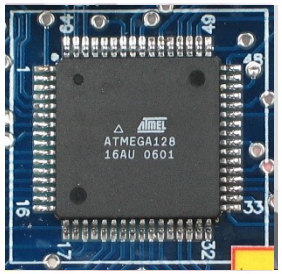
한국공학대학교
TECH UNIVERSITY OF KOREA



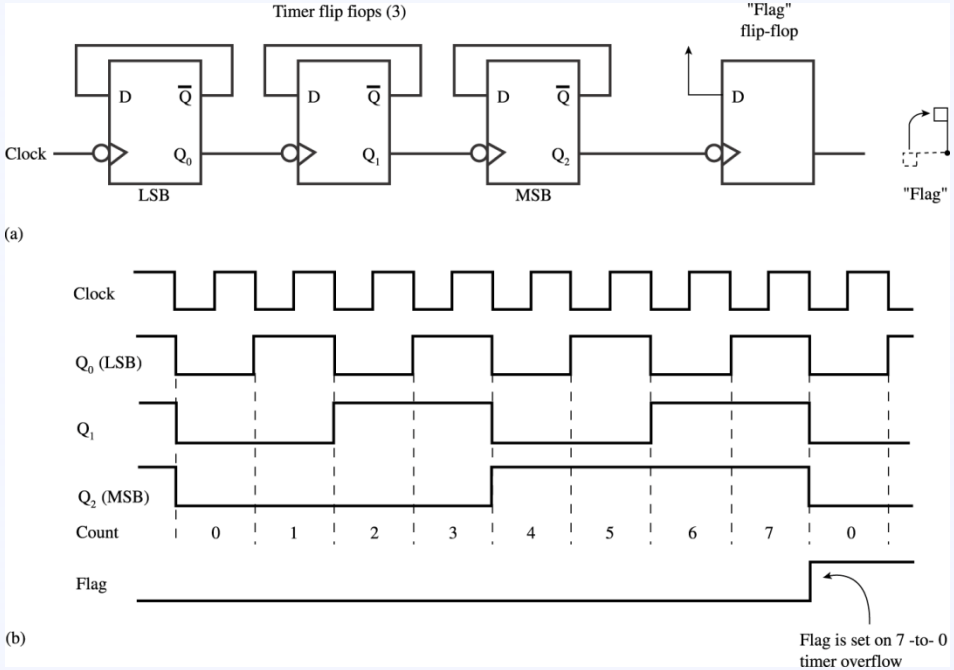
13 2022 May

타이머/카운터의 동작

타이머/카운터의 동작

타이머/카운터 개요

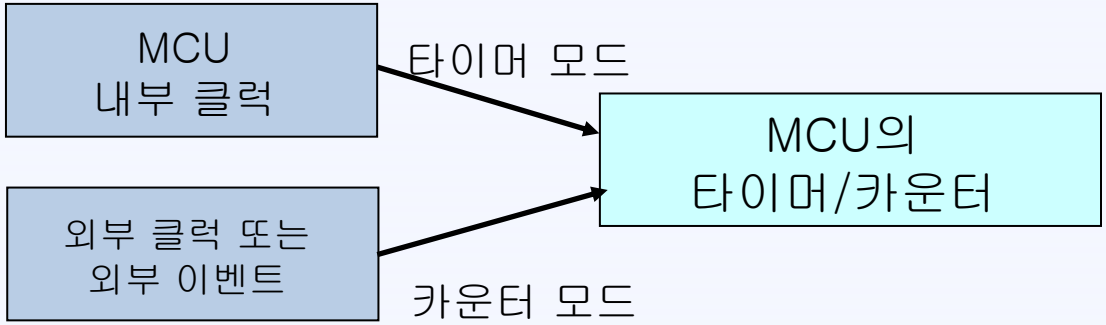
- 클럭이 입력되면 들어오는 클럭을 계수하는 기능
- 타이머/카운터의 각 단계는 입력 클럭을 2분주하여 동작하는 하나의 D 플립플롭 (D-type negative edge-triggered)으로 되어 있으며, D 플립플롭의 출력은 다음 단의 D 입력에 연결되어 있음.
- 오버플로우 : 3-비트 타이머/카운터의 경우 카운터의 상태가 111에서 000으로 변화할 때 발생함. → 플래그 플립플롭에 오버플로우가 발생했음을 알림.



타이머/카운터의 동작

타이머/카운터의 구분

- 타이머 모드 : MCU의 내부 클럭을 입력으로 사용하여 이를 분주해서 클럭 소스로 사용하는 경우
- 카운터 모드 : MCU의 외부에서 입력되는 클럭 신호를 받아서 이를 클럭 소스로 사용하는 경우



ATmega128의 타이머/카운터

- 2개의 8비트 타이머/카운터(T0, T2)
- 2개의 16비트 타이머/카운터(T1, T3)
- 다른 프로세서에 비해 PWM 기능이 추가

타이머/카운터의 동작

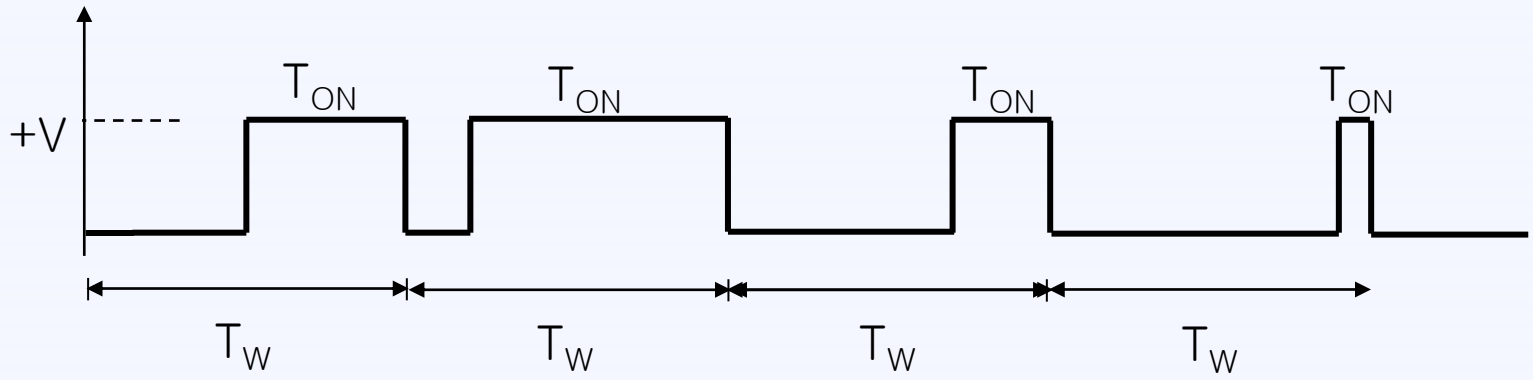
타이머/카운터의 기능

① 일정 시간 간격을 갖는 펄스의 발생
② 외부 사건의 계수하는 기능
③ PWM(Pulse Width Modulation) 펄스의 발생(모터 제어 등)-> 다음 페이지 참조

- **타이머 응용** :
  - 타이머는 주기적인 시간 간격으로 오버플로우가 발생되도록 프로그램되고 이 오버플로우가 발생할 때마다 오버플로우 플래그가 설정된다. 이 플래그는 펄스의 입력 상태를 확인하여 이 플래그가 발생하였으면 출력 포트에 데이터를 보내고, 또는 주기적으로 일정 시간 간격으로 어떠한 일을 수행하는 등의 동작을 동기화하는데 이용됨.
  - 외부에서 발생되는 펄스의 시간 간격을 계측하기 위해 타이머를 사용하여 외부에서 입력되는 펄스보다 작은 간격을 갖는 주기적인 펄스를 만들어 두 가지 상태(ON/OFF)의 시간을 측정하는 응용으로도 사용될 수 있음.
- **카운터 응용** : 외부 사건의 발생 회수를 계수하기 위해 사용됨.
  사건(event)은 어떤 외부적인 자극으로 ATmega128 IC의 핀에서 1 또는 0 상태로 공급되며, 이를 계수하여 사건의 발생 회수 및 사건의 발생 여부를 확인하는 응용 분야에 이용됨.
- **PWM 응용** : ATmega128의 타이머/카운터에는 비교 모듈(compare unit)을 내장 이 모듈을 이용하여 일반적인 파형의 발생과 PWM 신호를 발생.

타이머/카운터의 동작

PWM 신호



- **PWM 신호 : 일정한 주기를 갖는 신호를 T_W 시간 동안 T_{ON}의 폭으로 ON/OFF 제어를 하는 것을 의미함.**
- **즉, ON 되는 시간의 펄스폭을 넓게 하면 출력되는 평균 전압이 커지게 되고, ON 되는 시간의 펄스폭을 좁게 하면 평균 전압이 작아지게 되어 평균전압의 크기를 펄스 폭(T_{ON})을 디지털 값으로 제어함으로써 조절하는 방식을 PWM 제어라 함.**
- **모터의 속도를 변화시키거나, 솔레노이드와 같이 듀티비를 조정하여 구동하는 액츄에이터의 제어 분야에 PWM 제어 신호가 이용됨.**

Duty rate가 100% 면 +V peak 값, 50% 면 +V peak/2 값

타이머/카운터의 동작

한국공학대학교 TECH UNIVERSITY OF KOREA

ATmega128에 내장된 타이머/카운터의 기능 요약

타이머/카운터	타이머/카운터 0	타이머/카운터 1	타이머/카운터 2	타이머/카운터 3
비트 수	8비트	16비트	8비트	16비트
카운터 입력	타이머/카운터 OSC TOSC1	T1	T2	T3
관련 레지스터	TCCR0 TCNT0 OCR0 ASSR SFIOR TIMSK TIFR	TCNT1,TCCR1A, TCCR1B,TCCR1C, TCNT1H,TCNT1L, OCR1AH,OCR1AL, OCR1BH,OCR1BL, OCR1CH,OCR1CL ICR1H, ICR1L SFIOR TIMSK, ETIMSK, TIFR,ETIFR	TCCR2 TCNT2 OCR2 SFIOR TIMSK TIFR	TCNT3,TCCR3A, TCCR3B,TCCR3C, TCNT3H, TCNT3L, OCR3AH,OCR3AL, OCR3BH,OCR3BL, OCR3CH,OCR3CL ICR3H, ICR3L SFIOR TIMSK, ETIMSK, TIFR,ETIFR
동작 모드	일반모드 CTC 고속 PWM PC PWM	일반모드 CTC 고속 PWM PC PWM PFC PWM	일반모드 CTC 고속 PWM PC PWM	일반모드 CTC 고속 PWM PC PWM PFC PWM

타이머/카운터의 동작

타이머/카운터	타이머/카운터 0	타이머/카운터 1	타이머/카운터 2	타이머/카운터 3
입력 신호	TOSC1,TOSC2	T1, ICP1	T2	T3, ICP3
출력 신호	OC0	OC1A, OC1B, OC1C	OC2	OC3A, OC3B, OC3C
인터럽트	오버플로우, 출력 비교 일치	오버플로우, 출력 비교 일치 A/B,C 입력 캡쳐	오버플로우, 출력 비교 일치	오버플로우, 출력 비교 일치A/B,C 입력 캡쳐
기타	RTC 기능, 타이머/카운터 모두 프리스케일러를 사용	캡쳐 기능	OC2 단자는 OC1C 단자와 동일한 핀을 사용함.	캡쳐 기능

- **타이머/카운터0과 2 : 8비트 타이머/카운터**
- **타이머/카운터1과 3 : 16비트 타이머/카운터**
- **타이머/카운터0은 32.768kHz의 수정 발진자를 접속하는 TOSC1 및 TOSC2 단자를 가지고 있어서 RTC(Real Time Clock)의 기능을 구현할 수 있으며 외부의 클럭 입력단자 T0가 없는 대신에 TOSC1 단자를 통해 카운터 클럭을 인가할 수도 있음.**

8비트 타이머/카운터2의 동작

타이머/카운터2의 개요

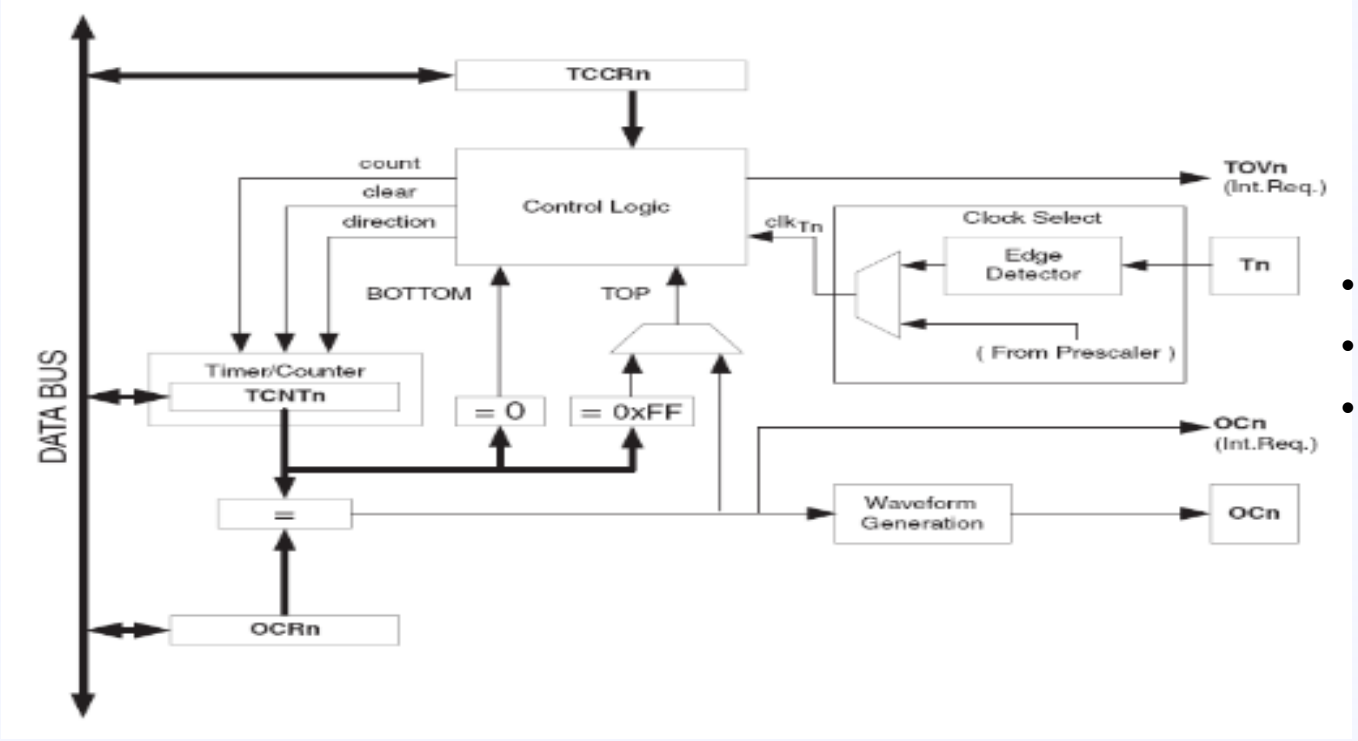
- 타이머/카운터2는 PWM 출력을 가지고 있는 8비트 타이머/카운터로서 프리스케일러를 통하여 내부 클럭을 입력으로 사용하여 동작하는 타이머 기능과 외부 클럭을 입력으로 사용하는 카운터 기능을 수행.

