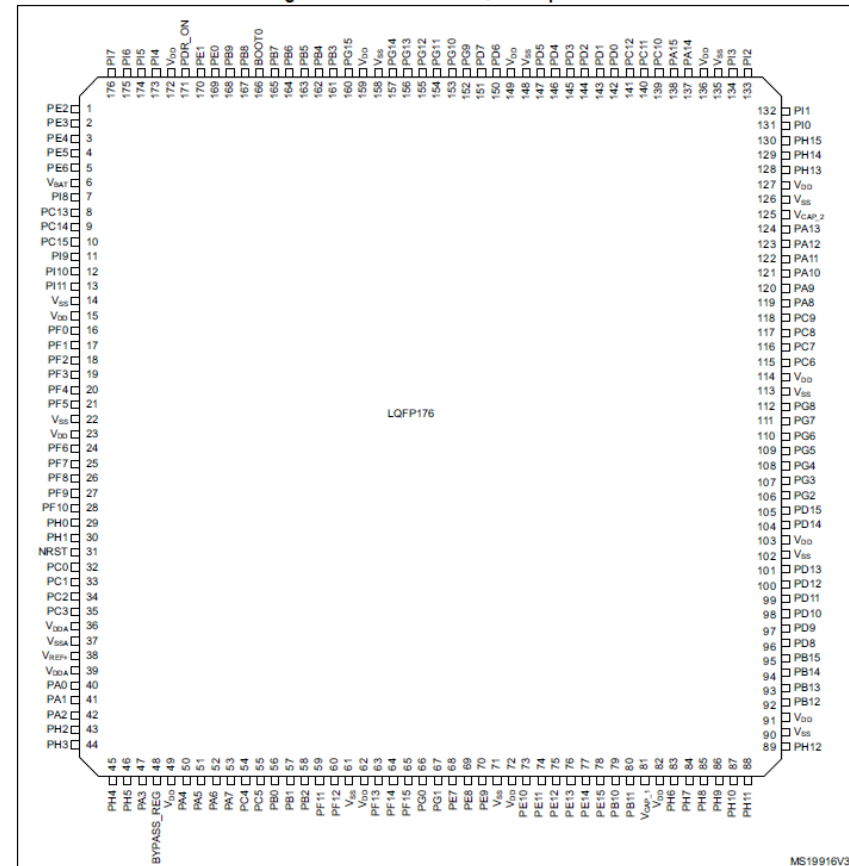


STM32F407 TIMER



한국산업기술대학교 메카트로닉스공학과
마이크로컴퓨터응용
담당교수: 남윤석

0. Timer 필요성(예:교통신호등)

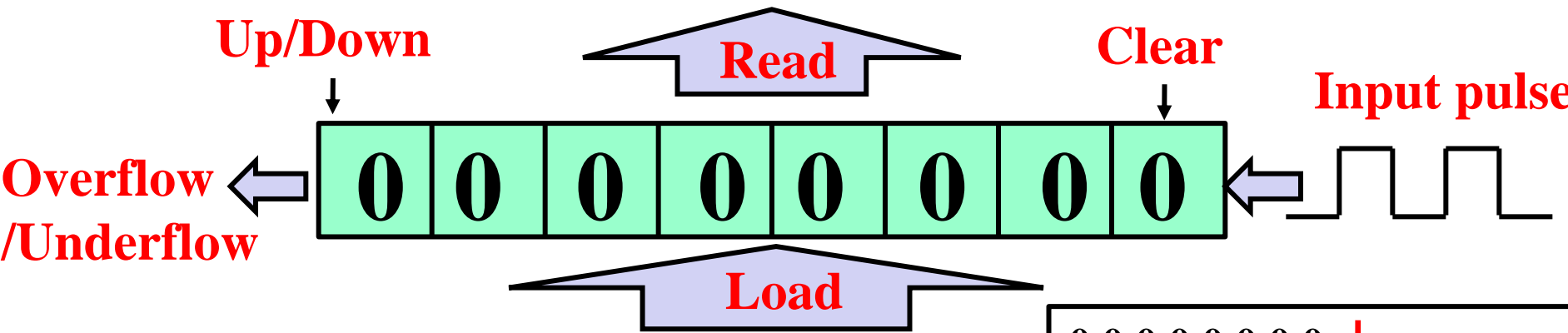


교체되는 LED교통신호등

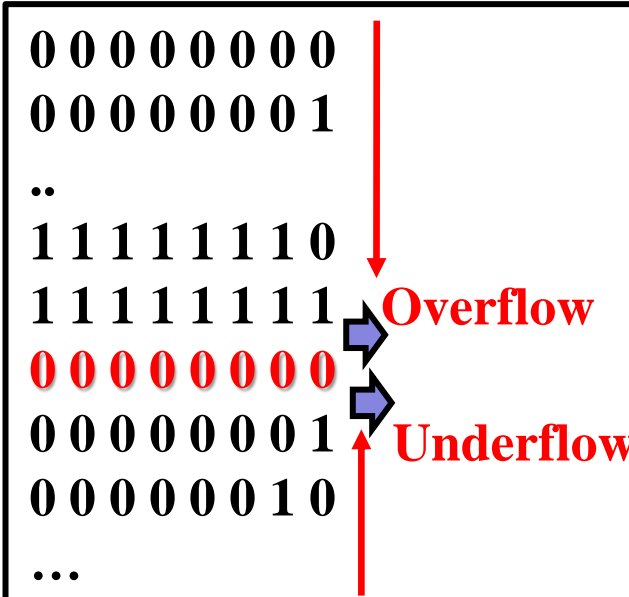


I. Timer 개요

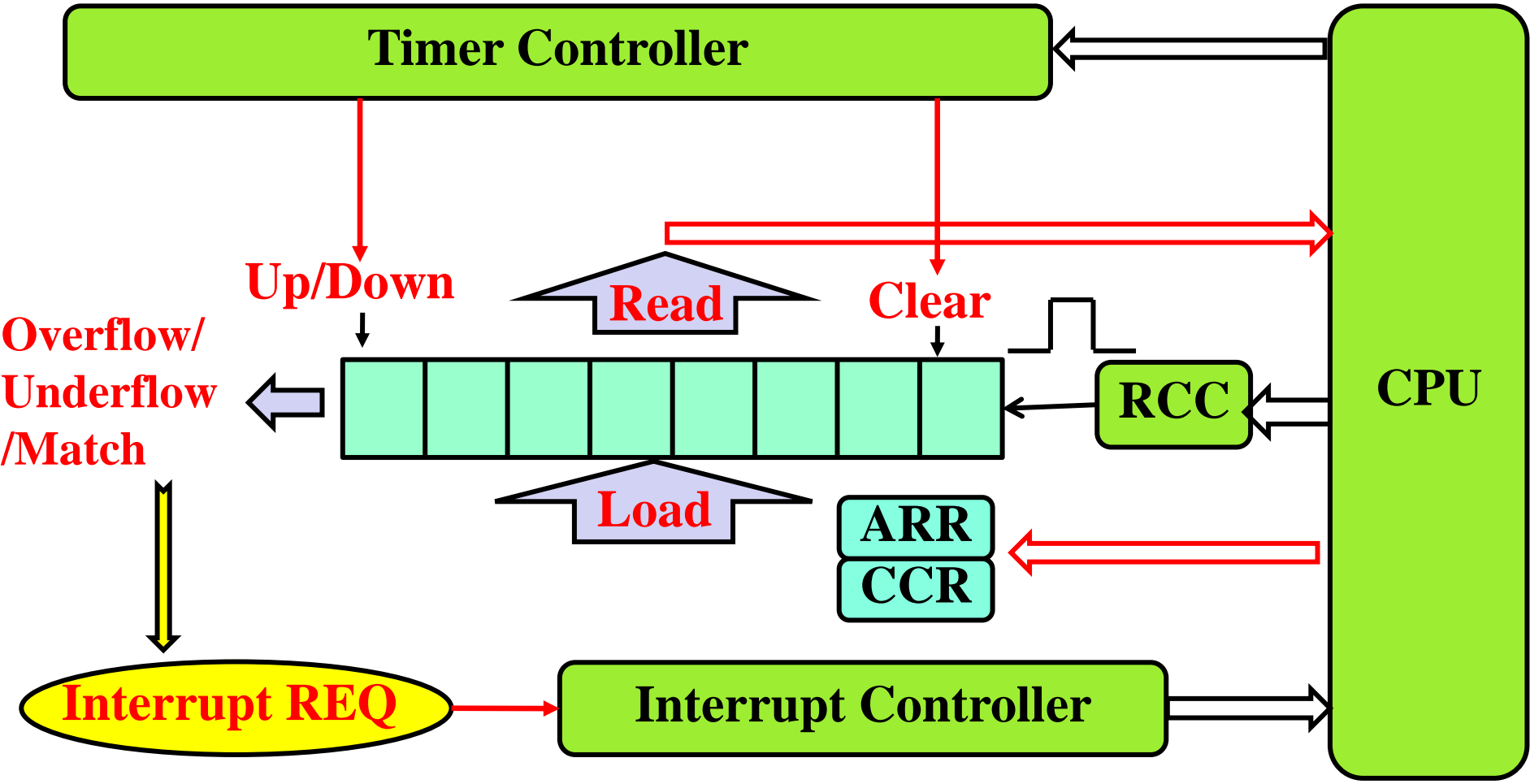
- ## 1. Timer의 Hardware : Counter (8,16,32,64 bit)
- Counter(8bit)의 구조와 동작



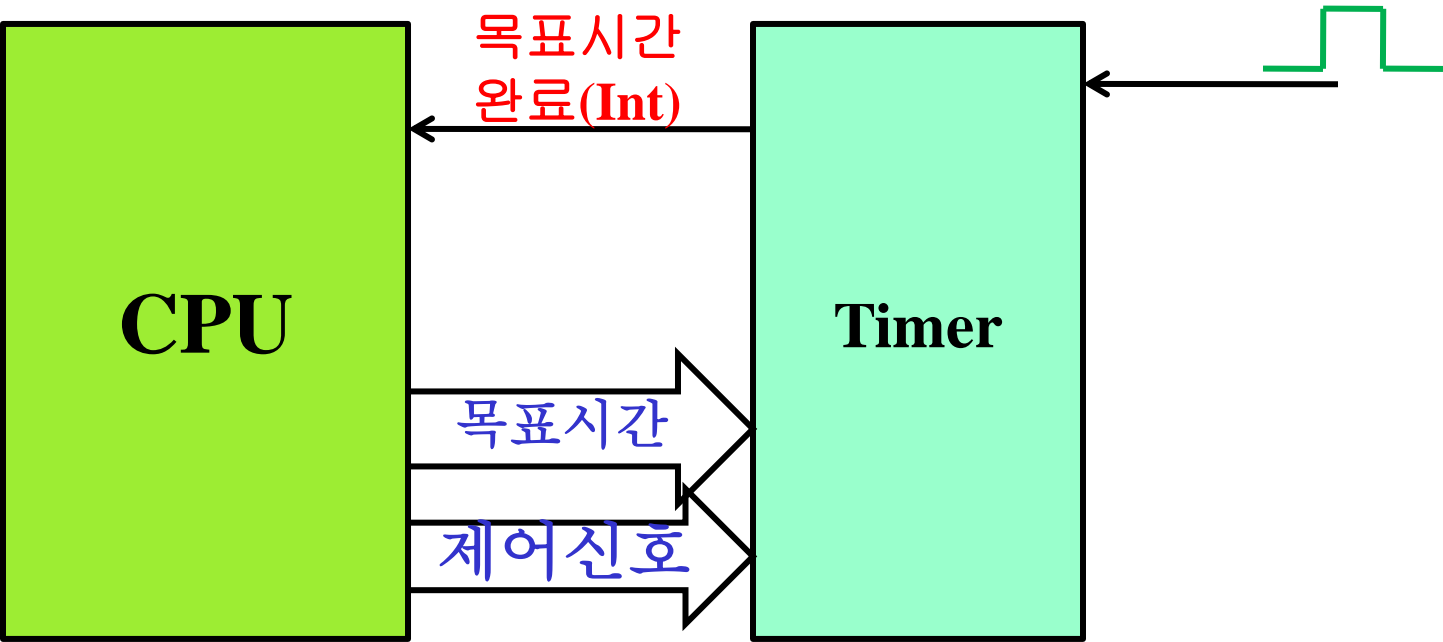
- Counter 정의: 입력 펄스(Clock/외부입력펄스 등)의 수를 세는 역할
- Timer 정의: Counter 동작을 이용하여 시간 설정(특정시간을 생성) 및 원하는 펄스를 발생하는 역할



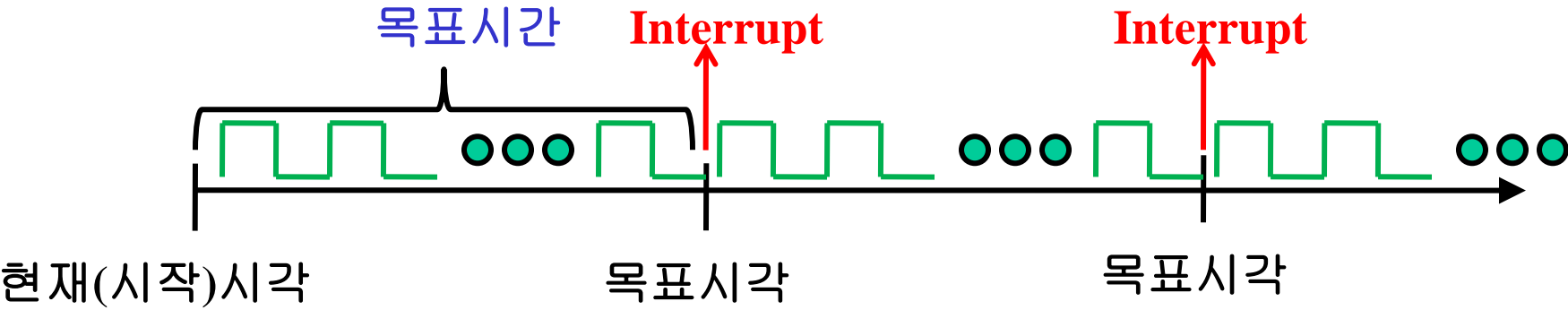
2. Timer 주변회로 및 상관관계



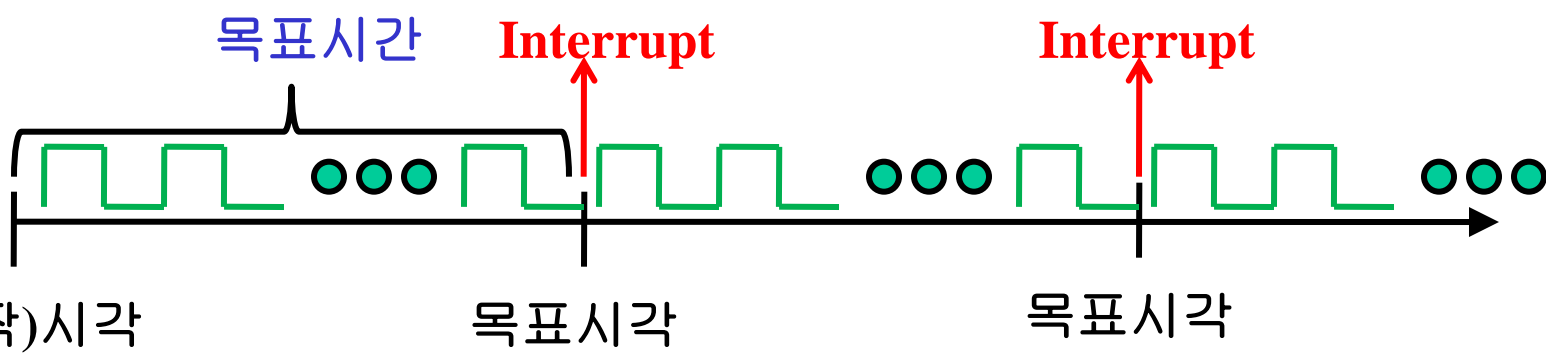
3. Timer의 제어구성도



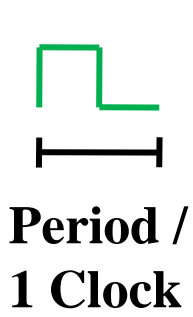
▪Timer의 주사용목적 : 주기적인 작업을 하기 위한 시간설정



4. Timer의 시간설정 동작



Solution: 목표시간을 clock의 총 개수로 환산



- (예) 목표시간: 1msec (period/1 clock = 100us 인 경우)
- 목표시간: 1sec → clock의 총 개수: 10 clocks
 - CPU는 '10' 을 Timer에 loading
 - 현재시각부터 10 clock 입력된 후 이벤트(interrupt) 발생

✓ 10 clock 입력된 것을 어떻게 알 수 있나????

➡ **Overflow(Underflow) 이용**

(예1) Overflow를 이용한 Timer의 시간 설정

- 목표시간 : 입력 펄스 수에 비례 (sec)
= 1 pulse의 주기(sec/pulse) X input pulse 수 (pulse)
- 목표시간: 1msec
 - 조건: 1 pulse의 주기(sec/pulse) = 100usec/pulse
 - input pulse 수 (pulse) : 10 input pulses
(근거: 1msec = 100usec/pulse X 10 pulses)

0 0 0 0 0 0 0 0 (start: Clear 상태)

0 0 0 0 0 0 0 1

.

.

0 0 0 0 1 0 0 1

0 0 0 0 0 0 0 0 (overflow 발생)

(10 pulses 입력)



(예2) Overflow를 이용한 Timer의 주기시간 설정

- 100usec/pulse X 100 pulses = 1 msec (주기)

0 0 0 0 0 0 0 0 (start: **Clear** 상태)

0 0 0 0 0 0 0 1

.

.

0 0 0 0 1 0 0 1

0 0 0 0 0 0 0 0 (**overflow** 발생) → 1msec 주기마다 인터럽트 발생

0 0 0 0 0 0 0 1

.

.

0 0 0 0 1 0 0 1

0 0 0 0 0 0 0 0 (**overflow** 발생) → 1msec 주기마다 인터럽트 발생

.

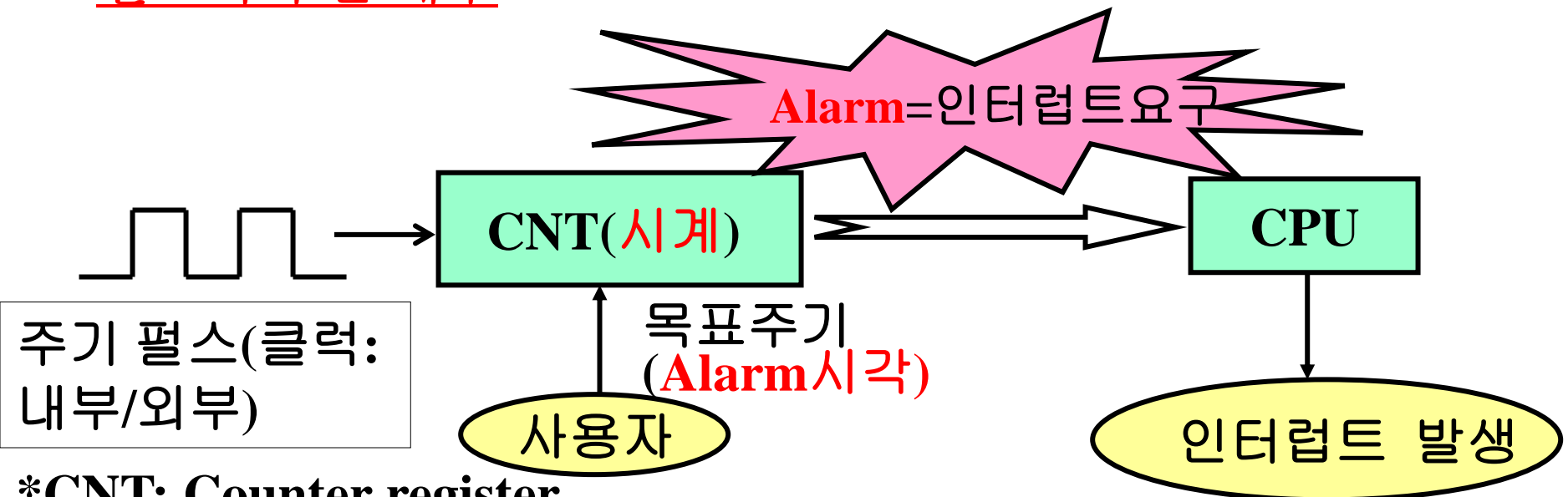
.

(10 pulses 입력)

(10 pulses 입력)

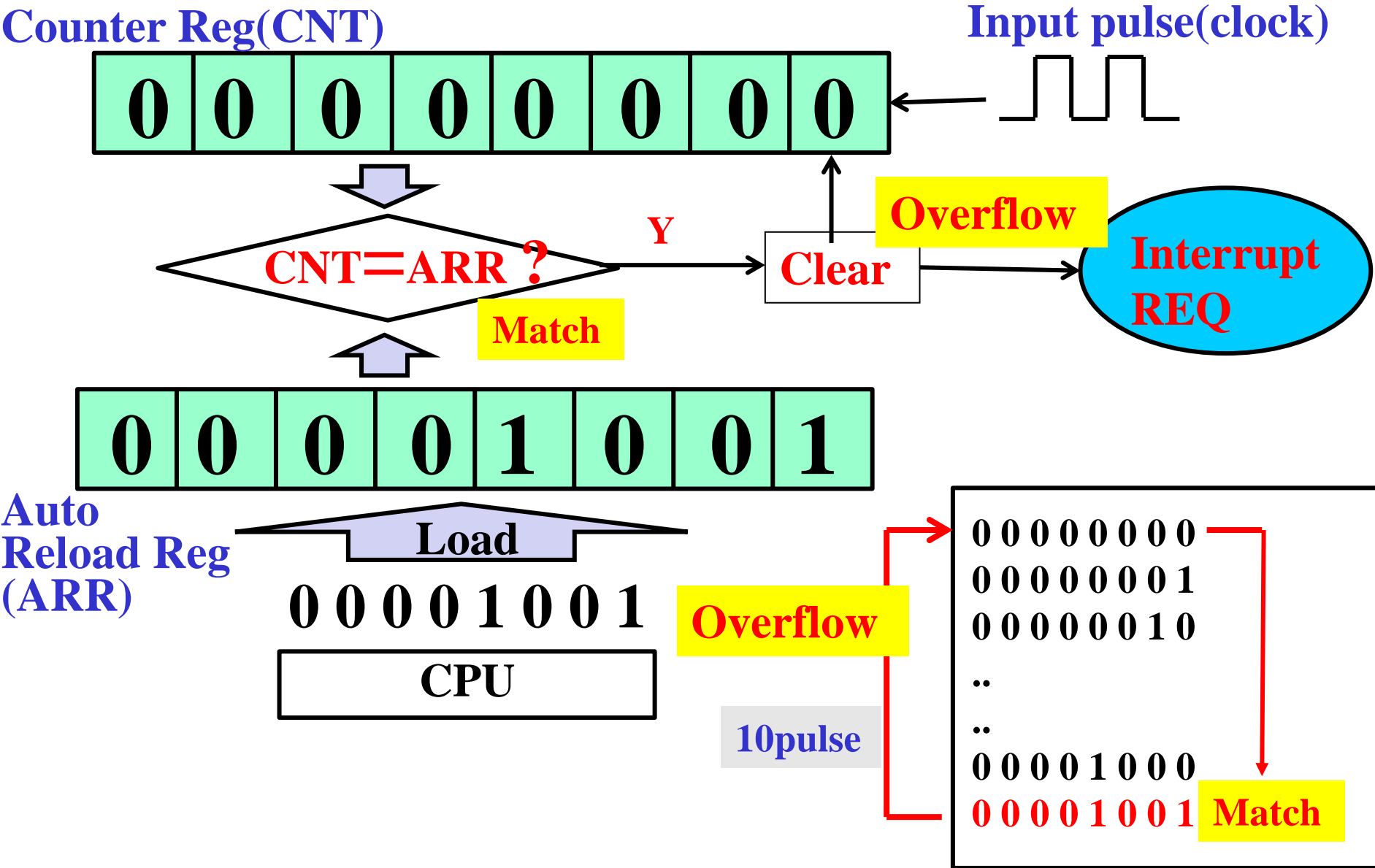
5. Timer의 주요 동작모드(mode)

- (1) Counter Mode** : Timer(Counter)에 외부펄스 또는 내부클럭을 입력하여 증가(UP) 또는 감소(DOWN)시키는 모드로서 다음 두가지 역할 수행
- 증가/감소하면서 어떤 특별한 사건(어떤 값과 비교하여 일치?, '0' ? 등)이 발생하면 인터럽트를 발생하게 함(주기적인 시간 발생 (시계의 Alarm과 유사))
 - 임의의 시각에 counter의 값을 읽어가서 특정시간동안 입력되는 펄스의 수를 계수

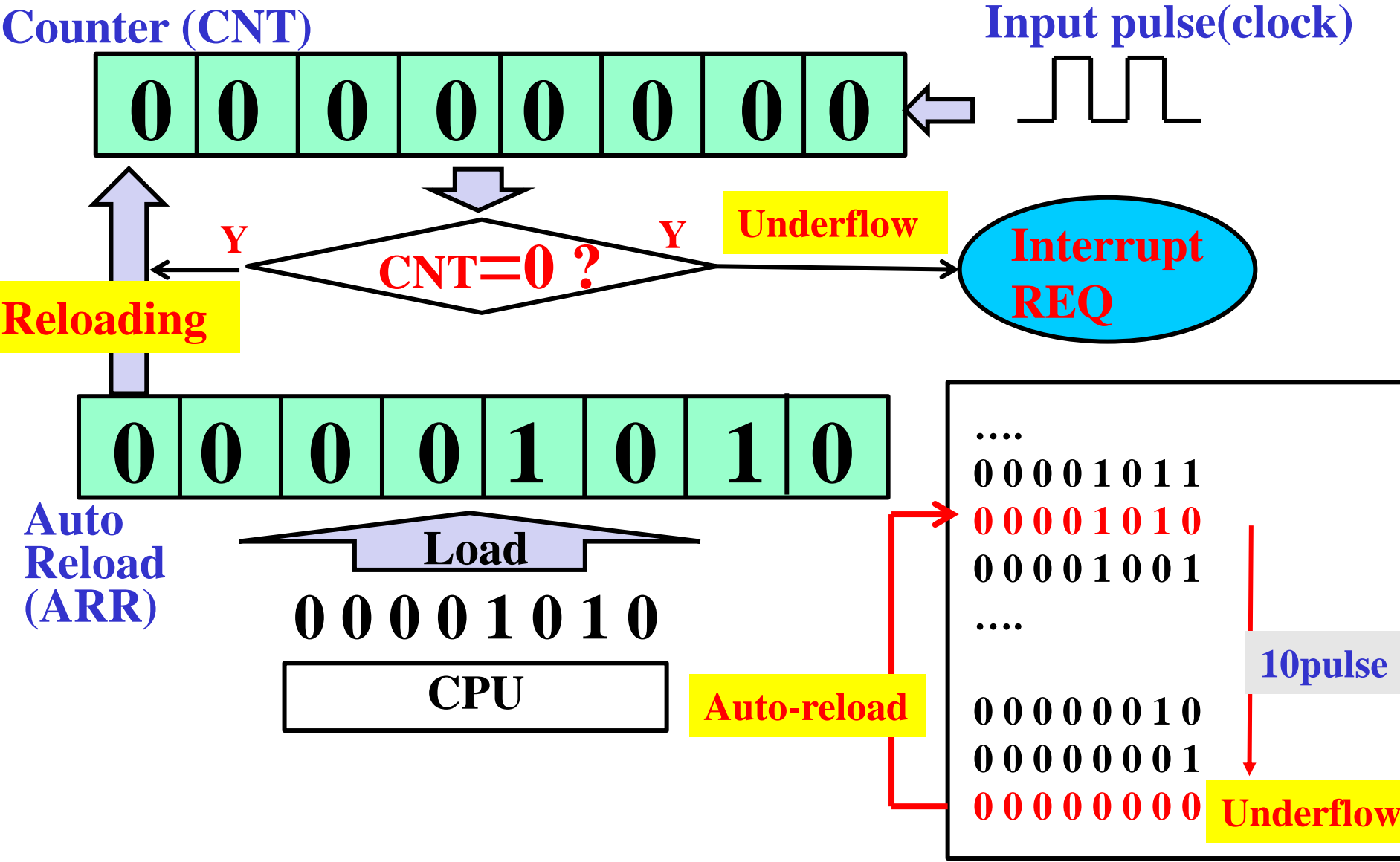


(1.1) Up Counting mode(Overflow를 이용한 주기시간 설정)

* 8bit 예



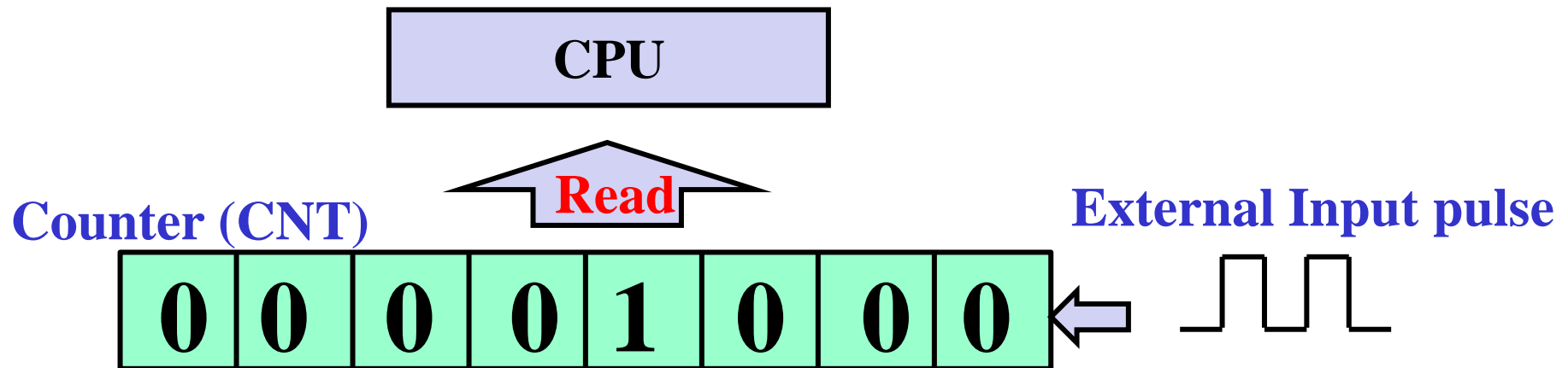
(1.2) Down Counting mode (Underflow를 이용한 주기시간 설정)
* 8bit 예



