드스쿨1기
Lv1과정
2020. 10.23
김인겸

1. Timer/Counter복습

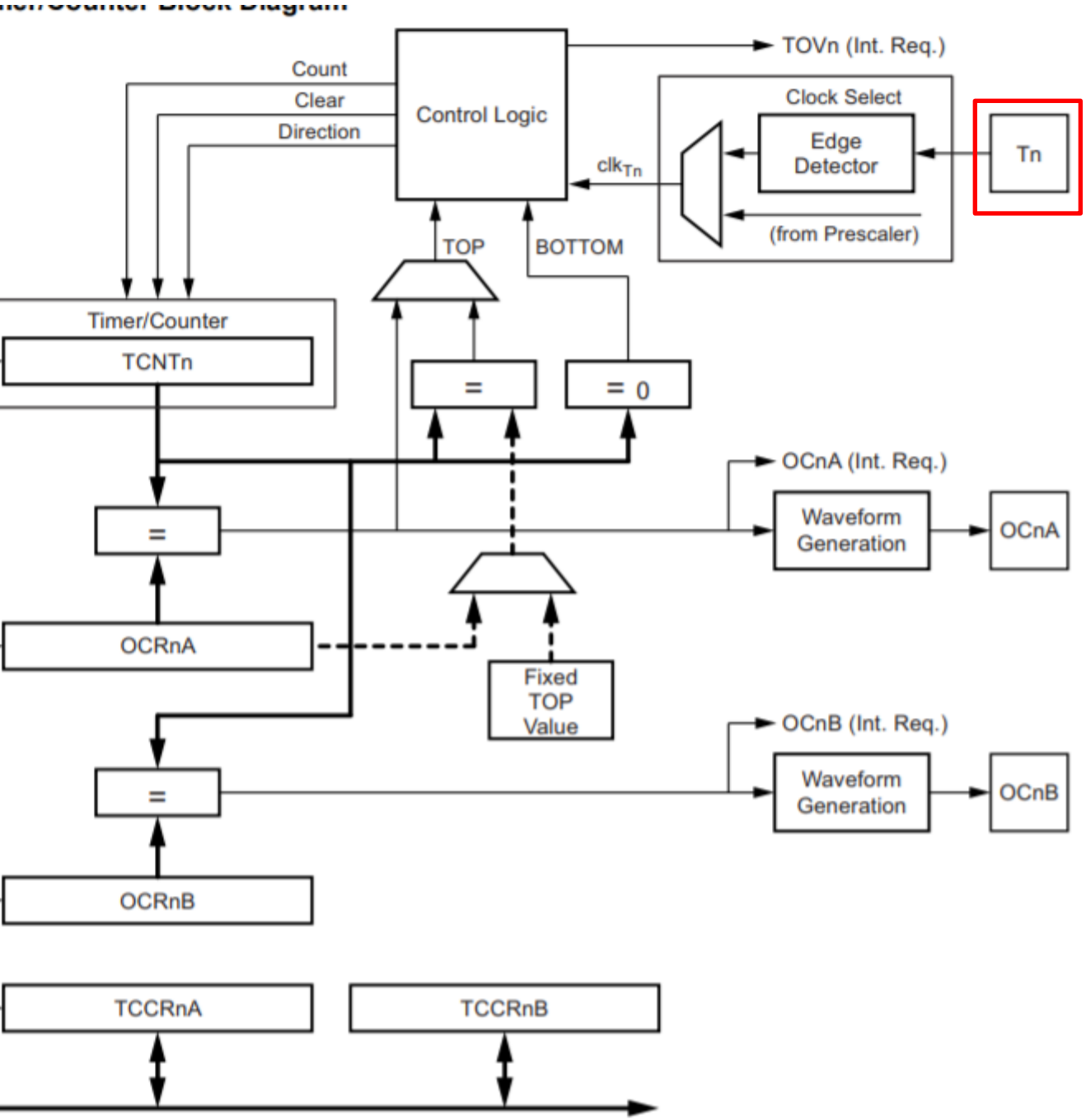
타이머 카운터 모드

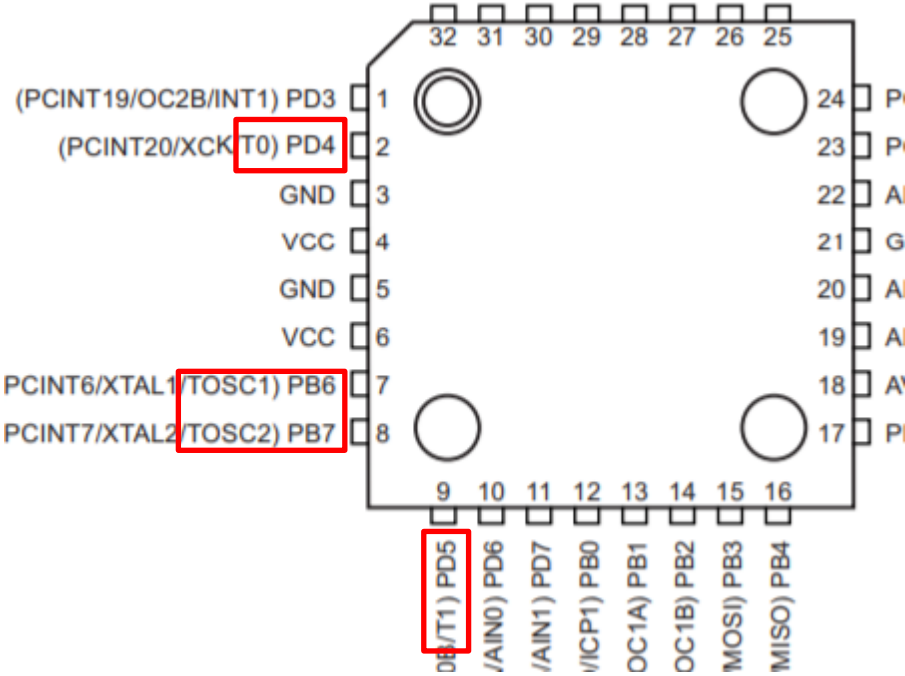
Normal Mode : 순수 Timer.Counter, 파형 생성 x

PWM Mode - Fast PWM : 주기일정, 듀티비 조절
- Phase Correct PWM : 주기조절, 듀티비 조절

CTC Mode : 주기를 조절해서 원하는 주파수를 생성하는데 이용됨

1. Timer/Counter복습





외부 클럭을 인가하는 부분

1. Timer/Counter복습

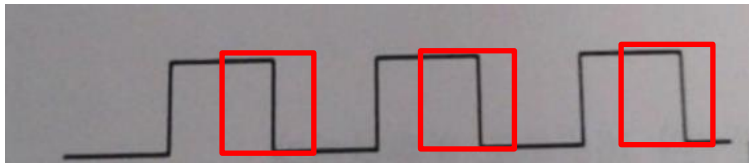
```
void EXT_falling_timer_Init(void); // 사실은 카운터 실습하는 예제임

int main(void)
{
    EXT_falling_timer_Init();

    while (1)
    {
        if(TCNT0 > 5)
        {
            PORTB = 0x10;
        }
    }
}
```

- PD4핀을 스위치에 연결하고 내부 풀업 저항 기능 활성화

- PD4핀은 타이머카운터의 외부 클럭으로 사용됨



스위치 falling edge에서 TCNT값 1증가

```
void EXT_falling_timer_Init(void)
{
    cbi(SREG, 7);
    TCCR0A = 0;
    TCCR0B = 0;
    DDRB = 0xFF;;
    DDRD = 0x00;
    PORTB = 0x00;
    PORTD = 0x10;    //스위치를 통해서 클럭을

    TCCR0B = (1 << CS02) | (1 << CS01);
    sbi(SREG, 7);
}
```