특징

- 단일 채널의 카운터
- 특정값과 비교하여 일치하면 타이머의 계수 값을 자동으로 클리어하는 CTC(Clear Timer on Compare match) 모드 -auto reload 모드
- 글리치없는 Phase Correct PWM 주파수 발생기
- 주파수 발생기
- 외부 사건 카운터
- 10비트 클럭 프리스케일러
- 2개의 인터럽트 소스 : 오버플로우, 비교 일치 인터럽트 발생(TOV2와 OCF2)

8비트 타이머/카운터2의 동작

타이머/카운터2의 내부 구성

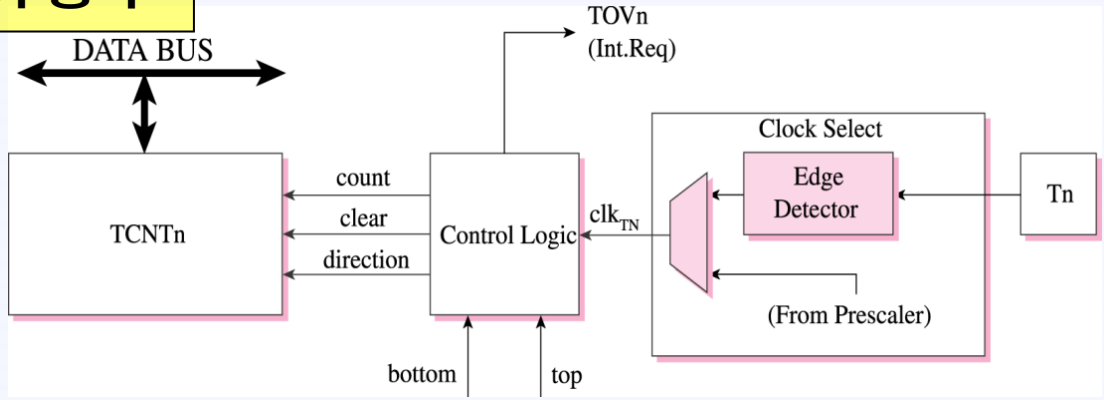


- 카운터
- 출력 비교부
- 비교 일치 출력부

- bottom : 8비트 타이머/카운터가 가질 수 있는 최소값(0x00)
- max : 8비트 타이머/카운터가 가질 수 있는 최대값(0xFF)
- top : 동작 모드에 따라 타이머/카운터가 도달할 수 있는 최대값으로 max 또는 OCR2 레지스터의 설정값

8비트 타이머/카운터2의 동작

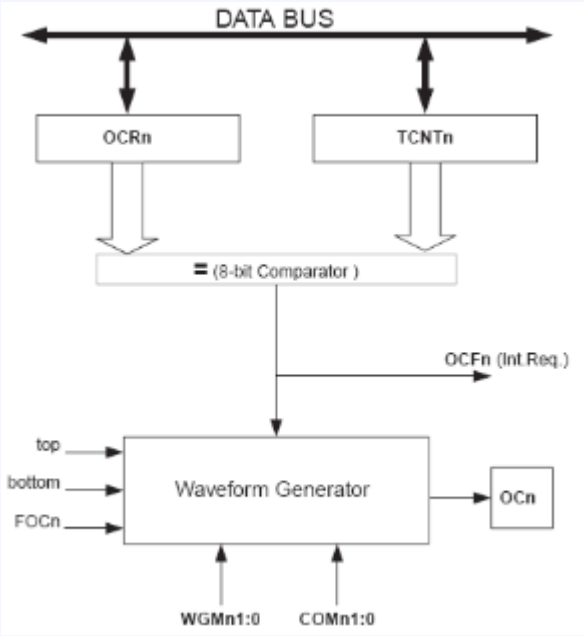
카운터부의 동작



- T2핀에 의해 공급되는 외부클럭과 프리스케일러를 통해 공급되는 시스템 클럭에 의해 구동
- 두 개의 클럭 신호는 클럭 선택 논리부에서 선택되어 카운터로 입력
- 즉, 타이머/카운터2는 클럭 선택 논리로부터 출력되는 클럭 신호 클럭 clk_{T2}를 입력받아 동작하는 8비트 업/다운 카운터(TCNT2)로 동작
- 업 카운터로 동작할 때에는 0xFF에서 0x00으로 계수의 값이 천이될 때 오버플로우(TOV2)인터럽트가 발생
- TCNT2카운터는 사용되는 동작 모드에 따라 클럭 clk_{T2}의 입력에 의해 값이 증가,감소하고, 경우에 따라서는 0x00으로 리셋 되기도 함
- 클럭의 선택 : TCCR2의 클럭 선택 비트(CS22 ~ CS20)에 의해 결정
- 카운터의 계수 동작과 TOV2 플래그 : TCCR2의 파형 발생 모드 비트(WGM21, WGM20)에 의해 결정
- TOV2 플래그는 ATmega128의 인터럽트를 발생시키는데 사용될 수 있음.

8비트 타이머/카운터2의 동작

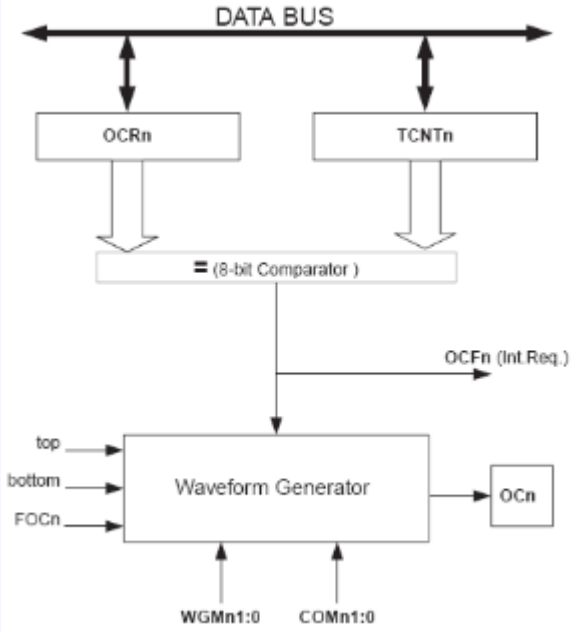
출력 비교부의 동작



- 타이머/카운터2 레지스터(TCNT2)의 값과 이중 버퍼의 구조를 가진 출력 비교 레지스터(OCR2)의 값은 계수 동작중에 항상 비교되며, 비교의 결과가 일치(compare match)하면 출력 비교 플래그(Output Compare Flag2, OCF2)를 세트하고, 이 신호에 의하여 출력 비교 인터럽트를 요청
- 또한 이 플래그를 이용하여 외부 핀 OC2 단자에 신호가 출력되도록 설정할 수 있음
- 타이머/카운터의 동작 모드를 적절히 설정하고 TCNT2 레지스터와 OCR2 레지스터를 활용하면 OC2 단자에 PWM 펄스를 발생하거나 가변 주파수의 펄스를 만들어 출력할 수 있게 됨
- OC2 단자의 출력 파형 결정 : TCCR2의 비교일치 출력 모드 비트(COM21 ~ 20)와 파형 발생 모드 비트 (WGM21, WGM20)의 설정에 의해 top과 bottom 신호를 결합하여 발생.

8비트 타이머/카운터2의 동작

출력 비교부의 동작(2)



- 출력비교 레지스터 OCR2는 PWM 동작 모드로 동작할 경우에는 이중 버퍼 구조로 동작하고, 일반 모드나 CTC 모드에서는 이중 버퍼 구조가 해제됨.
- PWM 모드에서는 이중 버퍼 구조에 의해 OCR2 레지스터의 값이 TCNT2가 top 또는 bottom에 도달하였을 때 갱신되므로, 이러한 동기화로 인해 홀수 길이 또는 비대칭의 PWM 펄스가 생성되지 않도록 하여 PWM 펄스에 글리치(glitch)가 발생하는 것을 방지할 수 있음.
- OCR2 레지스터를 액세스하는 과정 : CPU가 자동으로 처리함. 즉, 이중 버퍼 구조를 사용하는 모드에서는 OCR2 버퍼 레지스터를 액세스하고 이중 버퍼 구조가 사용되지 않는 모드에서는 OCR2 레지스터를 직접 액세스함.
- FOC2 신호 : 이는 PWM 파형 발생 모드가 아닌 경우에 사용되며, 이 신호는 TCCR2의 강제 출력 비교(FOC2) 비트를 1로 설정하면 발생하며, 비교기의 결과가 일치하였다고 강제로 설정하는 기능을 함.

타이머/카운터2의 레지스터

타이머/카운터2 레지스터	설 명
TCCR2	Timer/Counter 2 제어 레지스터
TCNT2	Timer/Counter 2 레지스터
OCR2	출력 비교 레지스터 2
SFIOR	특수 기능 I/O 레지스터
TIMSK	타이머/카운터 인터럽트 마스크 레지스터
TIFR	타이머/카운터 인터럽트 플래그 레지스터

8비트 타이머/카운터2의 동작

타이머/카운터2 제어 레지스터 : TCCR2

- TCCR2 레지스터(Timer/Counter Control Register 2) : 타이머/카운터0의 동작 모드, 프리스케일러의 분주비 설정 등의 기능을 수행

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
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	

- 비트7: FOC2(강제 출력 비교,Force Output Compare)
  - PWM 모드가 아닌 경우에만 유효, 이를 1로 설정하면 강제로 즉시 OC2 단자에 출력 비교가 일치된 것과 같은 출력을 내보내는 기능을 함
  - 유의할 점은 FOC2 비트는 스트로브 신호로 출력된다는 것과 특별한 경우가 아니라면 이 비트는 0으로 설정함.
- 비트 6,3 : WGM21~WGM20(파형 발생 모드, Waveform Generation Mode)
  - 카운터의 계수 순서와 카운터의 최대값, 파형 발생의 형태 등의 카운터의 동작 모드를 설정
  - 카운터/타이머2는 일반 모드, CTC모드, PWM모드, 고속PWM모드가 있음

8비트 타이머/카운터2의 동작

< WGM21~20비트에 의한 파형 발생모드의 설정 >

모드	WGM21 (CTC)	WGM20 (PWM)	동작모드	최대값	OCR2레지스터의 업데이트 시기	TOV2 플래그의 세트 시점
0	0	0	일반	0xFF	설정 즉시	MAX
1	0	1	PWM Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR2	설정 즉시	MAX
3	1	1	고속 PWM	0xFF	TOP	MAX

- **비트 5,4 : COM21~COM20(비교 일치 출력 모드, Compar Match Output Mode)**
  - **OC2 핀의 동작을 설정하는 기능을 담당**
  - **OC2 단자를 통해 PWM과 같은 출력 신호를 I/O포트 핀으로 출력하려면 이 두 비트 중에 하나 또는 모든 비트가 1로 세트되어 있어야 함.**
    (단, 이 경우에는 해당 병렬 I/O포트를 입출력 방향 제어 레지스터DDRx에서 출력 방향으로 미리 설정하여야 함.)
  - **WGM21~WGM20비트의 설정에 따라 타이머/카운터2의 동작 모드가 다름**

8비트 타이머/카운터2의 동작

한국공학대학교 TECH UNIVERSITY OF KOREA

< PWM 모드가 아닌 경우의 비교 출력 모드 >

COM21	COM20	OC2 핀의 기능
0	0	범용 I/O포트로 동작(OC2출력을 차단)
0	1	비교 일치에서 OC2출력을 토글
1	0	비교 일치에서 OC2출력을 0으로 클리어
1	1	비교 일치에서 OC2 출력을 1로 세트

< 고속 PWM 모드에서의 비교 출력 모드 >

COM21	COM20	OC2 핀의 기능
0	0	범용 I/O포트로 동작(OC2출력을 차단)
0	1	사용하지 않음(reserved)
1	0	비교 일치에서 OC2출력을 0으로 클리어 하고 top에서 OC2를 1로 세트함(비반전 비교 출력 모드)
1	1	비교 일치에서 OC2 출력을 1로 세트하고 top에서 OC2를 0으로 클리어함(반전 비교 출력 모드)

8비트 타이머/카운터2의 동작

< PC PWM 모드에서의 비교 출력 모드 >

COM21	COM20	OC2 핀의 기능
0	0	범용 I/O포트로 동작(OC2출력을 차단)
0	1	사용하지 않음(reserved)
1	0	상향 카운팅 경우에서는 비교 일치에서 OC2을 0으로 클리어 하고 하향 카운팅 경우에서는 비교 일치에서 OC2을 1로 세트함
1	1	상향 카운팅 경우에서는 비교 일치에서 OC2을 0으로 클리어 하고 하향 카운팅 경우에서는 비교 일치에서 OC2을 1로 세트함

➢ **비트 2~0 : CS22~CS20(클럭 선택, clock select)**

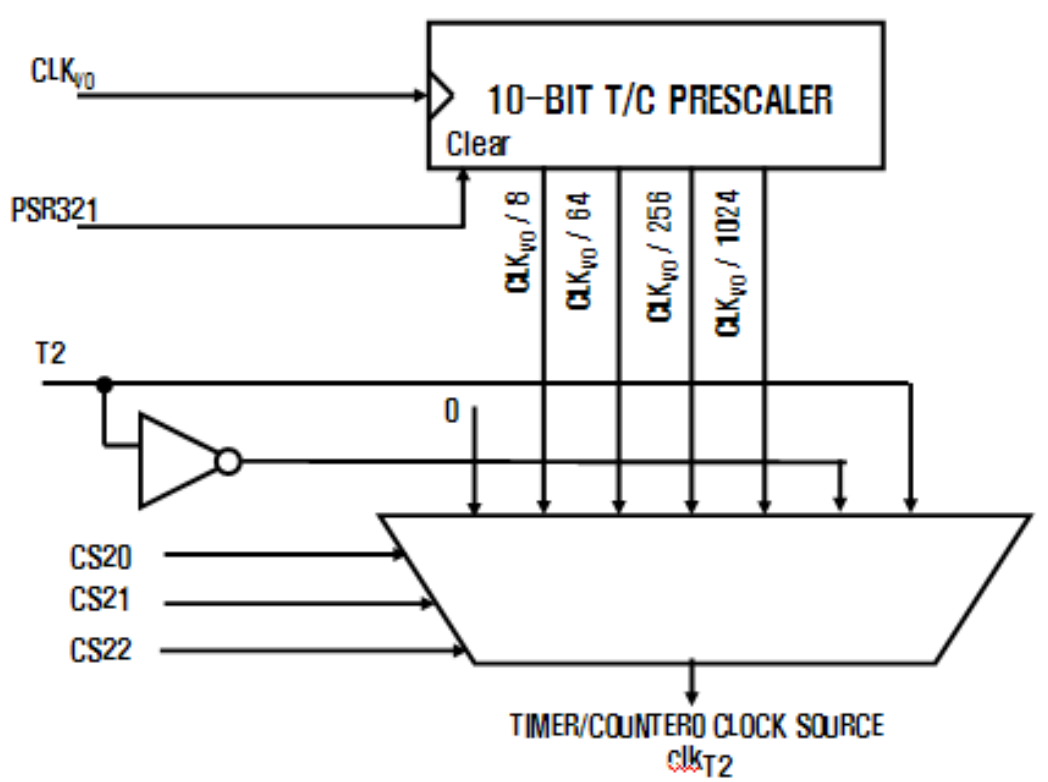
- **타이머/카운터2에서 사용할 클럭을 선택하는 기능을 담당**

CS22	CS21	CS20	클럭 소스의 기능
0	0	0	클럭 소스 차단(타이머/카운터 기능이 정지)
0	0	1	$ClK_{I/O}$
0	1	0	$ClK_{I/O}/8$
0	1	1	$ClK_{I/O}/64$
1	0	0	$ClK_{I/O}/256$
1	0	1	$ClK_{I/O}/1024$
1	1	0	T2 핀에 연결된 외부 클럭 소스, 클럭은 하향 에지에서 동작
1	1	1	T2 핀에 연결된 외부 클럭 소스, 클럭은 상승 에지에서 동작

8비트 타이머/카운터2의 동작

한국공학대학교
TECH UNIVERSITY OF KOREA

< 타이머/카운터2의 프리스케일러 구성도 >



8비트 타이머/카운터2의 동작

타이머/카운터0 레지스터 : TCNT2

- **TCNT2(Timer/Counter Register 2) 레지스터 : 타이머/카운터2의 8비트 카운터 값을 저장하는 레지스터**
  - 언제나 읽기 및 쓰기 동작이 가능하지만 카운터가 동작하고 있을 동안에 이 값을 수정하면 TCNT2값과 OCR2값을 비교하여 출력 신호를 발생하는 비교 일치 기능에 문제를 일으킬 수도 있음.