(1.3) Up/Down Counting mode (Overflow/Underflow를 이용한 주기시간 설정)

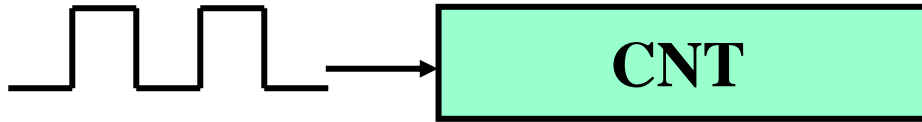
•그림 및 설명 생략

(1.4) Original Counting mode (임의의 시각에 CNT 값을 읽어 입력 펄스 수 계수) * 8bit 예



(2) Input Capture Mode : Pulse 발생 시간 측정

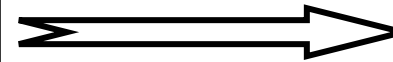
Input pulse(clock)



CNT



CCR



CPU



Signal Detector



Pulse 발생 시간
계산



Interrupt
REQ

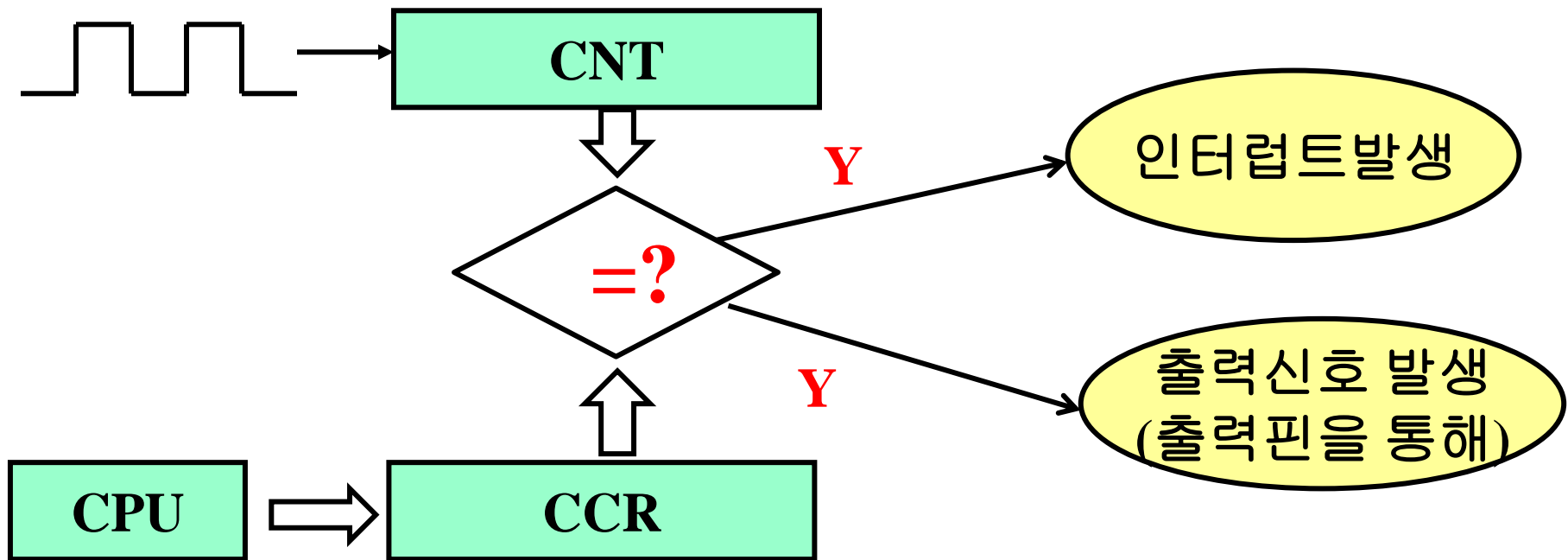


External Input pulse

* CCR: Capture /Compare Register

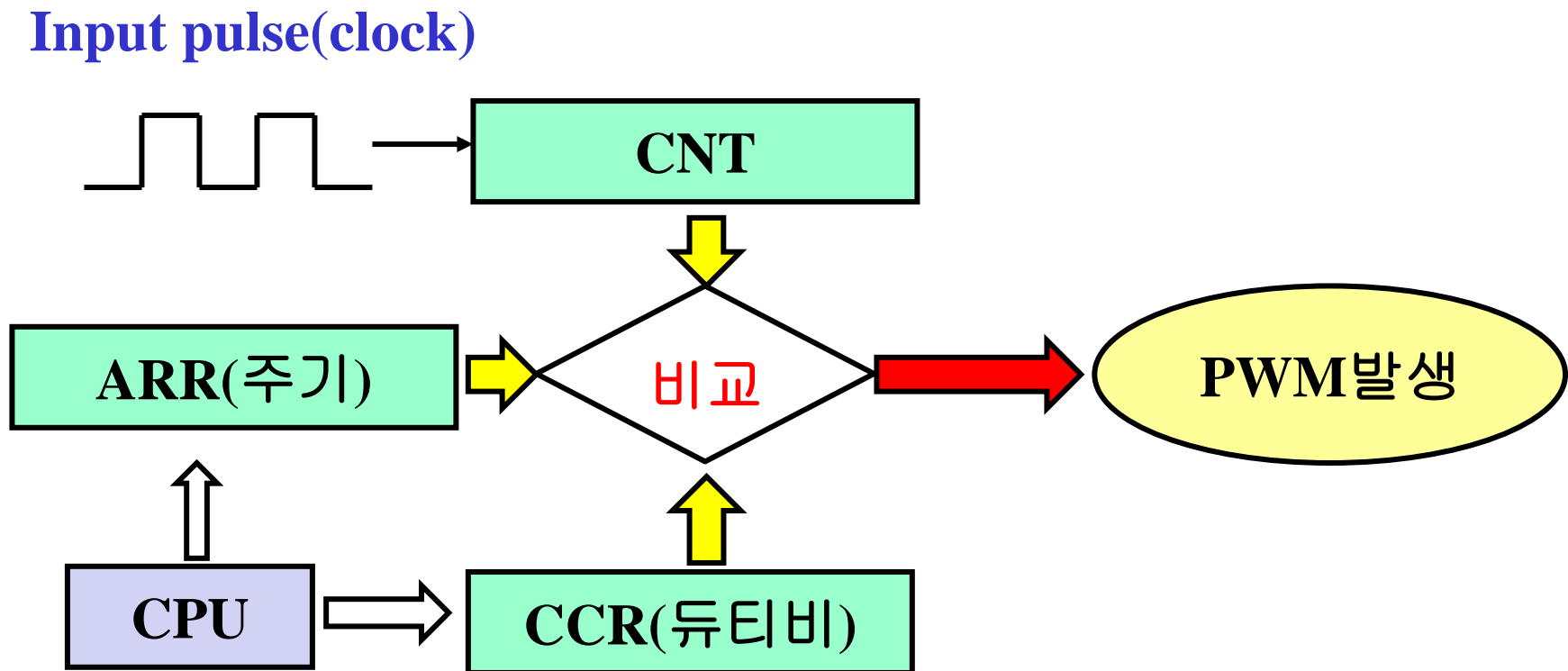
(3) Output Compare Mode : 기준값(CCR)과 비교하여 인터럽트나 외부출력Pulse 발생

Input pulse(clock)



* CCR: Capture /Compare Register

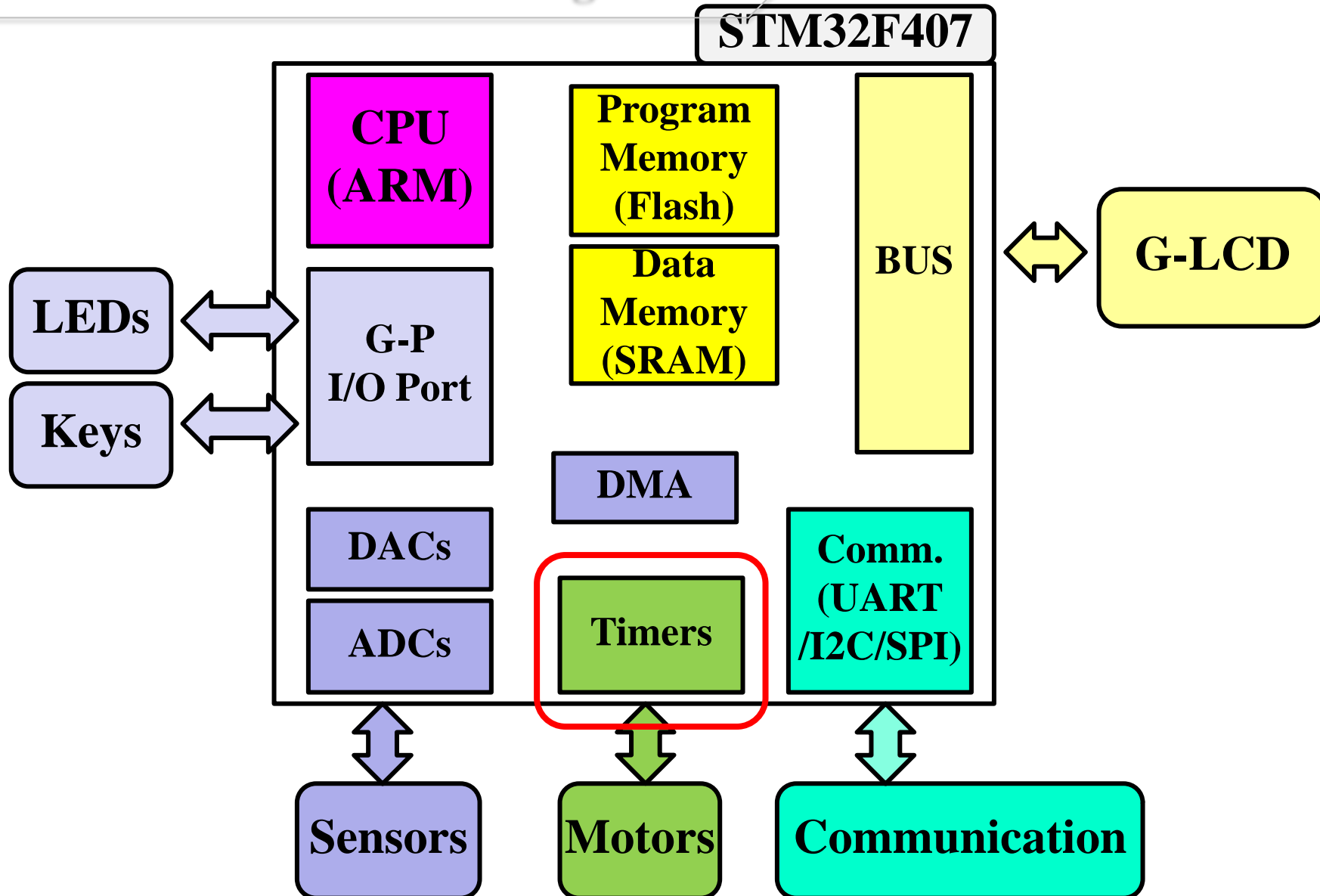
(4) PWM Mode : PWM 신호 발생 (주파수, 듀티비 선택 가능)



***PWM: Pulse Width Modulation**
ARR :Auto Reload Register
CCR : Capture/Compare Register

II. STM32F407의 Timer 응용

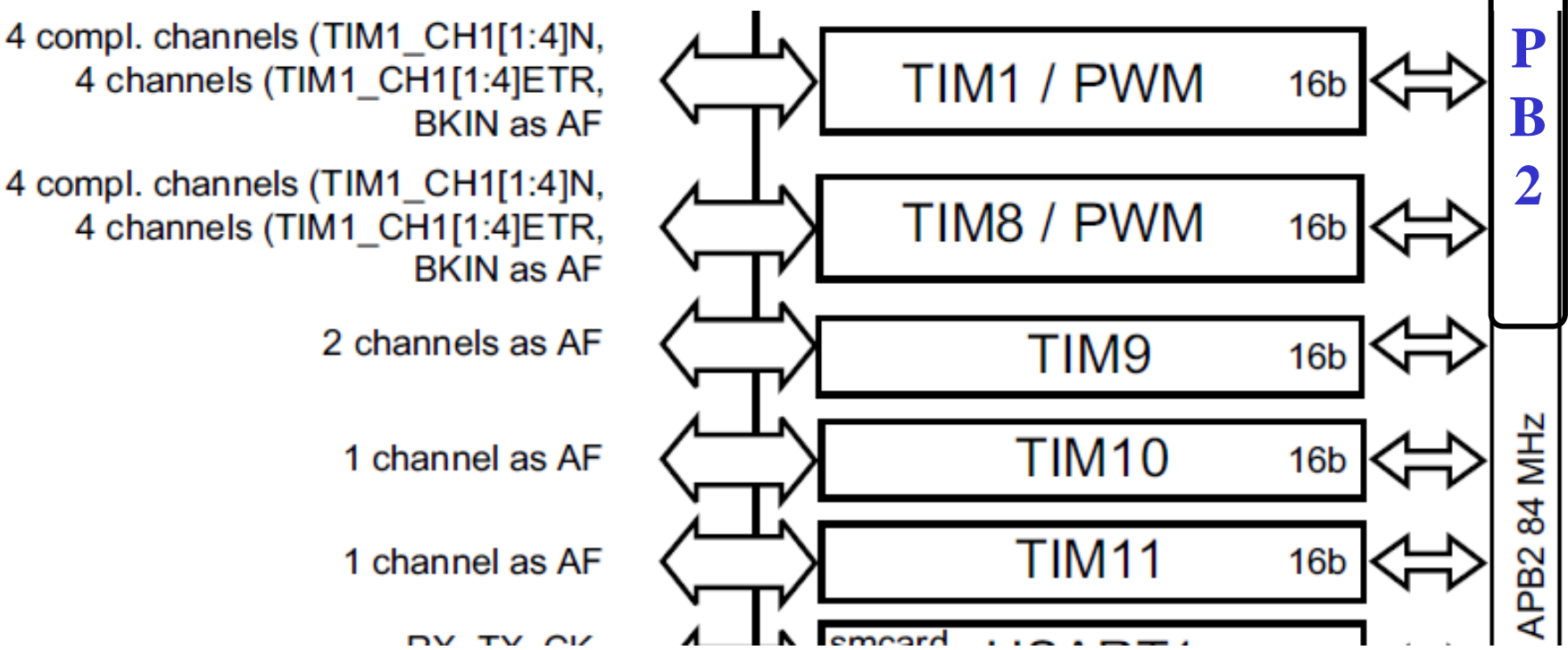
1. STM32F407 Timer : Block diagram



(1)

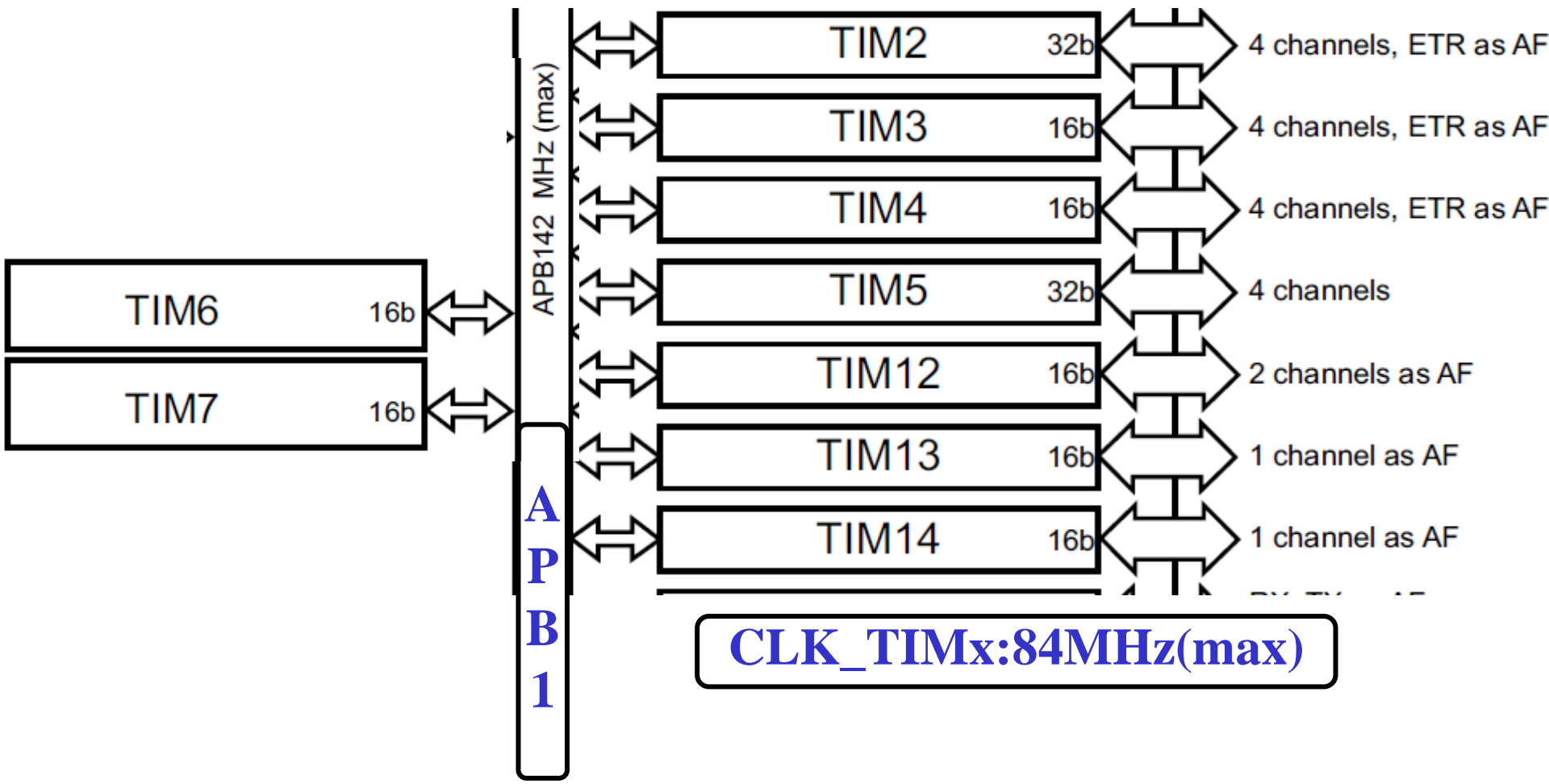


2.1 STM32F407의 Timer: Pin 및 기능(1)



CLK_TIMx: 168MHz(max)

2.2 STM32F407의 Timer: Pin 및 기능(2)



3. STM32F407 타이머의 개요

- **범용(General-purpose) 타이머** : 출력비교, 원펄스, 입력캡처, 센서 인터페이스(엔코더, 홀 센서 등) 등의 용도로 사용할 수 있는 범용 기능을 가지는 타이머
- **고급제어(Advanced-control) 타이머** : 범용 타이머 보다 더 많은 기능을 가지는 타이머. 주로 모터 제어와 디지털 파워 변환(power conversion) 용도로 사용 가능한 데드타임을 가지는 3상 출력, 비상 셧다운 입력 등의 기능이 더 추가됨
- **기본(Basic) 타이머** : 입출력 기능은 없고 시간기반 타이머(timebase timer)나 DAC의 트리거 용도로만 사용되는 타이머

*채널(Channel) : 1개의 타이머 모듈안에 여러 개의 동일한 입출력 회로가 있는 경우, 각 입출력 회로를 구분하기 위한 용어

* 대부분의 타이머는 16비트 타이머(단, 일부 범용타이머는 32비트)

* 고급제어타이머, 범용타이머: 업(up), 다운(down), 센터 얼라인(center aligned) 카운팅이 가능함

* 기본타이머: 업(up) 카운팅만 가능함

<표1> STM32F4 패밀리의 타이머의 종류

| Timer type | Counter resolution | Counter type | DMA | Channels | Comp. channels | Synchronization | |
|---|---------------------------------|-----------------------------|-----|----------|----------------|-----------------|---------------|
| | | | | | | Master config. | Slave config. |
| Advanced | 16 bit | up, down and center aligned | Yes | 4 | 3 | Yes | Yes |
| General-purpose | 16 bit 32 bit ⁽¹⁾ | up, down and center aligned | Yes | 4 | 0 | Yes | Yes |
| Basic | 16 bit | up | Yes | 0 | 0 | Yes | No |
| 1-channel | 16 bit | up | No | 1 | 0 | Yes (OC signal) | No |
| 2-channel | 16 bit | up | No | 2 | 0 | Yes | Yes |
| 1-channel with one complementary output | 16 bit | up | Yes | 1 | 1 | Yes (OC signal) | No |
| 2-channel with one complementary output | 16 bit | up | Yes | 2 | 1 | No | Yes |

<표2> STM32F407의 타이머의 구성

| Timer type | timer | resolution | Count type | Prescaler |
|------------------------|--------------------------|---|------------------------------|-----------|
| Advanced-control Timer | TIM1,TIM8 | 16bit | Up, Down | 1 ~ 65535 |
| General-purpose timer | TIM2~TIM5, TIM9~TIM14 | TIM2,5 = 32bit TIM3,4 = 16bit TIM9~14=16bit | TIM2~5=Up,Down TIM9~14=Up | 1 ~ 65535 |
| Basic Timer | TIM6,TIM7 | 16bit | Up | 1 ~ 65535 |

- 4채널 타이머 : TIM1~5,8
- 2채널 타이머 : TIM9,12
- 1채널 타이머 : TIM10,11,13,14
- 0채널 타이머(기본타이머: 외부로 발생하는 신호 및 회로, 핀 등이 없는 타이머, 즉, Counter mode만 있는 경우) : TIM6,7

4. 범용 타이머

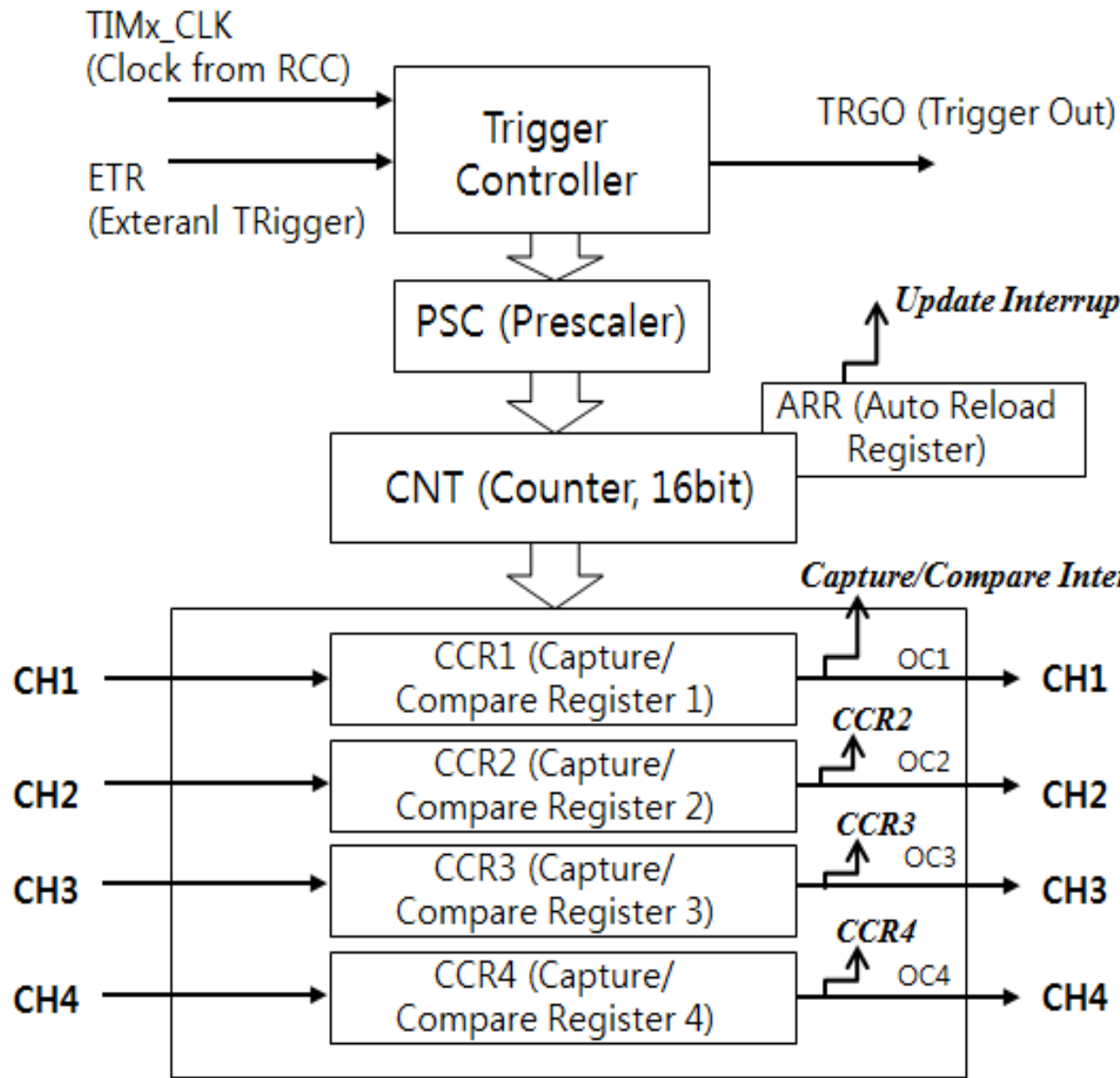
4.1 개요

4.1.1 범용 타이머(TIM2~TIM5, TIM9~TIM14) 특징

- 1개의 16 비트 카운터(Counter)를 가짐 (TIM2,5 는 32bit)
 - 업, 다운, 업/다운 모드로 동작이 가능
 - Auto-reload 기능이 있음
- 16비트의 프리스케일러(Prescaler)를 이용하여 카운터의 동작 클럭을 1~1/65,536까지 분주가능
- 4개의 16비트 Capture/Compare 채널을 가짐
- 동작모드 종류
 - 카운터(Counter) 모드
 - 입력 캡처(Input Capture) 모드
 - 출력 비교(Output Compare) 모드
 - PWM 출력 모드, PWM 입력 모드
 - 원 펄스(One-pulse) 모드
 - 강제 출력(Forced output) 모드
 - 엔코더 인터페이스 모드, 타이머 동기화(Timer synchronization)

- **엔코더 및 홀(Hall) 센서 인터페이스가 가능**
- **다음의 경우 6 개의 독립적인 IRQ/DMA 요청 신호를 발생**
 - **Update event (Overflow, Underflow, Counter initialization)**
 - **Input Capture**
 - **Output Compare match**
 - **Trigger event(Counter start, stop, initialization or count by internal / external trigger)**
- **TIM2~5: APB1 Bus (42MHz(max) clock : CK_APB1=CK_CNT)**
TIM12~14: APB1 Bus (42MHz(max) clock : CK_APB1=CK_CNT)
TIM9~11: APB2 Bus (84MHz(max) clock : CK_APB2=CK_CNT)
- **인터럽트 종류**
 - **Overflow, Underflow, Input Capture, Compare Match**
Counter initialization etc.
- **TIM2~5 : 4채널 타이머**
TIM12 : 2채널 타이머
TIM10,11 : 1채널 타이머

4.1.2 범용타이머의 구조 (TIMx (x = 2~5))

