Lecture 08~09

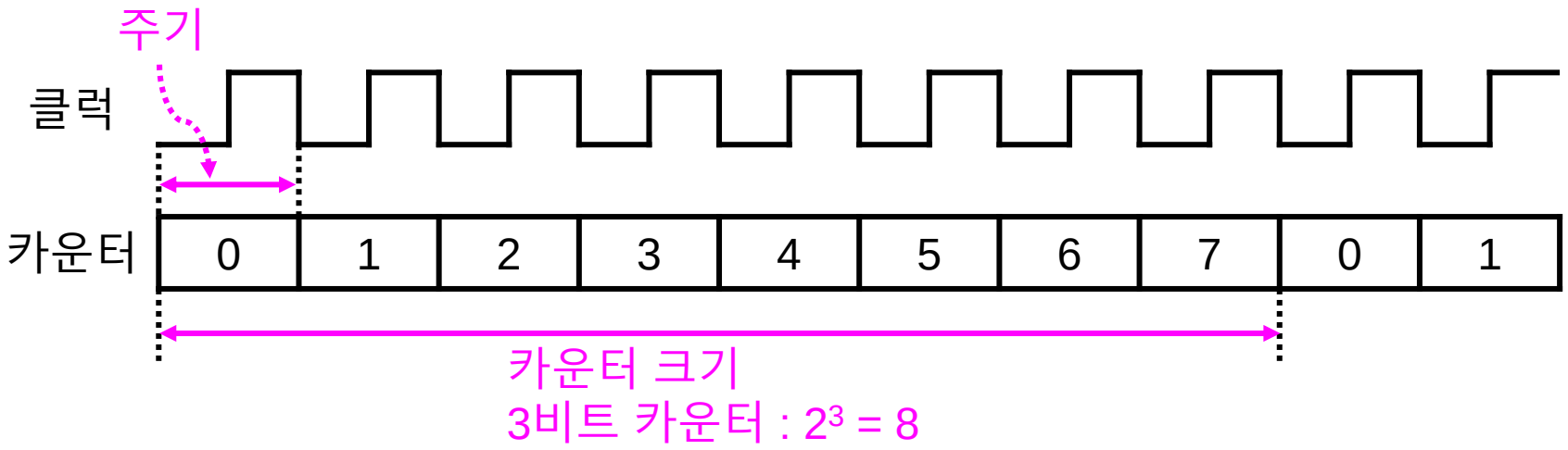
타이머/카운터

타이머/카운터

- 기계 동작에 있어서 정확한 시간과 측정이 필요함

→ 타이머와 카운터 기능을 사용함

- 클럭(clock)과 카운터 개념
  - 클럭 : 일정한 시간 간격으로 0이나 1의 값이 주어지는 신호
  - 카운터 : 클럭을 세는 장치

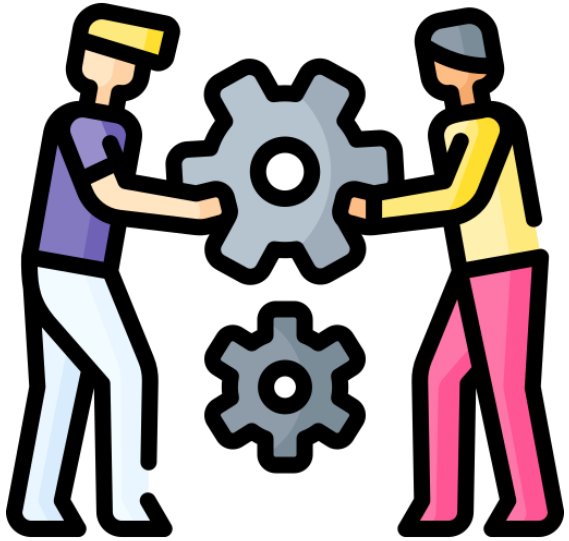


타이머/카운터

- 타이머/카운터 왜 사용?
  - 시간에 의한 일처리를 더 정확하게 할 수 있음
  - CPU를 풀어 줄 수 있음

CPU 단독 일처리

CPU가 타이머/카운터에 시간에 의한 일처리를 맡김



타이머/카운터

- 타이머/카운터 왜 사용?
  - 시간에 의한 일처리를 더 정확하게 할 수 있음
  - CPU를 풀어 줄 수 있음

```
int main(void) {
  DDRA = 0xff;
  while (1) {
    PORTA = 0x07;
    delay(2000);
    PORTA = 0x00;
    delay(2000);
  }
}

void delay(int d) {
  int i;
  for (i=0; i<d; i++) _delay_ms(1);
}
```

delay() 및 _delay_ms() 함수들은 for loop에 기반을 둬서 CPU가 계속 돌여 야 함
→ **고 CPU 사용량**

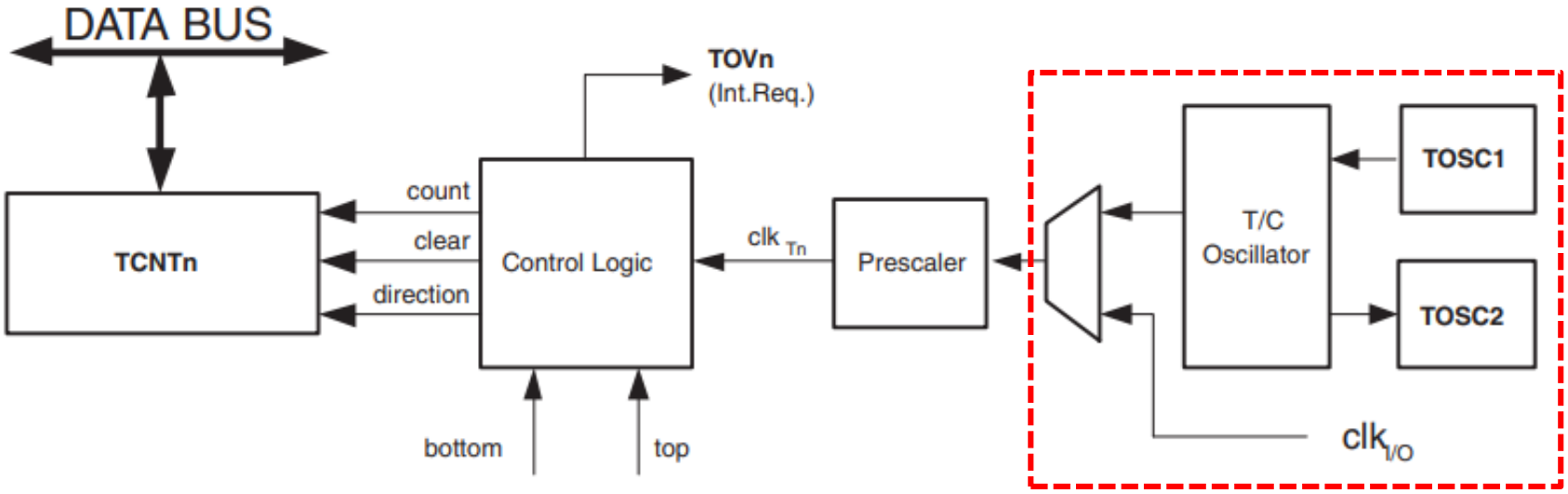
delay() 및 _delay_ms() 함수들을 타이머/카운터에 의해 설계하면 CPU가 일처리를 하지 않은 시간 간격이 생김 → **CPU 사용량을 줄임**
(예, 리눅스 sleep 함수들은 타이머에 기반을 둬서 구현됨)

타이머/카운터

- ATmega128
  - **4개의 타이머/카운터**
    - 타이머/카운터 0, 2 : 8비트 (0부터 255까지 셀 수 있음)
    - 타이머/카운터 1, 3 : 16비트 (0부터 65535까지 셀 수 있음)
  - **인터럽트 기능**
    - 오버플로우 인터럽트 : 카운터의 값이 오버플로우되는 경우 발생
    - 출력비교 인터럽트 : 카운터의 값이 출력비교 레지스터의 값과 같게 되는 순간에 발생
    - 입력 캡쳐 인터럽트 : 외부로부터의 트리거 신호에 의해서 카운터의 초기값을 입력캡쳐
  - **PWM(Pulse Width Modulation) 출력 기능**

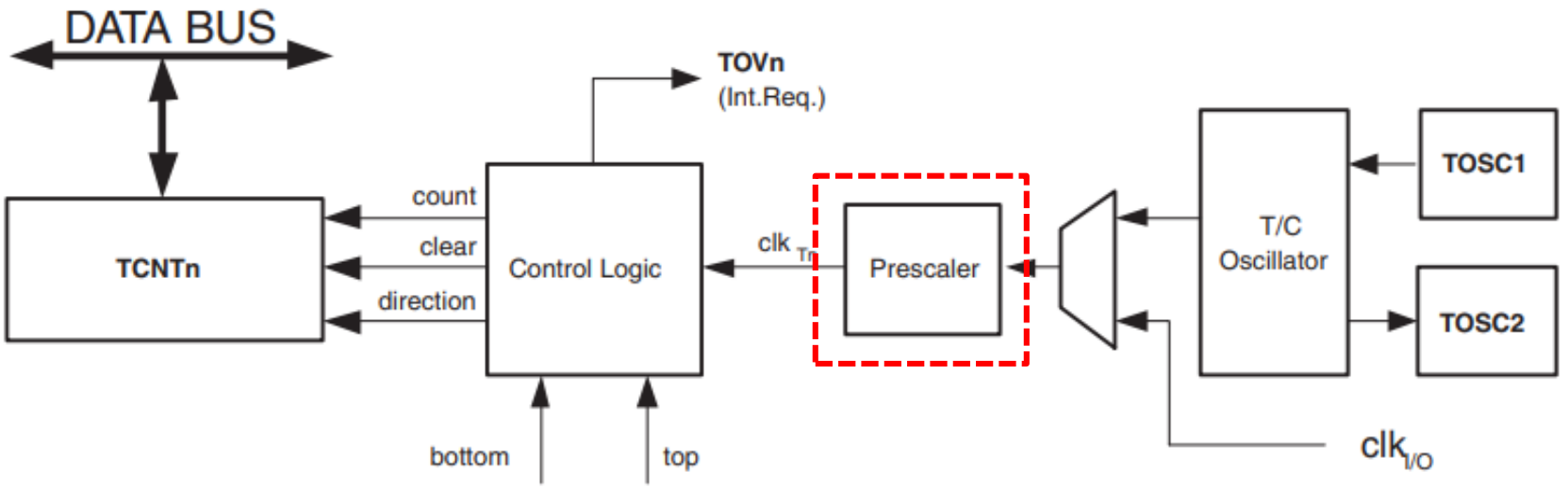
타이머/카운터

- 타이머/카운터 개념도
  - **클럭 소스**
    - $clk_{I/O}$ MCU 클럭(기본 설정)
    - TOSC1(Timer/Counter Oscillator)
    - TOSC2(Timer/Counter Oscillator)