Bit	7	6	5	4	3	2	1	0	
	TCNT2[7:0]								TCNT2
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	

8비트 타이머/카운터2의 동작

타이머/카운터2 출력 비교 레지스터 : OCR2

- OCR2(Timer/Counter Output Compare Register 2)레지스터 : TCNT2값과 비교하여 OC2단자에 출력 신호를 발생하기 위해 8비트의 값을 저장하는 레지스터
  - OCR2 레지스터는 모든 PWM모드에서 이중 버퍼 구조로 동작 → TCNT20이 top 또는 bottom에 도달하였을 때 그 값이 갱신되므로, PWM 출력 신호에는 글리치가 발생하지 않음
  - 일반 CTC모드에서는 OCR2 레지스터는 이중 버퍼 구조로 동작하지 않음

Bit	7	6	5	4	3	2	1	0	
	OCR2[7:0]								OCR2
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	

8비트 타이머/카운터2의 동작

타이머/카운터 인터럽트 마스크 레지스터 : TIMSK

- TIMSK(timer/Counter Interrupt Mask Register)레지스터 : 타이머/카운터0과 타이머/카운터2의 인터럽트의 발생 여부를 개별적으로 허용하는 기능을 담당(OCIE2와 TOIE2비트)

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	

- 비트6: TOIE2(타이머/카운터2 오버플로우 인터럽트 허가 비트)
  - TOIE2 비트가 1로 설정되고 상태 레지스터(SREG)의 I 비트가 1로 설정되면 타이머/카운터2의 오버플로우 인터럽트가 허용상태로 됨.
  - 이 상태에서 타이머/카운터2의 오버플로우 인터럽트가 발생하면, TIFR레지스터의 TOV2 비트가 1로 설정되어 인터럽트가 발생하고, 이에 해당하는 인터럽트 벡터를 참조함.
- 비트 7 : OCIE2 (타이머/카운터2 출력 비교 인터럽트 허가 비트)
  - OCIE2 비트가 1로 설정되고 상태 레지스터(SREG)의 I 비트가 1로 설정되면 타이머/카운터2의 출력 비교 인터럽트가 허용상태로 됨.
  - 이 상태에서 타이머/카운터2의 출력 비교 인터럽트가 발생하면, TIFR 레지스터의 OCF2 비트가 1로 설정되어 인터럽트가 발생하고, 이에 해당하는 인터럽트 벡터를 참조함.

8비트 타이머/카운터2의 동작

타이머/카운터 인터럽트 플래그 레지스터 : TIFR

- TIFR(Timer/Counter Interrupt Flag Register) 레지스터 : 타이머/카운터0과 타이머/카운터2의 인터럽트가 발생하면 인터럽트의 상태를 저장하고 있는 레지스터.
  - 타이머/카운터0의 인터럽트는 TOV2와 OCF2 비트임.

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	

➢ 비트 6 : TOV2(타이머/카운터2 오버플로우 플래그)

- TOV2 비트는 타이머/카운터2에서 오버플로우가 발생하면 이 비트는 1로 세트되면서 오버플로우 인터럽트를 요청함.
- 인터럽트 벡터에 의해 인터럽트가 실행되면 이 비트는 하드웨어에 의해 자동적으로 클리어됨.
- TOV2 비트는 소프트웨어에 의해 클리어될 수 있음. TOV2 비트에 1을 쓰면 클리어됨.
- PC PWM 모드에서는 타이머/카운터2이 0x00에서 계수방향을 바꿀 때 이 비트가 세트된다는 것도 주의하여야 할 사항임.

- 비트 7 : OCF2 (출력 비교 플래그 2)
  - OCF2 비트는 타이머/카운터0의 TCNT2 레지스터와 출력 비교 레지스터 OCR2의 값을 비교하여 이것이 같으면 다음 주기에서 이 비트가 1로 세트되면서 인터럽트가 요청됨.
  - 인터럽트 벡터에 의해 인터럽트가 실행되면 이 비트는 하드웨어에 의해 자동적으로 클리어됨.
  - OCF2 비트는 TOV2 비트와 마찬가지로 소프트웨어에 의해 클리어될 수 있음. → OCF2 비트에 1을 쓰면 클리어됨.
  - 타이머/카운터2 출력 비교 인터럽트는 SREG의 I비트, OCIE2 비트와 OCF2 비트가 모두 1로 되어 있을 때에 만 실행 가능하다는 것을 유의를 하여야 함.

특수 기능 I/O 레지스터 : SFIOR

- SFIOR 레지스터는 타이머/카운터들을 동기화 시키는데 관련된 기능을 수행

Bit	7	6	5	4	3	2	1	0	
	TSM	–	–	–	ACME	PUD	PSR0	PSR321	SFIOR
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- 비트 0 : PSR321(Prescaler Reset Timer/Counter3,2,1)

타이머/카운터2의 동작 모드

- 동작 모드 : 4가지 동작 모드(표 7.3 참조)
- TCCR2 레지스터의 파형 발생 모드 비트(WGM21 ~ WGM20)비트에 의해 결정
- 출력파형의 형태는 비교 출력 모드 비트 (COM21~COM20)에 의해 출력 신호의 동작이 결정됨.
  - 비교 출력 모드 비트(COM21~COM20)는 계수 순서에는 영향을 주지 않고, PWM 출력을 반전시킬 것인지 아니면 그대로 출력할 것인지를 제어하는데 활용됨.
  - PWM 모드가 아닌 경우에는, COM21~20 비트는 비교 일치시에 출력이 세트가 되는지, 클리어되는지 또는 토글되는지를 제어하는데 사용됨.

8비트 타이머/카운터2의 동작

일반 모드

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
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	

- WGM21~WGM20 비트를 ‘00’ 으로 설정
- TNCT2은 항상 상향 카운터로만 동작
- 타이머/카운터2의 값이 0xFF에서 0x00으로 바뀌는 순간에 TIFR레지스터의 TOV2비트가 1로 세트 되면서 오버플로우 인터럽트가 발생하며 인터럽트 서비스 루틴을 수행하면, 자동으로 TOV2비트는 0으로 리셋됨.
- 일반 모드 동작에서 출력 비교 인터럽트가 사용될 수는 있으나 CPU가 이를 처리하는데 너무 많은 시간이 필요하므로 이 모드에서는 출력 비교 인터럽트를 사용하는 것은 바람직하지 않음. → 일반 모드는 타이머/카운터를 외부 클럭을 계수하는 목적으로 사용하는 것이 일반적임.

8비트 타이머/카운터2의 동작

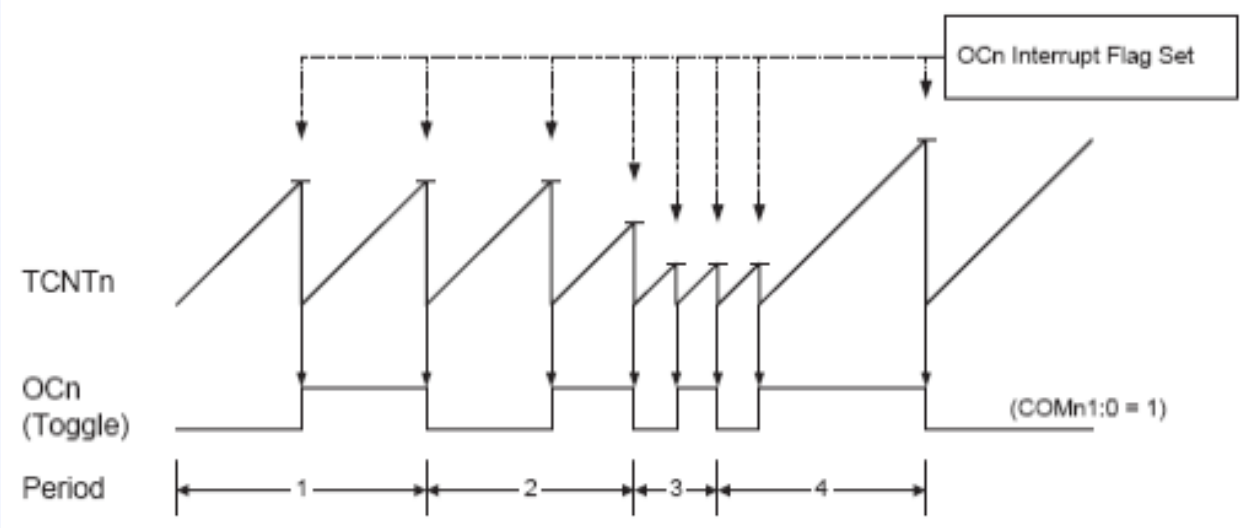
한국공학대학교 TECH UNIVERSITY OF KOREA

CTC 모드

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
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	

- WGM21~WGM20비트를 ‘10’으로 설정
- TCNT2의 값이 클럭의 입력에 따라서 증가하여 출력 비교 레지스터OCR2의 값과 같아지면 그 다음 클럭 사이클에서 0으로 클리어 됨
- 즉 CTC 모드의 동작은 클럭 입력에 의해서 타이머/카운터 레지스터 TCNT2의 값이 항상 0x00~OCR2의 값으로 계수동작이 반복하여 수행되며, 이러한 동작을 반복할 경우에 주기적인 펄스를 만들 수 있음(자동 재적재 모드) → 주파수 분주의 용도로 사용

8비트 타이머/카운터2의 동작



- TCNT2 값은 펄스의 입력 됨에 따라 OCR2의 값과 비교하여 일치할 때 까지 증가
- TCNT2와 OCR2의 값이 일치 하면 TCNT2의 값은 0으로 클리어됨.
- 인터럽트가 허가되어 있으면 카운터의 값이 TOP->0x00으로 바뀌는 순간 OCF2가 1로 되면서 출력 비교 인터럽트가 발생
- 인터럽트 서비스 루틴에서는 TOP의 값을 갱신할 수도 있는데, 이 경우에 OCR2 레지스터는 이중 버퍼링 기능이 없으므로 OCR2 레지스터에 현재 계수된 TCNT2의 값보다 작은 값을 쓰게 되면 해당 주기에서는 출력 비교가 일어나기 전에 카운터의 값이 MAX인 0xFF까지 증가하였다가 0x00으로 되는 상황이 발생한다. 따라서 이 경우에는 출력 비교 인터럽트가 발생하기 전에 TIFR 레지스터의 TOV2 비트가 1로 되면서 오버플로우 인터럽트가 발생하게 되므로 조심하여야 한다.

- CTC 모드에서의 OC2 단자로의 파형의 출력
  - COM21~COM20비트 = ’ 01’ 로 설정하면, OC2 단자에서 출력되는 파형의 출력 주파수는 다음과 같이 계산됨

$$f_{OC2} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCR2)}$$

  - N은 프리스케일러의 분주비로 1,8,64,256,1024 중의 하나를 사용
  - OC2의 출력 단자에 파형을 출력하기 위해서는 I/O 포트의 입출력 방향 제어 레지스터(DDRx)를 미리 출력 방향으로 지정을 하여야 함
  - CTC 모드에서 파형의 최대 주파수는 OCR2가 0으로 설정될 때 발생하며, 이때의 주파수는 다음과 같음.

$$f_{OC2} = f_{clk\_I/O}/2$$

  - CTC 모드에서는 출력 비교 인터럽트가 2번 발생하여야만 1주기의 출력 파형을 만들 수 있으므로 인터럽트에 의한 발생 주파수는 이 보다 2배 높게 된다는 것을 유념하여야 함.

8비트 타이머/카운터2의 동작

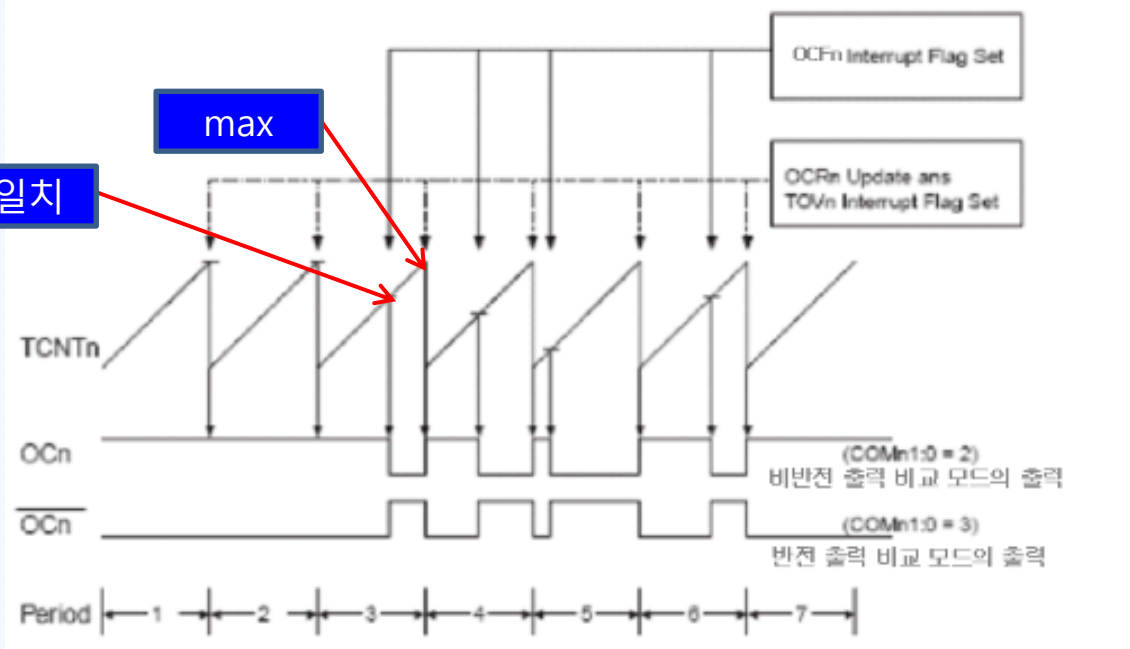
고속 PWM 모드

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
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	

- WGM21~WGM20 비트를 ‘11’로 설정
- 높은 주파수의 PWM 출력 파형을 발생하는데 유용하게 사용됨.
- TCNT2 레지스터의 값이 항상 bottom에서 시작하여 max까지 증가하는 방향으로만 반복하여 수행됨. : 단방향 경사 동작(single-slope operation)
- COM21~COM20 비트의 설정에 따라 비반전 비교 출력 모드와 반전 비교 출력 모드로 동작

8비트 타이머/카운터2의 동작

- **비반전 출력 비교 모드**
  - **COM21~COM20 = ‘10’**
  - **TCNT2값이 bottom에서 max까지 증가하며, OCR2의 값과 비교하여 일치하면 OCF2인터럽트 플래그가 세트되고 OC2의 출력은 0으로 클리어됨.**
  - **또한 TCNT2 값이 max로 되면, TOV2 인터럽트 플래그는 1로 되면서 오버플로우 인터럽트가 요청되고, 이때 OC2의 출력은 1로 세트됨.**
  - **이 때 인터럽트가 허가되어 있으면, 인터럽트 서비스 루틴에서 이 비교값은 변경될 수 있음.**



8비트 타이머/카운터2의 동작

- 반전 출력 비교 모드
  - COM21~COM20 = ‘11’
  - TCNT2의 값과 OCR2의 값이 일치할 때 $\overline{OC2}$의 출력은 1로 세트되고, TCNT2 값이 max로 될 때 TOV2 플래그는 1로 되면서 오버플로우 인터럽트가 요청되고, 이때 $\overline{OC2}$의 출력은 0으로 클리어됨.

- OC2 단자로의 파형의 출력
  - 표 7.5에 나타난 것과 같이 출력 비교 모드를 COM21 ~ COM20 비트를 미리 설정하여 놓아야 함.
  - 비반전 출력 비교 모드에서는 정상적인 PWM 신호가 출력되고, 반전 출력 비교 모드에서는 반전된 PWM 신호가 출력됨.
  - OC2의 출력 단자에서 파형을 출력하기 위해서는 I/O 포트의 입출력 방향 제어 레지스터를 미리 출력 방향으로 설정하여 놓아야 함.

8비트 타이머/카운터2의 동작

- PWM 파형의 주파수

$$f_{OC2PWM} = \frac{f_{clk\_I/O}}{N \cdot 256}$$

  - PWM 파형은 TCNT2 레지스터와 OCR2 레지스터가 비교 일치될 때 OC2 레지스터를 세트 또는 클리어하고, TCNT2의 값이 0으로 되는 시점에서 OC2 레지스터를 클리어 또는 세트하는 과정으로 발생됨.
  - PC PWM 모드에 비해 약 2배 정도 높은 주파수를 얻을 수 있음.
  - PWM 사이클은 OCR2 레지스터의 값에 의해 결정
    OCR2 =0x00으로 설정, TCNT2=0x00으로 되는 1타이머 사이클 동안에만 좁은 스파이크로 나타난다(듀티비 1/256)
    OCR2=0xFF으로 설정, 출력파형 OC2는 계속 1로 출력 (듀티비 100%)
  - 50%의 듀티비를 갖는 출력 신호를 얻는 방법
  - COM21 비트가 1로 설정되고, OCR2가 max 값으로 설정된 경우에는 비교 일치 동작은 무시되고, top에서 OC2의 값을 1 또는 0으로 설정할 수 있음. → 비교 일치가 발생할 때마다 OC2의 출력을 토글시키면 됨.