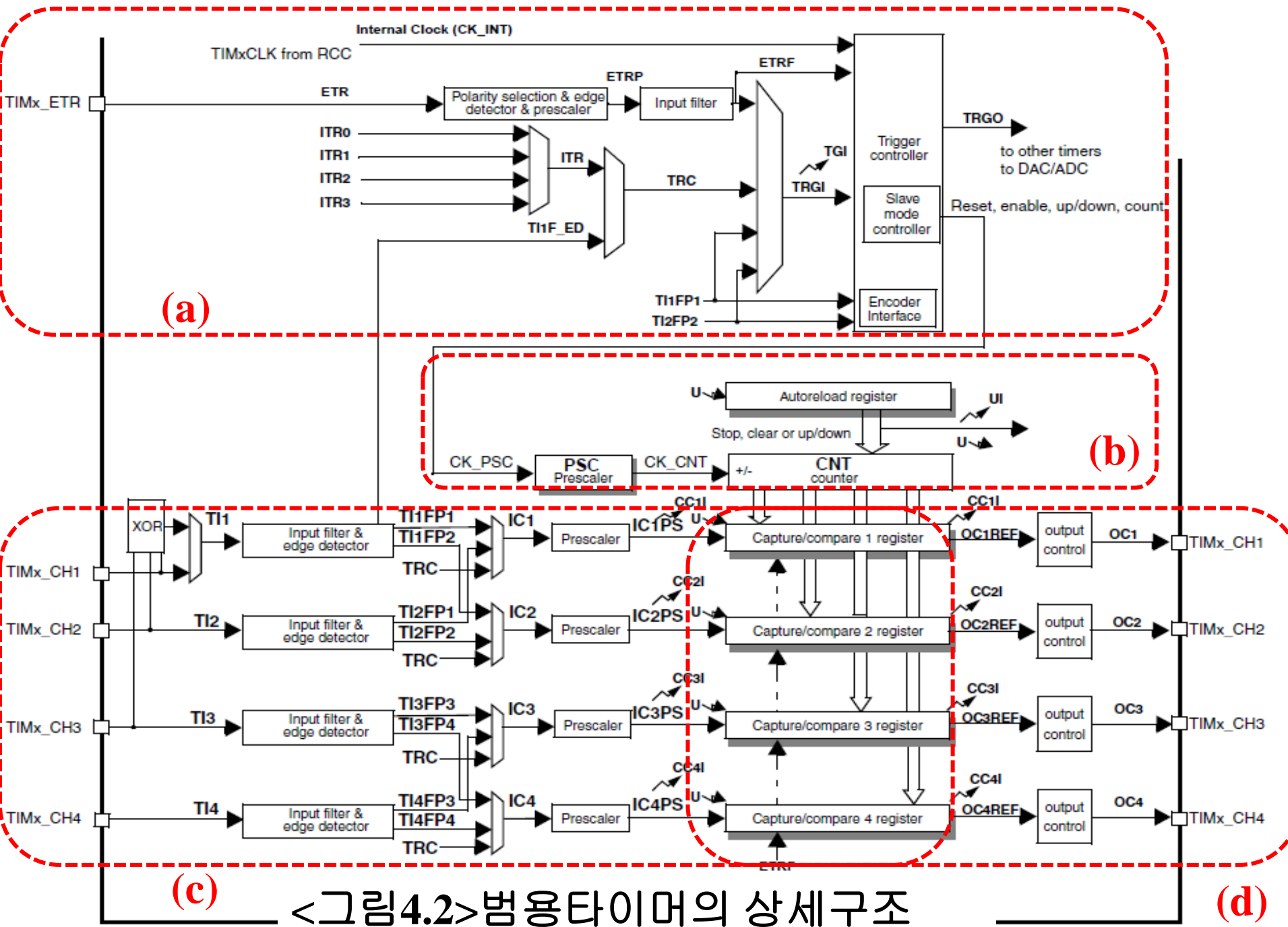


***TIM9,12: CH1,2 only**
***TIM10,11,13,14: CH1 only**

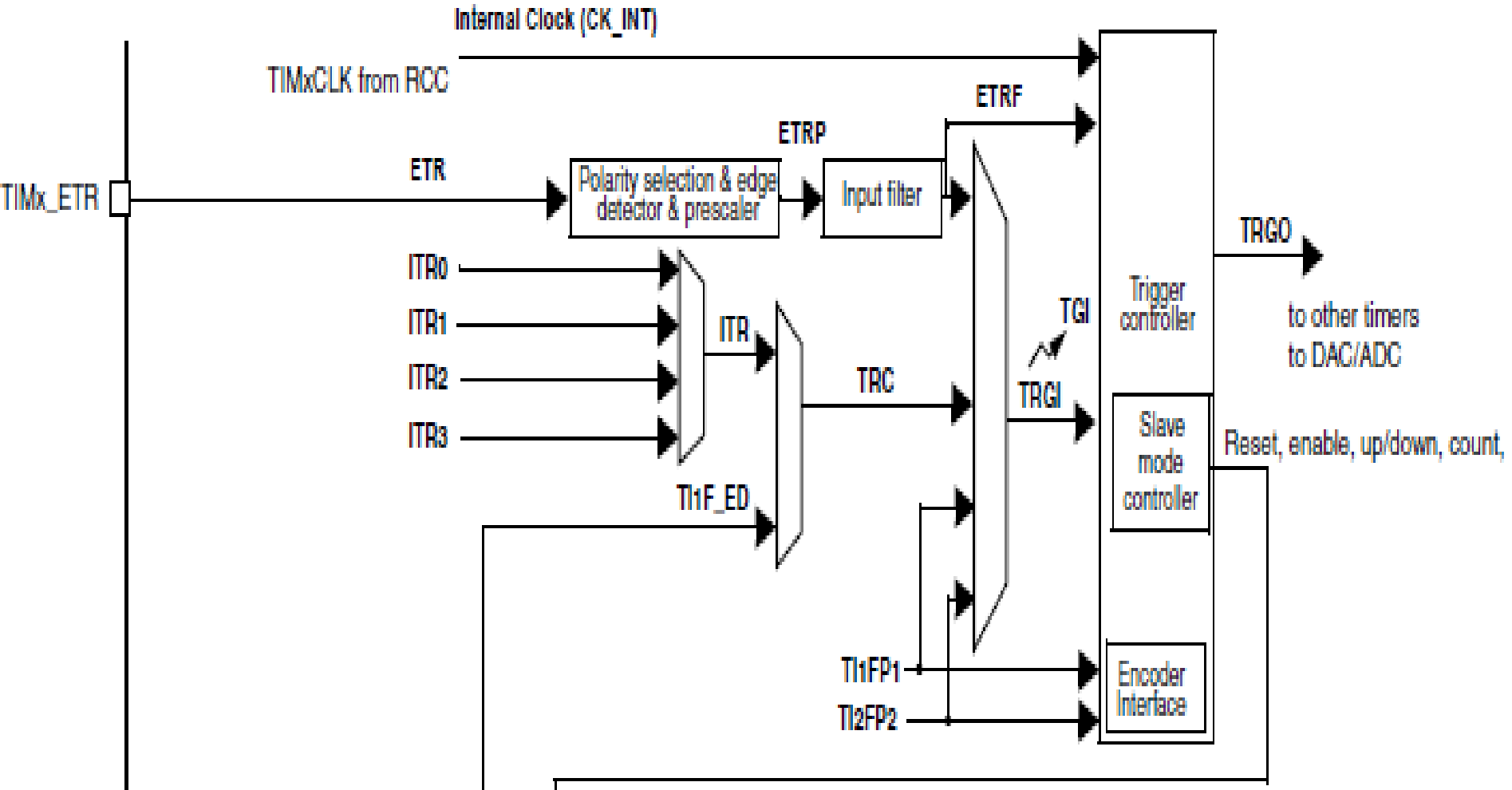
***Update event(Interrupt)
: Overflow or Underflow**

-CCR section은 4개의 CCRx Channel로 구성
-각 channel은 Input이 나 Output mode로 작동, 하나의 외부핀과 연결

<그림4.1>범용타이머의 간략구조



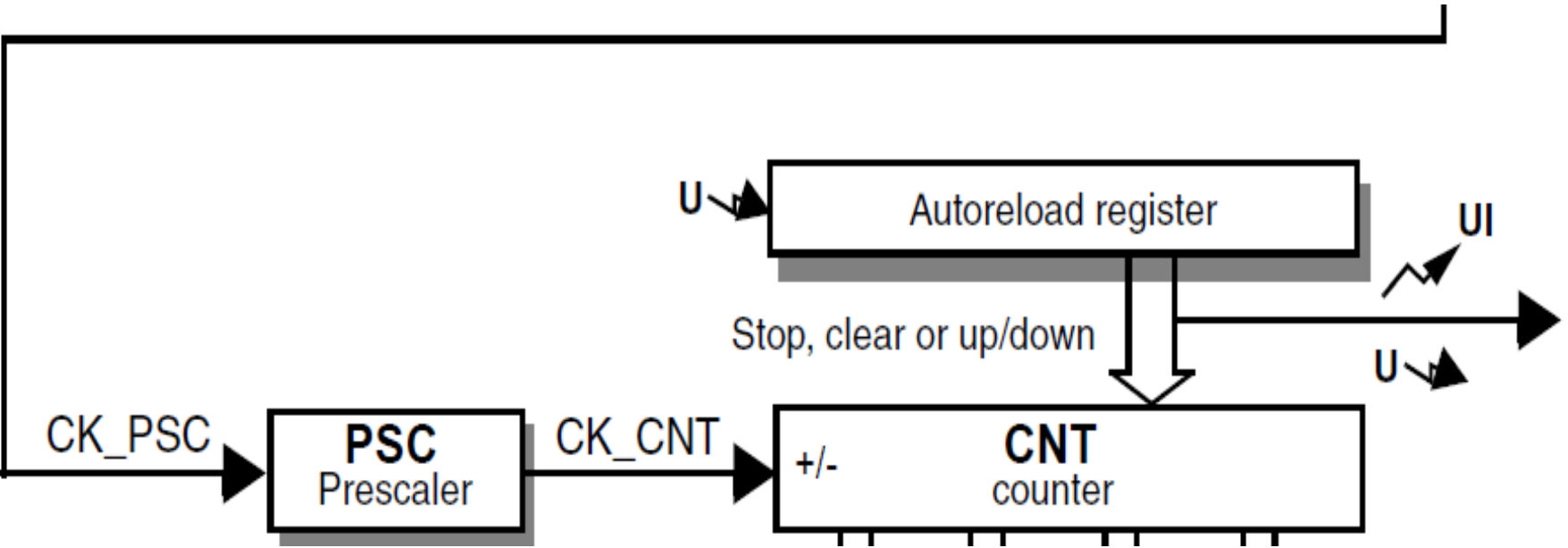
<그림4.2>범용타이머의 상세구조



(a) Pulse Input, Trigger controller Unit

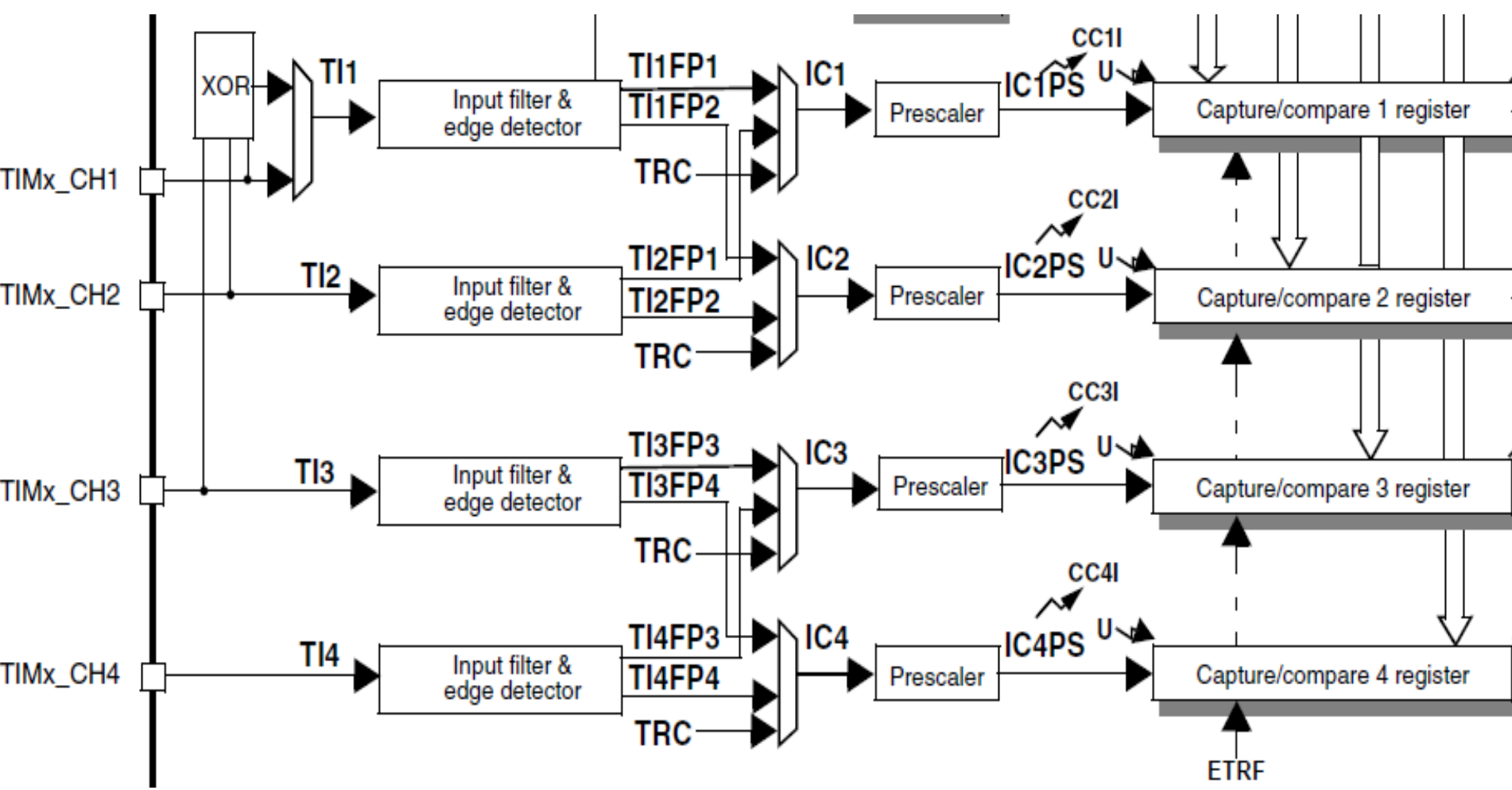
***TIMx_ETR: 외부 트리거 입력핀으로 이용**

<그림 4.2> 범용 타이머의 상세 구조 (pulse Input, trigger controller)



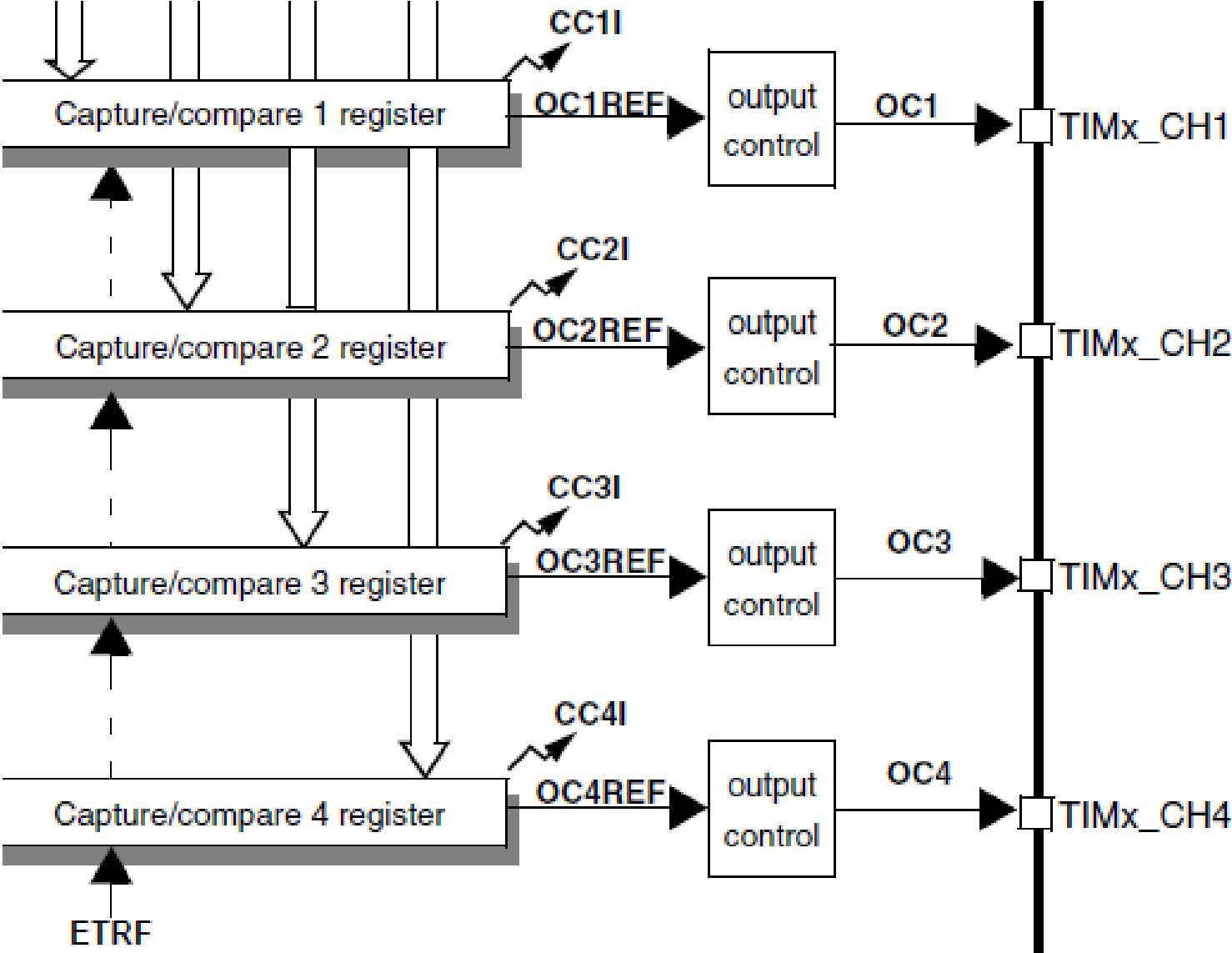
(b) Time base Unit

<그림 4.2> 범용 타이머의 상세 구조 (Time base Unit)



(c) Input Capture Unit

<그림 4.2>범용타이머의 상세구조(Input Capture Unit)



(d) Output Compare Unit

<그림4.2>범용타이머의 상세구조(Output Compare Unit)

4.1.3 범용 타이머 주요 구성요소

- 프리스케일러(PSC : Prescaler) : 16비트의 프리스케일러는 공급되는 클럭(**TIMx_CLK**, 또는 **CK_PSC**)을 1~65,536의 값으로 나누어 카운터의 동작 클럭(**CK_CNT**)을 만들어내는 분주기
 - 카운터(CNT : Counter) : 16비트의 카운터는 타이머의 핵심적인 요소로써, 공급되는 클럭을 이용한 업, 다운, 업/다운 카운팅이 가능
 - 오토 리로드 레지스터(ARR : Auto Reload Register)
 - 업 카운터의 경우 '카운터(CNT) 값 = ARR의 설정값'이 되면 CNT는 0부터 다시 업 카운팅
 - 다운 카운터의 경우 '카운터(CNT) 값 = 0'이 되면 CNT는 ARR의 설정값부터 다시 다운 카운팅
 - 캡처/비교기(CCR : Capture/Compare Register)
 - 입력 신호가 주어질 때 카운터(CNT) 값을 캡처하거나,
 - 또는 '카운터(CNT) 값 = CCR의 설정값'이 되면 인터럽트를 발생하거나 출력 채널로 0 또는 1을 출력
- * Slave mode controller: 외부 신호에 의해 타이머의 동작을 제어하는 제어기 (관련레지스터: SMCR(Slave Mode Control Register))

[참고] 프리스케일러에서 카운터의 동작 클럭(CK_CNT)의 결정

- $CK_CNT = \text{프리스케일러로 공급되는 클럭}(CK_PSC) /$

(프리스케일러의 설정 값 +1)

(예) 프리스케일러로 들어오는 클럭이 72MHz이고, 프리스케일러의 설정 값을 5999로 하면, 카운터(CNT)로 공급되는 클럭은

$$72,000,000 / (5,999 + 1) = 12,000(\text{Hz})$$

단, 프리스케일러도 16비트이기 때문에 설정 값이 0xFFFF(65535)를 넘어서는 안된다는 점을 유의

- 외부입력: 4개의 채널 입력(TIMx_CH1 ~ TIMx_CH4)과 외부 트리거(ETR : External TRigger)
- 외부출력: 4개의 채널(TIMx_CH1 ~ TIMx_CH4)

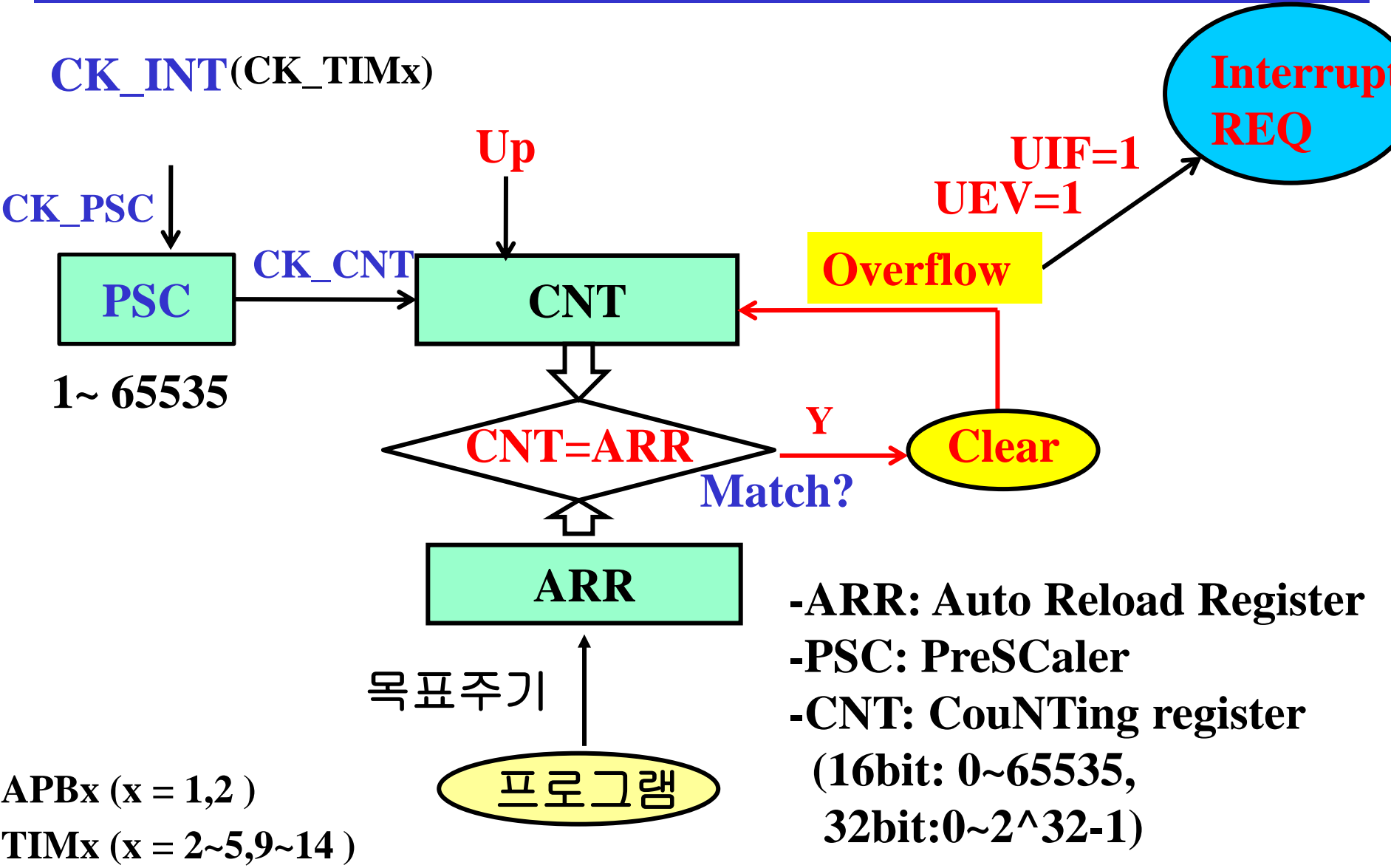
4.2 범용 타이머의 주요 기능(모드)

4.2.1 카운터 모드

- 정의: Timer(Counter)에 외부펄스 또는 내부클럭을 입력하여 증가(UP) 또는 감소(DOWN)시키는 모드로서 다음 두가지 역할 수행
- 증가/감소하면서 어떤 특별한 사건(어떤 값과 비교하여 일치?, '0' ? 등)이 발생하면 인터럽트를 발생하게 함(주기적인 시간 발생 (시계의 Alarm과 유사))
- 임의의 시각에 counter의 값을 읽어가서 특정시간동안 입력되는 펄스의 수를 계수 (original counter 역할)

(1) 업 카운팅(Up counting) 모드

- 정의: 카운터(CNT)의 값이 증가하면서 카운팅하는 모드
- 카운터는 $CNT = 0$ 부터 시작하여 $CNT = ARR(\text{Auto-Reload})$ 값까지 증가한 후에, 다시 0부터 카운팅을 시작하는 작업을 계속하여 반복
- $CNT = 0$ 이 될 때, **Overflow** 즉, 업데이트 이벤트(UEV)와 업데이트 인터럽트(UI)가 발생(그림 4.3~4.5참조)



<그림4.3> 범용타이머의 Up Counting 모드 동작도

(예) CK_INT사용한 범용타이머의 Up Counting mode를 이용한 Overflow 인터럽트 발생

(Q) 1초 Overflow INT 만들기

- **Timer: TIM2**

- * TIM2는 APB1(42MHz)이므로 최대주파수는 84MHz

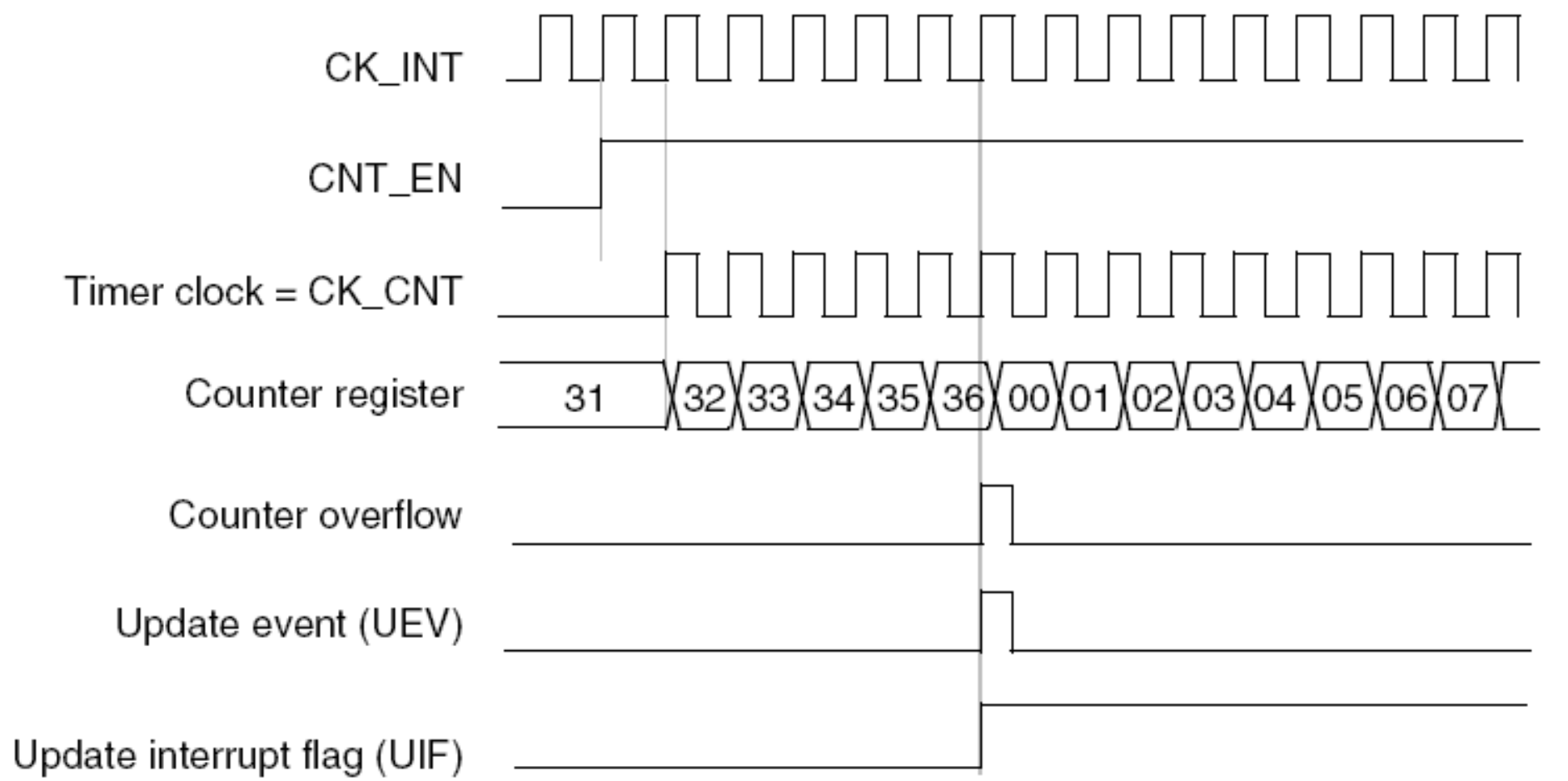
- * CK_INT(CK_TIM2= CK_PSC) = 84MHz 라고 가정

(A)

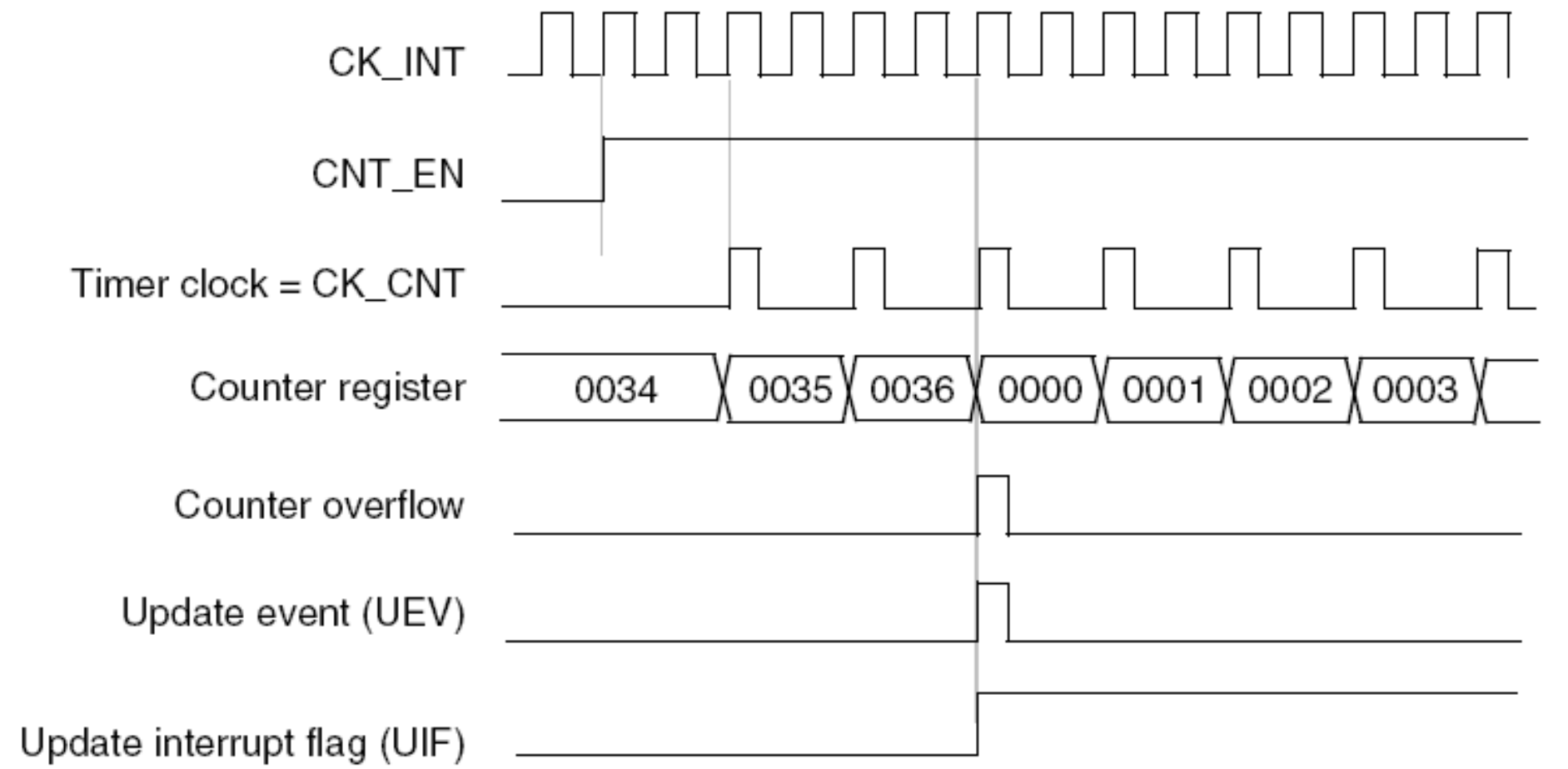
- **PSC = 8400 - 1 로 설정**

- **CK_CNT = 10KHz 가 됨 ($84\text{M} / 8400 = 10000$)**

- **ARR = 10000 - 1 로 설정하면 10000개의 pulse(CK_INT)가 입력되면 1초가 됨(1 pulse의 주기는 0.1ms)**