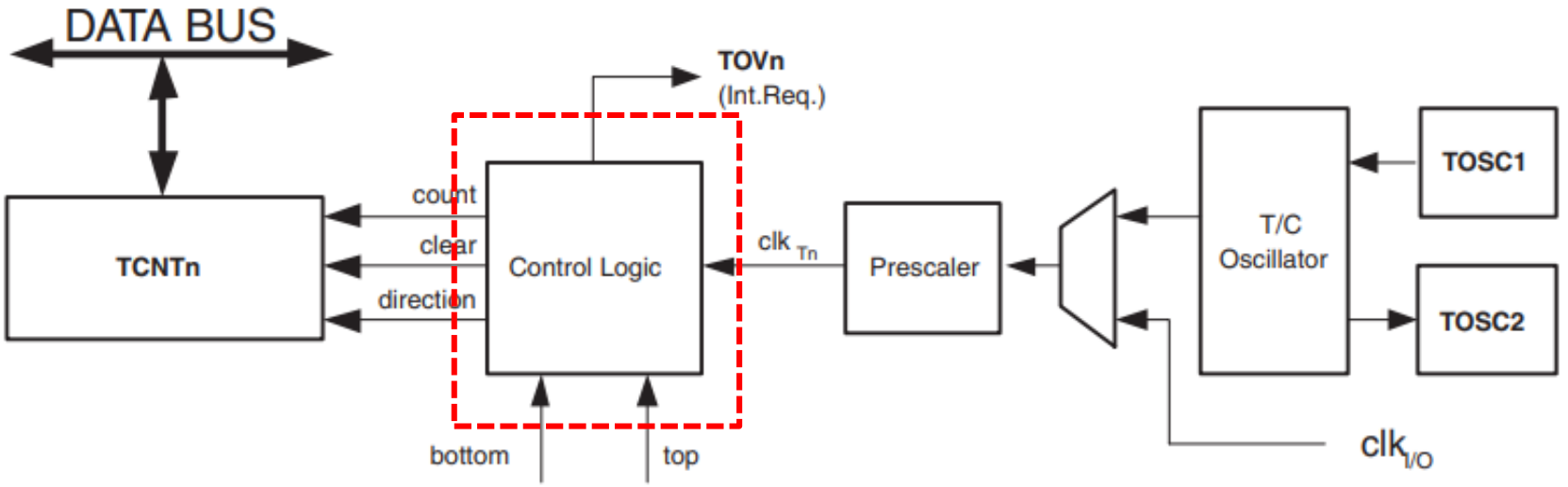
타이머/카운터

- 타이머/카운터 개념도
  - **프리스케일러(Prescaler)**
    - 타이머/카운터에 들어오는 클럭을 분주하는 장치
    - 1/1, 1/8, 1/32, 1/64, 1/128, 1/256, 1/1024까지 분주할 수 있음



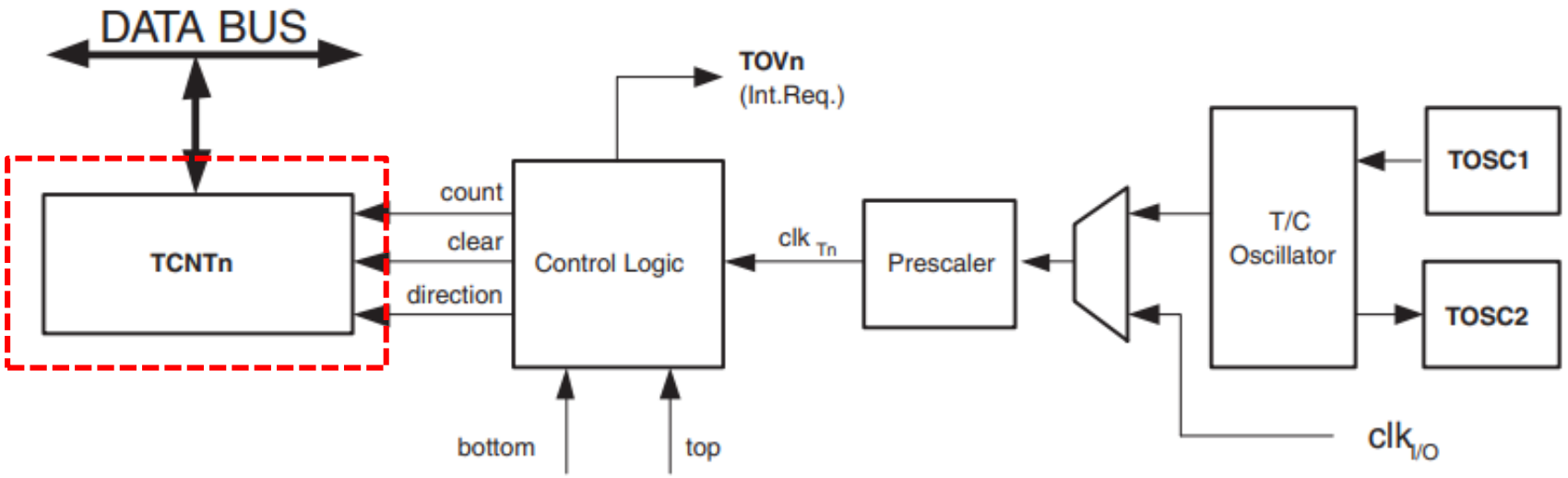
타이머/카운터

- 타이머/카운터 개념도
  - **컨트롤러 로직(Control Logic)**
    - 타이머/카운터 동작 제어
    - 오버플로우 인터럽트 발생



타이머/카운터

- 타이머/카운터 개념도
  - **TCNTn**
    - 클럭을 세는 값을 저장하는 타이머/카운터 레지스터



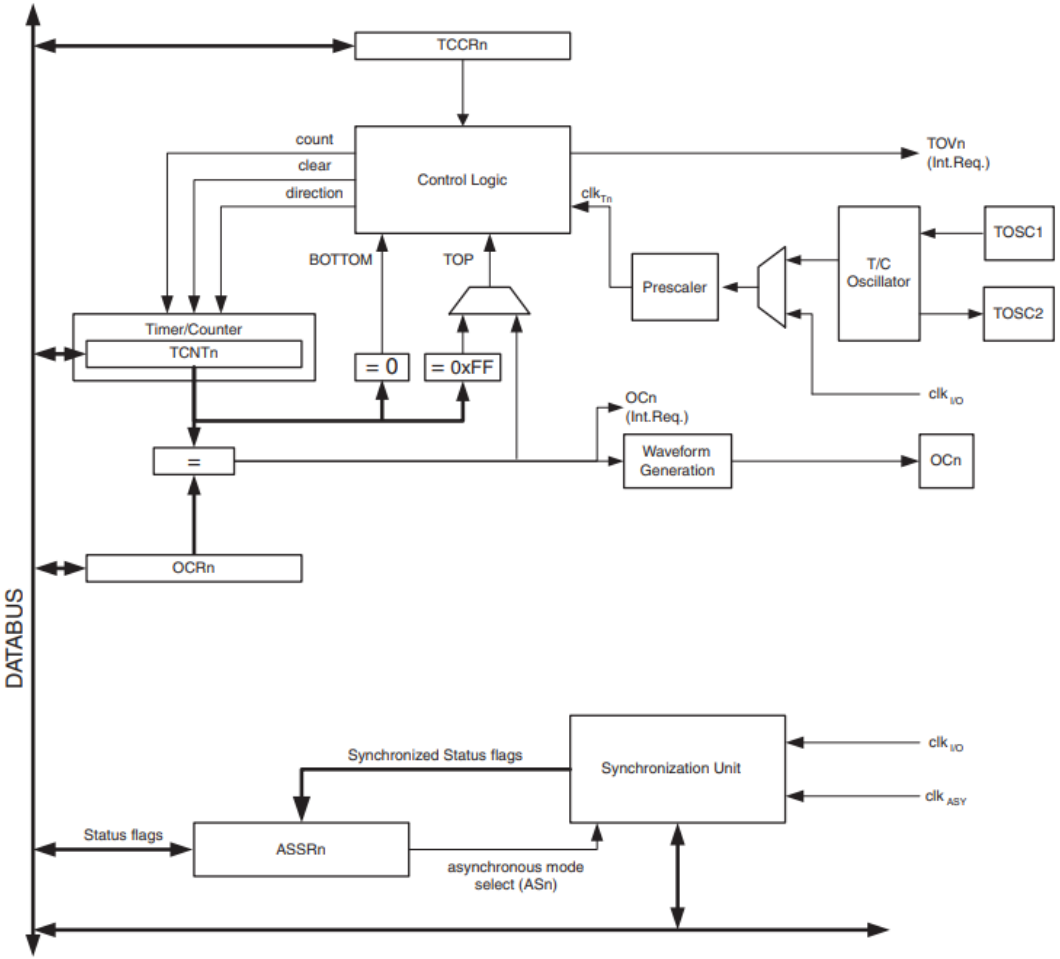
타이머/카운터

- ATmega128 타이머/카운터 요약

	타이머/카운터0	타이머/카운터1	타이머/카운터2	타이머/카운터3
구조	8비트	16비트	8비트	16비트
외부클럭 입력핀	TOSC1	T1	T2	T3
프리스케일러	1, 8, 32, 64, 128, 256, 1024	1, 8, 64, 256, 1024	1, 8, 64, 256, 1024	1, 8, 64, 256, 1024
관련 출력핀	OC0	OC1A, OC1B, OC1C	OC2	OC3A, OC3B, OC3C
인터럽트	오버플로우, 출력비교매치	오버플로우, 출력비교매치, 입력캡쳐	오버플로우, 출력비교매치	오버플로우, 출력비교매치, 입력캡쳐
특징	RTC 기능, 타이머/카운터 프리스케일러 사용	캡쳐기능	–	캡쳐기능

타이머/카운터0

- Real time counter(RTC) 기능
- 카운터로 동작할 경우의 클럭 입력단자 : TOSC1
- 비교일치시 타이머 자동 클리어(Auto Reloader)
- PWM 파형 발생 기능
- 10비트 클럭 프리스케일러
- 오버플로우 인터럽트와 Output Compare Match 인터럽트 발생 기능
- IO 클럭과 독립된 32kHz의 시계 클럭 접속 단자

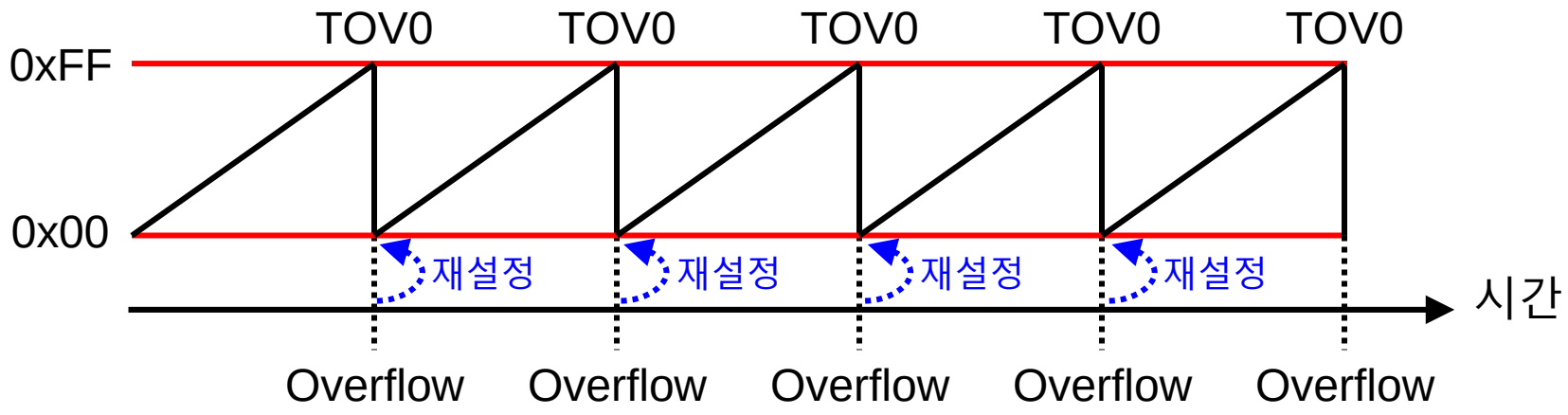


타이머/카운터0

- 동작 모드
  - **Normal mode**
    - 0(0x00)부터 255(0xFF)까지 셈
    - 255를 넘어갈 때 오버플로우 발생 → 0부터 시작되면서 오버플로우 인터럽트 발생됨
  - **Clear Timer on Compare Match(CTC) mode**
    - 카운터 값(TCNT0)이 OCR0 레지스터 값과 같게 될 때 TCNT0은 클리어되면서 인터럽트 발생됨
  - **Fast PWM mode**
    - Single-slope 동작
    - 0(0x00)부터 255(0xFF)까지 셈
  - **Phase Correct PWM mode**
    - Dual-slope 동작
    - 0(0x00) → 255(0xFF) → 0(0x00) → 255(0xFF) → …