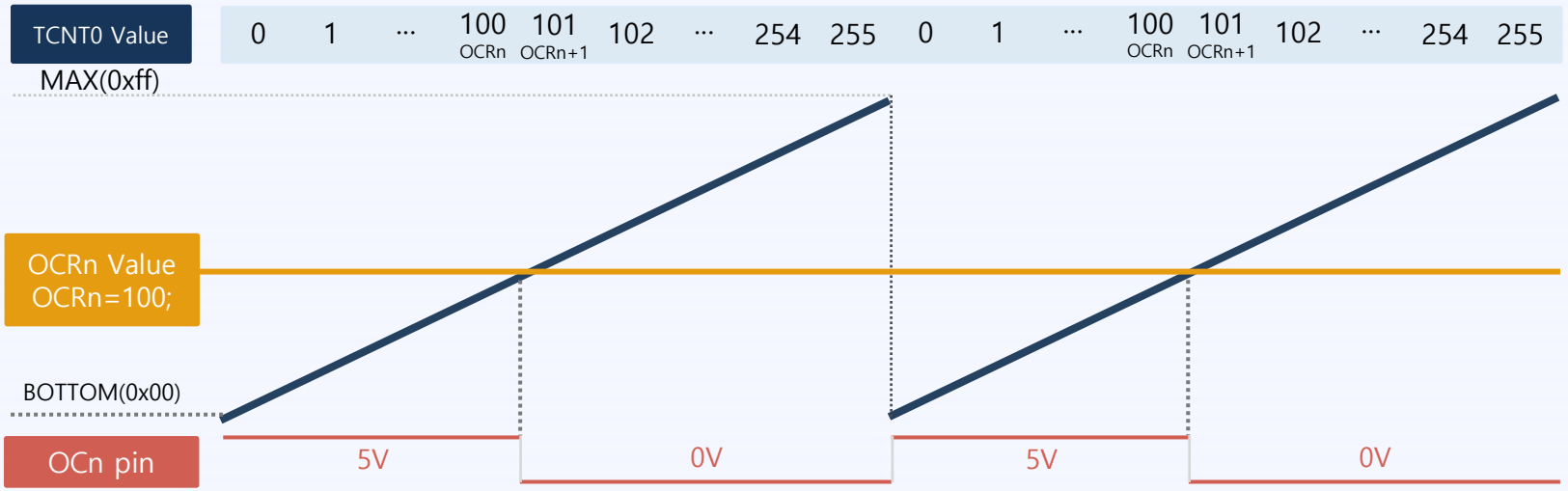
$$f_{OC2} = f_{clk\_I/O}/2$$

8비트 타이머/카운터2의 동작

한국공학대학교 TECH UNIVERSITY OF KOREA

Fast PWM 모드 정리

- TCNTn: 0~255, 0~255・・・
- OCn pin: 0~OCRn → 5V 출력, OCRn + 1 ~255 → 0V 출력 (단, COMn 1:0 이'10'일 때)
- 예) OCRn = 100 (단, 비반전 비교 출력 모드인 경우, COMn 1:0 이 '10'일 때)



LED	ON	OFF	ON	OFF
	0~100 count (101개)	101~255 count (155개)	0~100 count (101개)	101~255 count (155개)
LED ON-OFF Time	$\frac{1}{16MHz}$X101 = 6.3125[μS]	$\frac{1}{16MHz}$X155 = 9.6875[μS]	$\frac{1}{16MHz}$X101 = 6.3125[μS]	$\frac{1}{16MHz}$X155 = 9.6875[μS]

Period (cycle Time)	$T = \frac{1}{16MHz}$X256 = 16[μS]	$T = \frac{1}{16MHz}$X256 = 16[μS]

Frequency

$$f = \frac{f_{dk\_I/O}}{N \cdot 256}$$

- 여기서 N은 분주비로 1, 8, 64, 256, 1024(Timer/Counte2) / 1, 8, 32, 64, 128, 256, 1024(Timer/Counte0)
- 여기 $f_{dk_I/O}$는 시스템 클럭과 같은 주파수 값(16MHz)

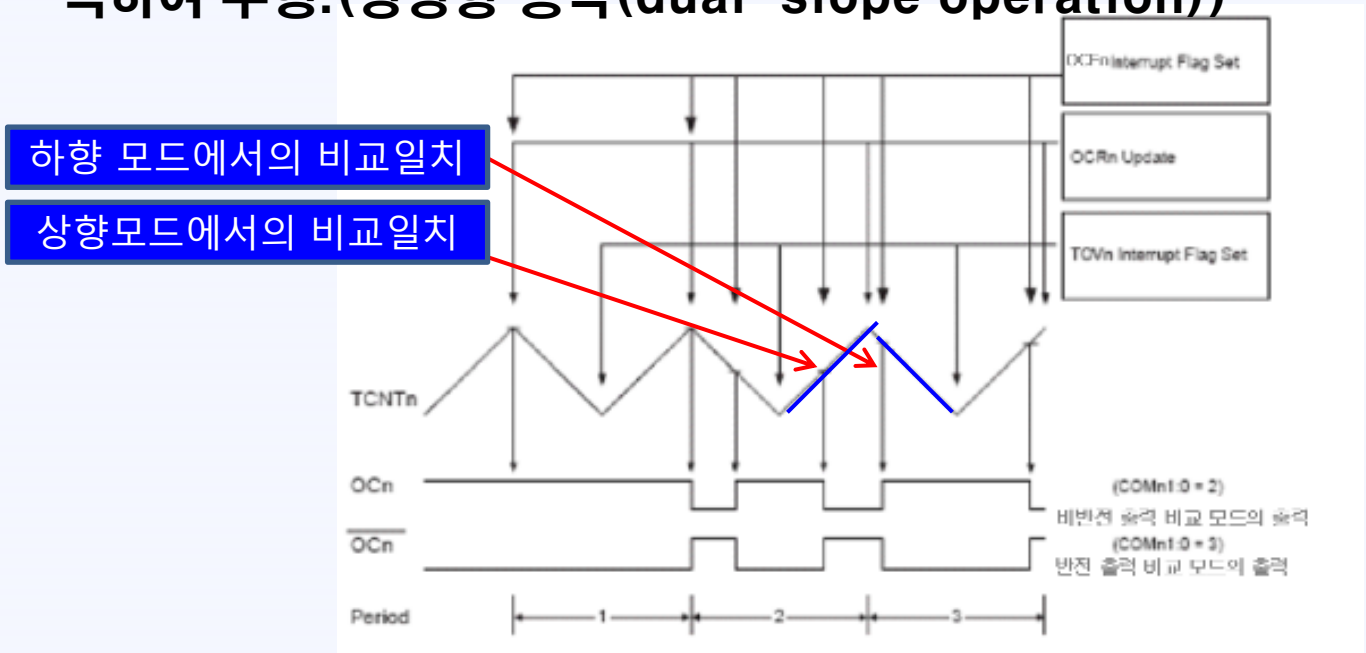
8비트 타이머/카운터2의 동작

한국공학대학교 TECH UNIVERSITY OF KOREA

PC(Phase Correct) PWM 모드

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
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	

- WGM21~WGM20 비트를 '01' 로 설정
- 높은 분해능의 PWM 출력 파형을 발생하는데 유용하게 사용됨.
- TCNT2가 상향 카운터로서 bootom(0x00)에서 max(0xFF)까지 증가하였다가 다시 하향 카운터로서 max서 bootom으로 감소를 하는 동작을 반복하여 수행.(양방향 동작(dual-slope operation))



- 타이머/카운터0의 PC PWM 모드 동작 타이밍도 -

8비트 타이머/카운터2의 동작

- **비반전 출력 비교 모드**
  - COM21~COM20 = ‘10’
  - 상향 모드와 하향 모드로 동작함.
  - 상향 모드 : TCNT2 = OCR20이 되면 -> OC2 = 0, OCF2 인터럽트 플래그가 세트됨.
  - 하향 모드 : TCNT2 = OCR20이 되면 -> OC2 = 1, OCF2 인터럽트 플래그가 세트됨.
  - 이 과정에서 TOV2 인터럽트 플래그는 TCNT2 값이 bottom으로 될 때마다 1로 세트되면서 오버플로우 인터럽트가 요청됨.

- **반전 출력 비교 모드**
  - COM21~COM20 = ‘11’
  - 상향 모드 : TCNT2 = OCR20이 되면 -> OC2 = 1, OCF2 인터럽트 플래그가 세트됨.
  - 하향 모드 : TCNT2 = OCR20이 되면 -> OC2 = 0, OCF2 인터럽트 플래그가 세트됨.
  - 이 과정에서 TOV2 인터럽트 플래그는 TCNT2 값이 bottom으로 될 때마다 1로 세트되면서 오버플로우 인터럽트가 요청됨.

8비트 타이머/카운터2의 동작

PWM 파형의 주파수

- PWM 파형은 카운터가 상향동작을 하고 있는 과정에서 TCNT2 레지스터와 OCR2 레지스터가 비교 일치될 때 OC2 레지스터를 클리어 또는 세트하고, 다시 하향동작을 하면 TCNT2과 OCR20이 비교 일치될 때 OC2 레지스터를 세트 또는 클리어하는 과정으로 발생됨.

$$f_{OC2PCPWM} = \frac{f_{clk\_IO}}{N \cdot 510}$$

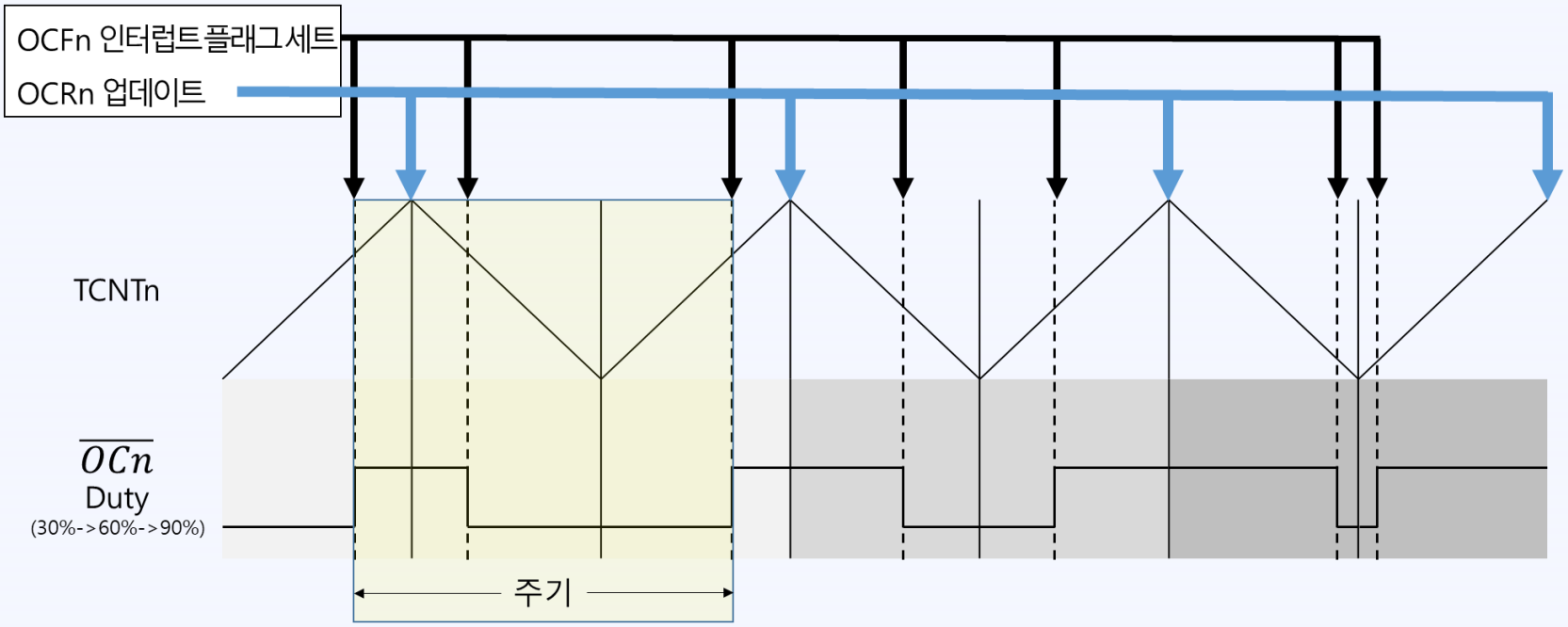
- N은 프리스케일러의 분주비로서 1, 8, 64, 256, 1024 중의 하나에 해당
- 이 모드에서의 PWM 주파수는 고속 PWM 모드의 경우에 비하여 약 1/2로 낮아지게 되는데, 이는 TCNT2의 값이 0x00 ~ 0xFF의 범위에서 증가하였다가 다시 0xFF ~ 0x00으로 감소하는 양방향 동작을 수행하기 때문.
- 이상과 같이 PC PWM 모드는 고속 PWM 모드에 비하여 주파수는 1/2로 낮아졌지만 고속 PWM 모드의 듀티비의 분해능이 8 비트인데 비하여 2배로 높아졌으며, 양방향의 PWM 펄스를 대칭적으로 만들 수 있는 장점으로 가지게 되어 모터 제어의 응용에 유용하게 사용됨.
- 듀티비는 OCR2=0x000이면 0%가 되고, OCR2=0xFF0이면, 100%가 됨.

비교 일치가 일어나지 않은 경우에도 OC2의 값이 바뀌는 경우

① OCR2 버퍼의 값이 max에서 다른 값으로 바뀌었을 때 OCR2의 값이 max일 때, OC2의 핀 값은 하향 계수시의 비교 일치일 때와 동일하다. 그러나 첫 번째 주기에서 OCR2 버퍼가 갱신된 값이 두 번째 주기의 상향모드의 값으로 바뀌었다면 두 번째 주기의 시작에서 OCR2가 버퍼의 값으로 갱신되었을 것이고, 따라서 그냥 두면 계속 1을 유지하므로 2번 주기의 bottom 근처에서 대칭성이 보장이 되지 않으므로, 대칭성을 보장하기 위해 max에서의 OC2값은 상향모드의 결과와 같아야 하므로 OC2가 1에서 0으로 바뀌게 됨

② 타이머가 OCR2 버퍼에 있는 값보다 큰 값에서 계수를 시작했을 때큰 값에서부터 계수를 시작했다면 이번 주기의 비교 일치를 한 번 놓치게 되고 그 다음 번에서도 대칭을 보장해야 하므로 OC2가 1에서 0으로 바뀌게 됨

PC PWM 보충자료



OCR 값 계산 방법 : [배경] PC PWM에서 발생되는 PWM 주기는 카운터 계수 510단계에서 비롯됨.

PC PWM은 업 카운터에서 OCn값은 '0'으로 클리어 되고, /OCn 값은 '1'로 세트 됨.

PC PWM은 다운 카운터에서 OCn값은 '1'로 세트 되고, /OCn 값은 '0'으로 클리어 됨.