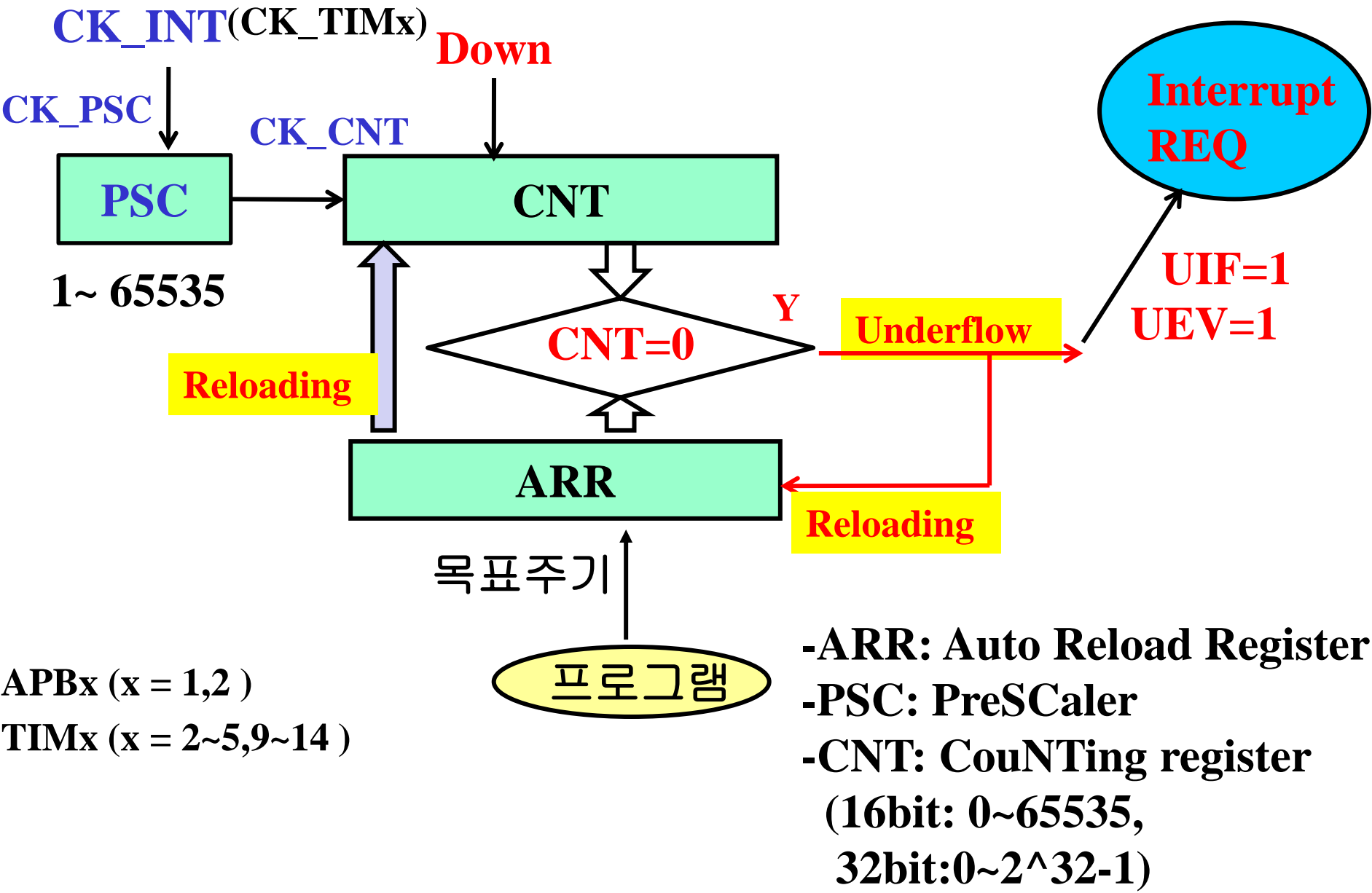
<그림4.4 > 업 카운팅 모드의 동작 시간도 (ARR=36, 분주비=1 인 경우)



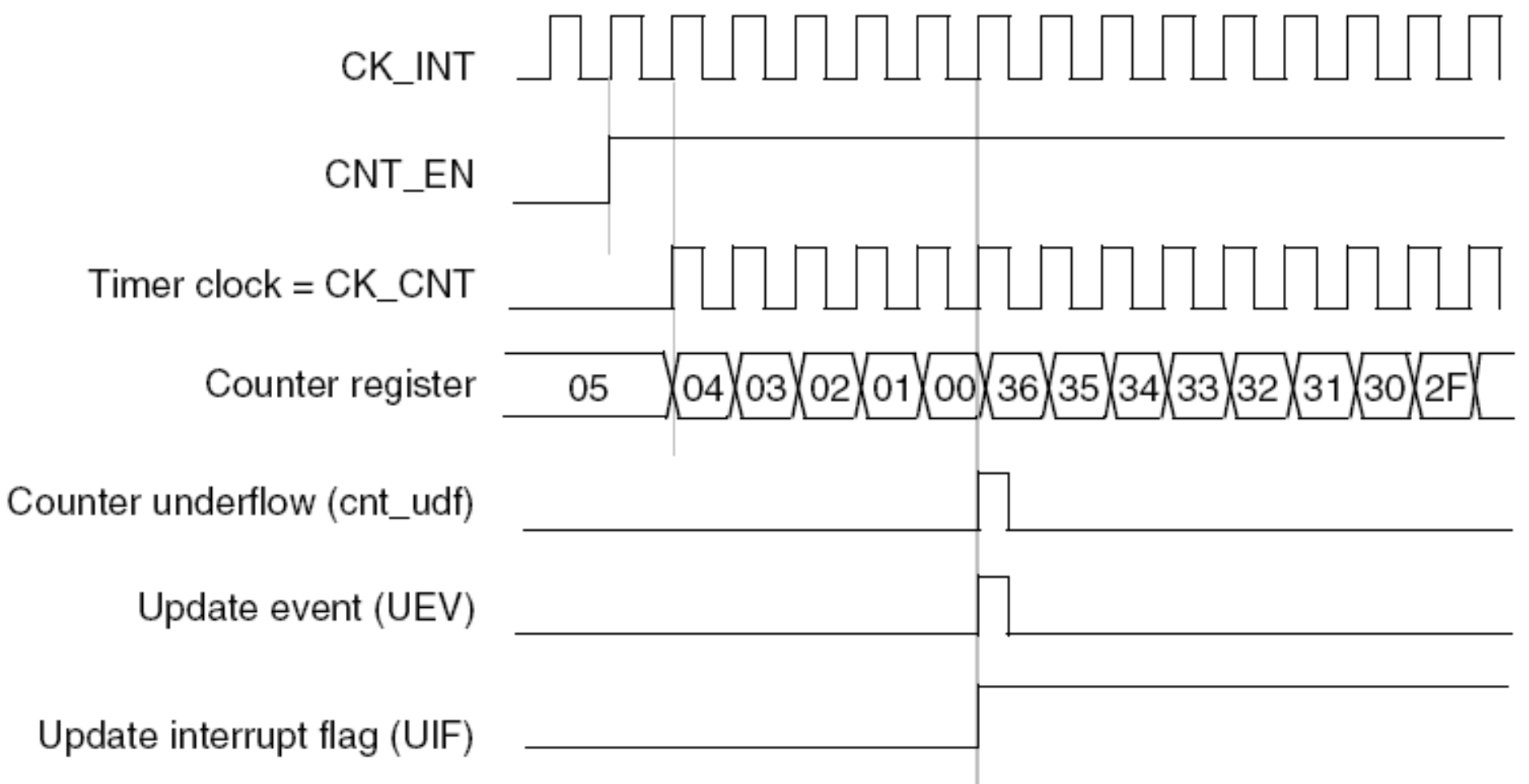
<그림4.5> 업 카운팅 모드의 동작 시간도 (ARR=36, 분주비=2인 경우)

(2) 다운 카운팅(Down counting) 모드

- 정의: 카운터(CNT)의 값이 감소하면서 카운팅을 하는 모드
- 카운터는 $CNT = ARR$ 부터 $CNT = 0$ 까지 감소한 후에, 다시 $CNT = ARR$ 부터 카운팅을 시작하는 작업을 계속하여 반복
- $CNT = ARR$ 이 될 때, 언더플로우(underflow), 즉, 업데이트 이벤트(UEV)와 업데이트 인터럽트(UI)가 발생 (그림 4.6~4.7 참조)



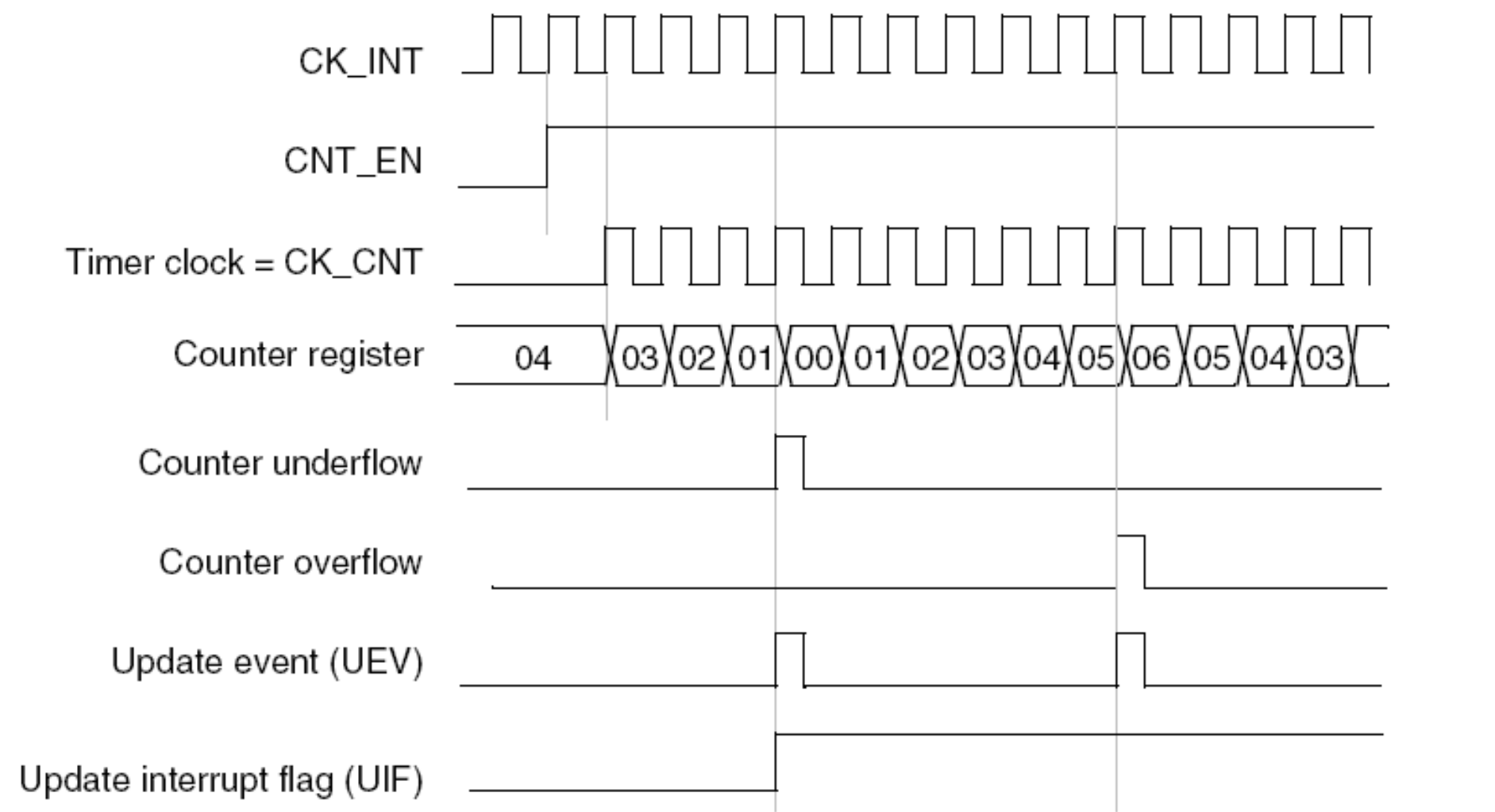
<그림4.6> 범용타이머의 Down Counting 모드 동작도



<그림4.7> 다운 카운팅 모드의 동작 시간도 (ARR=36, 분주비=1 경우)

(3) 업/다운 카운팅 (Up/Down counting) 모드 (center-aligned 모드)

- 정의: 카운터의 값(CNT)이 증가한 후 다시 감소하면서 카운팅을 하는 모드
- 카운터는 0부터 오토-리로드 값(ARR)까지 증가한 후에 다시 0까지 감소함. 이 후에는 다시 증가 → 감소 → 증가 → 감소.. 를 하며 계속 카운팅 반복
- 업 카운터는 0 ~ (ARR-1) 까지 카운팅
다운 카운터는 ARR ~ 1 까지 카운팅을 하는 개념으로 동작
- 업 카운팅을 할 때는 $CNT = ARR$ 이 될 때, 오버플로우, 업데이트 이벤트(UEV)와 업데이트 인터럽트(UI)가 발생
- 다운 카운팅을 할 때는 $CNT = 0$ 이 될 때 언더플로우, 업데이트 이벤트(UEV)와 업데이트 인터럽트(UI)가 발생
- <그림 4.8> 참조

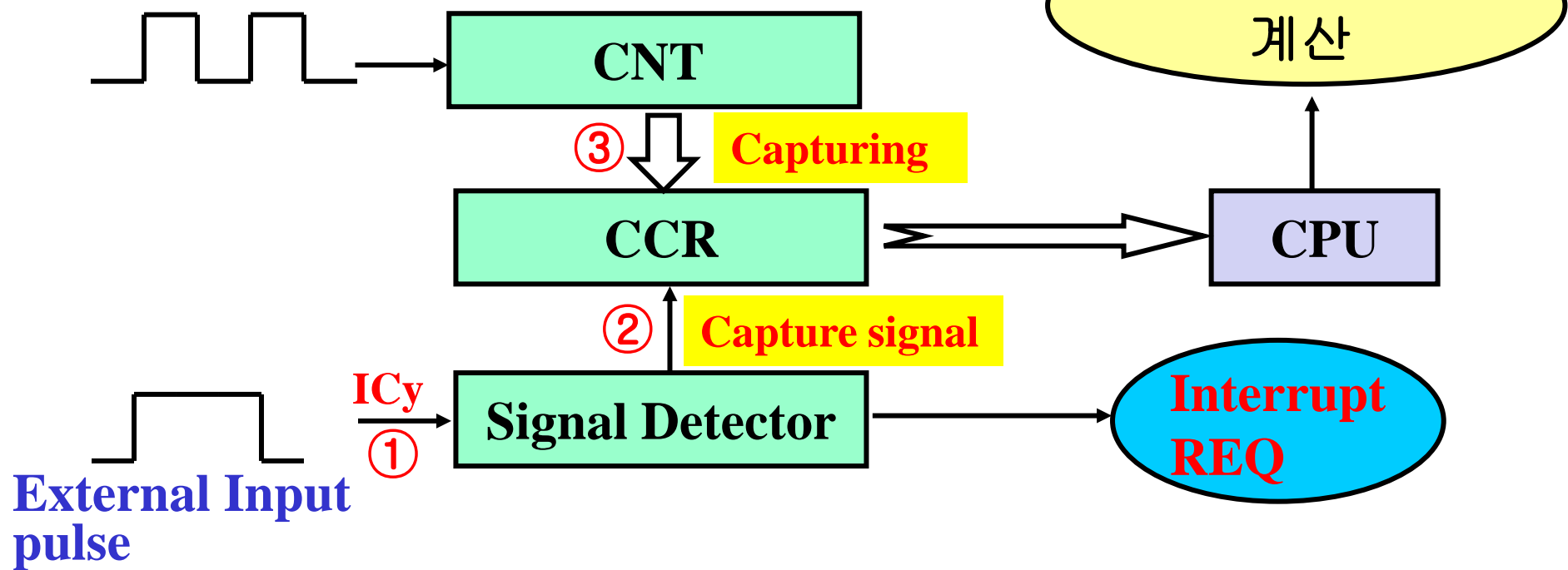


<그림4.8> 업/다운 카운팅 모드의 동작 시간도 (ARR=6, 분주비= 1 인 경우)

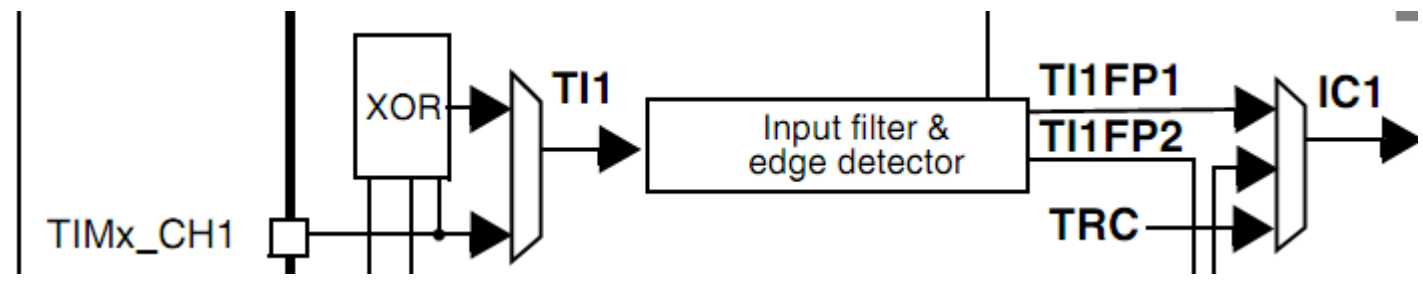
4.2.2 입력 캡처(Input Capture) 모드

- 정의: 외부의 입력신호가 들어오는 순간, 그 때의 카운터(CNT) 값을 캡처하는 동작을 하는 모드로서 외부입력신호의 입력시각 및 듀티비의 정보를 파악하는 기능
- 입력 캡처신호가 ICy (y=1~4)(STM32F407의 핀:TIMx_Chy(x:1~14(6,7제외),y=1~4))를 통해 입력되면,
 - 그 때의 카운터(CNT) 값이 캡처/비교레지스터(CCR)에 저장되며,
 - 캡처 인터럽트플래그(CCxIF)가 SET
 - 인터럽트가 인에이블되어 있는 경우는, 인터럽트(CCxI)가 발생
 - 캡처 인터럽트플래그(CCxIF)가 이미 설정되어 있는데 캡처가 또 발생하는 경우는 오버 캡처 플래그(CCxOF)가 설정

Input pulse(clock)



- * **CCR: Capture /Compare Register**
- * $ICy = TIMx_CHy$ (x: 1~14(6,7제외), y:1~4)

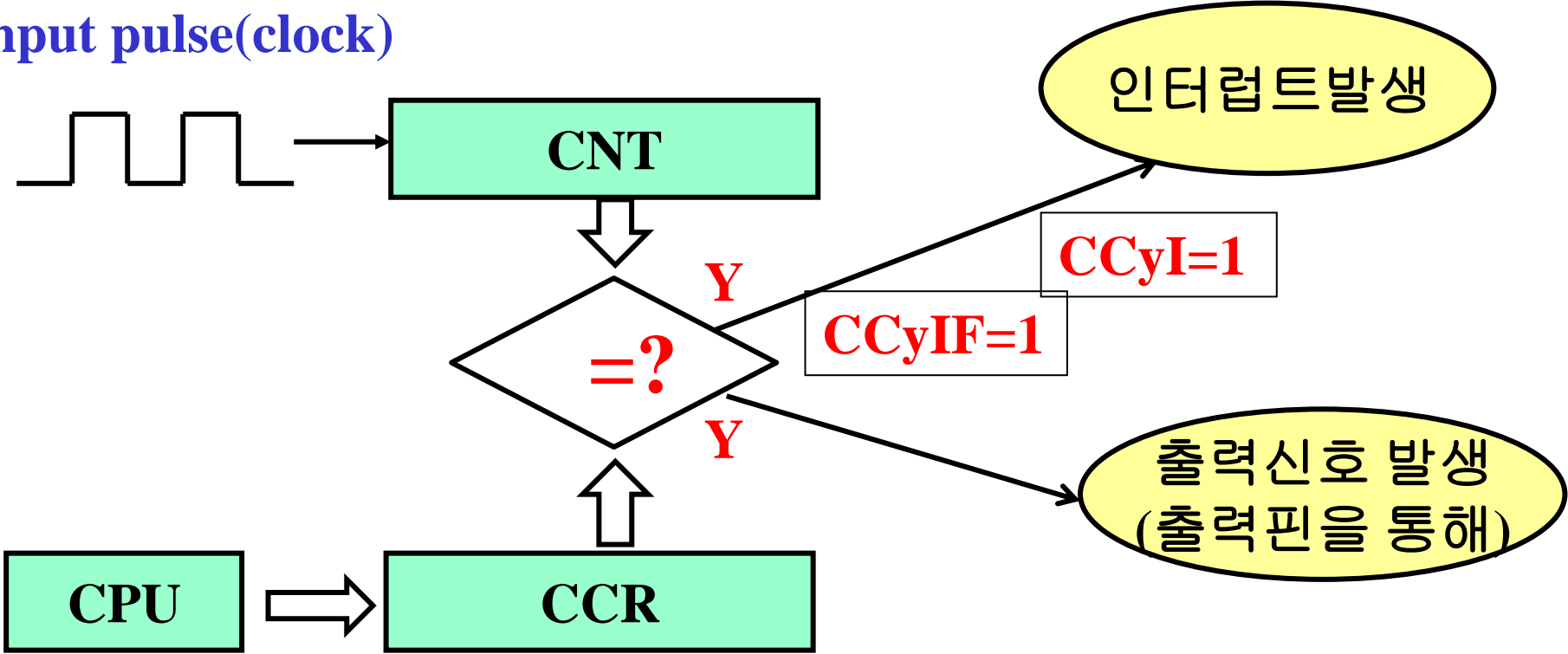


<그림4.9_1> 범용타이머의 Input Capture 회로의 입력회로

4.2.3 출력 비교(OC : Output Compare) 모드

•정의: 카운터(CNT)의 출력값이 캡처/비교레지스터에 설정된 비교값(CCRx)과 일치할 때, 인터럽트(CCx)나 해당 핀에 출력(OCx)이 발생하는 모드

Input pulse(clock)



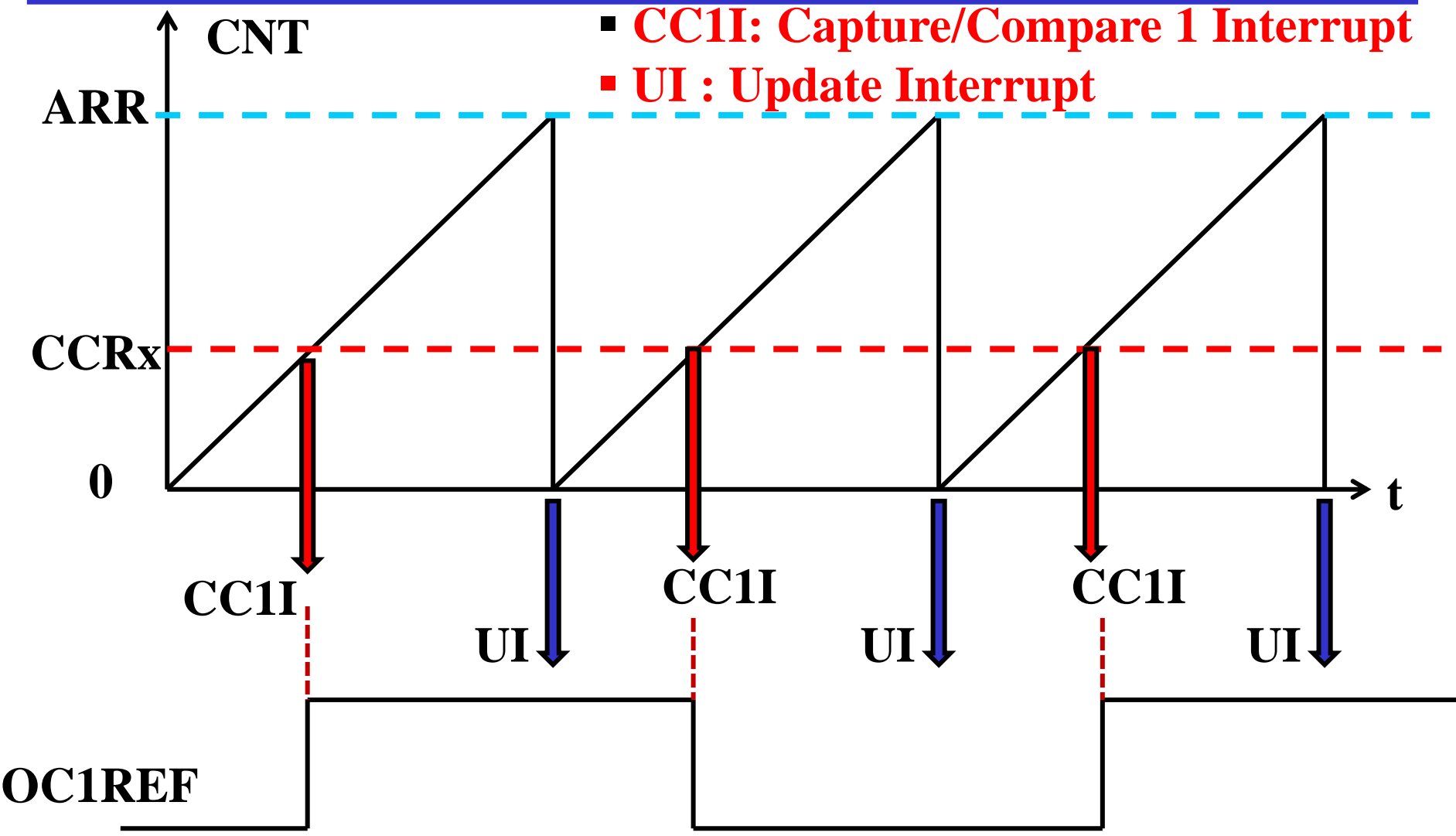
*CCyI : Channel y 에서 발생한 Output Compare Interrupt
<그림4.9_2> 범용타이머의 Input Capture 회로의 입력회로

•핀의 출력값은 다음과 같은 세부모드로 설정

| 출력비교 세부모드 | 효 과 |
|---------------------------------------|---|
| Output compare timing | CCR _x 이 출력에 영향없음. 일반적인 타이머 동작과 유사 |
| Output compare active | CNT = CCR _x 일때, OC _x 값이 ‘high’ |
| Output compare inactive | CNT = CCR _x 일때, OC _x 값이 ‘low’ |
| Output compare toggle | CNT = CCR _x 일때, OC _x 값이 ‘toggle’ |
| Output compare forced active/inactive | CNT의 값과 상관없이 OC _x 값이 ‘high (active)’ 또는 ‘low (inactive)’ |

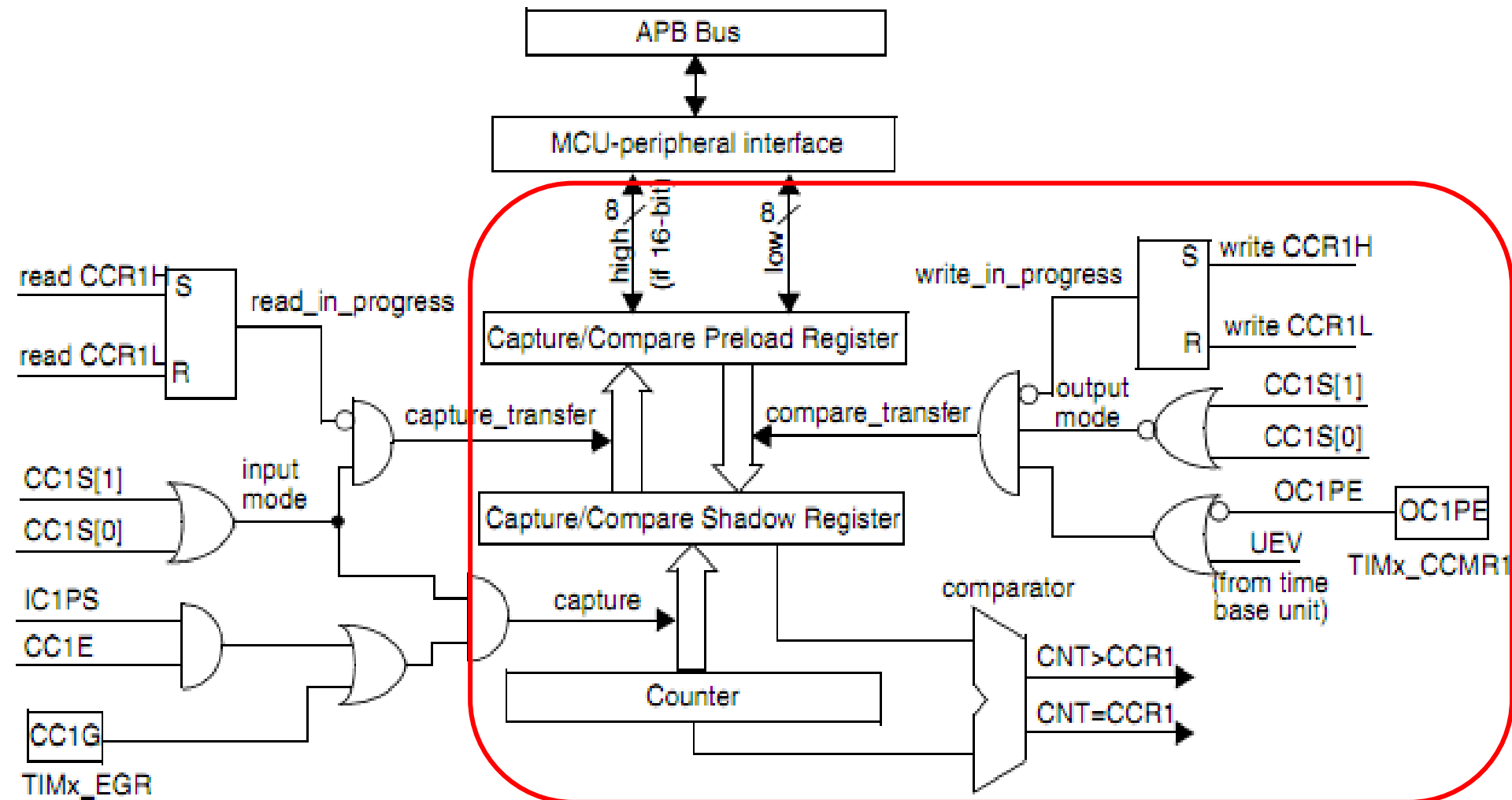
[참고] 범용 타이머(TIM2~5)는 4개의 채널을 가지므로 캡처/비교기도 4개, 따라서 OC_x (x= 1~4)는 OC1 ~OC4