타이머/카운터0

- Normal mode
  - 항상 업 카운터로만 동작하며, TCNT0 값이 8비트 최댓값 0xFF가 되면 0x00부터 다시 시작함
  - TCNT0 값이 0xFF에서 0x00으로 될 때 오버플로우 인터럽트 플래그(TOV0)은 1이 되어 인터럽트를 발생시킴
  - TCNT0의 상한값이 0xFF로 고정됨
  - 오버플로우 일어텁트 주기 = $(1/clk_{I/O})$ * 분주비 * 256



타이머/카운터0

- Normal mode : 관련 컨트롤 레지스터
  - TCCR0 : Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

  - WGM00(비트6), WGM01(비트3) : 파형 발생 모드 비트

모드	WGM01	WGM00	동작 모드	상한값	OCR0의 업데이트 시점	TOV0 플래그의 세트 시점
0	0	0	Normal	0xFF	설정즉시	상한값
1	0	1	Phase correct PWM	0xFF	상한값	0x00
2	1	0	CTC	OCR0	설정즉시	상한값
3	1	1	Fast PWM	0xFF	상한값	상한값

타이머/카운터0

- Normal mode : 관련 컨트롤 레지스터
  - TCCR0 : Timer/Counter Control Register 0

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

  - CS02-00 : 클럭 선택 비트

CS02	CS01	CS00	분주비 설정
0	0	0	클럭 입력 차단
0	0	1	clk_{T0S}/1
0	1	0	clk_{T0S}/8
0	1	1	clk_{T0S}/32

CS02	CS01	CS00	분주비 설정
1	0	0	clk_{T0S}/64
1	0	1	clk_{T0S}/128
1	1	0	clk_{T0S}/256
1	1	1	clk_{T0S}/1024

타이머/카운터0

- Normal mode : 관련 컨트롤 레지스터
  - TCNT0 : Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
	TCNT0[7:0]								TCNT0
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	

  - OCR0 : Timer/Counter Output Compare Register

Bit	7	6	5	4	3	2	1	0	
	OCR0[7:0]								OCR0
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	

  - OCR0와 TCNT0의 값이 일치했을 때 OC0 핀을 통해 설정된 값이 출력되거나 출력비교 인터럽트를 발생시킴

타이머/카운터0

- Normal mode : 관련 컨트롤 레지스터
  - TIMSK : Timer/Counter Interrupt Mask Register

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	

    - OCIE0 : Timer/Counter0 Output Compare Match Interrupt Enable
      - OCIE0은 1로 되고 SREG의 7번 비트는 1로 되면 출력비교 인터럽트가 허용상태로 됨
      - TCNT0과 OCR0의 값이 일치할 때 타이머/카운터 인터럽트 플래그 레지스터 TIFR의 OCF0 비트가 1로 되고 해당 ISR가 실행됨
    - TOIE0 : Timer/Counter0 Overflow Interrupt Enable
      - TOIE0은 1로 되고 SREG의 7번 비트는 1로 되면 오버플로우 인터럽트가 허용상토로 됨
      - 타이머/카운터가 오버플로우 될 때 타이머/카운터 인터럽트 플래그 레지스터 TIFR의 TOV0 비트가 1로 되고 해당 ISR가 실행됨

타이머/카운터0

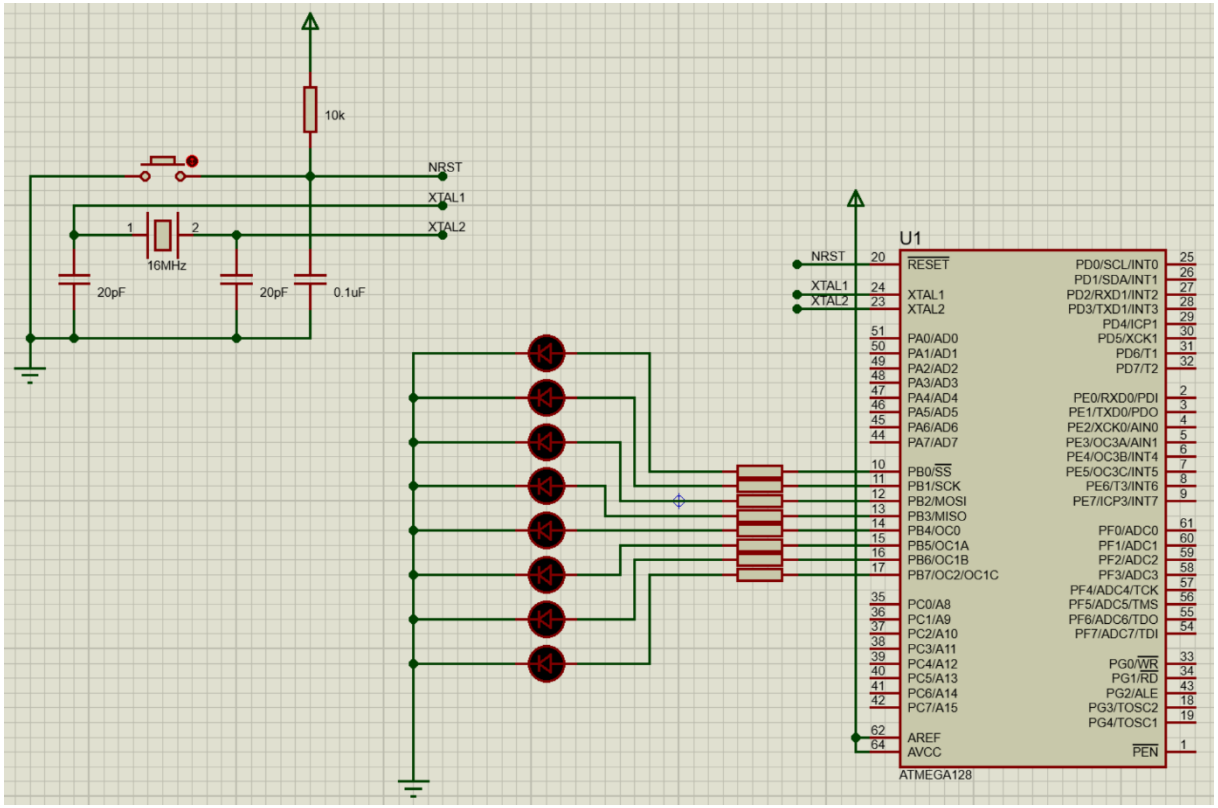
- Normal mode : 관련 컨트롤 레지스터
  - TIFR : Timer/Counter Interrupt Flag Register

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	

    - OCF0 : Output Compare Flag 0
      - TCNT0과 OCR0의 값이 일치할 때 OCF0 비트가 1로 됨
      - SREG의 7번 비트, OCIE0 비트 그리고 OCF0 비트가 1이 되면 타이머/카운터0 비교일치와 해당하는 ISR가 수행되게 되며, OCF0 비트가 자동적으로 0으로 클리어 됨
    - TOV0 : Timer/Counter0 Overflow Flag
      - 타이머/카운터가 오버플로우 될 때 TOV0 비트가 1로 됨
      - SREG의 7번 비트, TOIE0 비트 그리고 TOV0 비트가 1이 되면 타이머/카운터0 오버플로우와 해당하는 ISR가 수행되게 되며, TOV0 비트가 자동적으로 0으로 클리어 됨

타이머/카운터0

- Normal mode : 실습1



타이머/카운터0

- Normal mode : 실습1

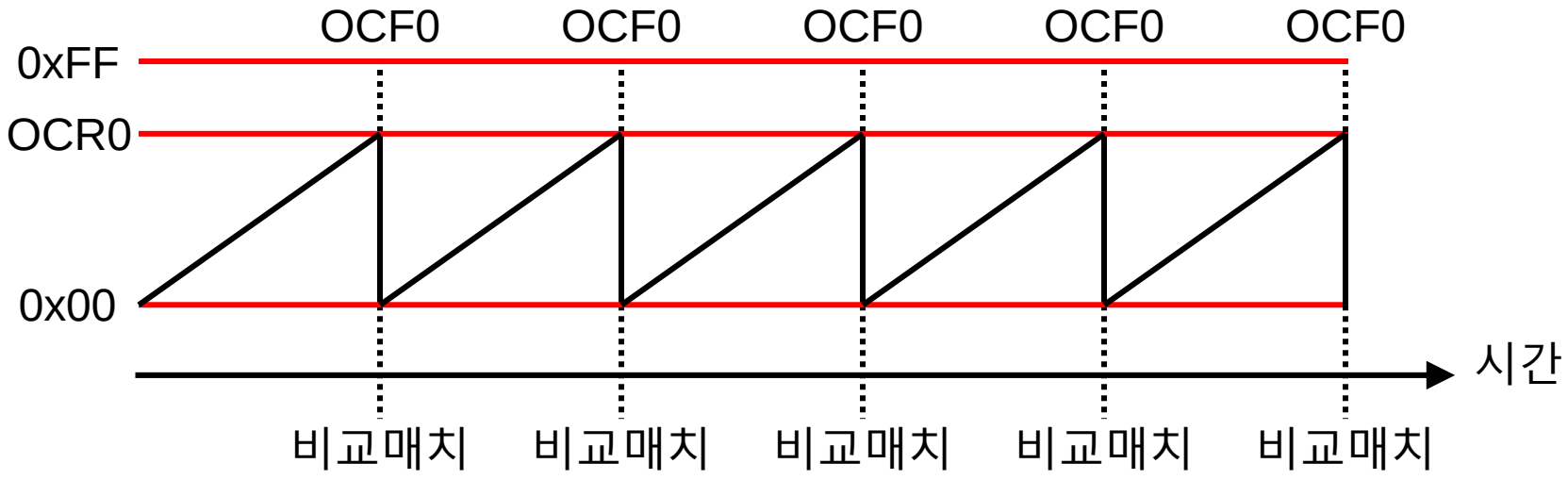
```
#include <xc.h>
#include <avr/io.h>
#include <avr/interrupt.h>

typedef unsigned char byte;
typedef unsigned int word;

#define cbi(REG8, BITNUM) REG8 &= ~(_BV(BITNUM))
#define sbi(REG8, BITNUM) REG8 |= _BV(BITNUM)

byte led=0x01, ledB=0x01;

// Timer 0 ISR
ISR(TIMER0_OVF_vect) {
   if (led>10) {
      led = 1;
      PORTB = ledB;
      ledB <<= 1;
      if (ledB==0x00) ledB = 0x01;
   } else led++;
}
```

```
int main(void) {
   DDRB = 0xff;
   PORTB = ledB;

   TCCR0 = 0x07;
   sbi(TIMSK, 0);
   sei(); // SREG |= 0x80;
   TCNT0 = 0x00;

   while(1);
}
```