1. 계수 총 단계인 510에 출력하고자 하는 듀티값(%)만큼 곱하여 비율적 단계를 계산함.
2. 타이머 카운터 TOP은 255이고 /OCn으로 출력되는 PWM 파형의 '1' 구간은 TOP을 기준으로 50:50 계수 단계를 갖음. 따라서 앞서 계산된 계수 단계 값을 2로 나눈다.
3. 단방향 카운터의 최대 값인 255에서 앞서 계산 된 값을 뺀다. (비반전의 경우 생략)

$\overline{OCn}$ 출력의 OCR 값 = 255 − (510 x Duty(%) / 2)

(ex: 듀티 90%), $26 \approx 255-(510 \times 0.9 / 2)$

PC PWM 보충자료

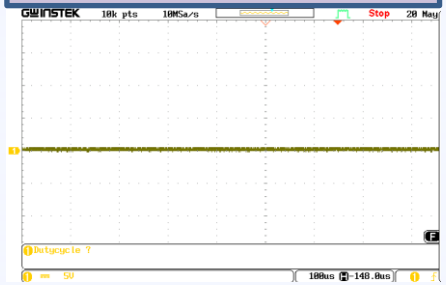
$\overline{OCn}$ 출력의 OCR 값 = 255 – (510 x Duty(%) / 2)
(ex: 듀티 90%), $26 \approx 255-(510 \times 0.9 / 2)$

OCn 출력의 OCR 값 = 510 x Duty(%) / 2
(ex: 듀티 90%), $229 \approx 510 \times 0.9 / 2$

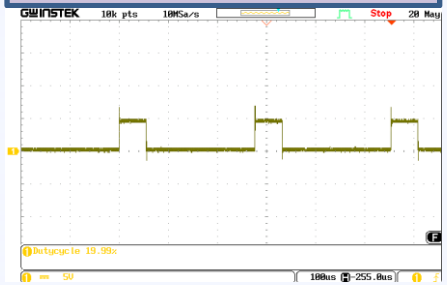
Duty Cycle	OCR	/OCR
0%	0	255
20%	51	204
30%	76	178
40%	102	153
60%	153	102
80%	204	51
90%	229	26
100%	255	0

OCR Duty Cycle 별 그래프

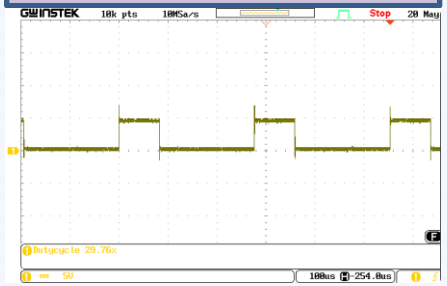
Duty:0%, /OCR:0



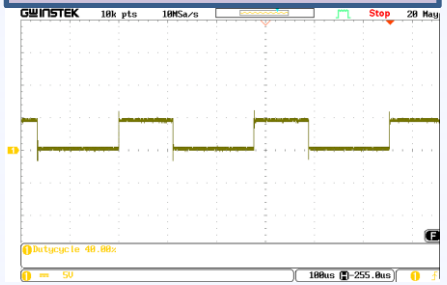
Duty:20%, /OCR:51



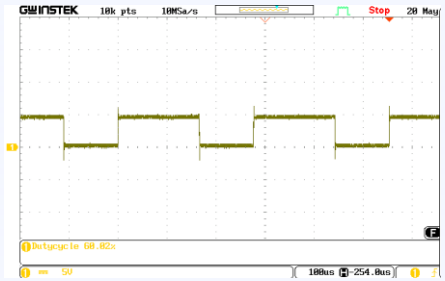
Duty:30%, /OCR:76



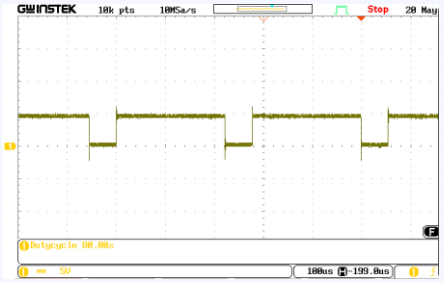
Duty:40%, /OCR:102



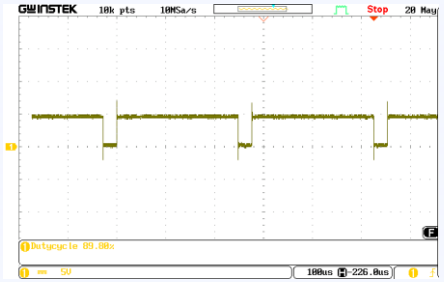
Duty:60%, /OCR:153



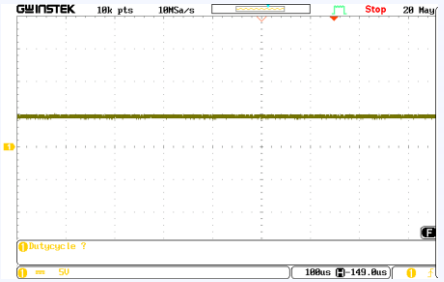
Duty:80%, /OCR:204



Duty:90%, /OCR:229

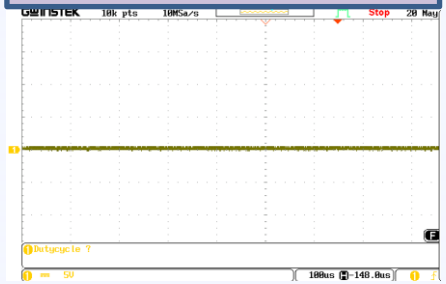


Duty:100%, /OCR:255



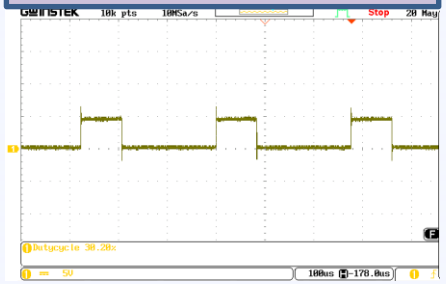
/OCR Duty Cycle 별 그래프

Duty:0%, /OCR:255

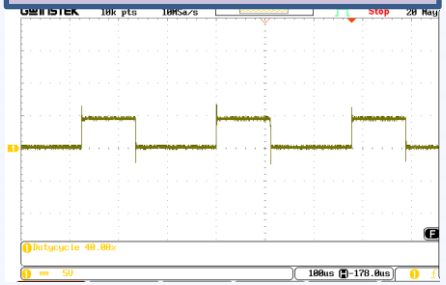


Duty:20%, /OCR:204

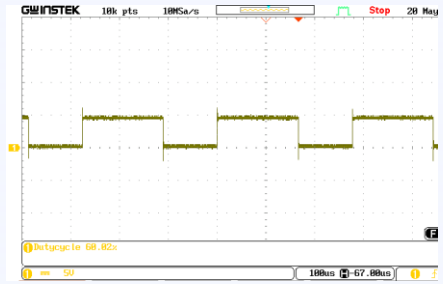
Duty:30%, /OCR:178



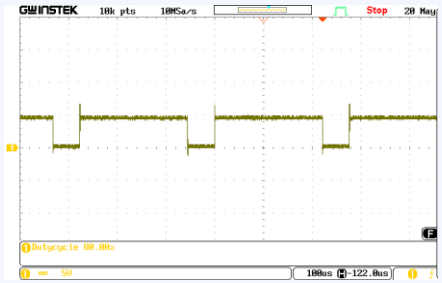
Duty:40%, /OCR:153



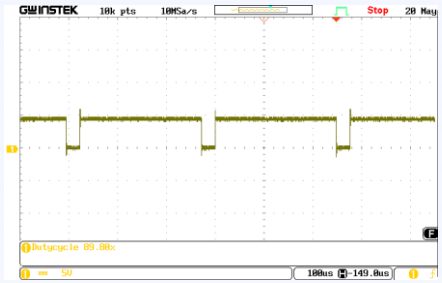
Duty:60%, /OCR:102



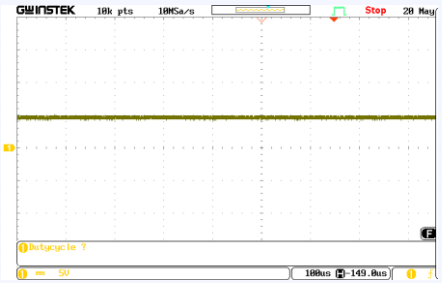
Duty:80%, /OCR:51



Duty:90%, /OCR:26



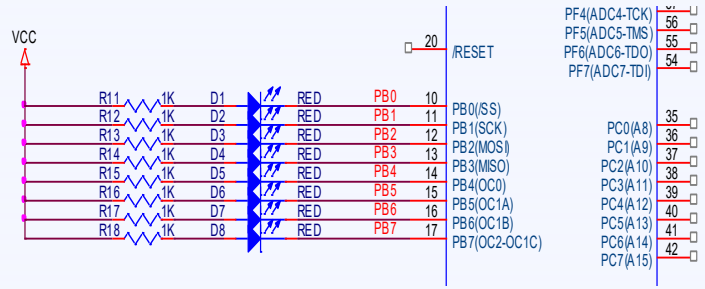
Duty:100%, /OCR:0



OCn, /OCn 의 선택적 활용 예시

OCn

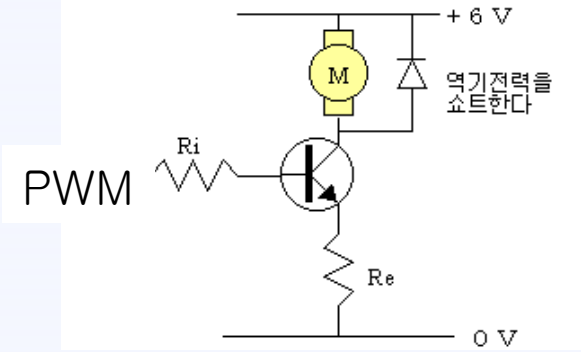
- 아래와 같은 회로의 LED 제어



-상기 회로에서 LED가 점등 되는 조건은 I/O출력이 '0' 이다. 이 같은 이유는 I/O 출력이 '1'일 경우 전위차가 발생하지 않기 때문이다. 따라서 OCRn 업데이트 시점 이후 출력 '0'인 구간을 조절하는 OCn 출력이 적합하다.

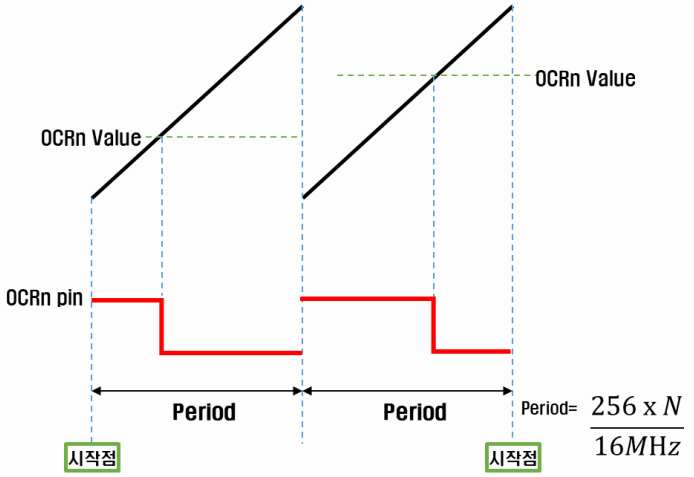
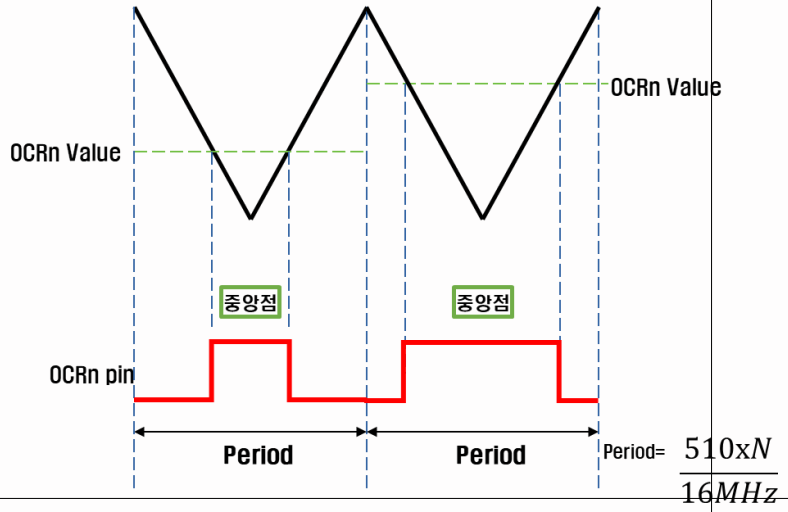
/OCn

- 아래와 같은 회로의 모터 제어



-상기 회로에서 모터가 구동 되는 조건은 I/O출력이 '1' 이다. 이 같은 이유는 I/O 출력이 '0'일 경우 전위차가 발생하지 않기 때문이다. 따라서 OCRn 업데이트 시점 이후 출력 '1'인 구간을 조절하는 /OCn 출력이 적합하다.

Fast/PC PWM Mode 비교

구분	Fast PWM Mode	Phase Correct PWM Mode	비고
TCNTn값	0.1…254.255. 0.1…254.255…	254,253 … 1.0.1. …254,255.254.253…	
Slope	Single Slope(Up Counter)	Double Slope(Up*Down Counter)	
OCn Pin 출력 주파수	$f_{oc_n\ pwm} = \frac{f_{clk\_I/O}}{N \cdot 256}$	$f_{oc_n\ pwm} = \frac{f_{clk\_I/O}}{N \cdot 510}$	주파수는 Fast PWM이 PC PWM보다 2배 크다.
출력 파형			
출력파형 특징	펄스이 시작이 동일하며, 시작점으로부터 펄스의 폭이 가변됨	펄스의 중앙이 일정하고 펄스의 중앙을 중심으로 펄스폭이 가변됨	

타이머/카운터0의 개요

- 타이머/카운터0 은 타이머/카운터2와 동일
- 프리스케일러를 통하여 내부/외부 클럭을 입력으로 사용하여 동작하는 타이머/카운터 기능을 수행
- 타이머/카운터0은 타이머/카운터2에 비해 32.678㎑의 외부 수정 발진자를 클럭 소스로 사용하여 계수할 수 있는 RTC(Real Time Clock or Counter) 기능과 비동기 레지스터(ASSR)를 가지고 있어 비동기 동작이 가능함.

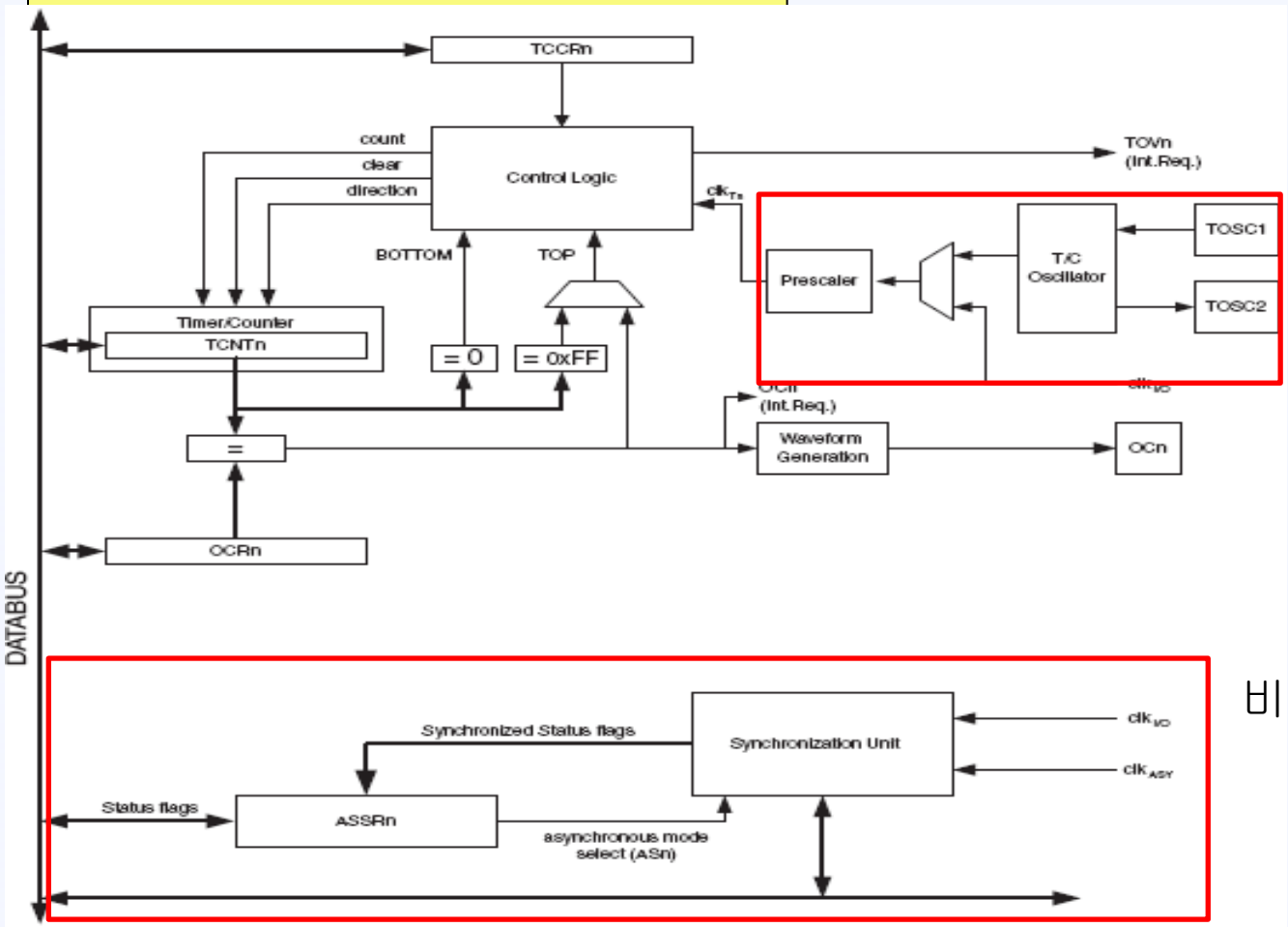
8비트 타이머/카운터0의 동작

특징

- 단일 채널의 카운터
- 특정값과 비교하여 일치하면 타이머의 계수 값을 자동으로 클리어하는 CTC(Clear Timer on Compare match) 모드-auto reload 모드
- 글리치없는 PC PWM 모드
- 주파수 발생기
- 10-비트의 클럭 프리스케일러
- 오버플로우, 비교 일치 인터럽트(TOV0와 OCF0) 발생
- 32.678㎑의 시계 크리스털을 클럭 소스로 사용하여 비동기적으로 계수하는 기능을 가지고 있음

8비트 타이머/카운터0의 동작

8-bit 타이머/카운터0 내부 구성도



비동기 동작

8비트 타이머/카운터0의 동작

- 32.678㎑의 수정 발진자를 접속하는 TOSC1 및 TOSC2 단자를 가지고 있어서 자체적인 발진에 의하여 RTC의 기능을 수행할 수 있으며, 외부의 T0 단자가 없는 대신에 TOSC1 단자에 외부 클럭을 인가할 수 있음.