단, TIM9,12: CH1,2 only, OC_x (x=1,2)
TIM10,11,13,14: CH1 only, OC1



➤ARR: CNT 의 maximum 값을 의미(타이머의 주기를 결정)
➤CCR: 출력신호의 변화시점을 결정

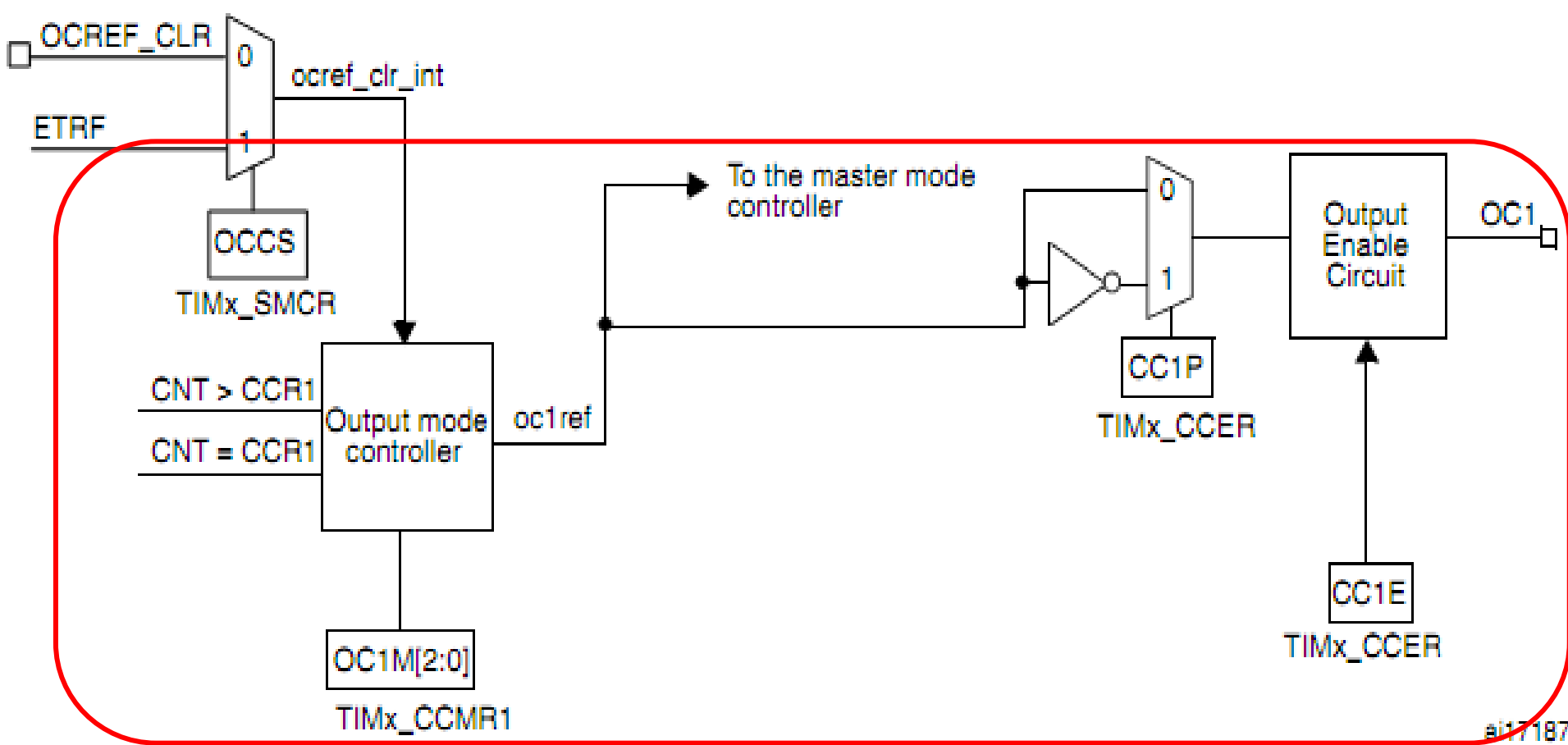
<그림4.9_3> Output Compare mode시 인터럽트 및 출력신호 발생



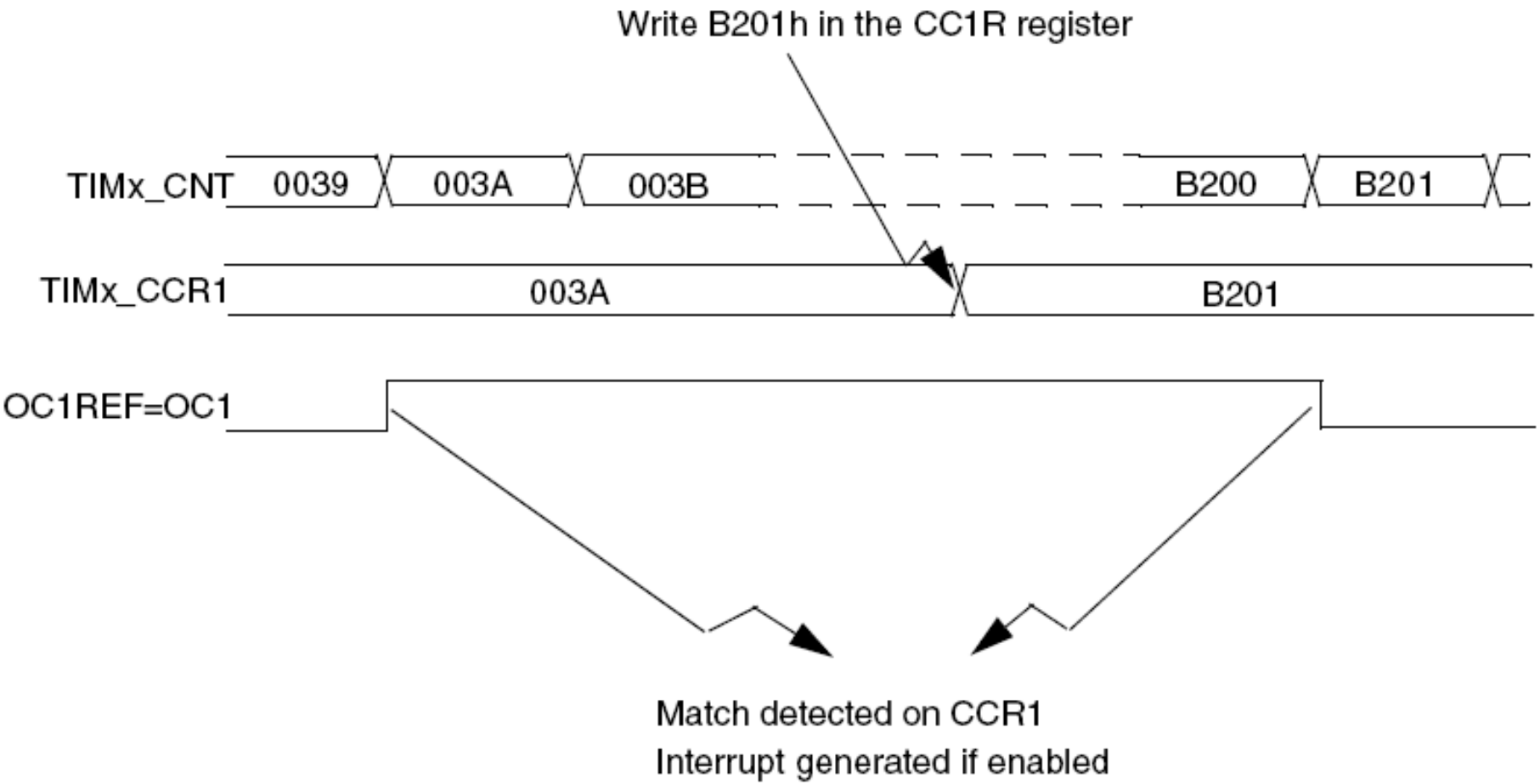
<그림 4.9_4> Capture/compare channel 1 main circuit

*** CC Shadow Register : 작업을 하는 active 레지스터**

*** CC Preload Register : write 나 read 등 access 용 레지스터**



<그림4.9_5> Output stage of capture/compare channel (channel 1)



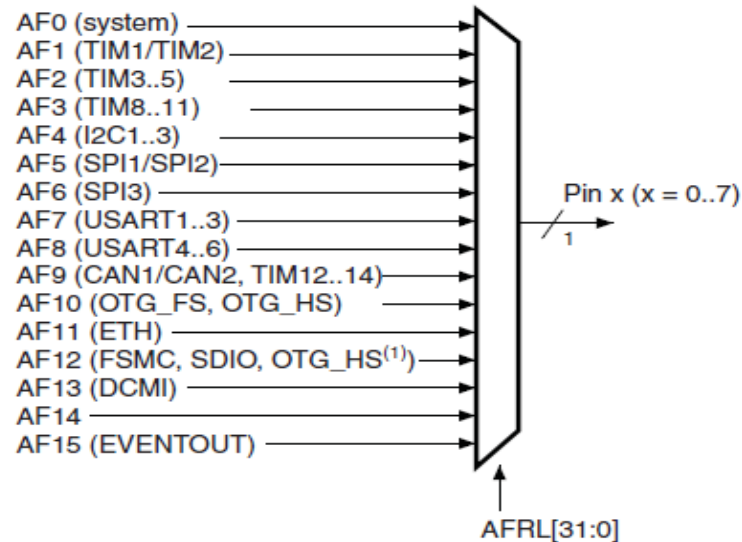
<그림 4.9_6> 출력 비교 모드의 동작 시간도 (Output compare toggle로 설정한 경우) : 비교값이 $TIMx \rightarrow CCR1 = 003A$ 으로 설정된 경우, 카운터 값이 $TIMx \rightarrow CNT = 003A$ 가 되는 순간에 출력값 OC1은 반전, 이후에 비교값을 $TIMx \rightarrow CCR1 = B201$ 로 바꾸면 카운터 값이 $TIMx \rightarrow CNT = B201$ 이 되는 순간 출력값 OC1이 다시 반전

- 출력 비교(OC : Output Compare) 설정 과정 (CCR1, OC1 사용)
- ① Select the counter clock (internal, external, prescaler).
 - ② **GPIO 핀 활성화**(Clock Enable, Mode, AFR지정)
 - ③ Write the desired data : **TIMx→ARR and TIMx→CCR1**
 - ④ **Set the CC1IE** bits if an interrupt request is to be generated.
 - ⑤ Select the output mode. For example, you must write
 - **GPIOz→AFR[]** : Define the Alternate function pin
 - **CC1S=00**, (OC1/CH1 **Output**: TIMx→CCMR1)
 - **OC1M=011**, (Output Compare 1 MODE **Toggle**: TIMx→CCMR1)
 - **OC1PE=0**, (Output Compare 1 Preload **Disable**: TIMx→CCMR1)
 - **CC1P=0**, (CC1 output Polarity **HIGH**: TIMx→CCER)
 - **CC1E=1** (CC1 output Enable **OC1 output**: TIMx→CCER)**to toggle OC1 output pin when CNT matches CCR1,**
CCR1 preload is not used, OC1 is enabled and active high.
 - ⑥ Enable the counter by **CEN=1 in the TIMx→CR1** register.

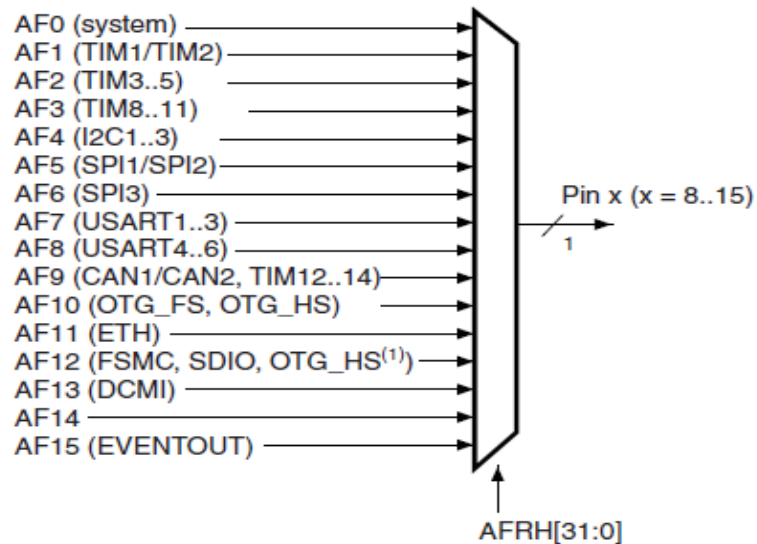
• Selecting an Alternate function

Selecting an alternate function on STM32F405xx/07xx and STM32F415xx/17xx

For pins 0 to 7, the GPIOx_AFRL[31:0] register selects the dedicated alternate function



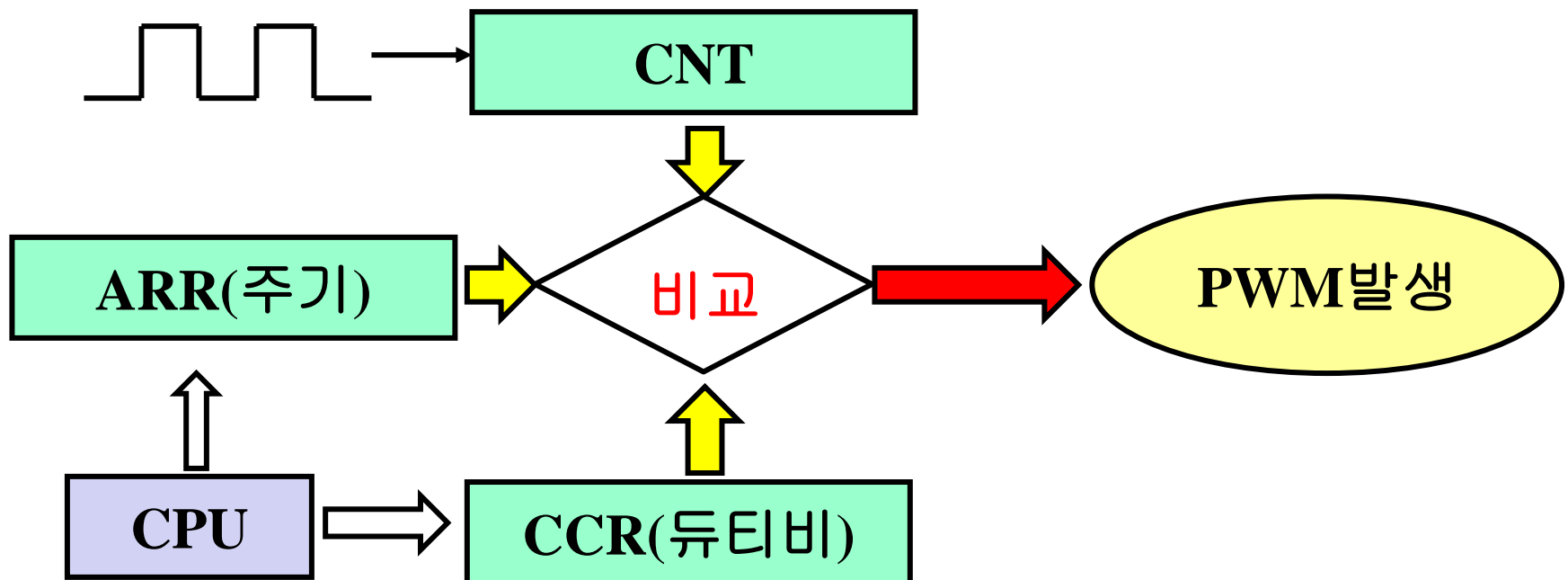
For pins 8 to 15, the GPIOx_AFRH[31:0] register selects the dedicated alternate function



4.2.5 PWM(Pulse Width Modulation) 출력 모드

- 정의: PWM 신호를 발생하여 외부에 출력하는 모드
- 출력의 주파수는 ARR에 의해 결정되며 듀티비(duty ratio)는 CCR에 의해 결정
- 세부 동작모드
 - * 세부 동작 모드는 타이머의 각 채널별로 별도의 설정이 가능

Input pulse(clock)



(1) PWM 출력 발생 모드에 따른 구분 : PWM mode 1과 PWM mode 2의 두 가지

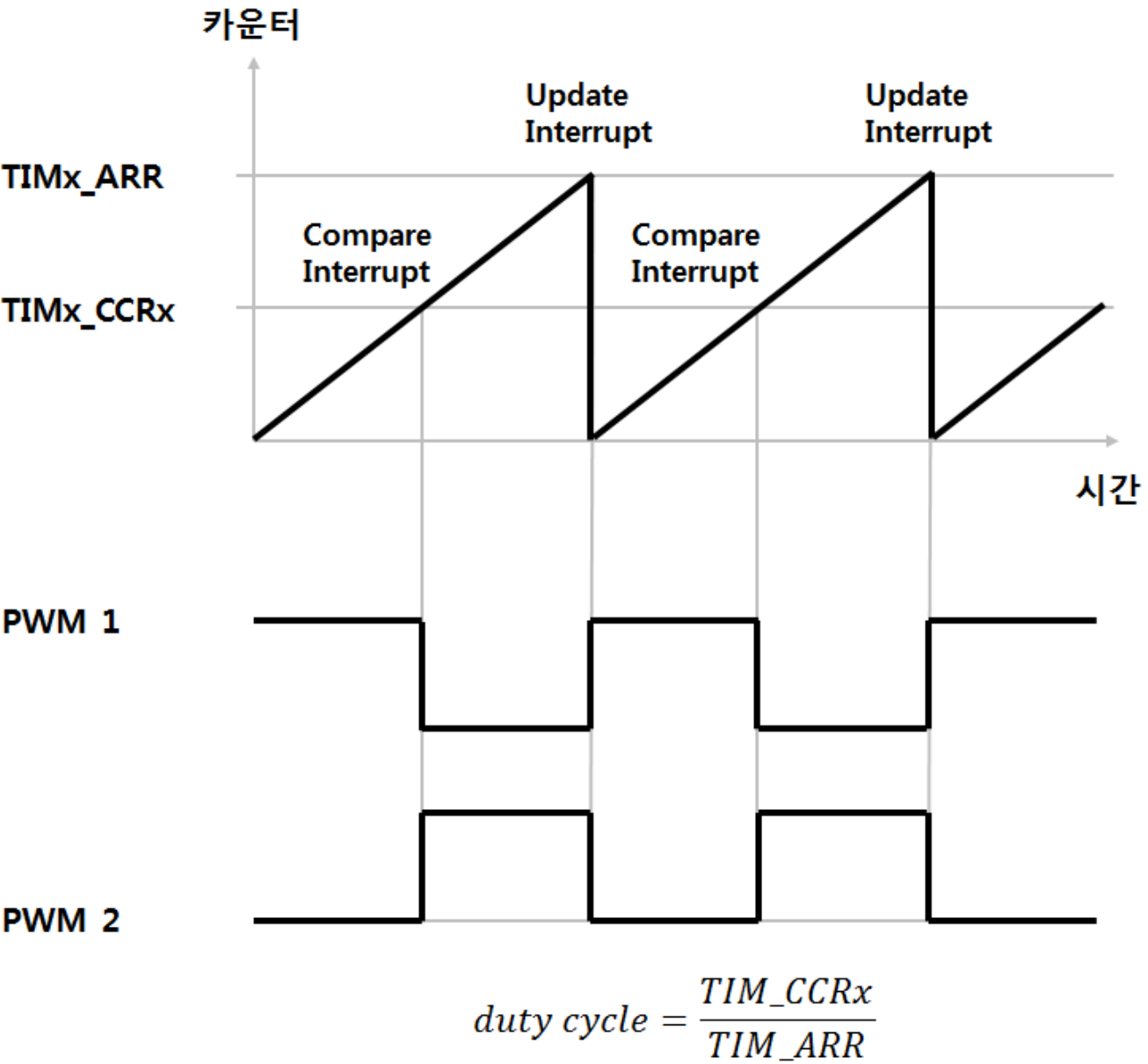
① **PWM mode 1**

- up-counting 일 때 : $CNT < CCRx$ 이면 : active(High) 출력
 $CNT \geq CCRx$ 이면 : inactive(Low) 출력
- down-counting 일 때 : $CNT \leq CCRx$ 이면 : active(High) 출력
 $CNT > CCRx$ 이면 : inactive(Low) 출력

② **PWM mode 2** : PWM mode 1과 출력이 반대

- up-counting 일 때 : $CNT < CCRx$: inactive 출력
 $CNT \geq CCRx$: active 출력
- down-counting 일 때 : $CNT \leq CCRx$: inactive 출력
 $CNT > CCRx$: active 출력

* Up counter는 0 \rightarrow (ARR-1) 까지 counting
Down counter는 ARR \rightarrow 1 까지 counting



<그림 4.10> PWM 모드의 동작

(2) 카운터의 동작 모드에 따른 구분 : Edge-aligned 모드와 Center-aligned 모드

[참고] Edge-aligned 모드는 Up 또는 Down 카운터일 때 사용이 가능하며, Center-aligned 모드는 Up/Down 카운터일 때 사용가능

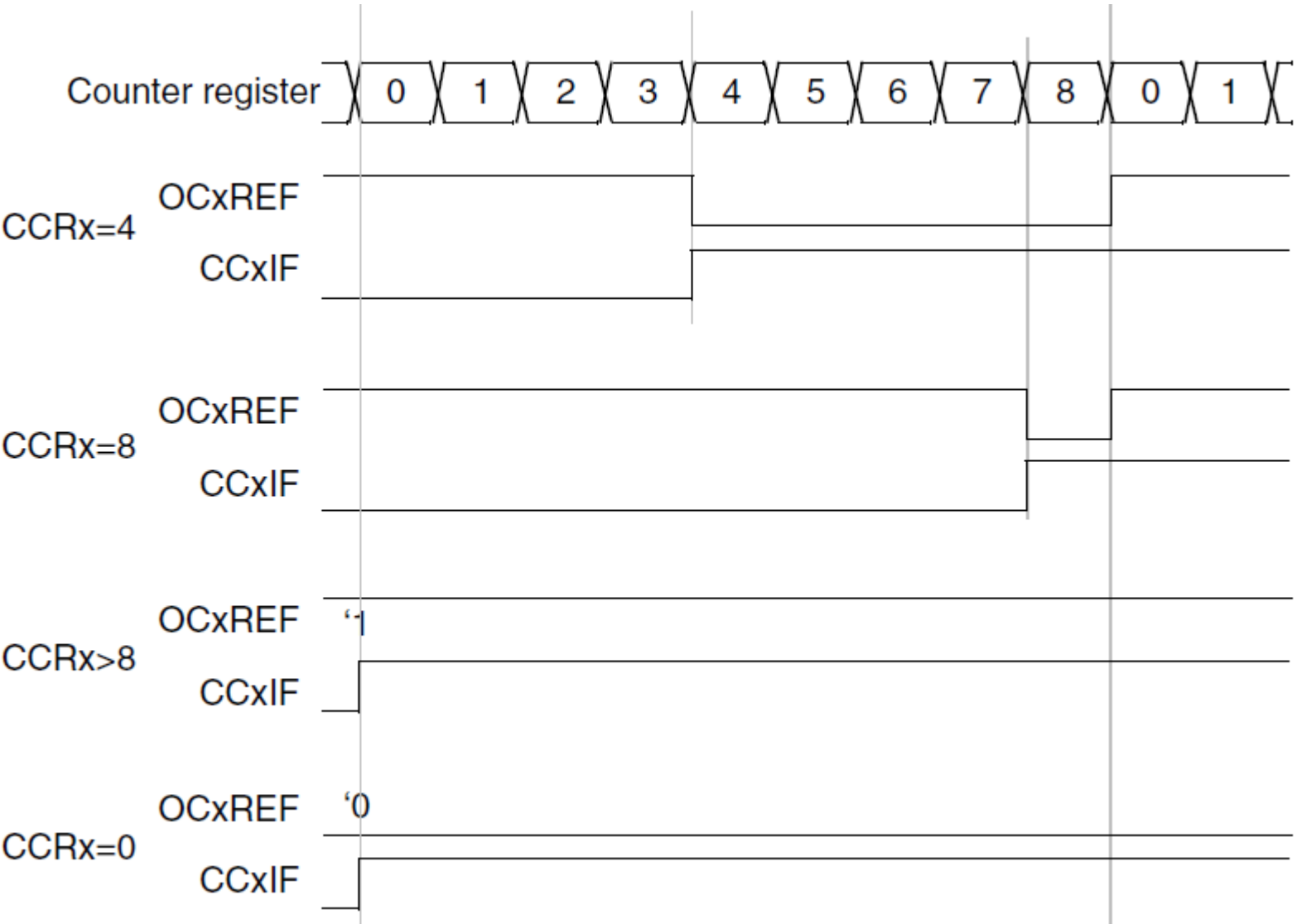
① **Edge-aligned PWM(UP) 모드** : 카운터가 업 카운팅으로 설정된 경우 (PWM 1 mode인 경우)

- $CNT < CCR_x$ 이면, $OC_xREF = 1$
- $CNT \geq CCR_x$ 이면, $OC_xREF = 0$

(예) <그림4.11>: ARR=8일 경우 PWM 신호 출력

[참고] <그림4.2>범용타이머의 상세구조도를 보면,

- Capture/Compare register의 출력이 OC_xREF ($x=1, 2, 3, 4$)이고 이것이 최종적으로 타이머의 출력 OC_x 및 $TIMy_CHx$ 가 됨.
- 따라서 OC_xREF 의 파형이 외부로 출력되는 PWM 파형이 됨



<그림 4.11> Edge-aligned PWM의 동작 시간도 (업 카운팅, ARR=8, PWM Mode 1의 경우)

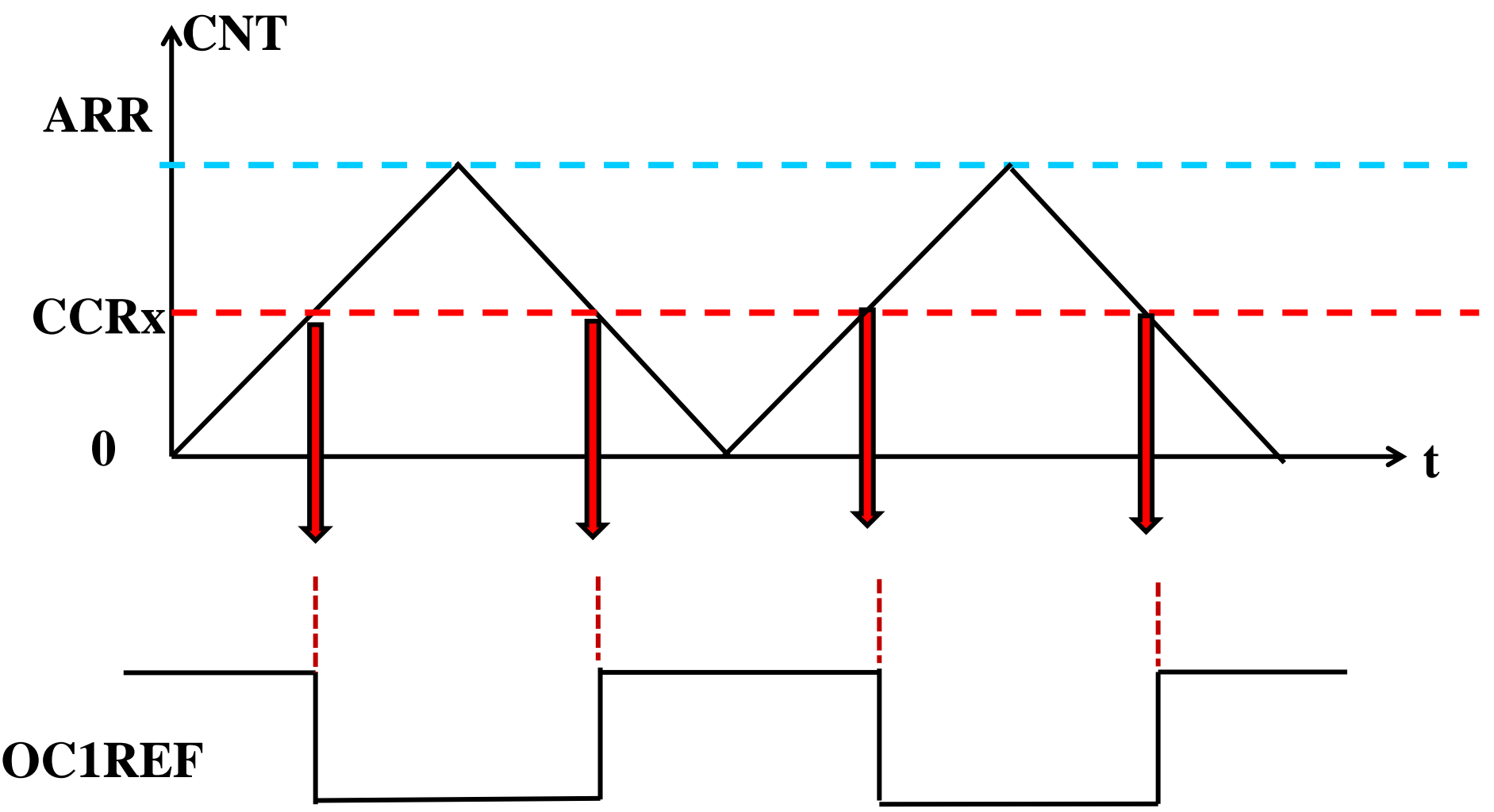
② **Edge-aligned PWM(DOWN) 모드** : 카운터가 다운 카운팅으로 설정된 경우 (PWM 1 mode인 경우)