타이머/카운터0

- CTC mode
  - TCNT0 값이 증가하여 OCR0 값과 일치하면 다시 0으로 클리어 됨
  - OCR0은 카운터의 최댓값을 나타냄 → OCR0로 주기 조정 가능
  - OC0 핀을 통해 파형을 발생시킬 수 있으며, 출력되는 파형의 주기는

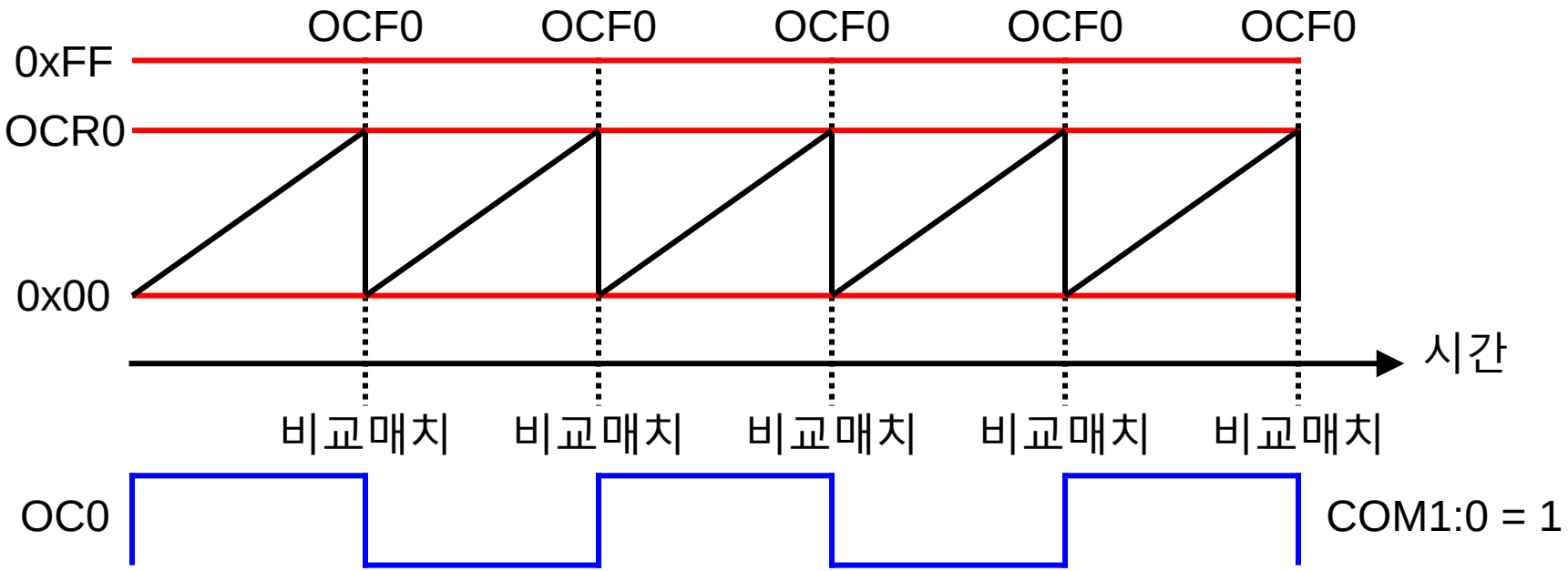
$$f_{OC0} = \frac{f_{clkI/O}}{2 * 분주비 * (1 + OCR0)}$$



타이머/카운터0

- CTC mode
  - OC0 핀을 통해 파형을 발생시킬 수 있으며, 출력되는 파형의 주기는

$$f_{OC0} = \frac{f_{clkI/O}}{2 * 분주비 * (1 + OCR0)}$$



타이머/카운터0

- CTC mode : 관련 컨트롤 레지스터
  - TCCR0 : Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

  - WGM00(비트6), WGM01(비트3) : 파형 발생 모드 비트

모드	WGM01	WGM00	동작 모드	상한값	OCR0의 업데이트 시점	TOV0 플래그의 세트 시점
0	0	0	Normal	0xFF	설정즉시	상한값
1	0	1	Phase correct PWM	0xFF	상한값	0x00
2	1	0	CTC	OCR0	설정즉시	상한값
3	1	1	Fast PWM	0xFF	상한값	상한값

타이머/카운터0

- CTC mode : 관련 컨트롤 레지스터
  - TCCR0 : Timer/Counter Control Register 0

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

  - CS02-00 : 클럭 선택 비트

CS02	CS01	CS00	분주비 설정
0	0	0	클럭 입력 차단
0	0	1	$clk_{T0S}/1$
0	1	0	$clk_{T0S}/8$
0	1	1	$clk_{T0S}/32$

CS02	CS01	CS00	분주비 설정
1	0	0	$clk_{T0S}/64$
1	0	1	$clk_{T0S}/128$
1	1	0	$clk_{T0S}/256$
1	1	1	$clk_{T0S}/1024$

타이머/카운터0

- CTC mode : 관련 컨트롤 레지스터
  - TCNT0 : Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
	TCNT0[7:0]								TCNT0
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	

  - OCR0 : Timer/Counter Output Compare Register

Bit	7	6	5	4	3	2	1	0	
	OCR0[7:0]								OCR0
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	

  - OCR0와 TCNT0의 값이 일치했을 때 OC0 핀을 통해 설정된 값이 출력되거나 출력비교 인터럽트를 발생시킴

타이머/카운터0

- CTC mode : 관련 컨트롤 레지스터
  - TIMSK : Timer/Counter Interrupt Mask Register

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	

    - OCIE0 : Timer/Counter0 Output Compare Match Interrupt Enable
      - OCIE0은 1로 되고 SREG의 7번 비트는 1로 되면 출력비교 인터럽트가 허용상태로 됨
      - TCNT0과 OCR0의 값이 일치할 때 타이머/카운터 인터럽트 플래그 레지스터 TIFR의 OCF0 비트가 1로 되고 해당 ISR가 실행됨
    - TOIE0 : Timer/Counter0 Overflow Interrupt Enable
      - TOIE0은 1로 되고 SREG의 7번 비트는 1로 되면 오버플로우 인터럽트가 허용상토로 됨
      - 타이머/카운터가 오버플로우 될 때 타이머/카운터 인터럽트 플래그 레지스터 TIFR의 TOV0 비트가 1로 되고 해당 ISR가 실행됨

타이머/카운터0

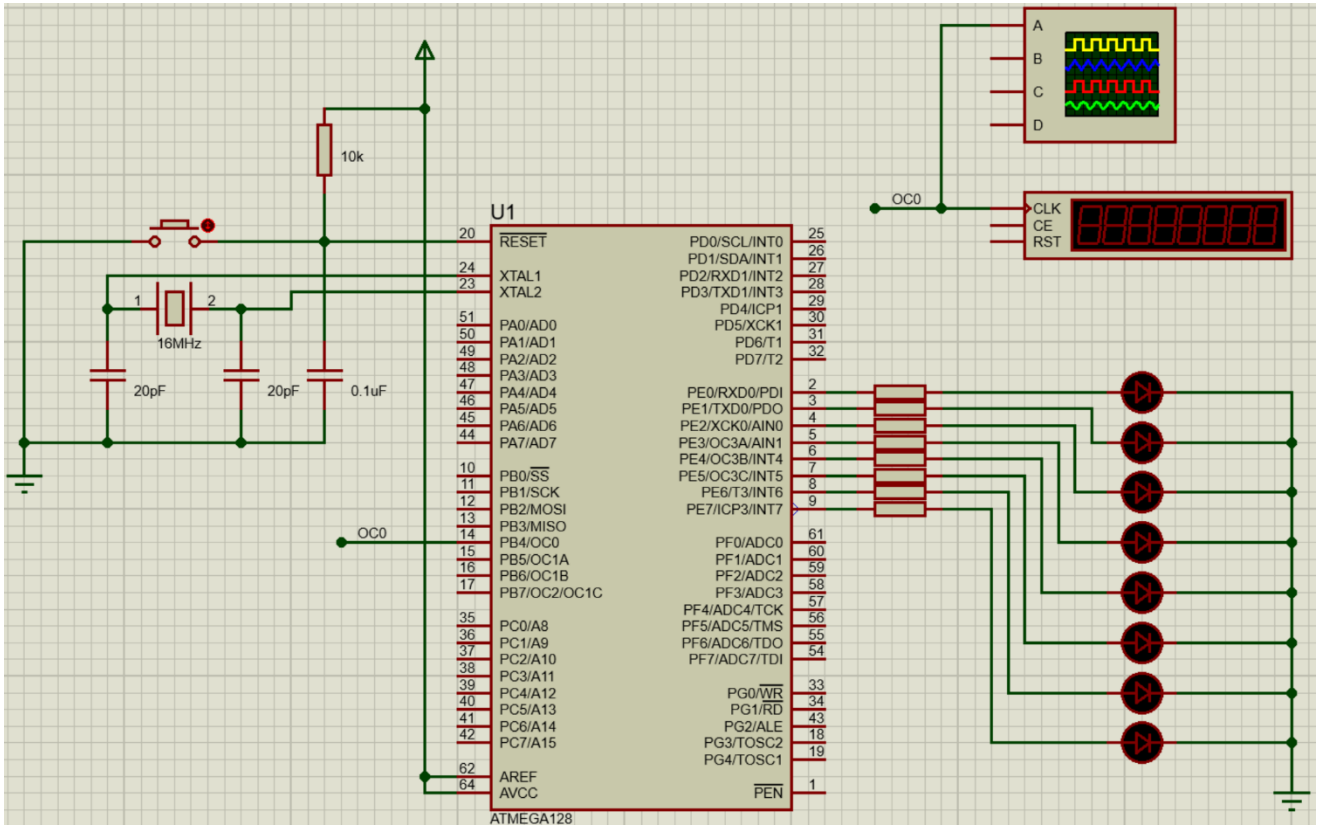
- CTC mode : 관련 컨트롤 레지스터
  - TIFR : Timer/Counter Interrupt Flag Register

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	

    - OCF0 : Output Compare Flag 0
      - TCNT0과 OCR0의 값이 일치할 때 OCF0 비트가 1로 됨
      - SREG의 7번 비트, OCIE0 비트 그리고 OCF0 비트가 1이 되면 타이머/카운터0 비교일치와 해당하는 ISR가 수행되게 되며, OCF0 비트가 자동적으로 0으로 클리어 됨
    - TOV0 : Timer/Counter0 Overflow Flag
      - 타이머/카운터가 오버플로우 될 때 TOV0 비트가 1로 됨
      - SREG의 7번 비트, TOIE0 비트 그리고 TOV0 비트가 1이 되면 타이머/카운터0 오버플로우와 해당하는 ISR가 수행되게 되며, TOV0 비트가 자동적으로 0으로 클리어 됨