- 타이머/카운터0는 내부 클럭과 외부 클럭을 사용하는 모든 카운터 동작에서 프리스케일러의 적용을 받으며, 사용할 수 있는 분주비의 종류도 더 많음.

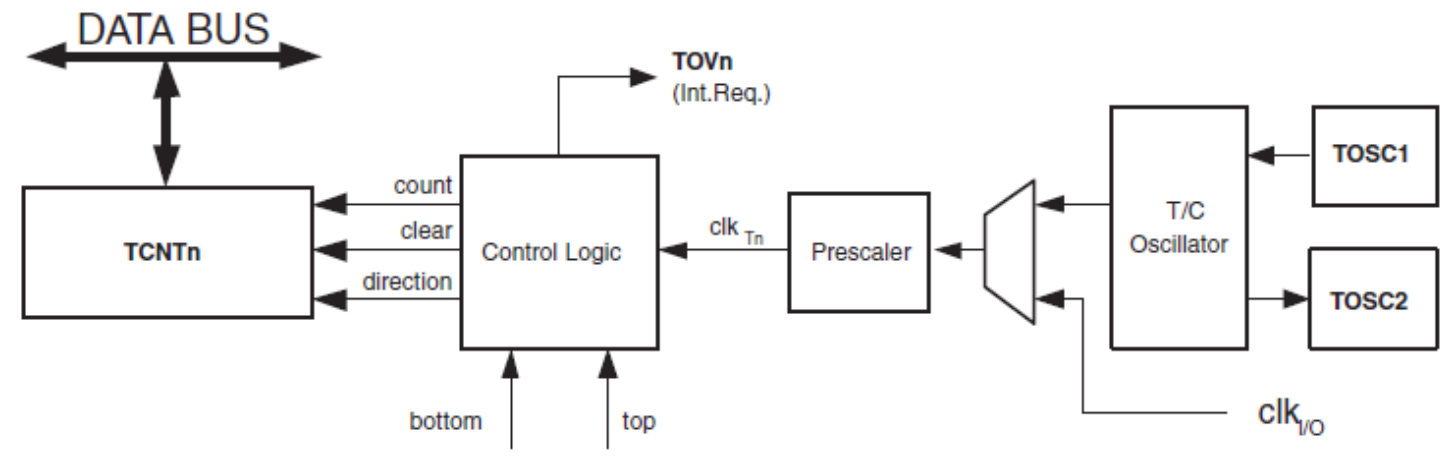
- 비동기 부분을 제어하기 위한 동기회로부가 별도로 내장되어 있음.

- 즉, 타이머/카운터0는 물론이고 외부 TOSC1과 TOSC2 단자에 32.678㎑의 수정 발진자를 접속하여 사용하는 카운터의 비동기 동작 모드에서도 프리스케일러가 사용되며, 프리스케일러의 분주비도 1, 8, 32, 64, 128, 256, 1024 중의 하나를 사용할 수 있어서 타이머/카운터2 와는 약간 차이가 있음

8비트 타이머/카운터0의 동작

타이머2와 타이머0의 차이점

- 타이머/카운터 부분의 구성
- 이를 제어하기 위한 동기 회로부



- **타이머/카운터0은 타이머 동작이나 카운터 동작에서 항상 프리스케일러를 사용하며 카운터로 동작할 때에는 비동기 동작의 특징을 가짐.**
- **즉, 타이머/카운터0은 물론이고 외부 TOSC1과 TOSC2 단자에 32.678 kHz의 수정발진자를 접속하여 사용하는 카운터의 비동기 동작 모드에서도 프리스케일러가 사용되며, 프리스케일러의 분주비도 1, 8, 32, 64, 128, 256, 1024 중의 하나를 사용할 수 있어서 타이머/카운터2와는 약간 차이가 있음.**

타이머/카운터0 의 레지스터

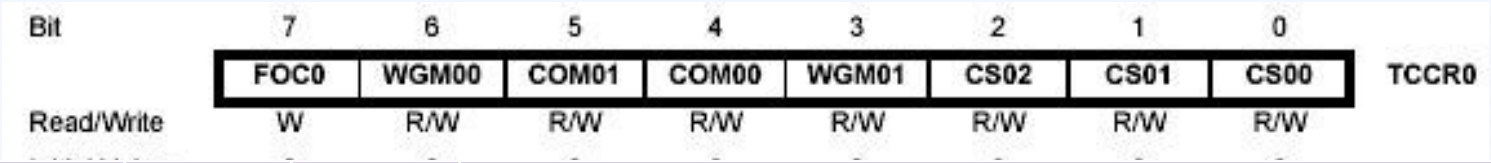
타이머/카운터0 레지스터	설 명
TCCR0	Timer/Counter 0 제어 레지스터
TCNT0	Timer/Counter 0 레지스터
OCR0	출력 비교 레지스터
ASSR	비동기 상태 레지스터
SFIOR	특수 기능 I/O 레지스터
TIMSK	타이머/카운터 인터럽트 마스크 레지스터
TIFR	타이머/카운터 인터럽트 플래그 레지스터

8비트 타이머/카운터0의 동작

한국공학대학교 TECH UNIVERSITY OF KOREA

TCCR0과 TCCCR2의 차이점

- TCCR0레지스터와 TCCR2레지스터의 차이점은 프리스케일러의 적용에 있음
- 비트2~0의 기능만 약간 차이가 있고 나머지 부분의 내용은 두 타이머/카운터가 동일함

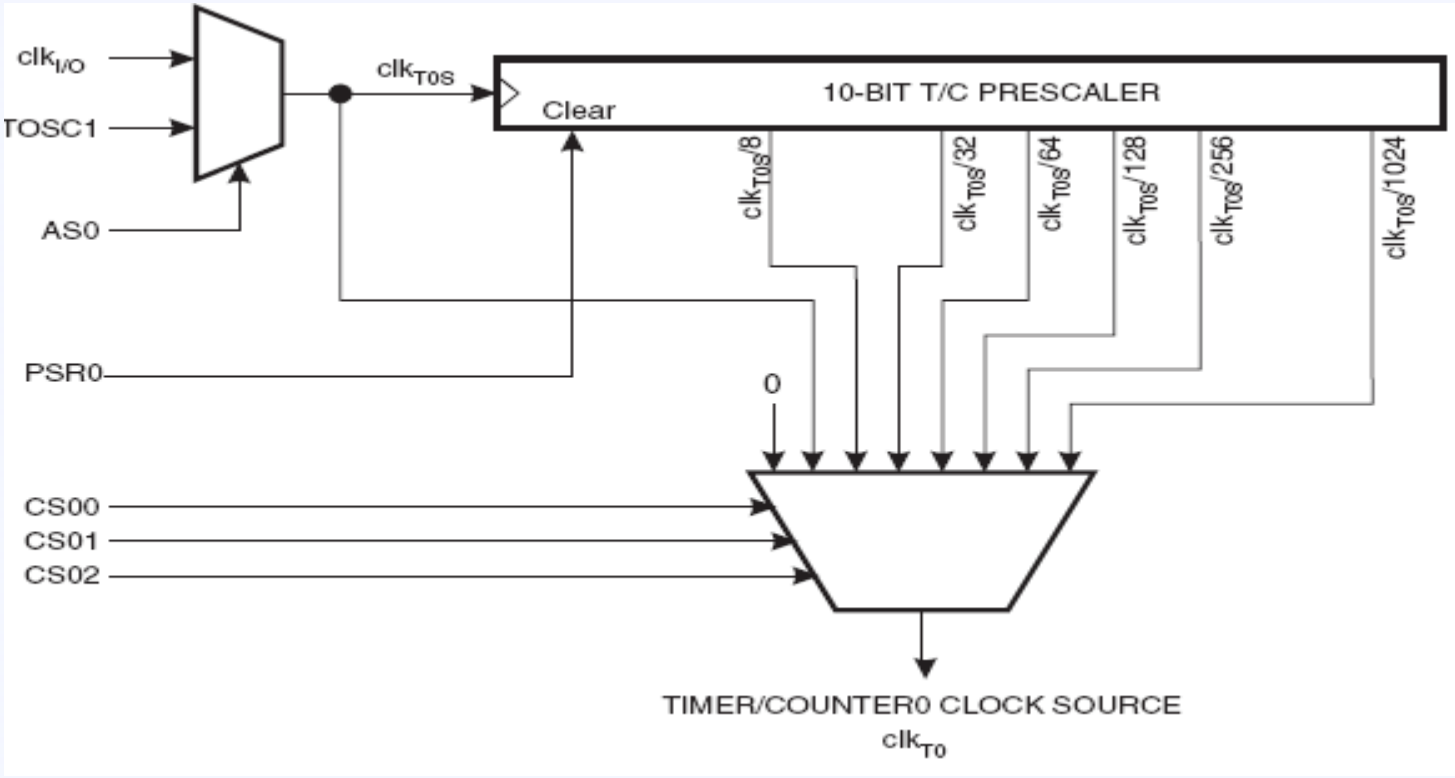


CS02	CS01	CS00	클럭 소스의 기능
0	0	0	클럭 소스 차단(타이머/카운터 기능이 정지)
0	0	1	ClK TOS
0	1	0	ClK TOS/8
0	1	1	ClK TOS/32
1	0	0	ClK TOS/64
1	0	1	ClK TOS/128
1	1	0	ClK TOS/256
1	1	1	ClK TOS/1024

8비트 타이머/카운터0의 동작

- 비트2~0 : CS02~CS00(클럭 선택, clock select)
  - 타이머/카운터0에서 사용할 클럭을 선택하는 기능을 담당

< 타이머/카운터0의 프리스케일러 구성도 >



8비트 타이머/카운터0의 동작

한국공학대학교 TECH UNIVERSITY OF KOREA

비동기 상태 레지스터 : ASSR

- ASSR 레지스터는 타이머/카운터0가 외부 클럭에 의해 비동기 모드로 동작하는 경우에 관련된 기능을 수행함.

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	AS0	TCN0UB	OCR0UB	TCR0UB	ASSR
Read/Write	R	R	R	R	R/W	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

➢ 비트 3 : AS0(비동기 타이머/카운터0 Asynchronous Timer/Counter0)
- 타이머/카운터0의 클럭 소스를 선택하는 비트
- AS2 = 0 -> 내부 클럭 $clk_{I/O}$가 선택되어 동기모드로 동작
- AS2 = 1 -> 외부의 TOSC1단자에서 입력되는 수정 발진자의 클럭이 선택되어, 비동기 모드인 RTC로 동작

➢ 비트 2 : TCN0UB (Timer/Counter0 Update BUSY)
- TCN0UB 비트는 타이머/카운터0가 비동기로 동작하고 있을 때 TCNT0 레지스터로 데이터를 쓰는 시점을 알려주기 위한 상태 비트
- TCNT0에 새로운 데이터를 쓰기 위해서는 이 비트가 0인 상태이어야 함

8비트 타이머/카운터0의 동작

- 비트 1 : OCR0UB(Output Compare Register 0 Update Busy)
  - OCR0UB 비트는 타이머/카운터0가 비동기로 동작하고 있을 때 OCR0 레지스터로 데이터를 쓰는 시점을 알려주기 위한 상태 비트
  - OCR0 레지스터에 새로운 값을 쓰면 이 비트가 1로 세트됨
  - TCN0UB에서와 마찬가지로 OCR0에 새로운 데이터를 쓰기 위해서는 이 비트가 0인 상태이어야 함

- 비트 0 : TCR0UB(Timer/Counter Control Register 0 Update Busy)
  - TCR0UB 비트는 타이머/카운터0가 비동기로 동작하고 있을 때, TCCR0 레지스터로 데이터를 쓰는 시점을 알려주기 위한 상태 비트
  - TCCR0 레지스터에 새로운 값을 쓰면 이 비트가 1로 세트됨
  - 임시 저장 레지스터로부터 TCCR0에 옮겨져서 TCCR0의 쓰기가 완료되면, 이 비트는 자동적으로 00이 됨

8비트 타이머/카운터0의 동작

- 타이머/카운터 동작 중에 동기모드에서 비동기 모드로 전환해야 하는 경우의 유의사항

① TIMSK 레지스터의 OCIE0=0 및 TOIE0=0으로 하여 인터럽트를 금지한다.
② ASSR 레지스터의 AS0 비트 값을 설정하여 클럭 소스를 선택한다.
③ TCNT0, OCR0, TCCR0 레지스터에 새로운 값을 기록한다.
④ 비동기 동작 모드로 전환하기 위해 ASSR 레지스터의 TCN0UB, OCR0UB, TCR0UB 비트가 0이 될 때까지 기다린다.
⑤ TIFR 레지스터의 인터럽트 플래그인 OCF0=0 및 TOV0=0으로 클리어한다.
⑥ 필요하다면, TIMSK 레지스터의 OCIE0=1 및 TOIE0=1로 하여 인터럽트를 허용 상태로 한다.

8비트 타이머/카운터0의 동작

특수 기능 I/O 레지스터 : SFIOR

- SFIOR 레지스터는 타이머/카운터들을 동기화 시키는데 관련된 기능을 수행

Bit	7	6	5	4	3	2	1	0	
	TSM	–	–	–	ACME	PUD	PSR0	PSR321	SFIOR
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- 비트 7 : TSM(Timer/Counter Synchronization Mode – 동기 모드)
  - 모든 타이머/카운터를 동기화 시키는 기능.
  - 1로 설정하면 PSR0 및 PSR321 비트 값이 유지하여, 이에 대응하는 프리스케일러 리셋 신호를 발생하고, 이는 해당 타이머/카운터의 동작을 정지시켜 모든 타이머/카운터를 똑같은 값으로 설정할 수 있도록 해줌.
  - 0으로 리셋하면, PSR0 및 PS321 비트는 하드웨어적으로 클리어되며, 타이머/카운터들이 동시에 계수 동작을 시작함. .