- $CNT > CCRx$ 이면, $OCxREF = 0$
- $CNT \leq CCRx$ 이면, $OCxREF = 1$

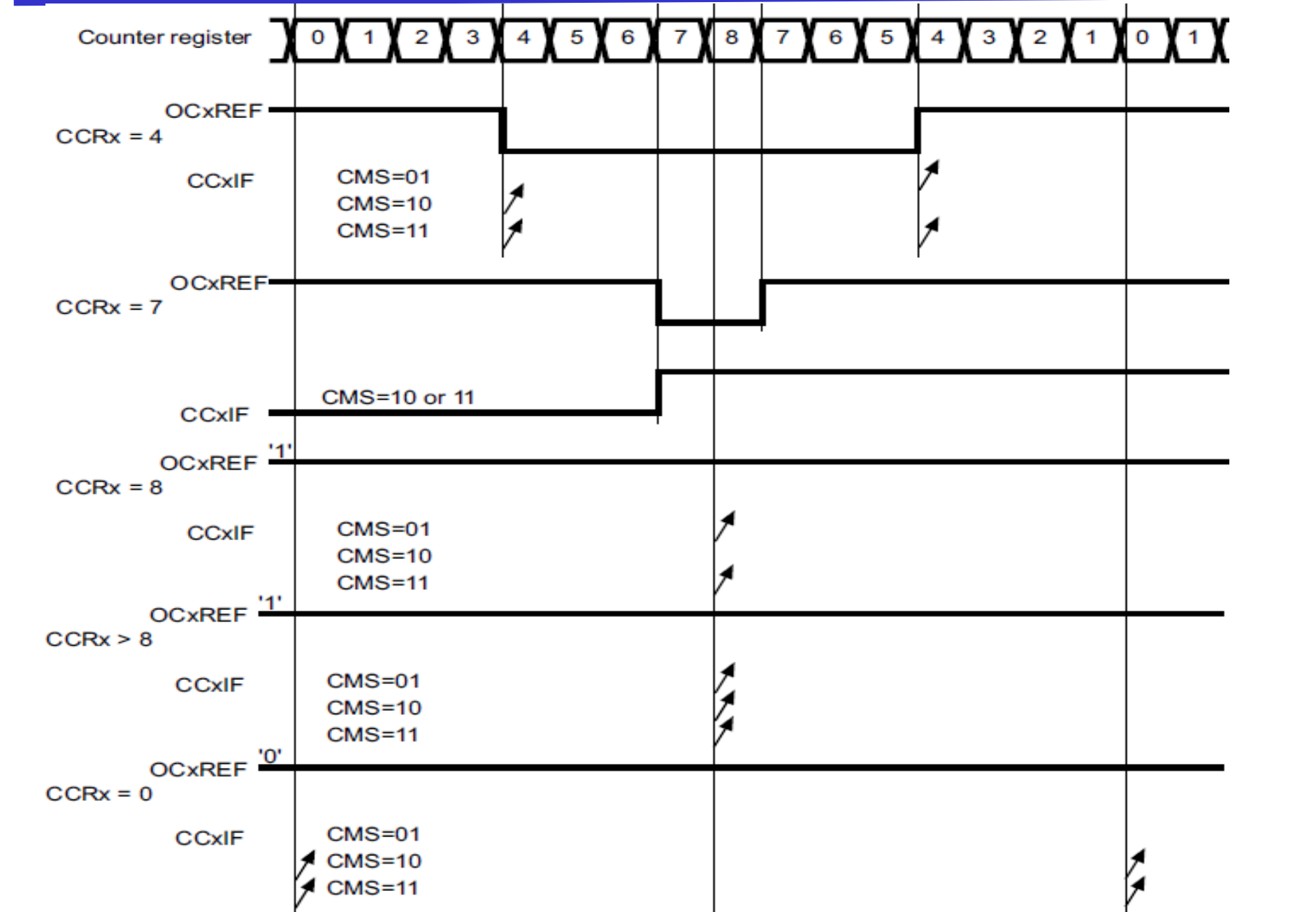
③ **Center-aligned PWM 모드** : 카운터가 업/다운 카운팅으로 설정된 경우 <그림 4.12_1>

- 업 카운터일때: Edge-aligned(UP) 모드와 동일한 동작
 - 다운 카운터일때: Edge-aligned(DOWN) 모드와 동일한 동작
- (예) <그림4.12_2> $ARR=80$ 이고, PWM mode = PWM Mode1 인 경우의 PWM 신호 출력의 예 ($CCRx = 4$ 인 경우)

- 업 카운터일때 : $CNT=4$ 일 경우(즉, 카운터가 3에서 4로 될 때)
 $OCxREF = 0$
- 다운 카운터일때 : $CNT = 4$ 일 경우(즉, 카운터가 5에서 4로 될 때)
 $OCxREF = 1$



<그림4.12_1> Output Compare mode시 인터럽트 발생 및 출력신호 (PWM 1 mode)



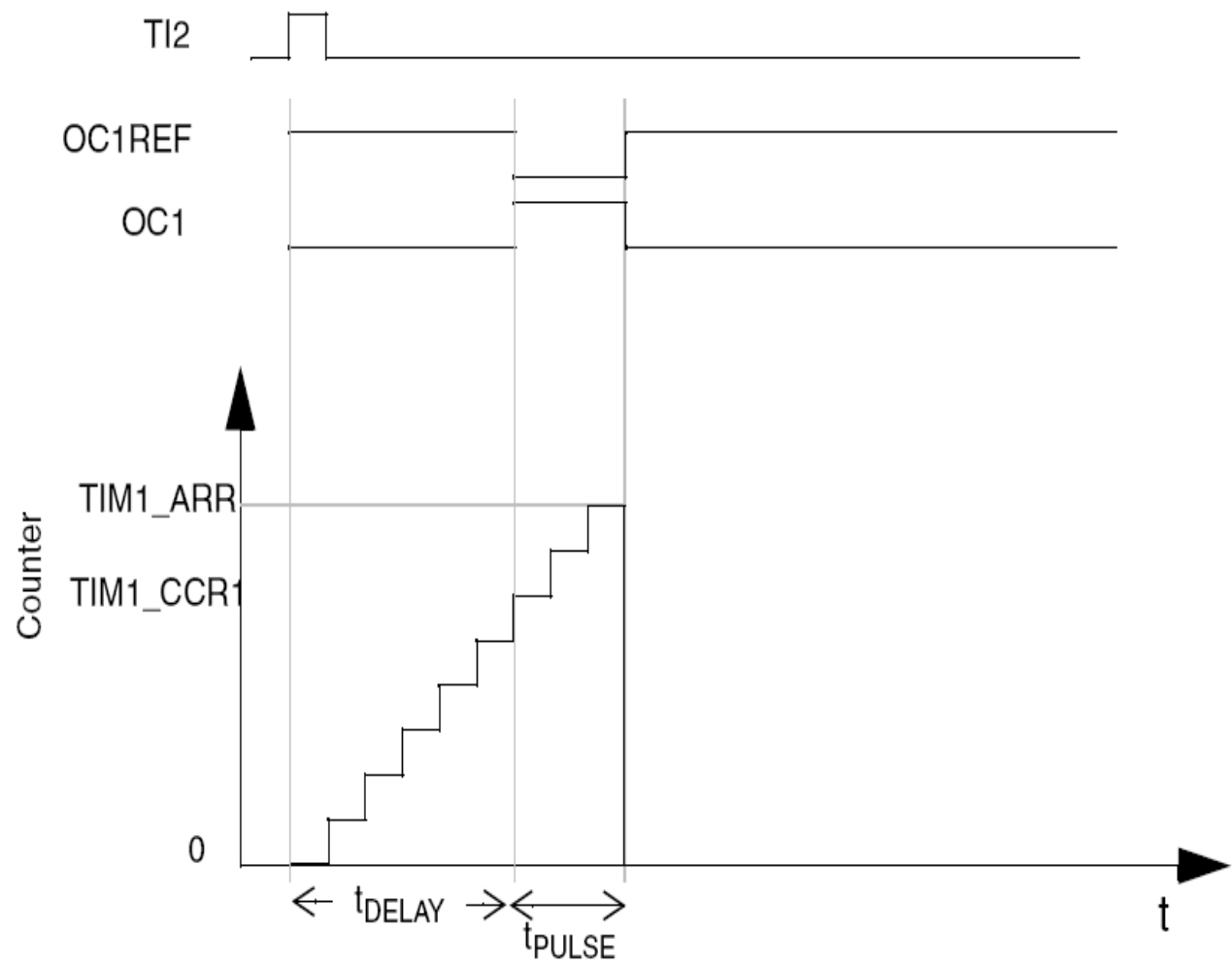
<그림 4.12_2>Center-aligned PWM 동작 시간도(ARR=8, PWM Mode 1)

- PWM mode 설정 과정 (CCR3, OC3 사용한 경우)

- ① Select the counter clock (internal, external, prescaler).
- ② **GPIO 핀 활성화**(Clock Enable, Mode, AFR지정)
- ③ Write the desired data : **TIMx→ARR(period) and TIMx→ CCR3 (Duty Ratio)**
- ④ Select the PWM 1 mode mode. For example, you must write
 - CC3S=00**, (OC3/CH3 **Output**: TIMx→CCMR2)
 - OC3M=110**, (OC3/CH3 **PWM 1 MODE**: TIMx→ CCMR2)
 - OC3PE=0**, (Output Compare 3 Preload **Disable**: TIMx→ CCMR2)
 - CC3P=0**, (CC3 output Polarity **HIGH**: TIMx→ CCER)
 - CC3E=1** (CC3 output Enable **OC3 output**: TIMx→ CCER)**to generate PWM signal on OC3(TIMx CH3) output pin**,
CCR3 preload is not used, OC3 is enabled and active high.
- ⑤ Enable the counter by **CEN=1 in the TIMx→CR1** register.

4.2.6 원 펄스(One pulse) 모드

- 정의: 외부 입력이 인가될 경우 이로부터 일정 시간이 지난 후에 펄스를 1번 발생시키는 모드
- 펄스가 발생하는 시간과 지속하는 시간은 프로그래밍 가능
- (예): <그림 4.13>
 - TI2 입력 핀에서 상승 에지가 검출되면 t_{Delay} 시간 후에 t_{PULSE} 폭의 펄스가 OC1 출력 핀에 발생
 - t_{Delay} 값과 t_{PULSE} 값은 ARR과 CCR 값의 설정에 따라 결정
 - 이 모드가 제대로 동작하기 위해서는 동작전에
 - 업 카운팅 모드의 경우, $\text{CNT} < \text{CCR}_x < \text{ARR}$,
 - 다운 카운팅 모드의 경우, $\text{CNT} > \text{CCR}_x$ 로 설정되어 있어야 함



<그림 4.13> One-pulse 모드의 동작 시간도

4.2.7 그 외의 동작 모드

- 범용 타이머는 이외에도 다음과 같은 기능들을 가지고 있음 (자세한 설명은 생략)
 - 입력 캡처(Input Capture) 모드
 - PWM 입력 모드
 - 강제 출력 (Forced output) 모드
 - 엔코더 인터페이스 모드
 - 타이머 동기화(Timer synchronization)

4.3 Timer(CNT)에 공급될 Clock의 선택

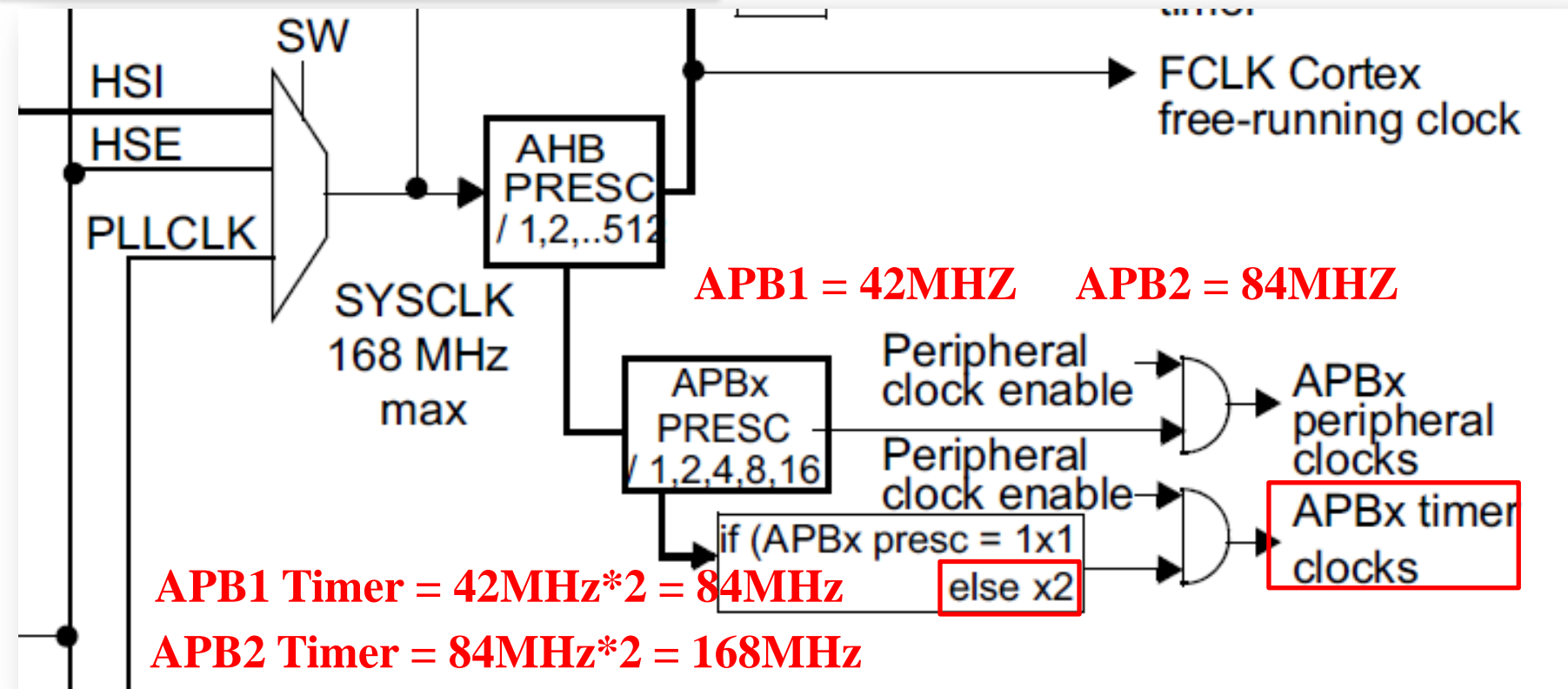
- Counter(CNT) clock의 clock sources:
 - Internal clock (CK_INT)
 - External clock mode1: external input pin (TlX)
 - External clock mode2: external trigger input (ETR) available on TIM2, TIM3 and TIM4 only.

① Internal clock source (CK_INT)

- SMS=000 (TIMx_SMCR register)
- Clock Init/Enable 관련 신호: **CEN, DIR** (TIMx_CR1 register), UG bits (TIMx_EGR register)
- CEN(Counter Enable) bit = '1'되면, prescaler 가 internal clock CK_INT에 의해 작동 (그림 4.14)

CK_INT(Internal Clock) 1

APB_x (x = 1,2)

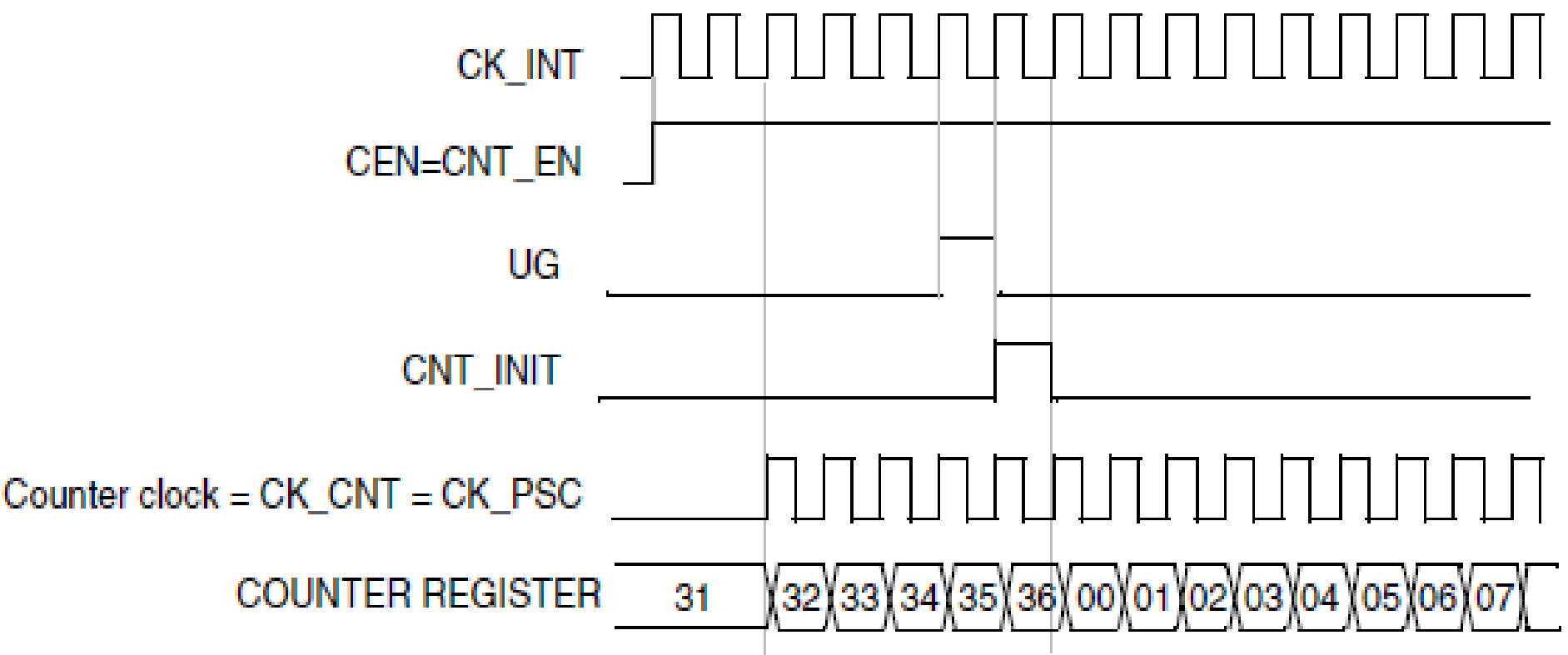


The timer clock frequencies for STM32F407 are automatically set by H/W.

Case 1: If the APB prescaler is 1, 타이머 클럭은 APB_x 주파수와 동일

Case 2: Otherwise, **they are set to twice (×2), 즉 APB_x 주파수의 2배**

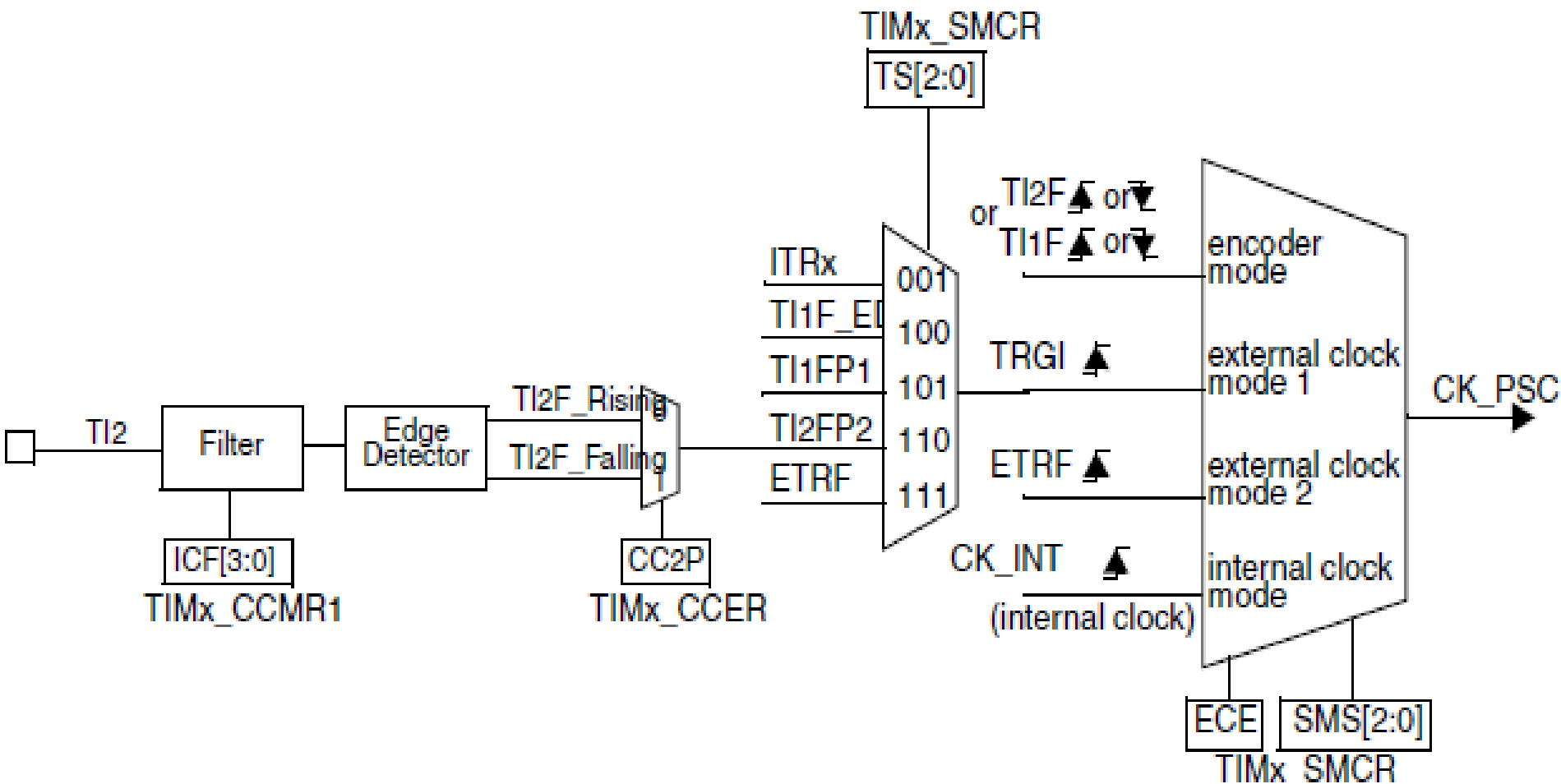
- AHB domain 최대 주파수: **168 MHz.**
- High-speed APB2 domain 최대 주파수 : **84 MHz**
- Low-speed APB1 domain 최대 주파수 : **42 MHz**



<그림4.14> Counter clock source로 Internal Clock(CK_INT)를 사용한 동작 시간도 (분주비= 1 인 경우)

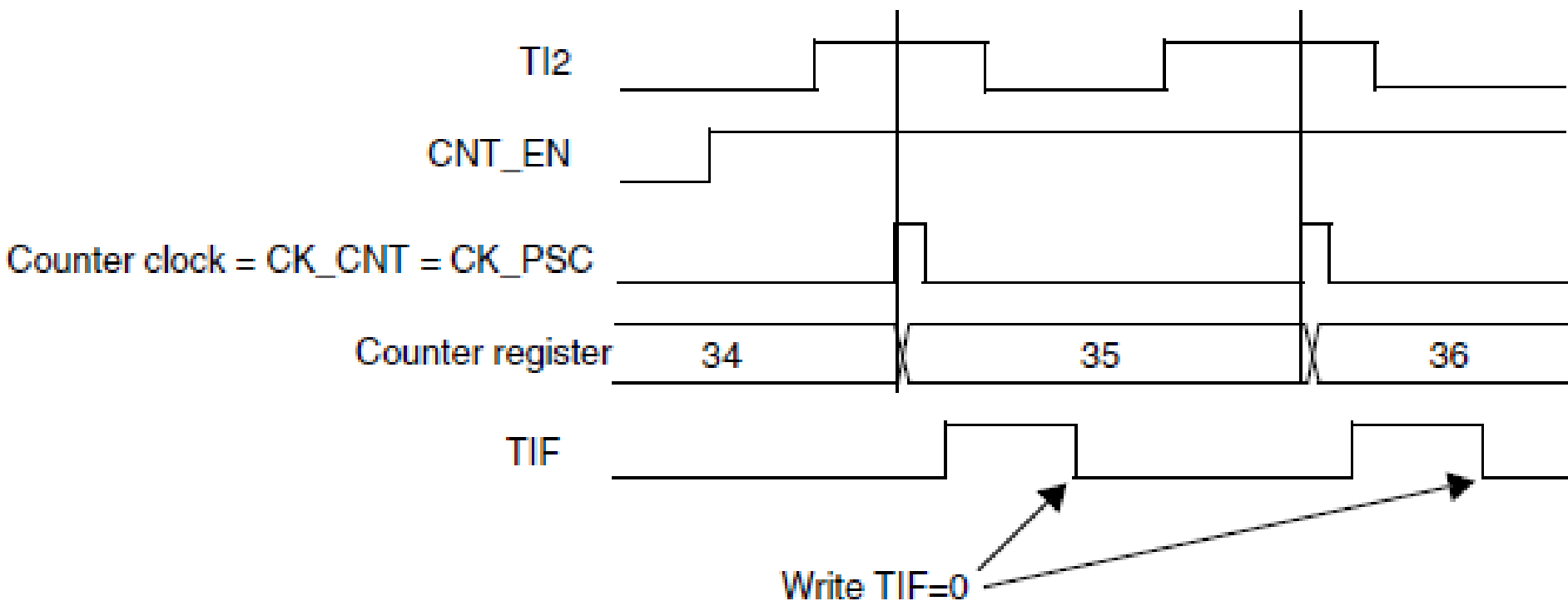
② External clock mode1: external input pin (TIx)

- SMS=111(TIMx→SMCR register)
- 외부클럭입력 허용과정(다음 그림은 TI2를 예로 설명)



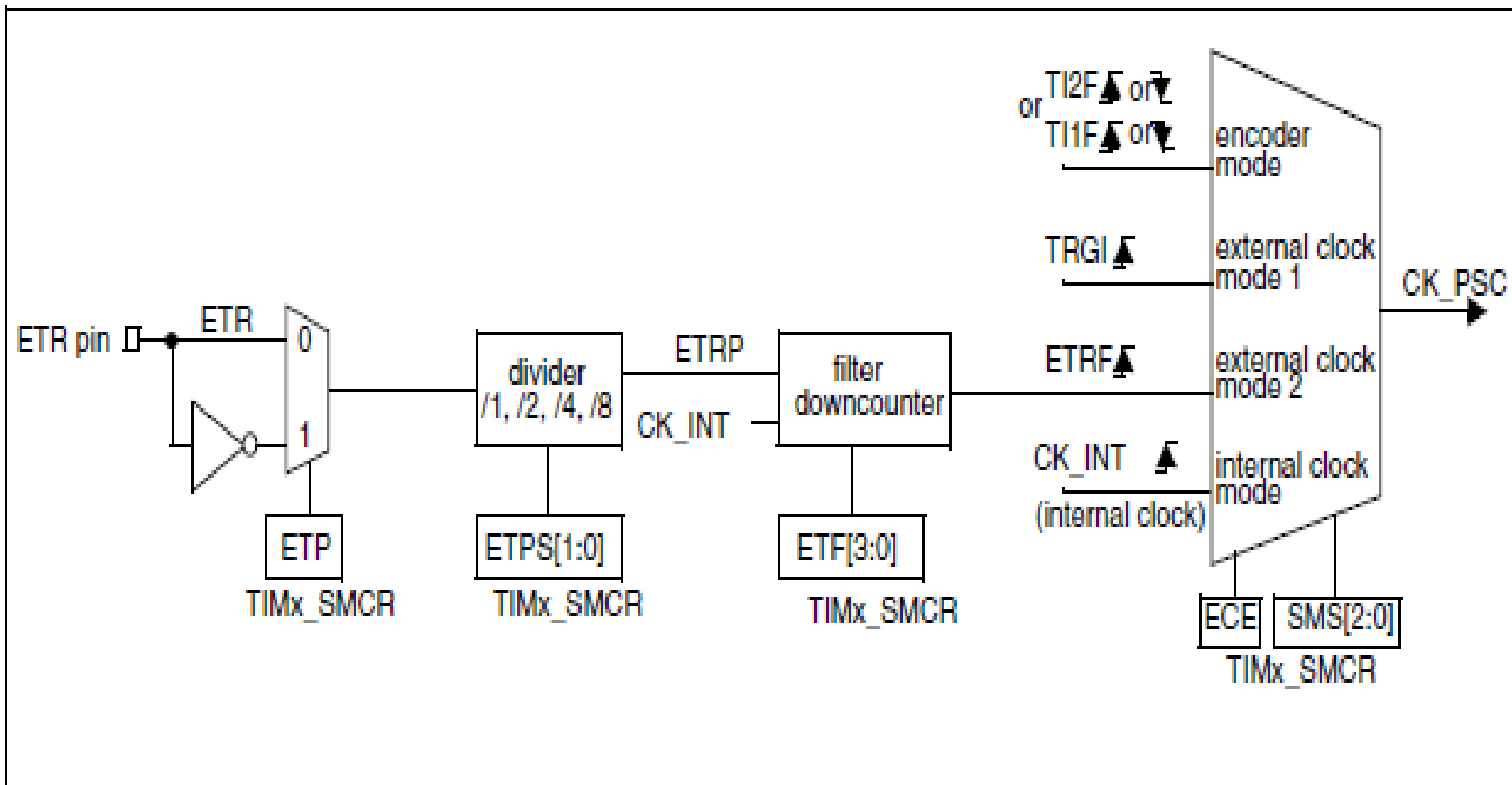
<그림4.15> TI2 external clock connection example

- ③ Configure channel 2 to detect rising edges on the TI2 input : CC2S= 01(TIM_x→CCMR1 register)
- ④ Input filter duration : IC2F[3:0] (TIM_x→CCMR1) (no filter : IC2F=0000)
- ⑤ Select rising edge polarity : CC2P=0 , CC2NP=0 (TIM_x→CCER)
- ⑥ Configure external clock mode 1 : SMS=111(TIM_x→SMCR)
- ⑦ Select TI2 as the input source : TS=110(TIM_x→SMCR)
- ⑧ Enable the counter : CEN=1(TIM_x→CR1)



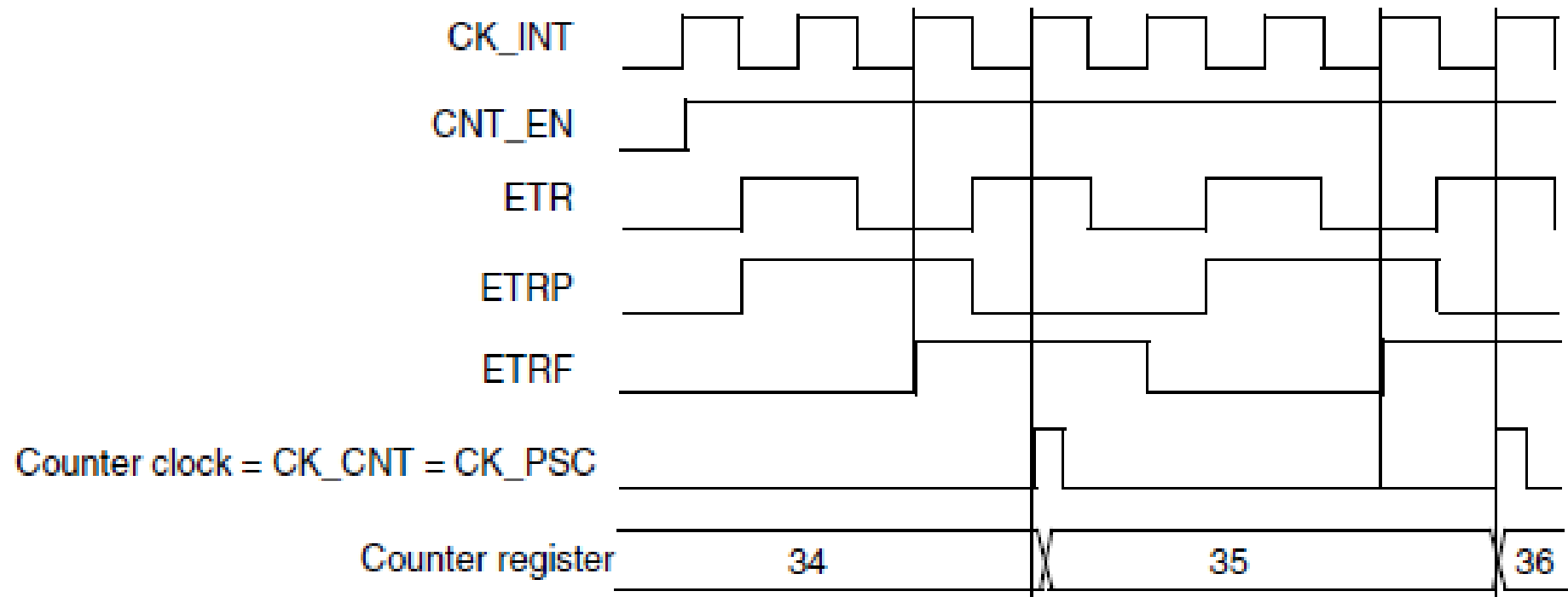
③ External clock mode2: external trigger input (ETR)

- ECE=1 (TIMx→SMCR register)
- ETR의 rising or falling edge 에서 Counter 동작.