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(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 T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

00/00

1. Timer/Counter복습

```
unsigned char counter = 0;
void EXT_falling_timer_Init(void);
SIGNAL(TIMER0_OVF_vect) // 타이머가 255를 초과하면 발생하는 인터럽트!
{
    if(counter == 125) //8ms가 125번이면 1000ms -> 1초
    {
        PORTB = 0xff; // 불이 깜빡
        counter = 0;  //초기화
    }
    else{
        PORTB = 0x00;
        counter++;
    }
}
int main(void)
{
    EXT_falling_timer_Init();

    while (1)
    {
    }
}
void EXT_falling_timer_Init(void)
{
    cbi(SREG, 7);
    TCCR0A = 0;
    TCCR0B = 0;
    PORTB = 0x00;
    DDRB = 0xFF;
    TCCR0B = (1 << CS02) | (1 << CS00); //101 : 분주비 1024
    TCNT0 = 131; //시작값131~256까지는 8ms
    sbi(TIMSK0, TOIE0); // 오버플로우 인터럽트 활성화
    sbi(SREG, 7);
}
```

CPU클럭 : 16MHz
분주비 : 1024

타이머 클럭 주파수
=16MHz/1024 = 15.625KHz

타이머 클럭 주기
=1 / 15.625KHz = 64us

TCNT값 131이면 256까지 총 125번 계수

64us * 125 = 8ms

8ms에 한 번씩 오버플로우 인터럽트를 발생시키게끔 설정한 거임.

1. Timer/Counter복습

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

위에서 사용한 인터럽트 이외에도 타이머카운터와 관련된 많은 인터럽트가 존재한다

1. PWM (UART로 받기?)

```c
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>
#include <stdlib.h>

#define sbi(x,y)    (x |= (1<<y))

void UART_init(void);
void TimerCounter_init(void);
unsigned char UART_receive(void);
void UART_transmit(unsigned char data);
void UART_string_transmit(char *string);

SIGNAL(TIMER0_OVF_vect)
{
    char data = TCNT0;
    unsigned char str[10] = {'\0', };
    itoa(data,str,10);
    UART_string_transmit(str);
    UART_string_transmit("\n");
}

SIGNAL(TIMER0_COMPA_vect)
{
    char data = OCR0A;
    unsigned char str[10] = {'\0', };
    itoa(data,str,10);
    UART_string_transmit(str);
    UART_string_transmit("\n");
}
```

```c
int main(void)
{
    UART_init();
    TimerCounter_init();

    while (1)
    {
    }
}

void UART_init(void)
{
    sbi(UCSR0A, U2X0);
    UBRR0H = 0;
    UBRR0L = 207;
    sbi(UCSR0B,RXEN0);
    sbi(UCSR0B,TXEN0);
}

void TimerCounter_init(void)
{
    TCCR0A = 0x83;  //1000 0011 Fast PWM모드 설정
    TCCR0B = 0x85;  //1000 0101 분주비1024 > 16MHz/1024 = 15.625KHz
    TIMSK0 = 0x03;  //오버플로우, 비교일치 인터럽트 Enable
    OCR0A = 128; //듀티비 50%

}
```

1. PWM (UART로 받기?)

```c
unsigned char UART_receive(void)
{
    while(!(UCSR0A & (1<<RXC0)));
    return UDR0;
}

void UART_transmit(unsigned char data)
{
    while(!(UCSR0A & (1<<UDRE0)));
    UDR0 = data;
}

void UART_string_transmit(char *string)
{
    while(*string != '\0')
    {
        UART_transmit(*string);
        string++;
    }
}
```

잘 모르겠어서

PWM설정하고

오버플로우 인터럽트 발생했을 때 TCNT값,
비교일치 인터럽트 발생했을 때 OCR값을
출력하려고 했습니다

빌드안되는데 문제점 찾아보겠습니다 ㅠ