8비트 타이머/카운터0의 동작

- 비트 1 : PSR0(타이머/카운터 0의 프리스케일러 리셋)
  - 1로 설정 시 프리스케일러 리셋.
  - 리셋 후 바로 자동 클리어.

  타이머/카운터0 이 비동기 모드 일 경우 이 비트를 1로 설정하면 프리스케일러가 리셋 될때 까지 1을 유지.

  TMS(bit 7) 비트가 셋 되어도 이 비트는 자동 클리어 되지 않는다.
- 비트 0 : PSR321 (프리스케일러 리셋 타이머/카운터3,2과 1, Prescaler Reset Timer/Counter3, 2 and 1)
  - PSR321 비트는 타이머/카운터3,2,10이 공통적으로 사용하고 있는 프리스케일러를 리셋시키며, TSM이 1로 되어 있지 않으면 동작후에 자동적으로 클리어된다.

8비트 타이머/카운터0의 동작

한국공학대학교 TECH UNIVERSITY OF KOREA

나머지 타이머/카운터0 레지스터

➢ TCNT0, OCR0

- 타이머/카운터0을 제어하는데 사용되는 레지스터는 타이머/카운터2와 동일함

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	

- TCNT0 레지스터 -

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	

- OCR0 레지스터 -

- 타이머/카운터0의 인터럽트를 제어하는 기능은 타이머/카운터2의 인터럽트를 제어하는 레지스터와 동일하며 비트의 위치만 다름
- TIMSK 레지스터의 비트 7과 비트 6인 OCIE0와 TOIE0가 인터럽트 발생을 개별적으로 허용하는 비트이고, TIFR 레지스터의 비트 7과 비트 6인 OCF0와 TOV00이 인터럽트 발생 여부를 알려주는 인터럽트 플래그 비트임. 이 비트들의 동작은 타이머/카운터0에서 설명한 것과 동일함

8비트 타이머/카운터의 제어를 위한 방법(요약)

- 8비트 타이머/카운터의 동작은 일반모드, CTC모드, 고속PWM 모드, PC PWM 모드가 있음 : 이는 타이머/카운터 제어 레지스터의 WGMn1~WGMn0 비트와 COMn1~COMn0 비트의 의해 결정
- 타이머/카운터가 동작 모드로 정확하게 동작을 하기 위해서는 프로그램 시작과 함께 초기화 되야함.
- 프로그램에서 타이머/카운터를 원하는 동작모드로 동작시키기 위해 제어 레지스터, 카운터 레지스터 또는 출력 비교 레지스터를 제어하여야 함.
- 타이머/카운터가 계수한 내용을 확인하기 위하여 레지스터를 읽거나 또는 시간 주기를 조정하기 위하여 레지스터의 내용을 수정하는 등의 필요한 동작을 수행하도록 타이머/카운터는 프로그램 내에서 제어되어야 함.
- 타이머/카운터를 활용하여 프로그램을 작성하는 과정에서는 다음의 3가지 방법
  ① 인터럽트 플래그의 발생여부를 감시하면서 사건이 발생하는 것을 감시하는 경우
  ② 인터럽트가 발생하면 ISR을 수행하는 경우
  ③ 출력 핀의 상태를 자동으로 변화시키는 경우

8비트 타이머/카운터 활용 실험

한국공학대학교 TECH UNIVERSITY OF KOREA

예제 1: 일반 모드의 활용

- **Port B0 I/O에 타이머/카운터2을 사용하여 10㎑ 구형파를 만드는 프로그램 작성**

일반 모드의 동작 요약

① 단순히 시스템 클럭을 계수하는 모드로서 시스템 클럭은 프리스케일러를 통해 입력된다.
② 입력된 클럭은 타이머/카운터 레지스터 TCNTn(0 또는 2)에 의해 상향 계수되고,
③ 최대값인 0xFF가 되면 오버플로우 플래그인 TOVn이 발생하여 인터럽트를 요청하게 된다.
④ 이렇게 발생된 TOVn 플래그는 인터럽트가 발생하여 ISR 루틴이 수행되면 자동으로 클리어 된다.

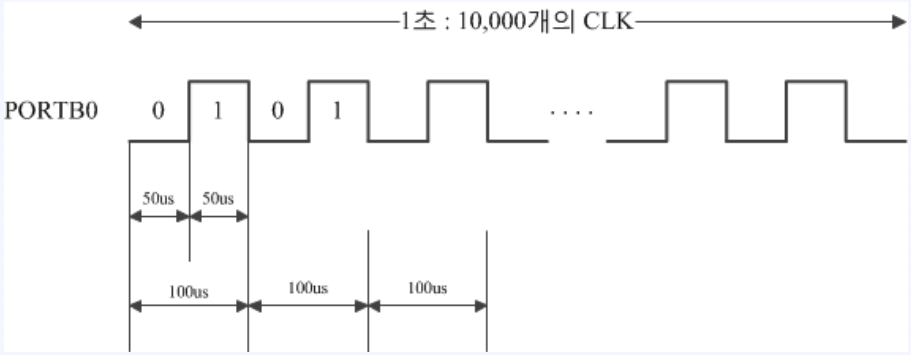
- **타이머/카운터2으로 입력되는 클럭 :** $t_{clk} = \frac{N}{f_{clk\_I/O}}$
- **타이머/카운터2의 최대 주기 :** $T_{max} = 256 \times t_{clk} = \frac{256 \times N}{f_{clk\_I/O}}$
- **예시) 시스템 클럭이 16MHz 일 때, 분주비에 따른 최대 주기 -> 수강생은 14.7456Mhz로 계산**

(실험 키트의 시스템 클럭은 14.765㎒를 사용하지만 예시는 계산의 편리상 16㎒를 사용하는 것으로 가정한다.)

분주비	1	8	64	256	1024
입력 클럭 (usec)	0.0625	0.5	4	16	64
최대 주기 (usec)	16	128	1,024	4,096	16,384

8비트 타이머/카운터 활용 실험

- 10KHz의 클럭(100us)을 만들기 위해서는 펄스의 ON/OFF가 각각 50us마다 이루어져야 함.
- PPT 17p(TCCR2 레지스터 CS22:20)참조하여 분주비를 80이하로 선정
- 10KHz의 구형파를 만들기 위해서는 50us 마다 오버플로우 발생을 하여야 함. → TCNT2에 -100을 적재하고, 오버플로우가 발생할 때 마다 -100을 재 적재한다.
- 매 50us 마다 값을 써 넣어야 하므로 인터럽트를 사용함.



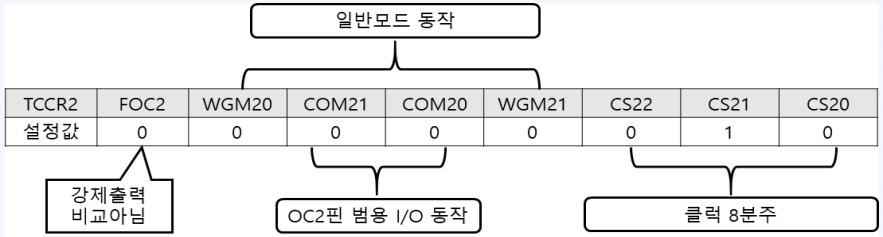
- 타이머/카운터2의 초기화 과정

```
#include <mega128.h>

void Init_Timer2(void)
{
        TCCR2 = 1<<CS21;     // 0b00000010, 일반 모드, 8 분주비 사용
        TCNT2 = -100;        // 카운터 값 설정(50㎲)
        //TIFR |= (1<<TOV2); // 필요시, 오버플로우 플래그 0으로 초기화
        TIMSK = (1<<TOIE2); // 오버플로우(TOV2) 인터럽트 허가
}
```

8비트 타이머/카운터 활용 실험

한국공학대학교 TECH UNIVERSITY OF KOREA

① <방법 1> 타이머/카운터2의 TOV2 인터럽트를 사용하는 경우




```
#include <avr/io.h>
#define F_CPU 14.7456E6
#include <avr/interrupt.h>
void Init_Timer2(void)
{
        TCCR2 = 1<<CS21;    // 0b00000010, 일반 모드, 8 분주비 사용
        TCNT2 = -100;       // 카운터 값 설정(50㎲)
        //TIFR |=  (1<<TOV2);// 필요시, 오버플로우 플래그 0으로 초기화
        TIMSK = (1<<TOIE2);// 오버플로우(TOV2) 인터럽트 허가
}
ISR(TIMER2_COMP_vect)
{
        TCNT2= -100;          // 카운터 값 설정(50㎲)
        PORTB ^= _BV(0);      // 50㎲
}
int main(void)
{
        Init_Timer2();        // 타이머/카운터2의 초기화
        DDRB = 0xff;          // PORTB 0~7 I/O 출력 포트로 설정
        PORTB &= ~_BV(0);     // Port B0 I/O를 클리어
        sei();                // 전체 인터럽트 허가
        while(1);             // 무한 루프
}
```

② <방법 2> 타이머/카운터2의 출력 비교 인터럽트를 사용하는 경우

```
#include <avr/io.h>
#define F_CPU 14.7456E6
#include <avr/interrupt.h>
void Init_Timer2(void)
{
        TCCR2 = 0x00;          // 타이머/카운터2 정지 프리스케일러 옵션 참조
        OCR2 = 100;            // 출력 비교 레지스터의 초기값 설정
        TIMSK = 1 << OCIE2;    // 출력비교(OCIEn) 인터럽트 허가
        TCCR2 = 1 << CS21;     // 8분주 프리스케일러, 일반 모드의 설정
}
ISR(TIMER2_COMP_vect)
{
        OCR2 += 100;           // 출력 비교 레지스터값의 갱신
        PORTB ^= _BV(0);       // 포트 비트 출력
}
void main(void)
{
        Init_Timer2();         // 타이머/카운터2의 초기화
        DDRB = 0xff;           // Port B 0~7 I/O를 출력으로 설정
        PORTB &= ~_BV(0);      // Port B0 클리어
        sei();                 // 전체 인터럽트 허가
        while(1);
}
```

조건 : 출력 비교 레지스터(OCR2)와 타이머/카운터 레지스터(TCNT2)가 일치할 때 발생함.

(1st : 0x64, 2nd : 0xC8, 3rd : 0x2C……)

주의 : 일반모드 에서도 주어진 시간마다 인터럽트를 발생은 가능하지만, 파형 발생을 위함이라면 CTC 모드를 사용하는 것을 권장.

예제 2: CTC 모드의 활용

- 타이머/카운터2의 CTC 모드를 사용하여 10㎑ 구형파를 출력 비교(OC2) 핀으로 출력하는 프로그램

CTC 모드의 동작 요약

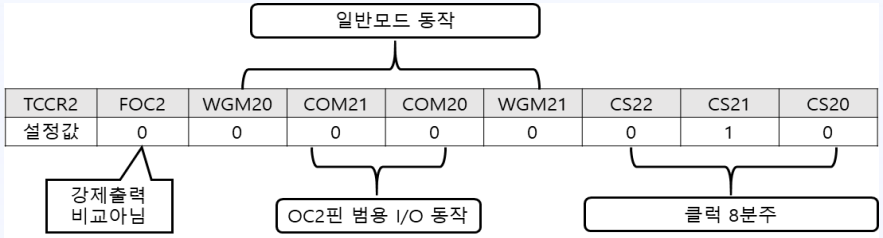
① TCNT2의 값이 0x00에서 시작하여 증가하다가 출력 비교 레지스터 OCR2에 설정된 값과 같아지면 리셋 되고 출력 비교 인터럽트가 발생한다.
② 이 동작은 예제 1에서 설명한 OVF2 인터럽트를 사용하는 것과는 달리 OCR2의 값이 인터럽트가 발생하면 재 적재되는 동작을 수행한다.
(단, 현재 TCNT 값보다 작은 OCR값을 입력하면 최대값(0xFF)까지 증가 하였다가 다음 상승 주기에서 정상 동작됨. ->글치치 현상 발생)

- 타이머/카운터2의 초기화 과정

① CTC 모드의 설정 : TCCR2 레지스터의 WGM21 ~ WGM20 비트를 '10' 으로 설정한다.
② OCR2 레지스터 값의 설정
③ 인터럽트의 설정 : TCNT2 레지스터와 OCR2 레지스터의 값을 비교하여 일치할 때 출력 비교 인터럽트를 사용하기 위해 TIMSK 레지스터의 OCIE2 비트를 1로 세트 하여야 한다.
④ 출력 핀의 설정 : OC2 핀으로 파형을 하기 위해 COM21 ~ COM20 비트를 동작 모드에 따라 설정한다.
⑤ 인터럽트의 사용 : 출력 비교 인터럽트가 발생하면 이에 해당하는 동작을 확인하기 위해 TIFR의 OCF2 비트를 사용하거나 TIM2_COMP ISR 루틴을 작성한다.

8비트 타이머/카운터 활용 실험

한국공학대학교 TECH UNIVERSITY OF KOREA




```
#include <avr/io.h>
#define F_CPU 14.7456E6
#include <avr/interrupt.h>
void Init_Timer2(void)
{
        TCCR2 = 0x00;
        //TCCR2 레지스터 0x08로 설정 CTC모드
        //TCCR2 레지스터 0x10로 설정 비교일치시 OC2 출력을 토글
        TCCR2 |= (1 << WGM21) | (1 << COM20);
        //출력비교 레지스터로 TCNT0 레지스터와 비교하여 OC2 단자에 출력신호 발생
        OCR2 = 100;
        TIMSK |= 1 << OCIE2; //출력 비교 인터럽트 허가상태
        TCCR2 |= 1<<CS21;    // 타이머/카운터2 동작
}
ISR(TIMER2_COMP_vect)
{
        asm volatile("nop");
}
int main(void)
{
        Init_Timer2();
        DDRB |= _BV(7);  //PORTB의 7번핀을 출력으로 설정
        sei();                  // 인터럽트 허가
        while(1);
}
```

nop = no processing

모드	WGM21 (CTC2)	WGM20 (PWM2)	동작모드
0	0	0	일반
1	0	1	PWM, Phase Correct
2	1	0	CTC
3	1	1	고속 PWM

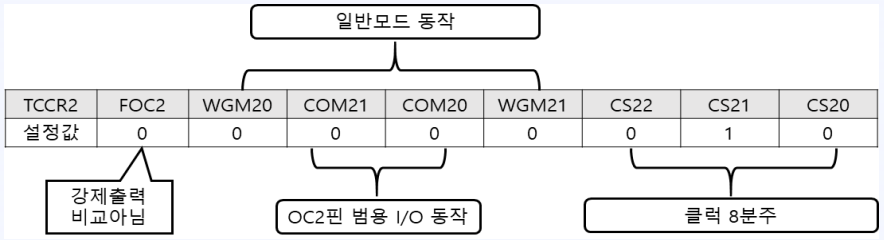
예제 3: CTC 모드의 활용: 소프트웨어 루프의 활용

- **예제 2의 프로그램을 활용하여 1㎑ 구형파를 Port B0상에 출력하는 프로그램 작성**
  - 1㎑의 구형파를 만들기 위해서는 펄스의 On 시간과 Off 시간이 각각 500㎲이어야 한다.

```
#include <avr/io.h>
#define F_CPU 14.7456E6
#include <avr/interrupt.h>
void Init_Timer2(void){
        TCCR2 = 0x00;           // 타이머/카운터 동작 금지
        TCCR2 |= (1<< WGM21); // TCCR2 레지스터 CTC 모드 설정
        OCR2 = 100;             // 출력비교 레지스터의 주기는 50uS
        TIMSK = (1<<OCIE2);     // 출력비교 인터럽트 허가상태
        TCCR2 |= 1<<CS21;// 타이머/카운터2 동작
}
// 출력비교 인터럽트가 발생했을 때 처리하는 인터럽트 함수
ISR(TIMER2_COMP_vect){
  c_cnt++; }

int main(void){
        DDRB |= _BV(0);         // PORTB의 0번핀을 출력으로 설정
        c_cnt = 0;              // c_cnt 변수 클리어
        Init_Timer2();          // 타이머2 초기화
        sei();                  // 전체 인터럽트 허가
        while(1){
                if(c_cnt == 10)// 50us×10 = 500us
                {       PORTB ^= _BV(0); // 500us이면 토글
                        c_cnt = 0;
                }
        }
}
```