<그림4.16> External trigger input block

- Ⓐ No filter : ETF[3:0]=0000 (TIM_x→SMCR)
 - Ⓑ Set the prescaler : ETPS[1:0]=01(TIM_x→SMCR)
 - Ⓒ Select rising edge detection on the ETR pin :ETP=0 (TIM_x→SMCR)
 - Ⓓ Enable external clock mode 2 : ECE=1(TIM_x→SMCR)
 - Ⓔ Enable the counter : CEN=1(TIM_x→CR1)
- The counter counts once each 2 ETR rising edges.



5.0 Register Map

[illegible]

[illegible]

| | | | | | | | | | | | | | | | | | | | | | | | |
|------|-------------|---|----------------------------------|--|--|--|--|--|--|--|--|--------------------------|--|----------|----------|----------|--|---------|--|----------|--|--|--|
| 0x38 | TIMx_CCR2 | CCR2[31:16] (TIM2 and TIM5 only, reserved on the other timers) | CCR2[15:0] | | | | | | | | | | | | | | | | | | | | |
| | Reset value | 00000000000000000000000000000000 | 00000000000000000000000000000000 | | | | | | | | | | | | | | | | | | | | |
| 0x3C | TIMx_CCR3 | CCR3[31:16] (TIM2 and TIM5 only, reserved on the other timers) | CCR3[15:0] | | | | | | | | | | | | | | | | | | | | |
| | Reset value | 00000000000000000000000000000000 | 00000000000000000000000000000000 | | | | | | | | | | | | | | | | | | | | |
| 0x40 | TIMx_CCR4 | CCR4[31:16] (TIM2 and TIM5 only, reserved on the other timers) | CCR4[15:0] | | | | | | | | | | | | | | | | | | | | |
| | Reset value | 00000000000000000000000000000000 | 00000000000000000000000000000000 | | | | | | | | | | | | | | | | | | | | |
| 0x44 | Reserved | | | | | | | | | | | | | | | | | | | | | | |
| 0x48 | TIMx_DCR | Reserved | | | | | | | | | | DBL[4:0] | | | Reserved | DBA[4:0] | | | | | | | |
| | Reset value | | | | | | | | | | | 00000 | | | | 00000 | | | | | | | |
| 0x4C | TIMx_DMAR | Reserved | | | | | | | | | | DMAB[15:0] | | | | | | | | | | | |
| | Reset value | | | | | | | | | | | 000000000000000000000000 | | | | | | | | | | | |
| 0x50 | TIM2_OR | Reserved | | | | | | | | | | Reserved | | ITR1_RMP | | Reserved | | | | | | | |
| | Reset value | | | | | | | | | | | | | 00 | | | | | | | | | |
| 0x50 | TIM5_OR | Reserved | | | | | | | | | | Reserved | | | | | | IT4_RMP | | Reserved | | | |
| | Reset value | | | | | | | | | | | | | | | | | 00 | | | | | |

5.1 TIMx control register 1 (TIMx_CR1)

Address offset: 0x00 *타이머의 주요기능을 설정하는 역할

Reset value: 0x0000

| | | | | | | | | | | | | | | | |
|----------|----|----|----|----|----|----------|-----|------|-----|-----|-----|-----|-----|------|-----|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | CKD[1:0] | | ARPE | CMS | | DIR | OPM | URS | UDIS | CEN |
| | | | | | | r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w |

5.2 TIMx DMA/Interrupt enable resister (TIMx_DIER)

Address offset: 0x0C *인터럽트 Enable (MASK Register)하는 역할

Reset value: 0x0000

| | | | | | | | | | | | | | | | |
|------|-----|-----|-------|-------|-------|-------|-----|------|-----|-----|-------|-------|-------|-------|-----|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Res. | TDE | Res | CC4DE | CC3DE | CC2DE | CC1DE | UDE | Res. | TIE | Res | CC4IE | CC3IE | CC2IE | CC1IE | UIE |
| | r/w | | r/w | r/w | r/w | r/w | r/w | | r/w | | r/w | r/w | r/w | r/w | r/w |

5.7 TIMx event generation register (TIMx_EGR)

Address offset: 0x14 *소프트웨어에 의해 Event 발생케 하는 역할
Reset value: 0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|----|----|----|----|----|---|---|---|----|------|------|------|------|------|----|
| Reserved | | | | | | | | | TG | Res. | CC4G | CC3G | CC2G | CC1G | UG |
| | | | | | | | | | W | | W | W | W | W | W |

5.8 TIMx slave mode control register (TIMx_SMCR)

Address offset: 0x08 *외부신호에 의해 타이머를 동작케 하는 역할
Reset value: 0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|-----|-----------|----|----------|----|----|----|-----|---------|----|----|------|----------|----|----|
| ETP | ECE | ETPS[1:0] | | ETF[3:0] | | | | MSM | TS[2:0] | | | Res. | SMS[2:0] | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | rw | rw | rw |

5.9 TIMx capture/compare mode register 1 (TIMx_CCMR1)

Address offset: 0x18 *각 채널마다의 입출력기능을 결정하는 역할
Reset value: 0x0000 (Input or Output)

[illegible]

5.10 TIMx capture/compare mode register 2 (TIMx_CCMR2)

Address offset: 0x1C

Reset value: 0x0000

[illegible]

5.11 TIMx capture/compare enable register (TIMx_CCER)

Address offset: 0x20 *각 채널의 입출력 핀의 기능을 **Enable**하는 역할

Reset value: 0x0000

| | | | | | | | | | | | | | | | |
|-------|------|------|------|-------|------|------|------|-------|------|------|------|-------|------|------|------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CC4NP | Res. | CC4P | CC4E | CC3NP | Res. | CC3P | CC3E | CC2NP | Res. | CC2P | CC2E | CC1NP | Res. | CC1P | CC1E |
| rw | | rw | rw | rw | | rw | rw | rw | | rw | rw | rw | | rw | rw |

6. STM32F407의 TIMER 프로그래밍 실습

6.1 프로그램에서의 TIMx set-up 과정 및 레지스터 설정

6.1.1 Counter mode

RCC 설정

- **RCC→CR,CFGR,PLLCFGR (Clock소스/주파수설정)**
- **RCC→APB1ENR(TIMx Clock Enable)**

INT Enable

- **NVIC→ISER[]**

TIMx 초기 설정

- **TIMx→PSC, TIMx→ARR**
- **TIMx→CR1(counter EN, Mode, Dir)**
- **TIMx→DIER(Update INT enable)**

Int Handler 설정

- **TIMx_IRQHandler()**

6.1.2 Output Compare mode

RCC 설정

- **RCC**→**CR,CFGR,PLLCFGR** (Clock소스/주파수 설정)
- **RCC**→**APB1ENR**(**TIMx** Clock Enable)

AF Pin 설정

- **GPIOy**→**AFR[]** (**GPIO CK_EN, Mode**)

INT Enable

- **NVIC**→**ISER[]** : 인터럽트 사용할 때

TIMx 초기 설정

- **TIMx**→**PSC**, **TIMx**→**ARR**(Output signal Period)
- **TIMx**→**CR1**(counter EN, Dir etc)
- **TIMx**→**CCER**(Output Compare mode etc)
- **TIMx**→**CCR1..4**(Compare value)
- **TIMx**→**CCMR1/CCMR2**(Output toggle etc)

Int Handler 설정

- **TIMx_IRQHandler()** : 인터럽트 사용할 때

6.1.3 PWM mode

RCC 설정

- **RCC→CR,CFGR,PLLCFGR** (Clock소스/주파수 설정)
- **RCC→APB1ENR**(TIMx Clock Enable)

AF Pin 설정

- **GPIOy→AFR[]** (GPIO CK_EN, Mode)

TIMx 초기 설정

- **TIMx→PSC, TIMx→ARR**(PWM Period)
- **TIMx→CR1**(counter EN, Mode)
- **TIMx→CCER**(output pin En etc)
- **TIMx→CCR1..4**(Duty Ratio)
- **TIMx→CCMR1/CCMR2**(PWM mode etc)

6.2 STM32F407의 TIMx의 Address(Memory map)

| Bus | Boundary address | Peripheral |
|------|---------------------------|--------------------|
| APB2 | 0x4001 4C00 - 0x4001 57FF | Reserved |
| | 0x4001 4800 - 0x4001 4BFF | TIM11 |
| | 0x4001 4400 - 0x4001 47FF | TIM10 |
| | 0x4001 4000 - 0x4001 43FF | TIM9 |
| | 0x4001 3C00 - 0x4001 3FFF | EXTI |
| | 0x4001 3800 - 0x4001 3BFF | SYSCFG |
| | 0x4001 3400 - 0x4001 37FF | Reserved |
| | 0x4001 3000 - 0x4001 33FF | SPI1 |
| | 0x4001 2C00 - 0x4001 2FFF | SDIO |
| | 0x4001 2400 - 0x4001 2BFF | Reserved |
| | 0x4001 2000 - 0x4001 23FF | ADC1 - ADC2 - ADC3 |
| | 0x4001 1800 - 0x4001 1FFF | Reserved |
| | 0x4001 1400 - 0x4001 17FF | USART6 |
| | 0x4001 1000 - 0x4001 13FF | USART1 |
| | 0x4001 0800 - 0x4001 0FFF | Reserved |
| | 0x4001 0400 - 0x4001 07FF | TIM8 |
| | 0x4001 0000 - 0x4001 03FF | TIM1 |
| | 0x4000 7800 - 0x4000 FFFF | Reserved |

| Bus | Boundary address | Peripheral |
|------|---------------------------|---------------------|
| APB1 | 0x4000 7800 - 0x4000 7FFF | Reserved |
| | 0x4000 7400 - 0x4000 77FF | DAC |
| | 0x4000 7000 - 0x4000 73FF | PWR |
| | 0x4000 6C00 - 0x4000 6FFF | Reserved |
| | 0x4000 6800 - 0x4000 6BFF | CAN2 |
| | 0x4000 6400 - 0x4000 67FF | CAN1 |
| | 0x4000 6000 - 0x4000 63FF | Reserved |
| | 0x4000 5C00 - 0x4000 5FFF | I2C3 |
| | 0x4000 5800 - 0x4000 5BFF | I2C2 |
| | 0x4000 5400 - 0x4000 57FF | I2C1 |
| | 0x4000 5000 - 0x4000 53FF | UART5 |
| | 0x4000 4C00 - 0x4000 4FFF | UART4 |
| | 0x4000 4800 - 0x4000 4BFF | USART3 |
| | 0x4000 4400 - 0x4000 47FF | USART2 |
| | 0x4000 4000 - 0x4000 43FF | I2S3ext |
| | 0x4000 3C00 - 0x4000 3FFF | SPI3 / I2S3 |
| | 0x4000 3800 - 0x4000 3BFF | SPI2 / I2S2 |
| | 0x4000 3400 - 0x4000 37FF | I2S2ext |
| | 0x4000 3000 - 0x4000 33FF | IWDG |
| | 0x4000 2C00 - 0x4000 2FFF | WWDG |
| | 0x4000 2800 - 0x4000 2BFF | RTC & BKP Registers |
| | 0x4000 2400 - 0x4000 27FF | Reserved |
| | 0x4000 2000 - 0x4000 23FF | TIM14 |
| | 0x4000 1C00 - 0x4000 1FFF | TIM13 |
| | 0x4000 1800 - 0x4000 1BFF | TIM12 |
| | 0x4000 1400 - 0x4000 17FF | TIM7 |
| | 0x4000 1000 - 0x4000 13FF | TIM6 |
| | 0x4000 0C00 - 0x4000 0FFF | TIM5 |
| | 0x4000 0800 - 0x4000 0BFF | TIM4 |
| | 0x4000 0400 - 0x4000 07FF | TIM3 |
| | 0x4000 0000 - 0x4000 03FF | TIM2 |

6.3 STM32F407의 TIMER 관련 header file(stm32f4xx.h) 주요 부분

```
/* Peripheral memory map */
#define PERIPH_BASE ((uint32_t)0x40000000) /* Peripheral base address */
#define APB1PERIPH_BASE PERIPH_BASE
#define APB2PERIPH_BASE (PERIPH_BASE + 0x00010000)

/*!< APB1 peripherals */
#define TIM2_BASE (APB1PERIPH_BASE + 0x0000)
#define TIM3_BASE (APB1PERIPH_BASE + 0x0400)
#define TIM4_BASE (APB1PERIPH_BASE + 0x0800)
#define TIM5_BASE (APB1PERIPH_BASE + 0x0C00)
#define TIM6_BASE (APB1PERIPH_BASE + 0x1000)
#define TIM7_BASE (APB1PERIPH_BASE + 0x1400)
#define TIM12_BASE (APB1PERIPH_BASE + 0x1800)
#define TIM13_BASE (APB1PERIPH_BASE + 0x1C00)
#define TIM14_BASE (APB1PERIPH_BASE + 0x2000)

/*!< APB2 peripherals */
#define TIM1_BASE (APB2PERIPH_BASE + 0x0000)
#define TIM8_BASE (APB2PERIPH_BASE + 0x0400)
#define TIM9_BASE (APB2PERIPH_BASE + 0x4000)
#define TIM10_BASE (APB2PERIPH_BASE + 0x4400)
#define TIM11_BASE (APB2PERIPH_BASE + 0x4800)
```

```
#define TIM2      ((TIM_TypeDef *) TIM2_BASE)
#define TIM3      ((TIM_TypeDef *) TIM3_BASE)
#define TIM4      ((TIM_TypeDef *) TIM4_BASE)
#define TIM5      ((TIM_TypeDef *) TIM5_BASE)
#define TIM6      ((TIM_TypeDef *) TIM6_BASE)
#define TIM7      ((TIM_TypeDef *) TIM7_BASE)
#define TIM12     ((TIM_TypeDef *) TIM12_BASE)
#define TIM13     ((TIM_TypeDef *) TIM13_BASE)

#define TIM1      ((TIM_TypeDef *) TIM1_BASE)
#define TIM8      ((TIM_TypeDef *) TIM8_BASE)
#define TIM9      ((TIM_TypeDef *) TIM9_BASE)
#define TIM10     ((TIM_TypeDef *) TIM10_BASE)
#define TIM11     ((TIM_TypeDef *) TIM11_BASE)
```



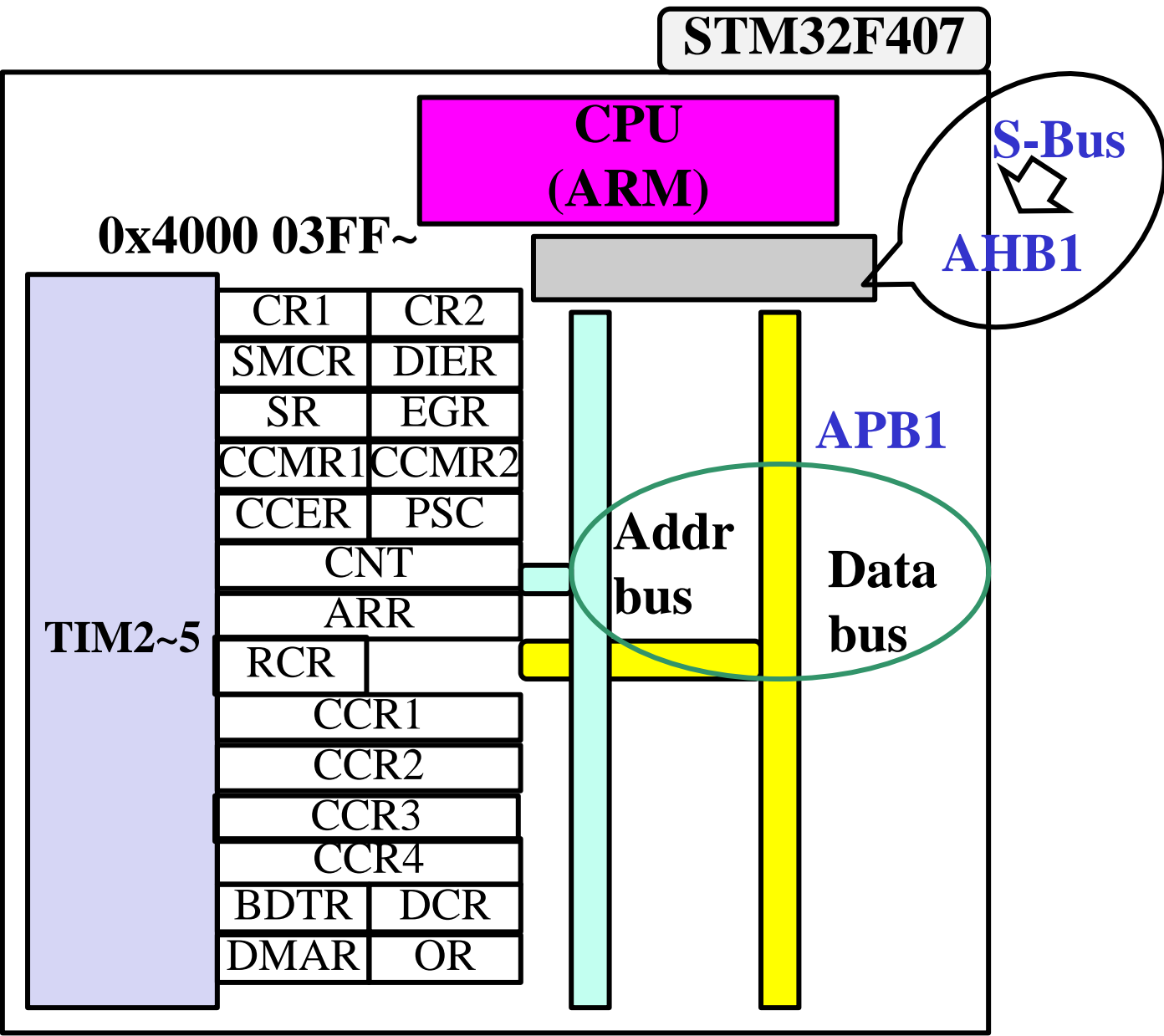
```
/* * @brief TIM */
```

```
typedef struct
```

```
{ __IO uint16_t CR1;      /*!< TIM control register 1, Address offset: 0x00 */  
  __IO uint16_t CR2;      /*!< TIM control register 2, offset: 0x04 */  
  __IO uint16_t SMCR;     /*!< TIM slave mode control register, offset: 0x08 */  
  __IO uint16_t DIER;     /*!< TIM DMA/interrupt enable register, offset: 0x0C */  
  __IO uint16_t SR;       /*!< TIM status register, Address offset: 0x10 */  
  __IO uint16_t EGR;      /*!< TIM event generation register, offset: 0x14 */  
  __IO uint16_t CCMR1;    /*!< TIM capture/compare mode register 1, offset: 0x18 */  
  __IO uint16_t CCMR2;    /*!< TIM capture/compare mode register 2, offset: 0x1C */  
  __IO uint16_t CCER;     /*!< TIM capture/compare enable register, offset: 0x20 */  
  __IO uint32_t CNT;      /*!< TIM counter register, offset: 0x24 */  
  __IO uint16_t PSC;      /*!< TIM prescaler, offset: 0x28 */  
  __IO uint32_t ARR;      /*!< TIM auto-reload register, offset: 0x2C */  
  __IO uint16_t RCR;      /*!< TIM repetition counter register, offset: 0x30 */  
  __IO uint32_t CCR1;     /*!< TIM capture/compare register 1, offset: 0x34 */  
  __IO uint32_t CCR2;     /*!< TIM capture/compare register 2, offset: 0x38 */  
  __IO uint32_t CCR3;     /*!< TIM capture/compare register 3, offset: 0x3C */  
  __IO uint32_t CCR4;     /*!< TIM capture/compare register 4, offset: 0x40 */  
  __IO uint16_t BDTR;     /*!< TIM break and dead-time register, offset: 0x44 */  
  __IO uint16_t DCR;      /*!< TIM DMA control register, offset: 0x48 */  
  __IO uint16_t DMAR;     /*!< TIM DMA address for full transfer, offset: 0x4C */  
  __IO uint16_t OR;       /*!< TIM option register, offset: 0x50 */
```

```
} TIM_TypeDef;
```

6.4 STM32F407의 범용 TIMER 내부 구조 (TIM2~5)



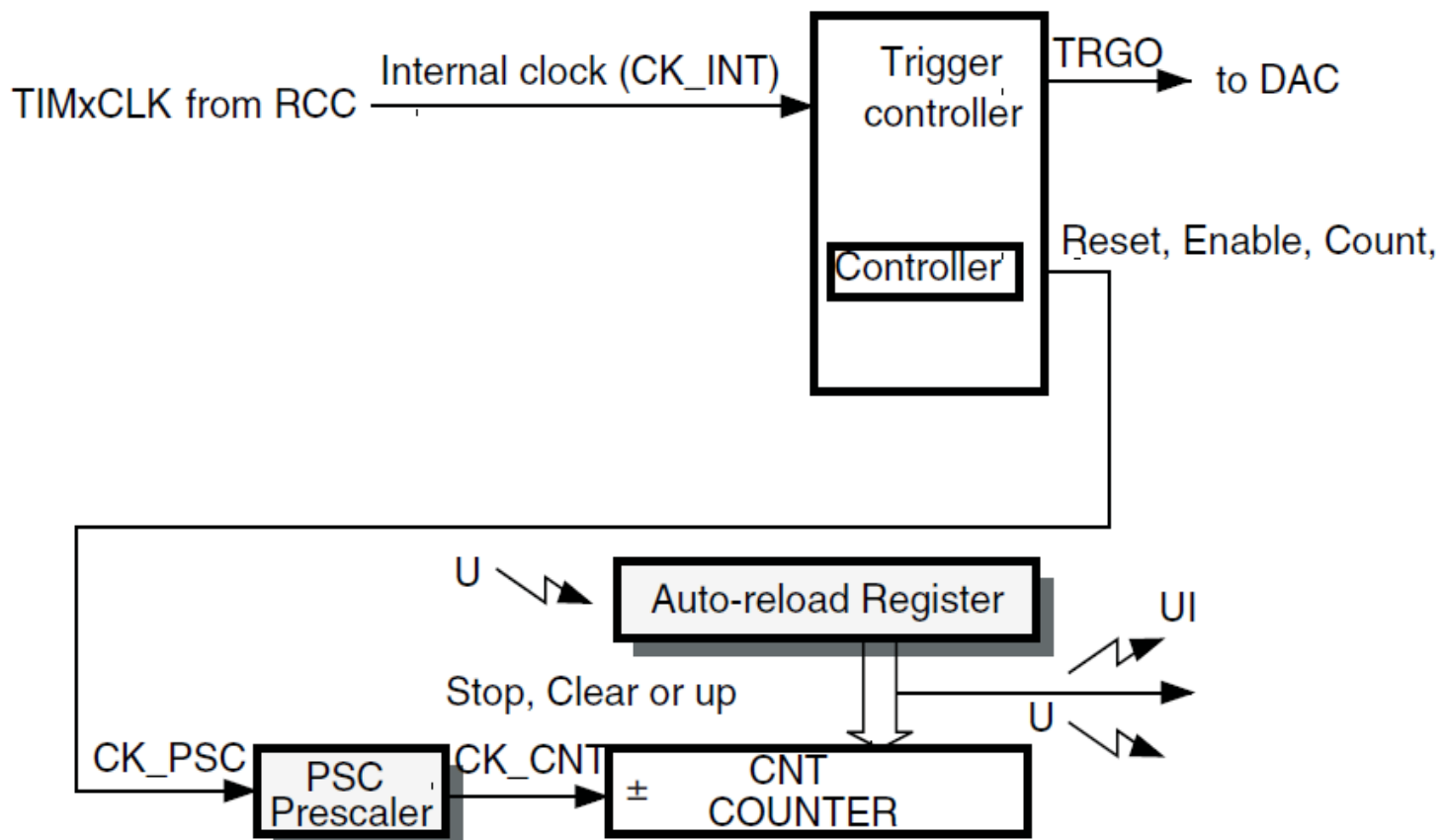
7. 기본 타이머

7.1 개요

- 기본 타이머(TIM6,TIM7) 특징
- 1개의 16 비트 카운터(Counter)를 가짐
 - Up 동작만 가능
 - Auto-reload 기능이 있음
- 16비트의 Prescaler를 이용하여 카운터의 동작클럭을 1~ 1/65,536까지 분주 가능
- 42MHz(Max)까지 동작이 가능한 APB1 버스에 연결
- 인터럽트/DMA 요청 신호를 발생
 - 업데이트 이벤트(overflow) 발생시

7.2 기본 타이머의 구조

Basic timer block diagram

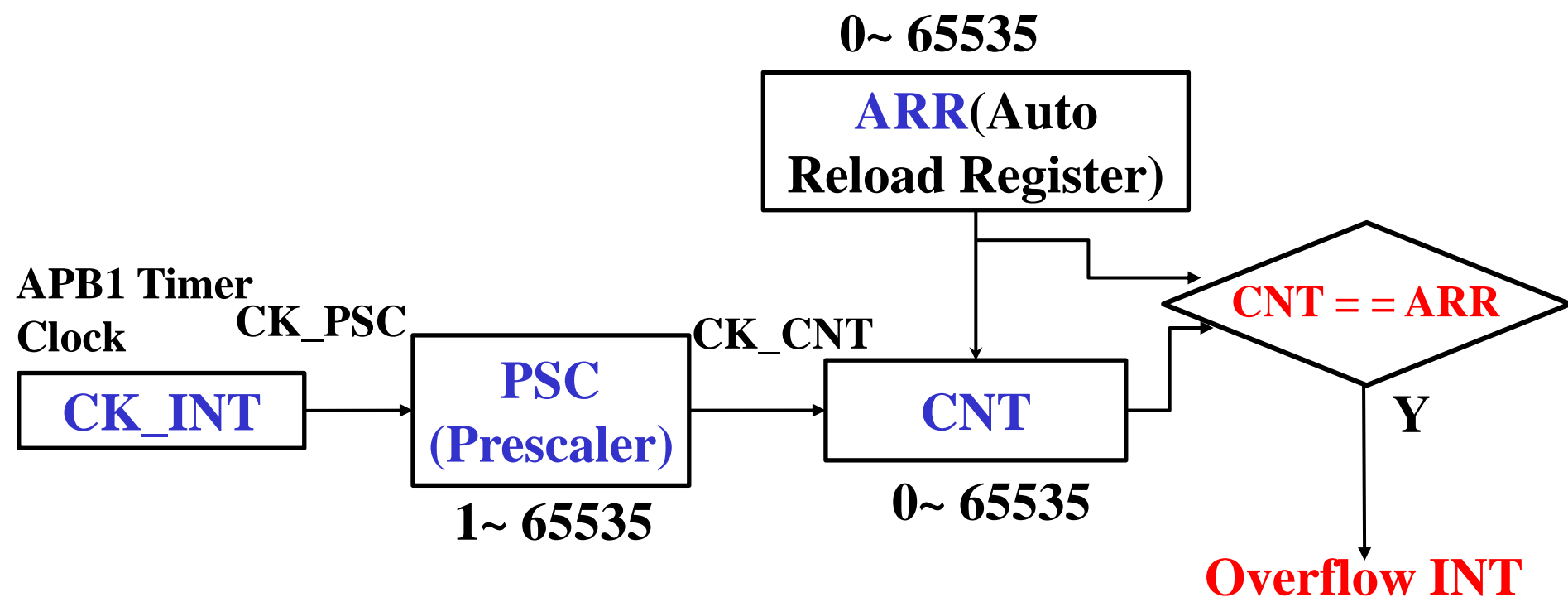


TIMx (x = 6,7)

7.3 기본 타이머의 주요 기능

- 카운터 모드
- Up 카운팅만 가능하며 그 동작은 범용 타이머와 기본적으로 유사
- * 기본 타이머는 캡처/비교기(Capture/Compare)가 없고 카운터(Counter)만 있으므로 업 카운터로만 동작함

(예) CK_INT사용한 기본타이머의 Overflow 인터럽트 발생



TIM_x (x = 6,7)

Ex) 1초 INT 만들기(APB1)

-CK_INT(CK_PSC) = 42MHz

-PSC = 4200 - 1

-CK_CNT = 10KHz

-ARR = 10000 - 1

7.4 기본타이머(TIM6,7) set-up 과정 및 레지스터 설정

RCC 설정

- **RCC→CR,CFGR,PLLCFGR (Clock소스/주파수설정)**
- **RCC→APB1ENR(TIMx Clock Enable)**

INT Enable

- **NVIC→ISER[]**

TIMx 초기 설정

- **TIMx→PSC(prescaler), TIMx→ARR(auto reload reg.)**
- **TIMx→CR1(counter EN, Dir etc)**
- **TIMx→DIER(update INT)**