타이머/카운터0

- CTC mode : 실습1



타이머/카운터0

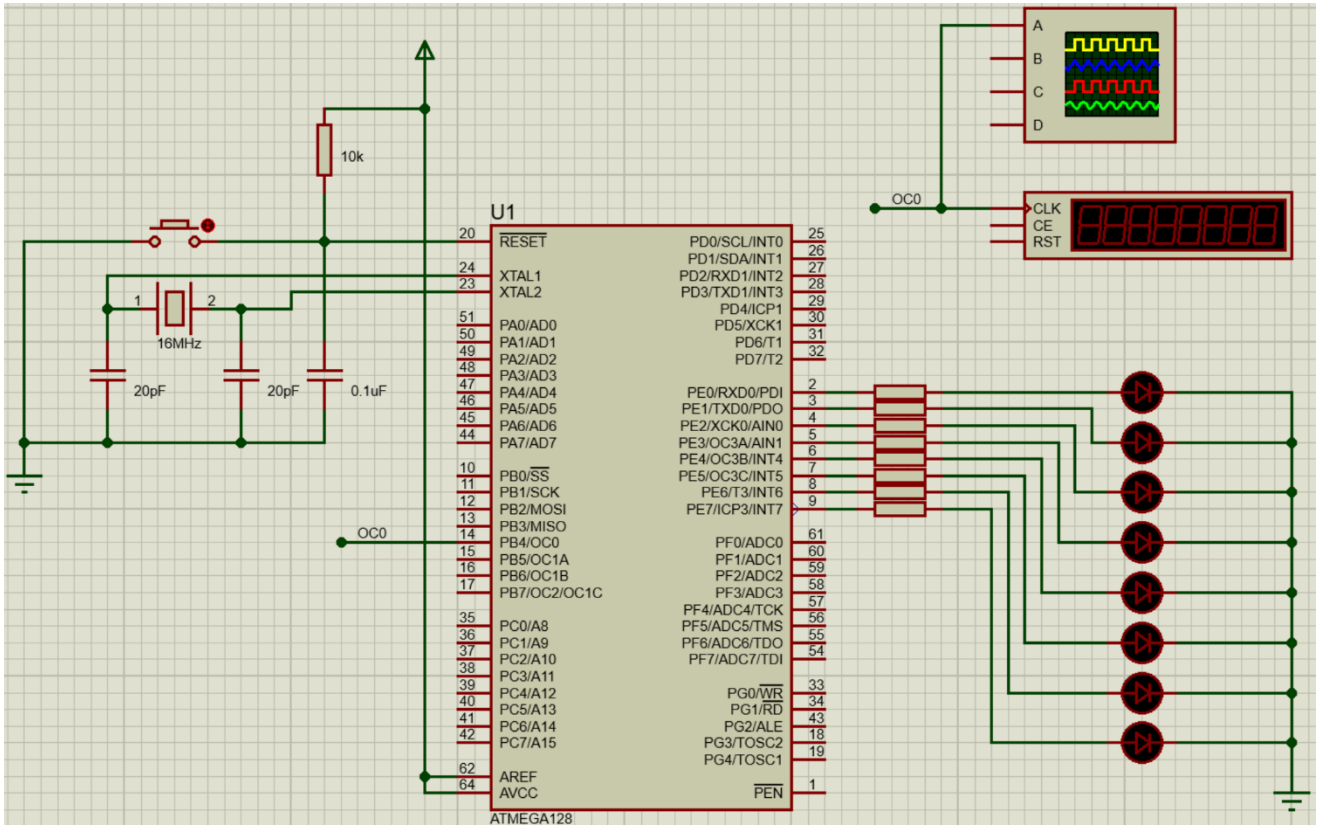
- CTC mode : 실습1

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

typedef unsigned char byte;
typedef unsigned int word;

#define cbi(REG8, BITNUM) REG8 &= ~(_BV(BITNUM))
#define sbi(REG8, BITNUM) REG8 |= _BV(BITNUM)

byte ledE = 0x01;
```

```
int main(void) {
    DDRE = 0xff;
    DDRB = 0xff;
    PORTE = ledE;

    TCCR0 = 0x1f;
    OCR0 = 0x80;
    TCNT0 = 0x00;

    while(1);
}
```

타이머/카운터0

- CTC mode : 실습2



타이머/카운터0

- CTC mode : 실습2

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

typedef unsigned char byte;
typedef unsigned int word;

#define cbi(REG8, BITNUM) REG8 &= ~(_BV(BITNUM))
#define sbi(REG8, BITNUM) REG8 |= _BV(BITNUM)

byte led=1, ledE=0x01;

ISR(TIMER0_OVF_vect) {
   if (led>10) {
      led = 0;
      PORTE = ledE;
      ledE <<= 1;
      if (ledE==0x00) ledE = 0x01;
   } else led++;
}
```

```
ISR(TIMER0_COMP_vect) {
   OCR0 -= 1;
   if (OCR0<0x1f) OCR0 = 0xff;
}

int main(void) {
   DDRE = 0xff;
   DDRB = 0xff;
   PORTE = ledE;

   sbi(TIMSK, 0);
   sbi(TIMSK, 1);
   TCCR0 = 0x1f;
   OCR0 = 0xf0;
   TCNT0 = 0x00;
   sei();

   while(1);
}
```

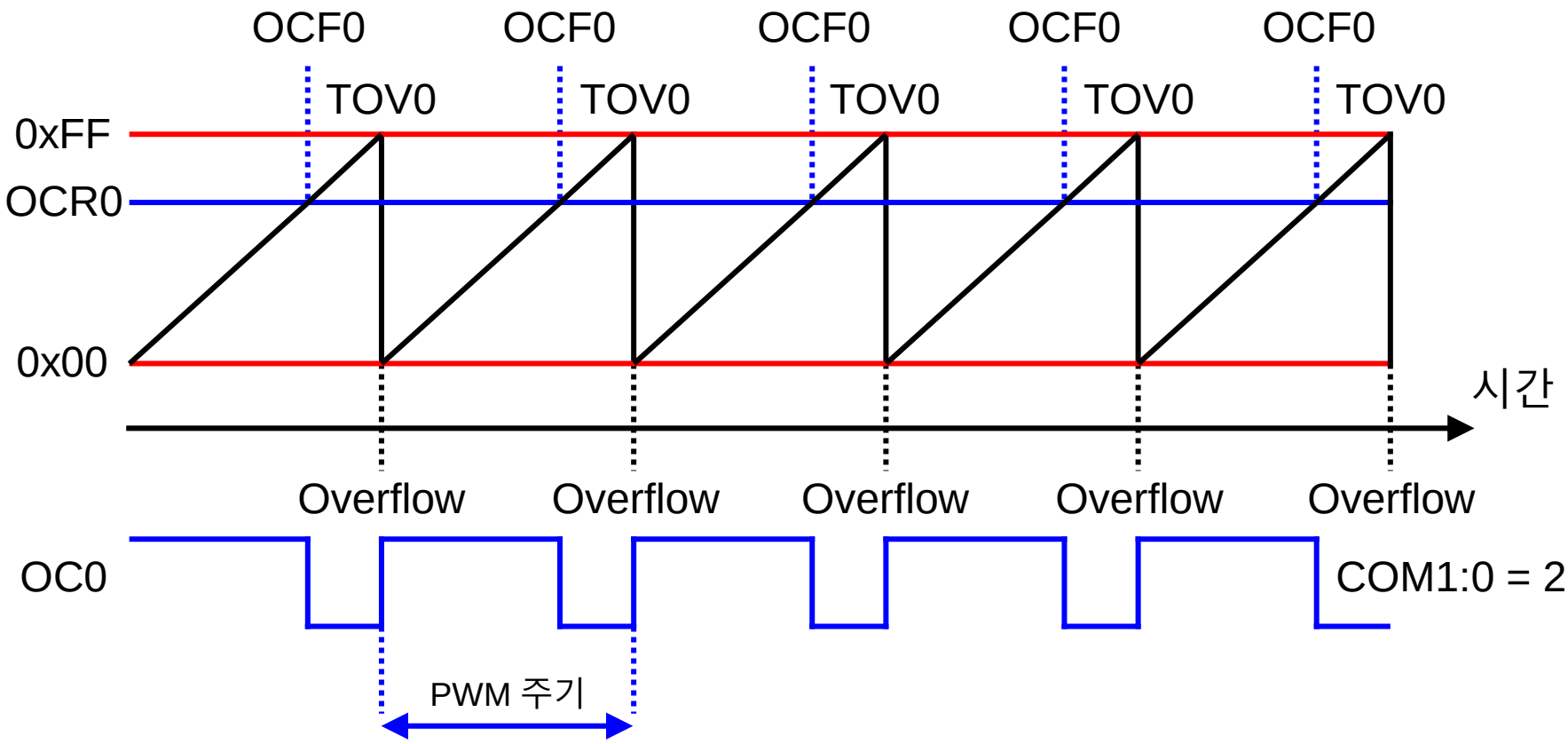
타이머/카운터0

- Fast PWM mode
  - 높은 주파수의 PWM 파형을 발생하는데 유용함
  - TCNT0 값은 0x00에서 0xFF까지 증가하다 다시 0x00부터 세는 동작을 반복함
  - 비반전 비교 출력모드(COM01 = 2)에서 TCNT0 값은 계속하여 OCR0과 비교되어 일치하면 OC0 핀을 통해 0이 출력되고, TCNT0 값이 0이 되면 1이 출력됨
  - OCR0 값에 따라 high (또는 low) 부분의 펄스의 펄스폭이 가변되어 PWM 파형을 출력됨

$$f_{OC0} = \frac{f_{clkI/O}}{\text{분주비} * 256}$$

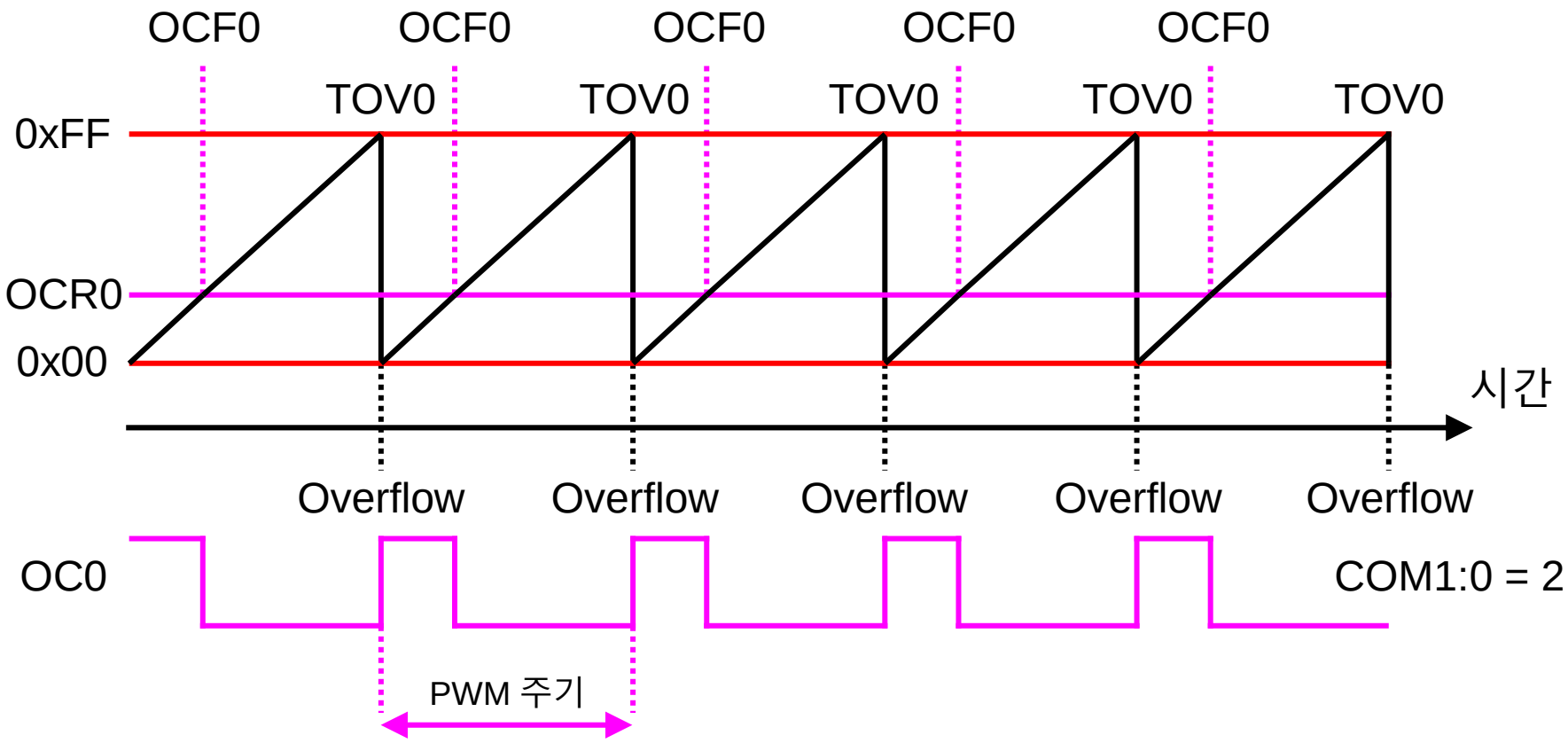
타이머/카운터0

- Fast PWM mode : OCR0 값이 큰 경우



타이머/카운터0

- Fast PWM mode : OCR0 값이 작은 경우



타이머/카운터0

- Fast PWM mode : 관련 컨트롤 레지스터
  - TCCR0 : Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