예제 4: Fast PWM 모드의 활용

```
// Timer/Counter2 이용한 Fast PWM Mode 실험 (듀티비 50% : OCR2=0x7F)
// 확인 사항 : Timer/Counter2를 Fast PWM Mode를 사용하여 비교일치 인터럽트가 발생할 때 마다
// OC2 핀(Port B7)에 PWM 파형의 듀티 변화 발생, 타이머 TCNTn Max->Bottom 마다 주기 갱신
```




```
#include <avr/io.h>
#define F_CPU 14.7456E6
#include <avr/interrupt.h>

void Init_Timer2(void)
{
     TCCR2 |= (1<<WGM20) | (1<<WGM21) | (1<<COM21) | (1 << CS21);
    // FAST PWM 모드 : WGM21 ~ WGM20 : '11',
    // OC2핀을 비반전 비교 출력 : COM21, COM20 : '10'
    // 시스템 클럭을 8분주 : CS22, CS21, CS20 =  '010'
    TIMSK = (1<<OCIE2);
    TCNT2 = 0x00;           // TCNT2 0부터 시작
    OCR2 = 0x7F;            // OCR2 값 설정(duty ratio 50%)
}
void Init_Port(void)
{
    DDRB = 0xff;            // PORTB I/O 설정 : 출력(Port B7 포함)
}
```

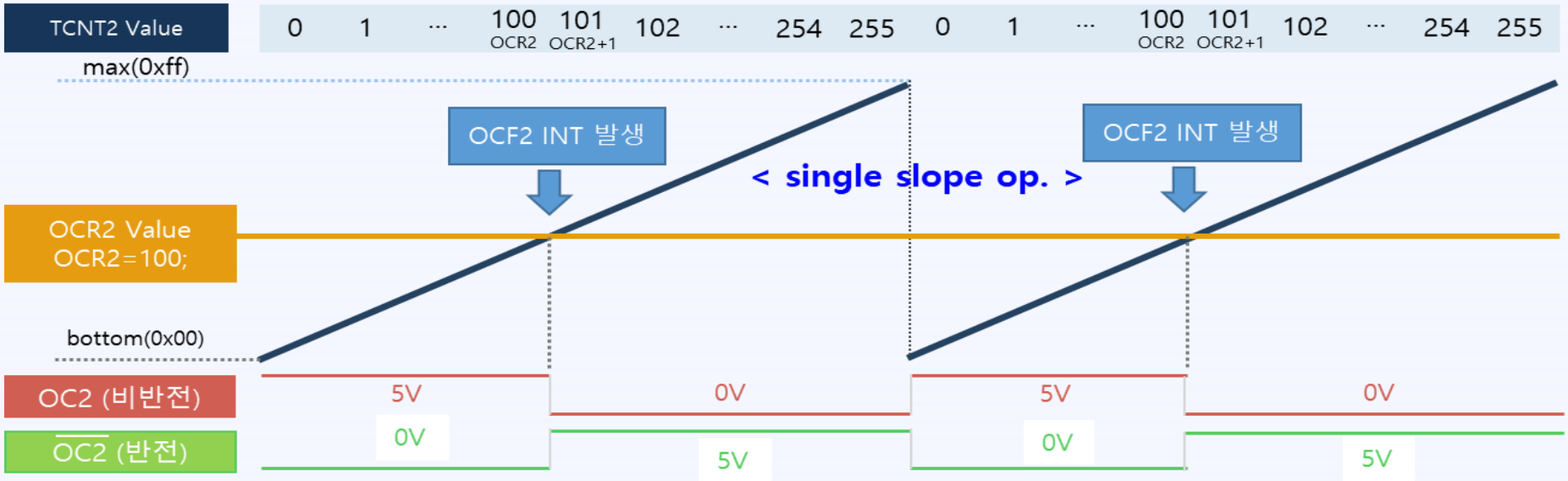
모드	WGM21 (CTC2)	WGM20 (PWM2)	동작모드
0	0	0	일반
1	0	1	PWM, Phase Correct
2	1	0	CTC
3	1	1	고속 PWM

예제 4: Fast PWM 모드의 활용

```
// Timer/Counter2 이용한 Fast PWM Mode 실험 (듀티비 50% : OCR2=0x7F)
// 확인 사항 : Timer/Counter2를 Fast PWM Mode를 사용하여 비교일치 인터럽트가 발생할 때 마다
// OC2 핀(Port B7)에 PWM 파형의 듀티 변화 발생, 타이머 TCNTn Max->Bottom 마다 주기 갱신
```

```
ISR(TIMER2_COMP_vect)
{
   asm volatile("nop");       // 1클럭 지연
}

void main(void)
{
    Init_Port();
    Init_Timer2();

    sei();                    // 전체 인터럽트 허가

    while(1);                 // 무한 루프
}
```

예제 4: Fast PWM 모드의 활용



	0~100 count (101개)	101~255 count (155개)	0~100 count (101개)	101~255 count (155개)
Oc2 pin Time	$\frac{1}{16MHz}X101 = 6.3125[\mu S]$	$\frac{1}{16MHz}X155 = 9.6875[\mu S]$	$\frac{1}{16MHz}X101 = 6.3125[\mu S]$	$\frac{1}{16MHz}X155 = 9.6875[\mu S]$
Period (cycle Time)	$T = \frac{1}{16MHz}X256 = 16[\mu S]$		$T = \frac{1}{16MHz}X256 = 16[\mu S]$	

Frequency

$$f_{oc2PWM} = \frac{f_{dk\_I/O}}{N \cdot 256}$$

- N은 분주비로 1, 8, 64, 256, 1024(Timer/Counter2)
- $f_{dk_I/O}$는 시스템 클럭과 같은 주파수값(16MHz)

예제 5: Fast PWM 모드의 응용

- **고속 PWM 모드를 이용하여 OC2 단자에 듀티비가 25%, 50%, 75%인 PWM 신호를 만드는 프로그램을 작성 하시오.**
- **듀티비 선택은 Port D에 연결되어 있는 스위치 입력에 따라 결정**
- **Port D7이 On되면 75%, Port D6이 On되면 50%, Port D5가 On되면 25%로 각각 설정되도록 프로그램 작성**

➢ PWM 신호의 기본 주파수 설정(16Mhz 기준으로 설명-교재)

$$f_{OC2\_FPWM} = \frac{f_{clkI/O}}{N(1+255)} = \frac{16\ X\ 10^6}{8\ X\ 256} = 7.81\ KHz$$ **(8분주의 경우)**

< 분주비에 따른 PWM 신호의 주파수 및 주기 >

분주비	1	8	32*	64	128*	256	1024
주파수 (㎑)	62.5	7.81	1.95	0.97	0.48	0.24	0.06
주기 (msec)	0.016	0.13	0.51	1.03	2.08	4.17	16.67

예제 5: Fast PWM 모드의 응용

- **듀티비의 변경 : PORTD에 있는 스위치 입력 SW7~5의 상태에 따라 결정**
- **PWM 신호의 듀티비의 결정 : OCR2 레지스터의 값에 의해 결정**
  - **SW7(PORTD7)이 ON되면, 듀티비가 75%**
  - **SW6(PORTD6)가 ON되면, 듀티비가 50%**
  - **SW5(PORTD5)이 ON되면, 듀티비가 25%로 설정되도록 프로그램을 작성.**

$$Duty\ Ratio(\%) = \frac{OCR2}{256}\ X\ 100(\%)\quad or\ OCR2\ Value = \frac{256\ X\ Duty\ Ratio}{100}$$

  - **듀티비가 25%일 때 : OCR2=64(0x40)**
  - **듀티비가 50%일 때 : OCR2=128(0x80)**
  - **듀티비가 75%일 때 : OCR2=192(0xC0)**
  - **고속 PWM 모드에서의 해상도는 8비트의 최대값인 256임.**
- **OCR2 레지스터의 값은 메인 함수에서 키 값을 사용하여 switch 문으로 구현**

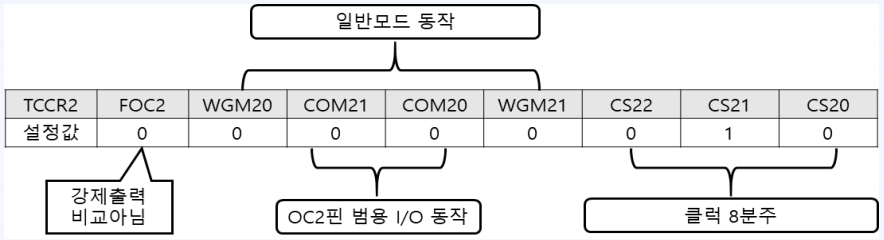
```
void Init_Timer2(void)
{
    TCCR2 |= (1<<WGM20 | 1<<WGM21 | 1<<COM21) | (1 << CS21);
    // FAST PWM 모드 : WGM20~WGM21 : '11',
    // OC2핀을 비반전 비교 출력 : COM21, COM20 : '10'
    // 시스템 클럭을 8분주, TCNT2 계수 시작 : CS22, CS21, CS20 =  '010'
    TIMSK |= (1<<OCIE2);              // 출력비교 인터럽트 허가
    OCR2 = 0x00;                      // OCR2 레지스터의 값을 0으로 설정
    TCNT2 = 0x00;
    DDRB |= _BV(7);                   // PORTB의 7번 비트를 출력으로 설정(OC2 출력)
    DDRD = 0x00;                      // 스위치 PORTD를 입력으로 설정
}
```

예제 5: Fast PWM 모드의 응용

```
ISR(TIMER2_COMP_vect){
   asm volatile("nop");       // 1클럭 지연
}

void main(void)
{
      unsigned char input_SW=0;
      Init_Timer2();                              // 타이머 2 초기화
      sei();                                      // 전체 인터럽트 허가
      while(1)
      {
            input_SW = (PIND & 0xf0) >> 4;  // PORTD의 상위비트의 스위치 입력을 읽어 4비트 우로 시프트
            switch(input_SW)
            {
                  case 0x07 :                 // PORTB.7 스위치가 ON인 경우 75% 듀티비
                     OCR2 = 0xC0;             // OCR2 값을 0xC0으로 설정
                     break;
                  case 0x0b :                 // PORTB.6 스위치가 ON인 경우 50% 듀티비
                     OCR2 = 0x80;             // OCR2 값을 0x80으로 설정
                     break;
                  case 0x0d :                 // PORTB.5 스위치가 ON인 경우 25% 듀티비
                     OCR2 = 0x40;             // OCR2 값을 0x40로 설정
                     break;
                  default:
                     break;
             }
      }
}
```

예제 6: PC(Phase Correct) PWM 모드의 활용

```
// Timer/Counter2 이용한 PC PWM Mode 실험 (듀티비 50% : OCR2=0x7F)
// 확인 사항 : Timer/Counter2를 PC PWM Mode를 사용하여 TCNTn과 OCRn이 일치 할 때 마다
// OC2 핀(Port B7)에 PWM 파형의 변화 발생
```

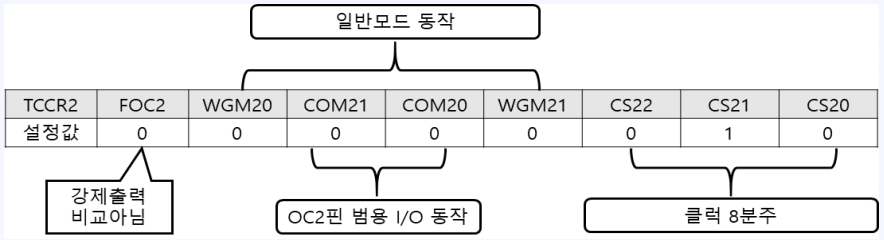



```
#include <avr/io.h>
#define F_CPU 14.7456E6
#include <avr/interrupt.h>

void Init_Timer2(void)
{
     TCCR2 |= (1<<WGM20 | (1<<COM21) | (1 << CS21);
    // PC PWM 모드 : WGM21 ~ WGM20 : '01',
    // OC2핀을 비반전 비교 출력 : COM21, COM20 : '10'
    // 시스템 클럭을 8분주 : CS22, CS21, CS20 =  '010'
    TIMSK =0x00;     // 인터럽트를 사용하지 않아도 파형 발생 가능함.
    TCNT2 = 0x00;    // TCNT2 0부터 시작
    OCR2 = 0x7F;     // OCR2 값 설정(duty ratio 50%)
}

void main(void)
{
    Init_Timer2();
    DDRB = 0xff;              // PORTB I/O 설정 : 출력(Port B7 포함)
    while(1);                 // 무한 루프
}
```

모드	WGM21 (CTC2)	WGM20 (PWM2)	동작모드
0	0	0	일반
1	0	1	PWM, Phase Correct
2	1	0	CTC
3	1	1	고속 PWM

예제 7: PC(Phase Correct) PWM 모드의 응용

```
// Timer/Counter2 이용한 PC PWM Mode 실험 (듀티비 50% : OCR2=0x7F)
// 확인 사항 : Timer/Counter2를 PC PWM Mode를 사용하여 TCNTn과 OCRn이 일치 할 때 마다
// OC2 핀(Port B7)에 PWM 파형의 변화 발생
```




```
#include <avr/io.h>
#define F_CPU 14.7456E6
#include <avr/interrupt.h>

void Init_Timer2(void)
{
     TCCR2 |= (1<<WGM20 | (1<<COM21) | (1 << CS21);
    // PC PWM 모드 : WGM21 ~ WGM20 : '01',
    // OC2핀을 비반전 비교 출력 : COM21, COM20 : '10'
    // 시스템 클럭을 8분주 : CS22, CS21, CS20 =  '010'
    TIMSK =0x00;     // 인터럽트를 사용하지 않아도 파형 발생 가능함.
    TCNT2 = 0x00;    // TCNT2 0부터 시작
    OCR2 = 0x7F;     // OCR2 값 설정(duty ratio 50%)
}

void main(void)
{
    Init_Timer2();
    DDRB = 0xff;              // PORTB I/O 설정 : 출력(Port B7 포함)
    while(1);                 // 무한 루프
}
```

모드	WGM21 (CTC2)	WGM20 (PWM2)	동작모드
0	0	0	일반
1	0	1	PWM, Phase Correct
2	1	0	CTC
3	1	1	고속 PWM

1초 계수 방법

- **1클럭의 주기는 $\frac{1}{14.7456MHz}$ = 67.8ns**

시스템 클럭	14.7465MHz	
분주비	1024	
카운터 입력 클록	0.014400879MHz	
1 클록의 시간	69.44us	
1초 계수를 위한 횟수	14405	
실제 시간	1.000초	

- **69.44us를 기준 클록으로 사용하여 1초를 만들기 위해서는 14405회를 계수하면 (69.44us × 14405 =1.000 sec) 됨.**
- **1차 계수를 5회, 2차 계수를 2881회로 설정하여 소프트웨어 루프를 활용하여 프로그램을 작성함.**
- **타이머/카운터 0의 오버플로우 인터럽트를 발생 주기를 TCNT0의 레지스터 값을 5로 조정하면 69.42μs(67.8ns × 1024) × 5 = 347.1μs 마다 인터럽트 발생되고, 인터럽트의 회수를 2881회 계수하면 약 1초가 됨.**

1초 계수 방법

타이머/카운터0, 일반모드, 분주비 1024, OC0 차단 모드 사용
오버플로우 INT 사용하여 작성

실험 보드의 시스템 클럭은 14.7456MHz 사용, 분주비 1024 사용
1개의 클럭의 주기 = $\frac{1}{14.7456MHz/1024} = 00694[\mathrm{msec}]$
14,400개의 펄스를 계수하면 1초가 됨.
144개의 클럭을 계수하는 루프를 100번 반복하여 14,400개의 클럭을 계숙하는 프로그램을 작성하여 활용

Report

요리용 디지털 타이머의 구현

1. FND와 키패드를 이용하여 타이머 설정 시간 결정 및 사용자 에게 표기
   FND 초기상태는 - - - - 으로 표기
2. 타이머의 설정은 키패드의 M1(A) 버튼을 눌러 모드 진입
   진입시 FND 는 0 0 0 0 으로 표기
3. 원하는 시간을 1초 단위로 기재 최대 9999초
4. 타이머의 시작은 키패드 M4(D) 버튼을 눌러서 시작
5. 타이머의 종료가 되면 FND에 - E n d 를 출력하고 종료.