  - WGM00(비트6), WGM01(비트3) : 파형 발생 모드 비트

모드	WGM01	WGM00	동작 모드	상한값	OCR0의 업데이트 시점	TOV0 플래그의 세트 시점
0	0	0	Normal	0xFF	설정즉시	상한값
1	0	1	Phase correct PWM	0xFF	상한값	0x00
2	1	0	CTC	OCR0	설정즉시	상한값
3	1	1	Fast PWM	0xFF	상한값	상한값

타이머/카운터0

- Fast PWM mode : 관련 컨트롤 레지스터
  - TCCR0 : Timer/Counter Control Register 0

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

  - CS02-00 : 클럭 선택 비트

CS02	CS01	CS00	분주비 설정
0	0	0	클럭 입력 차단
0	0	1	clk_{T0S}/1
0	1	0	clk_{T0S}/8
0	1	1	clk_{T0S}/32

CS02	CS01	CS00	분주비 설정
1	0	0	clk_{T0S}/64
1	0	1	clk_{T0S}/128
1	1	0	clk_{T0S}/256
1	1	1	clk_{T0S}/1024

타이머/카운터0

- Fast PWM mode : 관련 컨트롤 레지스터
  - TCNT0 : Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
	TCNT0[7:0]								TCNT0
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	

  - OCR0 : Timer/Counter Output Compare Register

Bit	7	6	5	4	3	2	1	0	
	OCR0[7:0]								OCR0
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	

  - OCR0와 TCNT0의 값이 일치했을 때 OC0 핀을 통해 설정된 값이 출력되거나 출력비교 인터럽트를 발생시킴

타이머/카운터0

- Fast PWM mode : 관련 컨트롤 레지스터
  - TIMSK : Timer/Counter Interrupt Mask Register

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	

    - OCIE0 : Timer/Counter0 Output Compare Match Interrupt Enable
      - OCIE0은 1로 되고 SREG의 7번 비트는 1로 되면 출력비교 인터럽트가 허용상태로 됨
      - TCNT0과 OCR0의 값이 일치할 때 타이머/카운터 인터럽트 플래그 레지스터 TIFR의 OCF0 비트가 1로 되고 해당 ISR가 실행됨
    - TOIE0 : Timer/Counter0 Overflow Interrupt Enable
      - TOIE0은 1로 되고 SREG의 7번 비트는 1로 되면 오버플로우 인터럽트가 허용상토로 됨
      - 타이머/카운터가 오버플로우 될 때 타이머/카운터 인터럽트 플래그 레지스터 TIFR의 TOV0 비트가 1로 되고 해당 ISR가 실행됨

타이머/카운터0

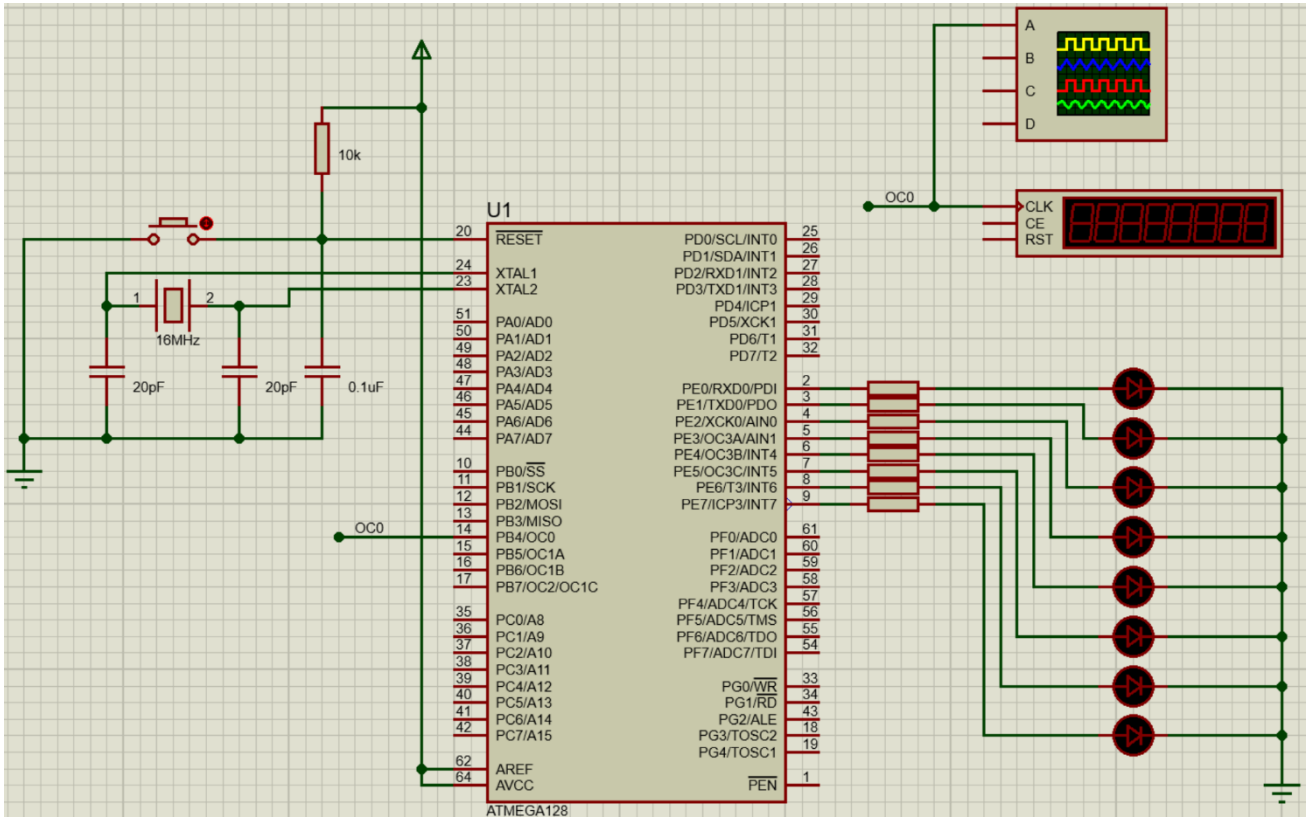
- Fast PWM mode : 관련 컨트롤 레지스터
  - TIFR : Timer/Counter Interrupt Flag Register

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	

    - OCF0 : Output Compare Flag 0
      - TCNT0과 OCR0의 값이 일치할 때 OCF0 비트가 1로 됨
      - SREG의 7번 비트, OCIE0 비트 그리고 OCF0 비트가 1이 되면 타이머/카운터0 비교일치와 해당하는 ISR가 수행되게 되며, OCF0 비트가 자동적으로 0으로 클리어 됨
    - TOV0 : Timer/Counter0 Overflow Flag
      - 타이머/카운터가 오버플로우 될 때 TOV0 비트가 1로 됨
      - SREG의 7번 비트, TOIE0 비트 그리고 TOV0 비트가 1이 되면 타이머/카운터0 오버플로우와 해당하는 ISR가 수행되게 되며, TOV0 비트가 자동적으로 0으로 클리어 됨

타이머/카운터0

- Fast PWM mode : 실습1



타이머/카운터0

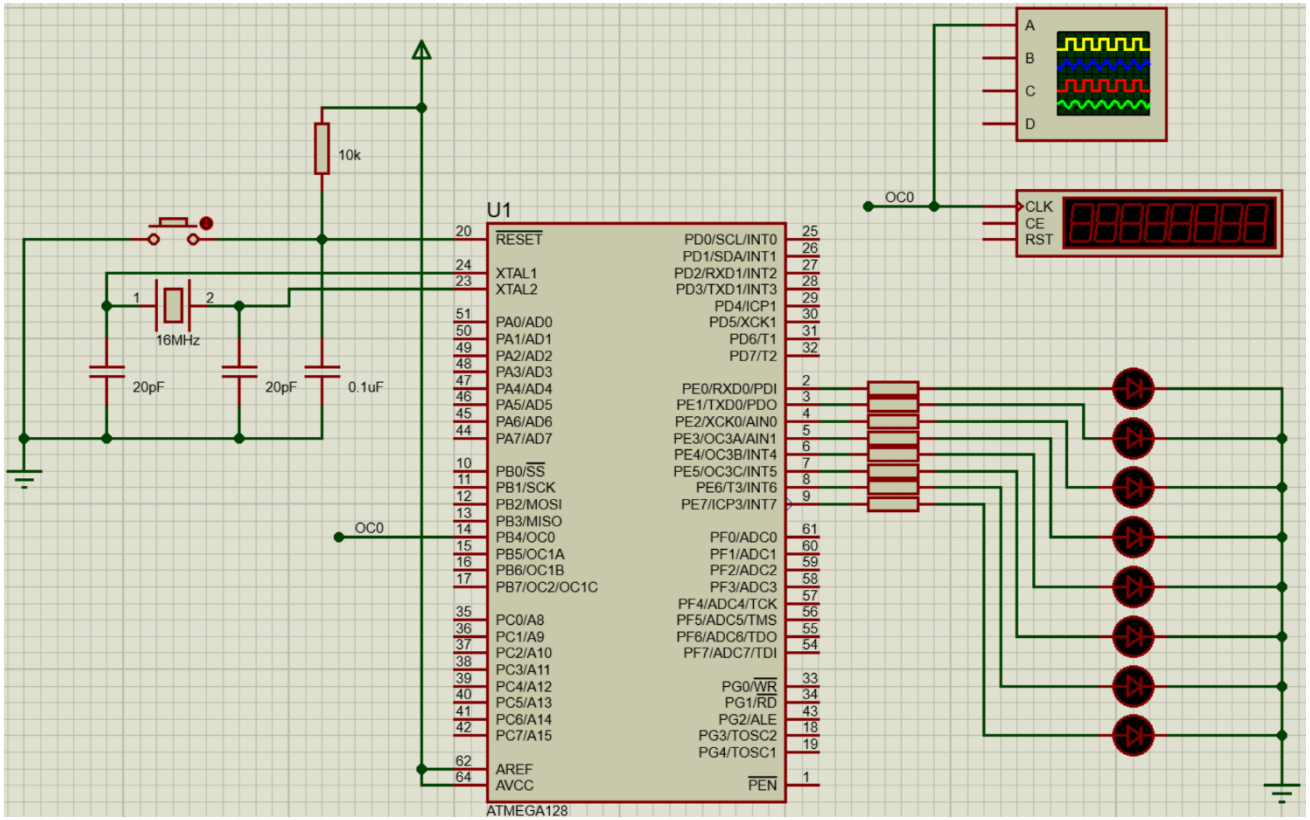
- Fast PWM mode : 실습1

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

typedef unsigned char byte;
typedef unsigned int word;

#define cbi(REG8, BITNUM) REG8 &= ~(_BV(BITNUM))
#define sbi(REG8, BITNUM) REG8 |= _BV(BITNUM)

byte ledE = 0x01;
```

```
int main(void) {
    DDRE = 0xff;
    DDRB = 0xff;
    PORTE = ledE;

    TCCR0 = 0x6f;
    // fast PWM, duty cycle 50%
    OCR0 = 0x7f;
    TCNT0 = 0x00;

    while(1);
}
```

타이머/카운터0

- Fast PWM mode : 실습2



타이머/카운터0

- Fast PWM mode : 실습2

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

typedef unsigned char byte;
typedef unsigned int word;

#define cbi(REG8, BITNUM) REG8 &= ~(_BV(BITNUM))
#define sbi(REG8, BITNUM) REG8 |= _BV(BITNUM)

byte led=1, ledE=0x01;

ISR(TIMER0_OVF_vect) {
   if (led>10) {
      led = 0;
      PORTE = ledE;
      ledE <<= 1;
      if (ledE==0x00) ledE = 0x01;
   } else led++;
}
```

```
ISR(TIMER0_COMP_vect) {
   OCR0 -= 1;
   if (OCR0<0x1f) OCR0 = 0xff;
}

int main(void) {
   DDRE = 0xff;
   DDRB = 0xff;
   PORTE = ledE;

   sbi(TIMSK, 0);
   sbi(TIMSK, 1);
   TCCR0 = 0x6f;
   OCR0 = 0x7f;
   TCNT0 = 0x00;
   sei();

   while(1);
}
```

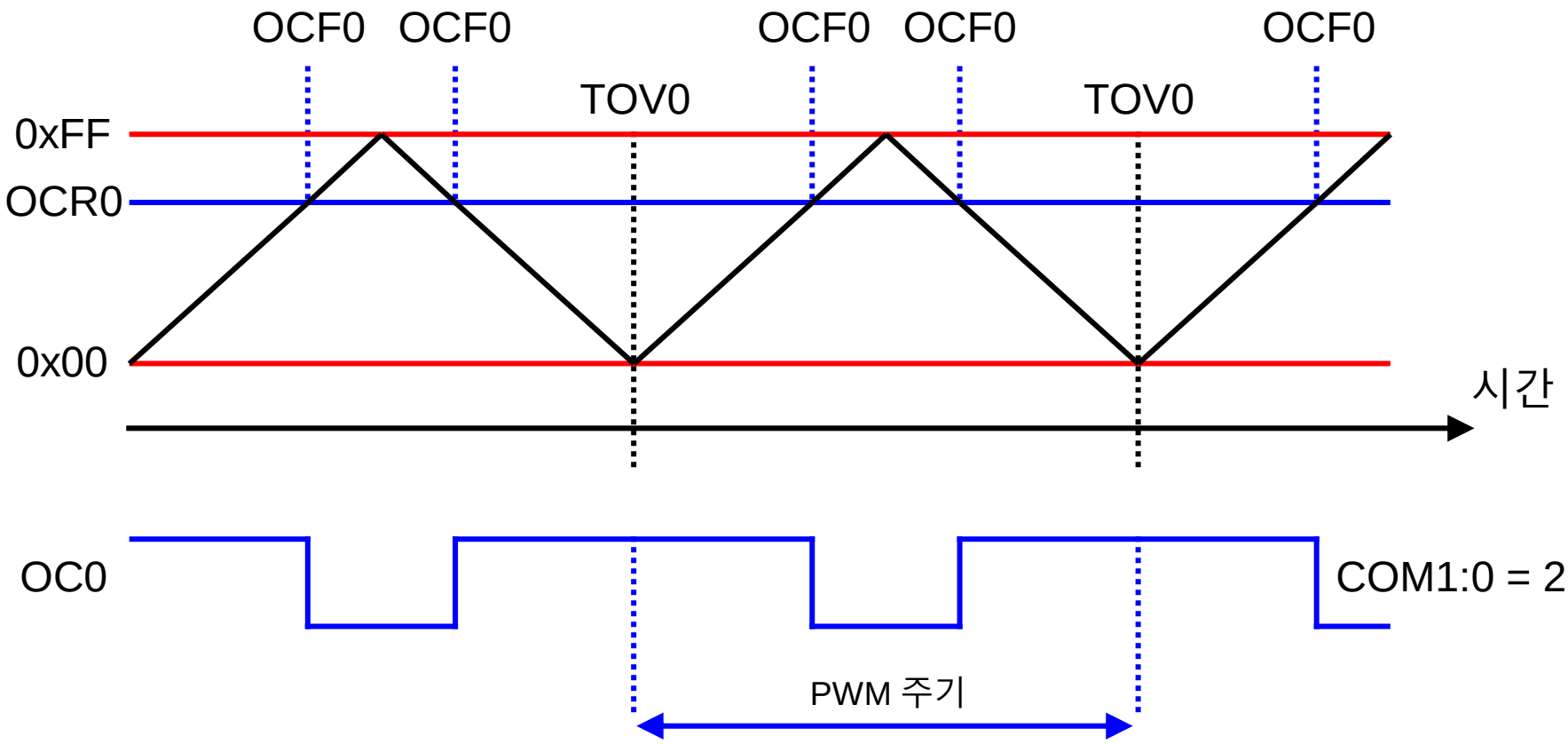
타이머/카운터0

- PC(phase correct) PWM mode
  - 높은 주파수의 PWM 파형을 발생하는데 유용함
  - TCNT0 값은 0x00에서 0xFF까지 증가하다 0x00으로 감소하는 동작을 반복함
  - 비반전 비교 출력모드(COM01 = 2)에서 업 카운터 중에 TCNT0과 OCR0의 값이 일치하면 OC0 핀을 통해 0이 출력되고, 다운 카운터 중에 TCNT0과 OCR0의 값이 일치하면 1이 출력됨
  - OCR0 값에 따라 high (또는 low) 부분의 펄스의 펄스폭이 가변되어 PWM 파형을 출력됨

$$f_{OC0} = \frac{f_{clkI/O}}{\text{분주비} * 510}$$

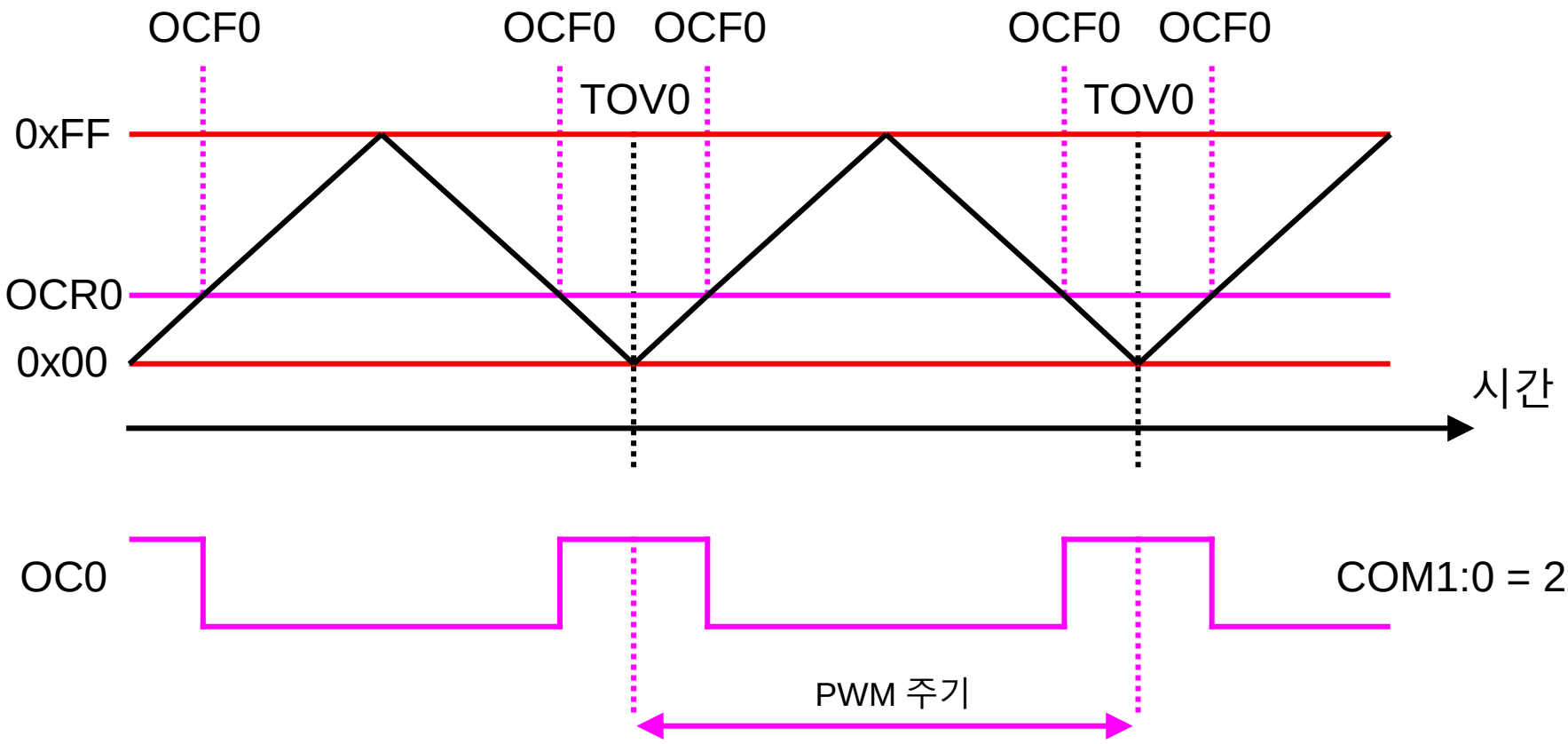
타이머/카운터0

- PC(phase correct) PWM mode : OCR0 값이 큰 경우



타이머/카운터0

- PC(phase correct) PWM mode : OCR0 값이 작은 경우



타이머/카운터0

- PC(phase correct) PWM mode : 관련 컨트롤 레지스터
  - TCCR0 : Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

  - WGM00(비트6), WGM01(비트3) : 파형 발생 모드 비트

모드	WGM01	WGM00	동작 모드	상한값	OCR0의 업데이트 시점	TOV0 플래그의 세트 시점
0	0	0	Normal	0xFF	설정즉시	상한값
1	0	1	Phase correct PWM	0xFF	상한값	0x00
2	1	0	CTC	OCR0	설정즉시	상한값
3	1	1	Fast PWM	0xFF	상한값	상한값

타이머/카운터0

- PC(phase correct) PWM mode : 관련 컨트롤 레지스터
  - TCCR0 : Timer/Counter Control Register 0

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

  - CS02-00 : 클럭 선택 비트

CS02	CS01	CS00	분주비 설정
0	0	0	클럭 입력 차단
0	0	1	clk_{T0S}/1
0	1	0	clk_{T0S}/8
0	1	1	clk_{T0S}/32

CS02	CS01	CS00	분주비 설정
1	0	0	clk_{T0S}/64
1	0	1	clk_{T0S}/128
1	1	0	clk_{T0S}/256
1	1	1	clk_{T0S}/1024

타이머/카운터0

- PC(phase correct) PWM mode : 관련 컨트롤 레지스터
  - TCNT0 : Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
	TCNT0[7:0]								TCNT0
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	

  - OCR0 : Timer/Counter Output Compare Register

Bit	7	6	5	4	3	2	1	0	
	OCR0[7:0]								OCR0
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	

  - OCR0와 TCNT0의 값이 일치했을 때 OC0 핀을 통해 설정된 값이 출력되거나 출력비교 인터럽트를 발생시킴

타이머/카운터0

- PC(phase correct) PWM mode : 관련 컨트롤 레지스터
  - TIMSK : Timer/Counter Interrupt Mask Register

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	

    - OCIE0 : Timer/Counter0 Output Compare Match Interrupt Enable
      - OCIE0은 1로 되고 SREG의 7번 비트는 1로 되면 출력비교 인터럽트가 허용상태로 됨
      - TCNT0과 OCR0의 값이 일치할 때 타이머/카운터 인터럽트 플래그 레지스터 TIFR의 OCF0 비트가 1로 되고 해당 ISR가 실행됨
    - TOIE0 : Timer/Counter0 Overflow Interrupt Enable
      - TOIE0은 1로 되고 SREG의 7번 비트는 1로 되면 오버플로우 인터럽트가 허용상토로 됨
      - 타이머/카운터가 오버플로우 될 때 타이머/카운터 인터럽트 플래그 레지스터 TIFR의 TOV0 비트가 1로 되고 해당 ISR가 실행됨

타이머/카운터0

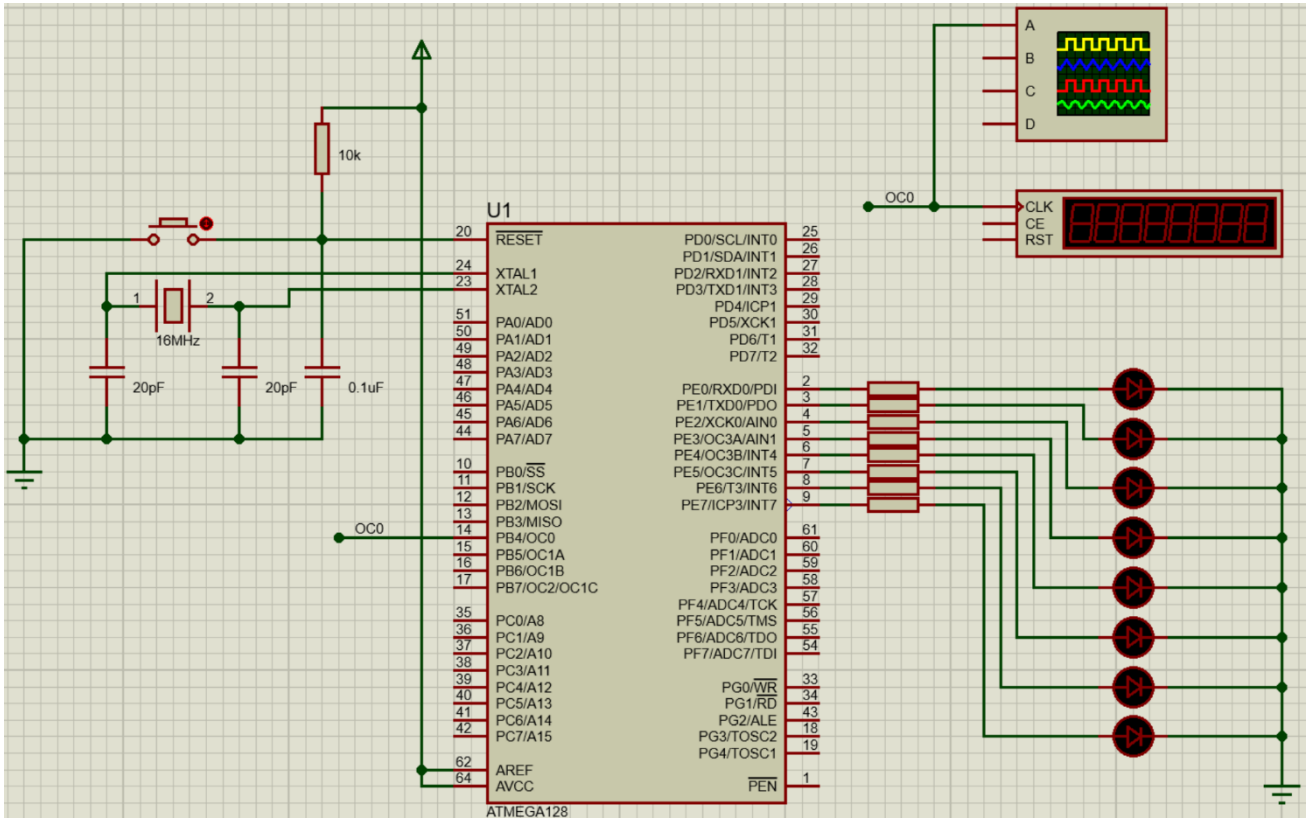
- PC(phase correct) PWM mode : 관련 컨트롤 레지스터
  - TIFR : Timer/Counter Interrupt Flag Register

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	

    - OCF0 : Output Compare Flag 0
      - TCNT0과 OCR0의 값이 일치할 때 OCF0 비트가 1로 됨
      - SREG의 7번 비트, OCIE0 비트 그리고 OCF0 비트가 1이 되면 타이머/카운터0 비교일치와 해당하는 ISR가 수행되게 되며, OCF0 비트가 자동적으로 0으로 클리어 됨
    - TOV0 : Timer/Counter0 Overflow Flag
      - 타이머/카운터가 오버플로우 될 때 TOV0 비트가 1로 됨
      - SREG의 7번 비트, TOIE0 비트 그리고 TOV0 비트가 1이 되면 타이머/카운터0 오버플로우와 해당하는 ISR가 수행되게 되며, TOV0 비트가 자동적으로 0으로 클리어 됨

타이머/카운터0

- PC(phase correct) PWM mode : 실습1



타이머/카운터0

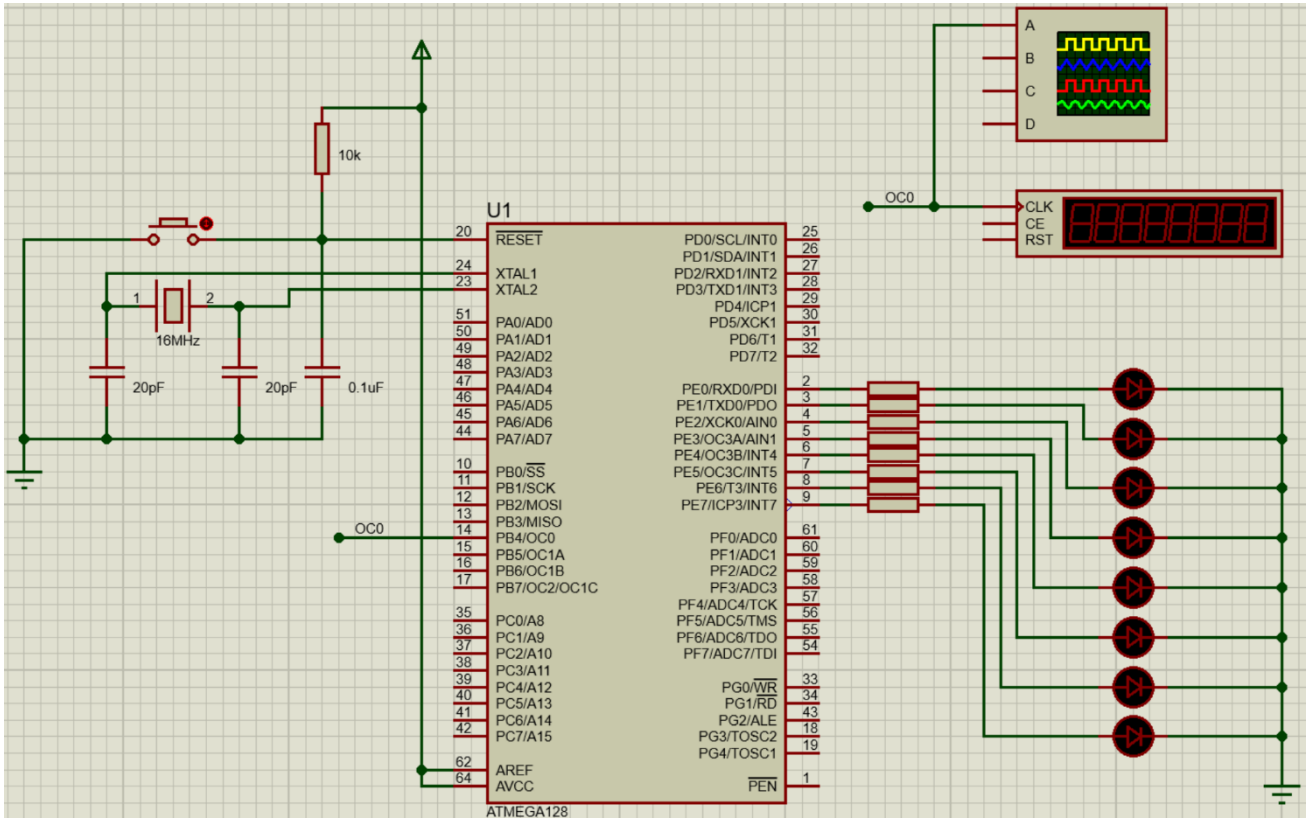
- PC(phase correct) PWM mode : 실습1

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

typedef unsigned char byte;
typedef unsigned int word;

#define cbi(REG8, BITNUM) REG8 &= ~(_BV(BITNUM))
#define sbi(REG8, BITNUM) REG8 |= _BV(BITNUM)

byte ledE = 0x01;
```

```
int main(void) {
    DDRE = 0xff;
    DDRB = 0xff;
    PORTE = ledE;

    TCCR0 = 0x61;
    OCR0 = 0x7f;
    TCNT0 = 0x00;

    while(1);
}
```

타이머/카운터0

- PC(phase correct) PWM mode : 실습2



타이머/카운터0

- PC(phase correct) PWM mode : 실습2

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

typedef unsigned char byte;
typedef unsigned int word;

#define cbi(REG8, BITNUM) REG8 &= (~_BV(BITNUM))
#define sbi(REG8, BITNUM) REG8 |= _BV(BITNUM)

byte led=1, ledE=0x01;

ISR(TIMER0_OVF_vect) {
   if (led>10) {
      led = 0;
      PORTE = ledE;
      ledE <<= 1;
      if (ledE==0x00) ledE = 0x01;
   } else led++;
}
```

```
ISR(TIMER0_COMP_vect) {
   OCR0 -= 1;
   if (OCR0<0x1f) OCR0 = 0xff;
}

int main(void) {
   DDRE = 0xff;
   DDRB = 0xff;
   PORTE = ledE;

   sbi(TIMSK, 0);
   sbi(TIMSK, 1);
   TCCR0 = 0x67;
   OCR0 = 0x7f;
   TCNT0 = 0x00;
   sei();

   while(1);
}
```