Int Handler 설정

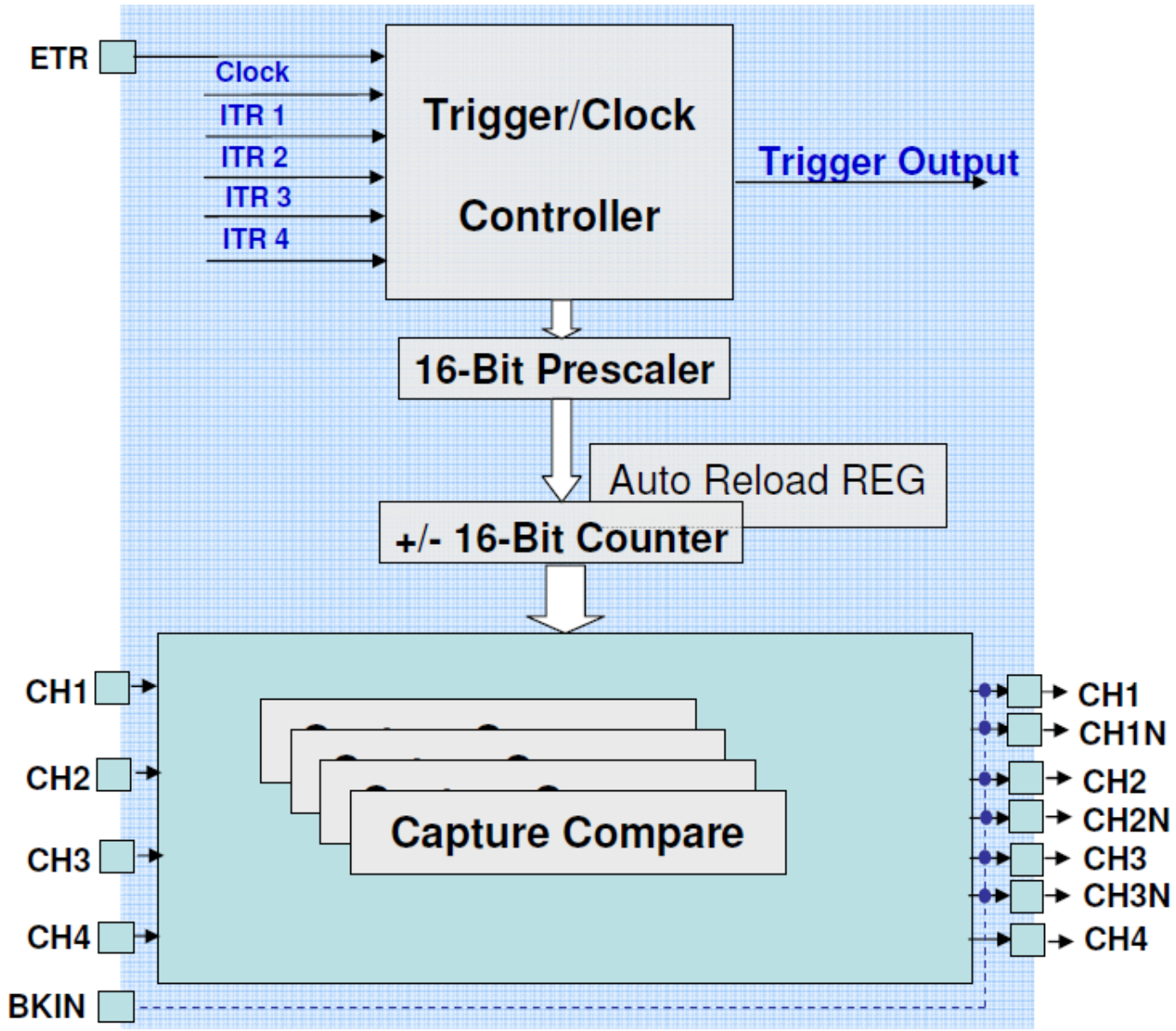
- **TIMx_IRQHandler()**

8. 고급제어 타이머

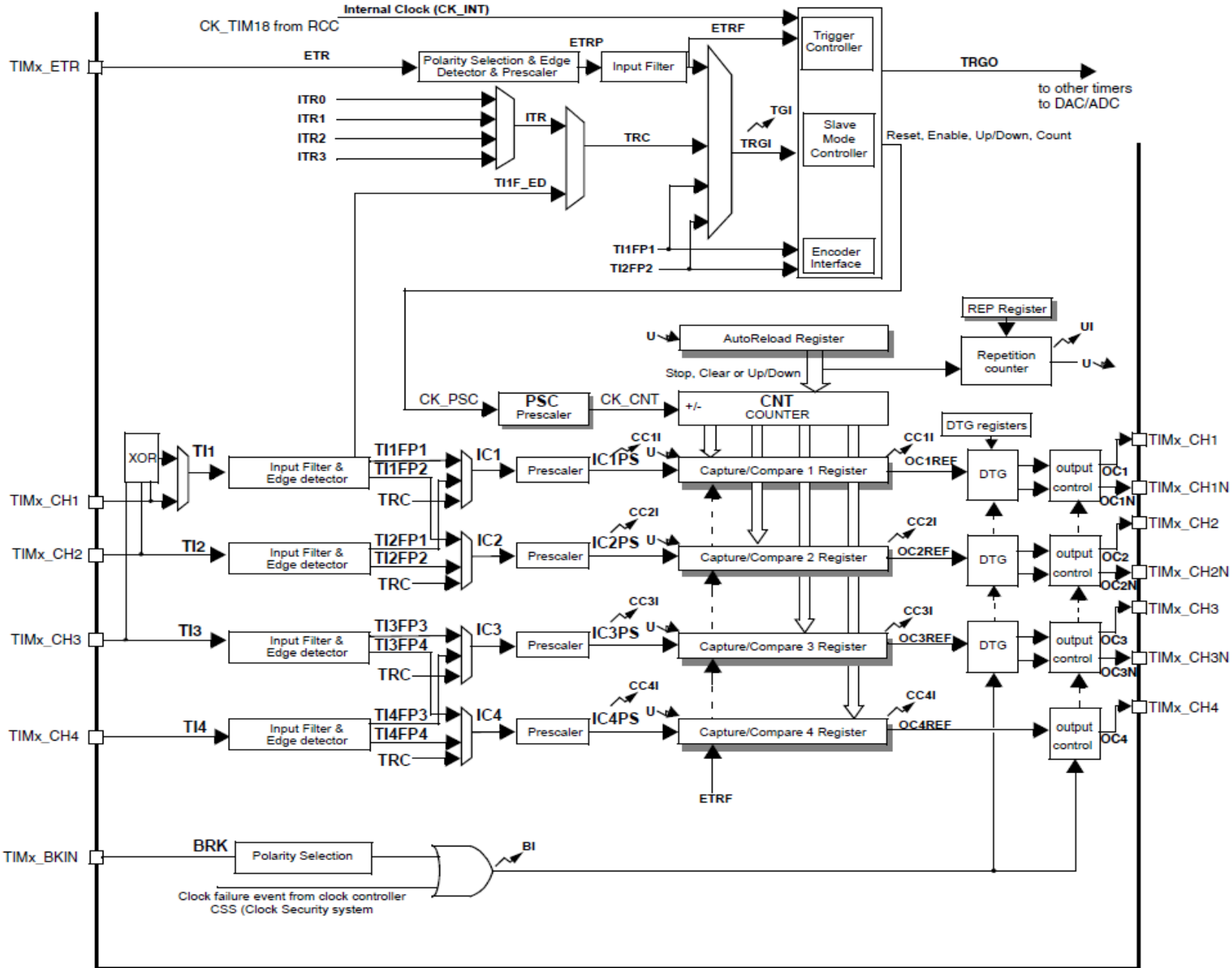
8.1 개요

- **TIM1, TIM8 (APB2)**
- 거의 모든 기능은 범용타이머와 동일
- 고급제어만의 기능
 - 3개의 반전된 출력 신호 발생 채널(**TIM_x_CH1N ~ TIM_x_CH3N**)을 가짐
 - 브레이크 입력 신호 채널(**BKIN**)을 가짐
 - 브레이크 입력시 **IRQ/DMA** 요청 신호를 발생
 - 반복(**Repeatition**) 타이머 기능을 가짐

8.2 고급제어 타이머의 간략구조



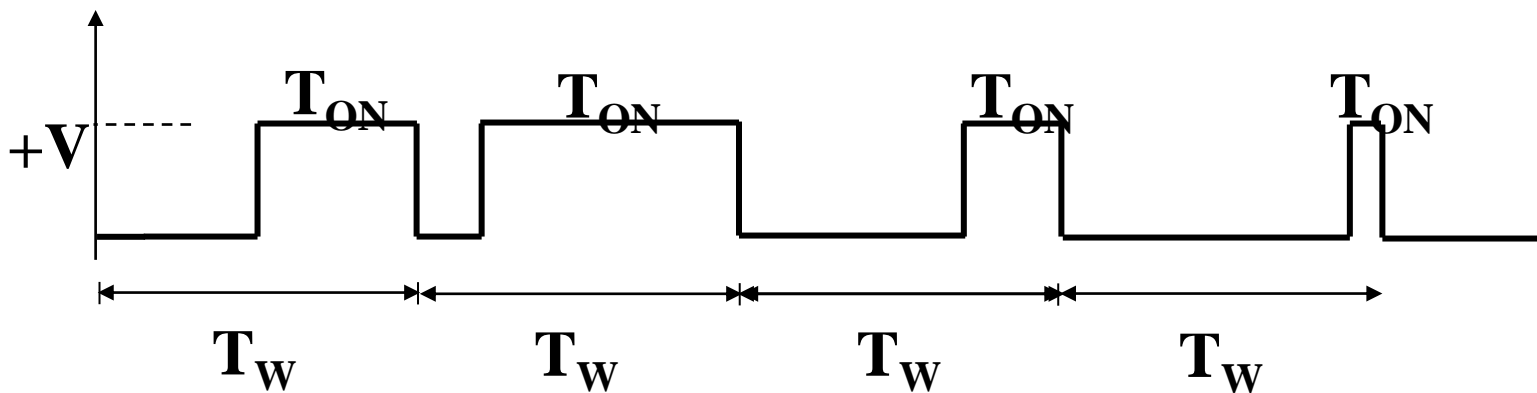
8.3 고급제어 타이머의 상세구조



◆부록1

PWM 신호와 모터제어

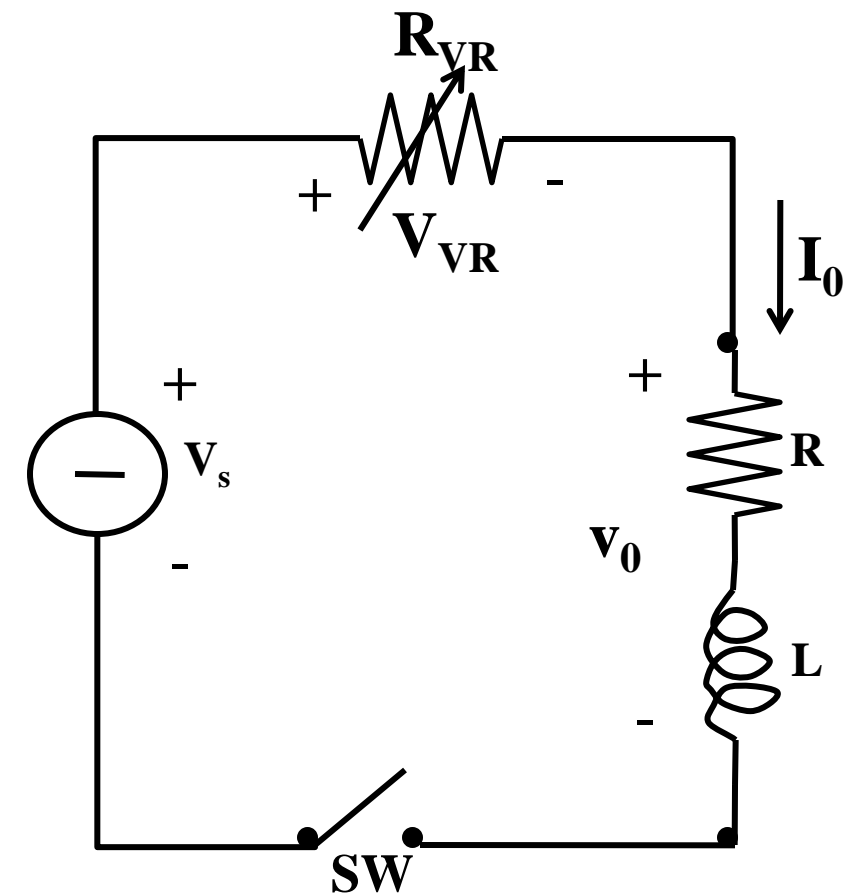
◆ PWM 신호 정의



- 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 액추에이터의 제어 분야에 PWM 제어 신호가 이용됨.

DC Motor 제어 : 가변저항사용

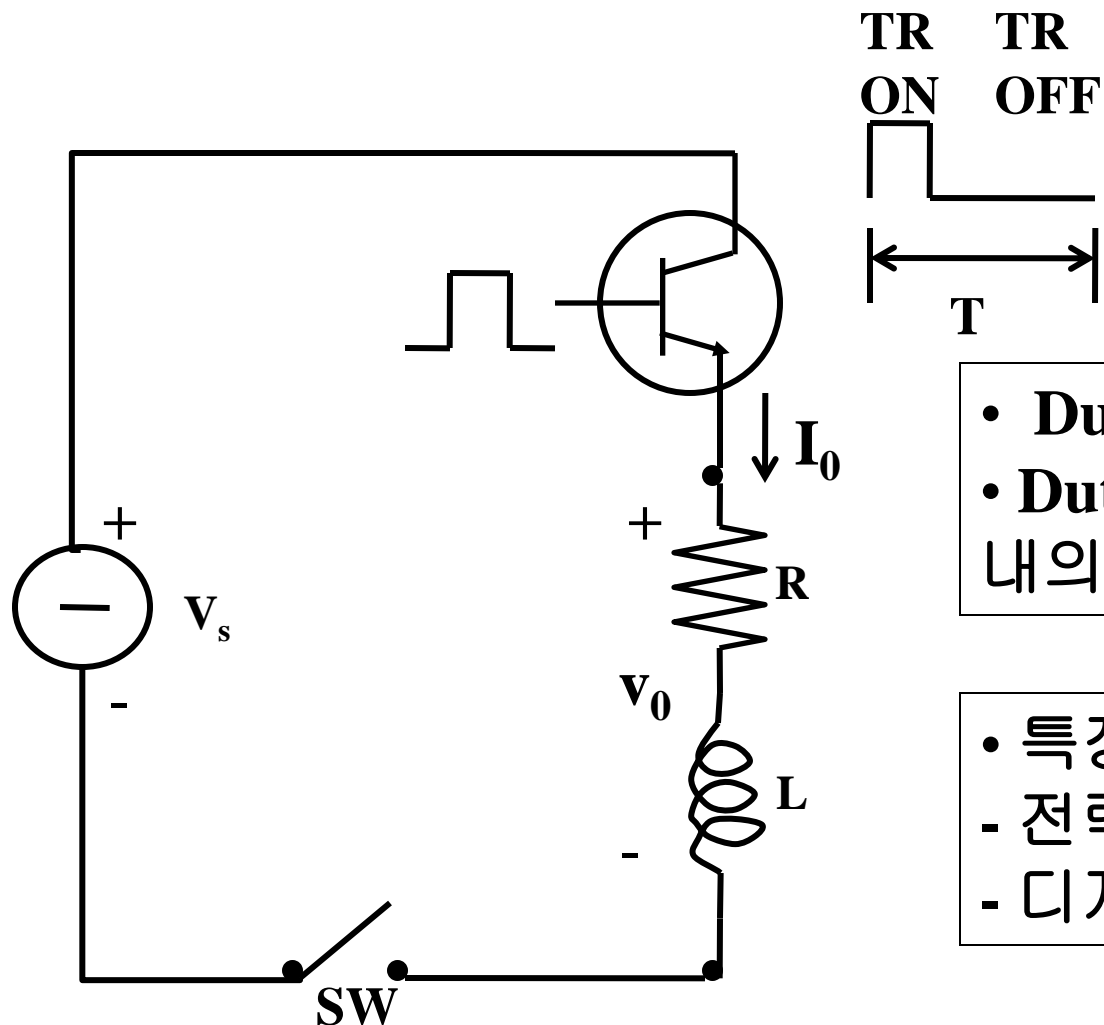
- R_{VR} 증가 $\rightarrow V_{VR}$ 증가 $\rightarrow I_0$ 감소 \rightarrow 모터 토크 감소
- R_{VR} 감소 $\rightarrow V_{VR}$ 감소 $\rightarrow I_0$ 증가 \rightarrow 모터 토크 증가



- 특징:
 - 항상 R_{VR} 에 전압이 걸려있어 열로 전력낭비
 - 아날로그 제어

DC Motor 제어 : TR/PWM 사용

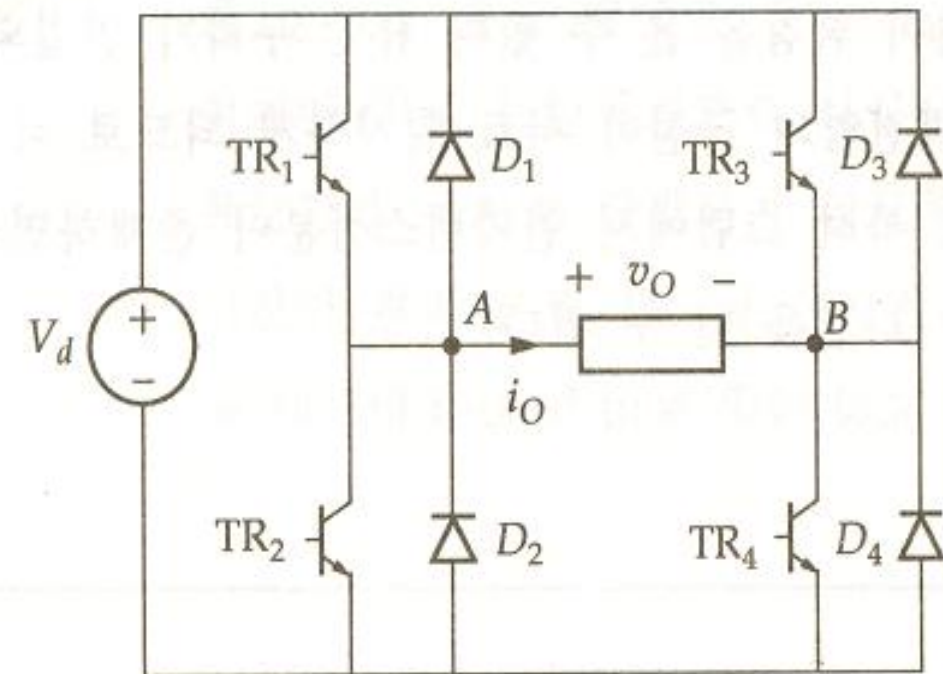
- PWM 신호의 듀티비에 비례한 전류 I_0 (전류 토크)증가



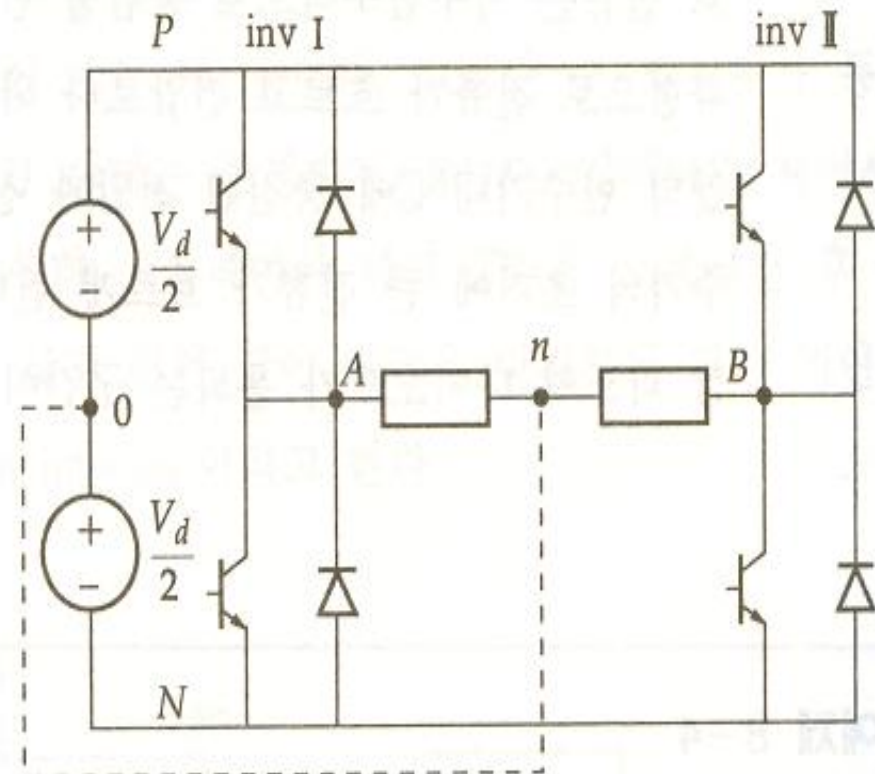
- Duty Ratio = $t_{on} / (t_{on} + t_{off})$
- Duty Ratio 증가 \rightarrow 한 주기(T) 내의 평균 전류 I_0 증가

- 특징:
 - 전력낭비 거의 없음
 - 디지털(마이컴) 제어 가능

AC 단상 Motor 제어 : TR/PWM 사용

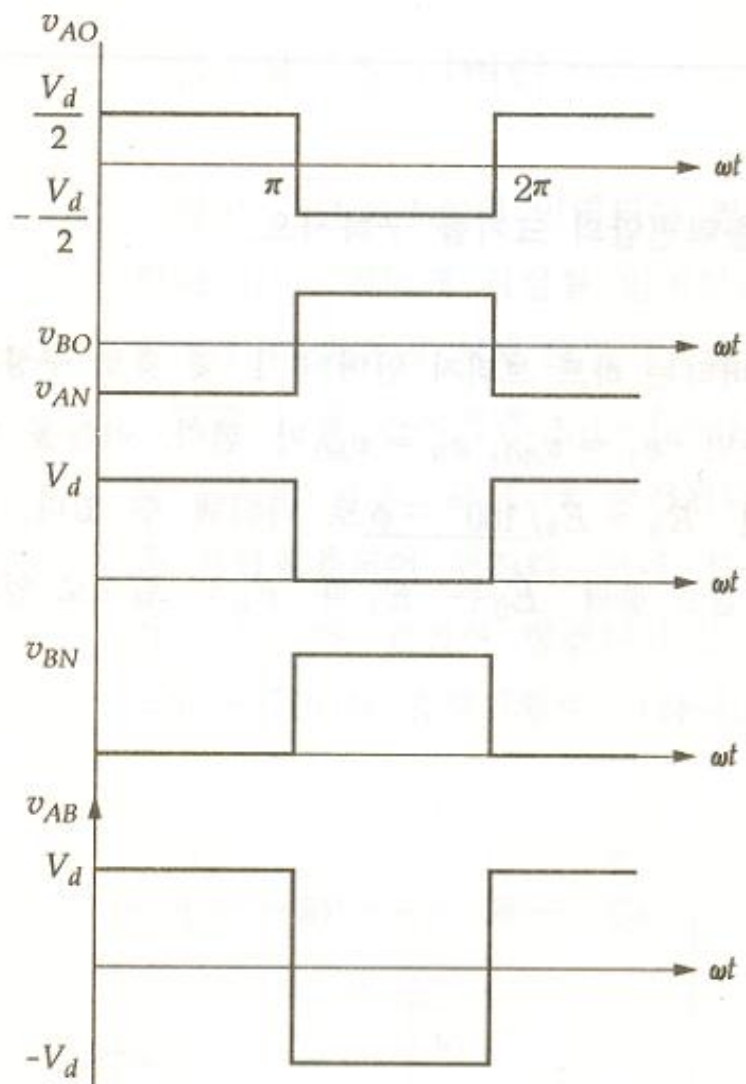


(a)

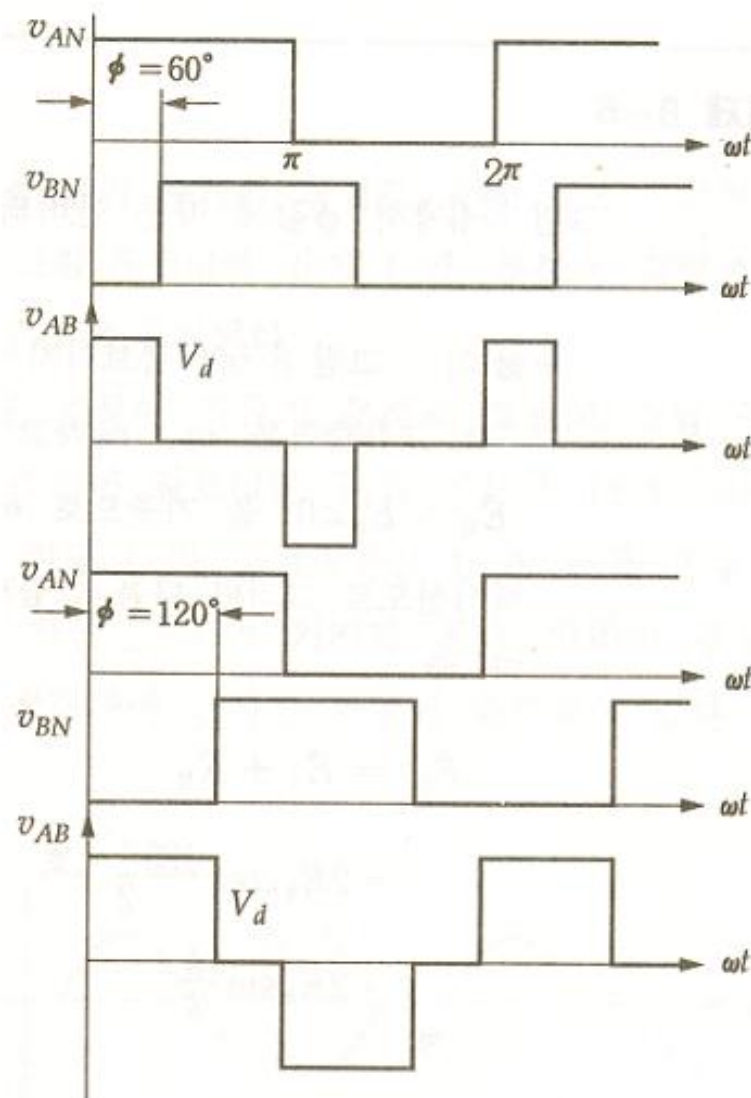


(b)

AC 단상 Motor 제어 : TR/PWM 사용

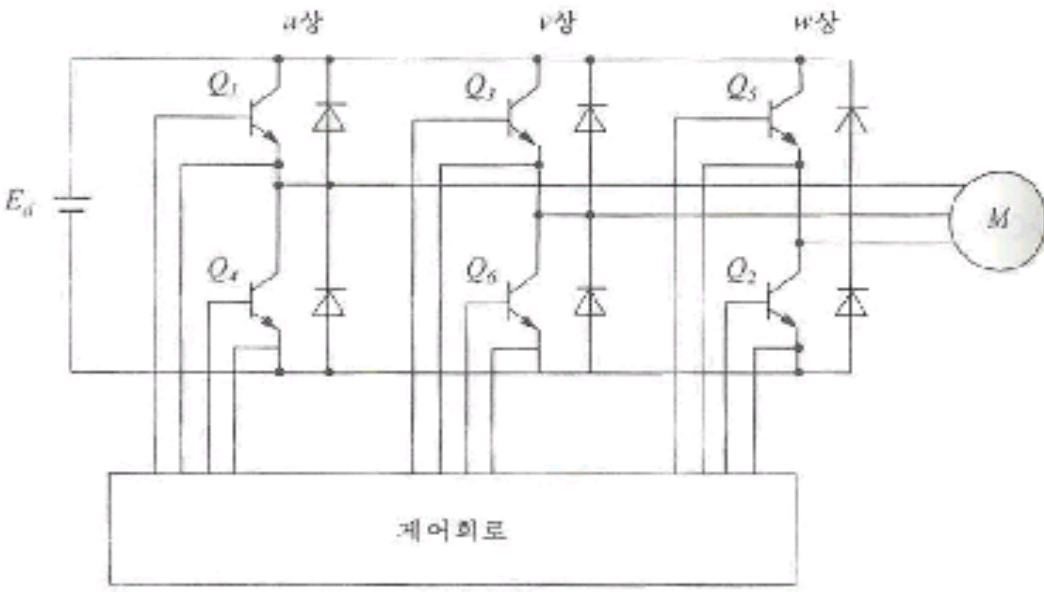
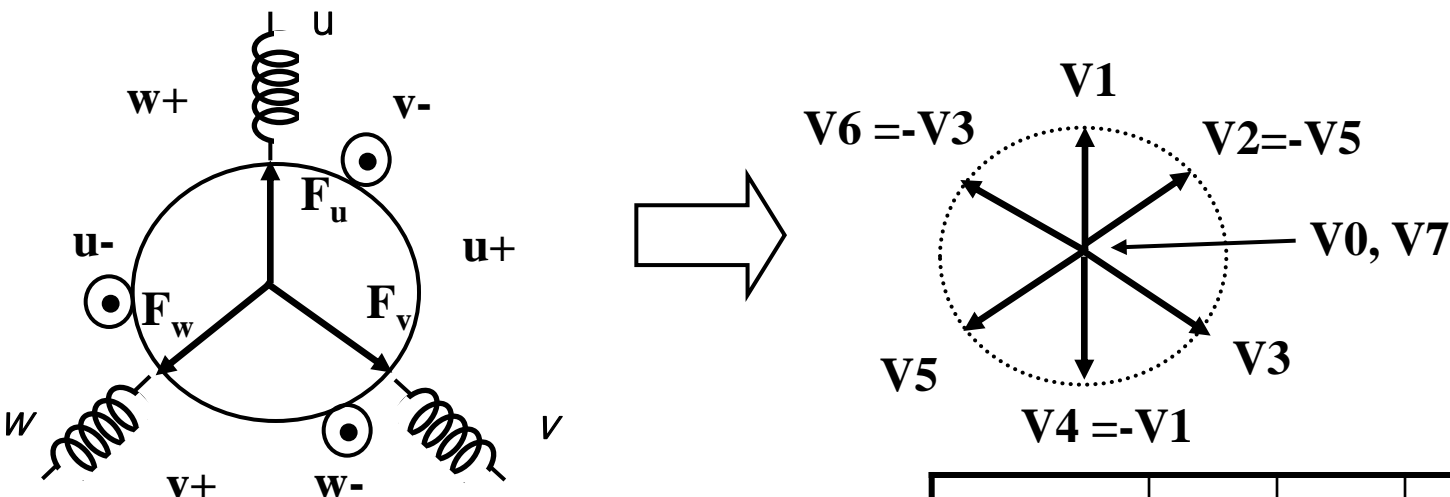


(a) $\phi = 180^\circ$



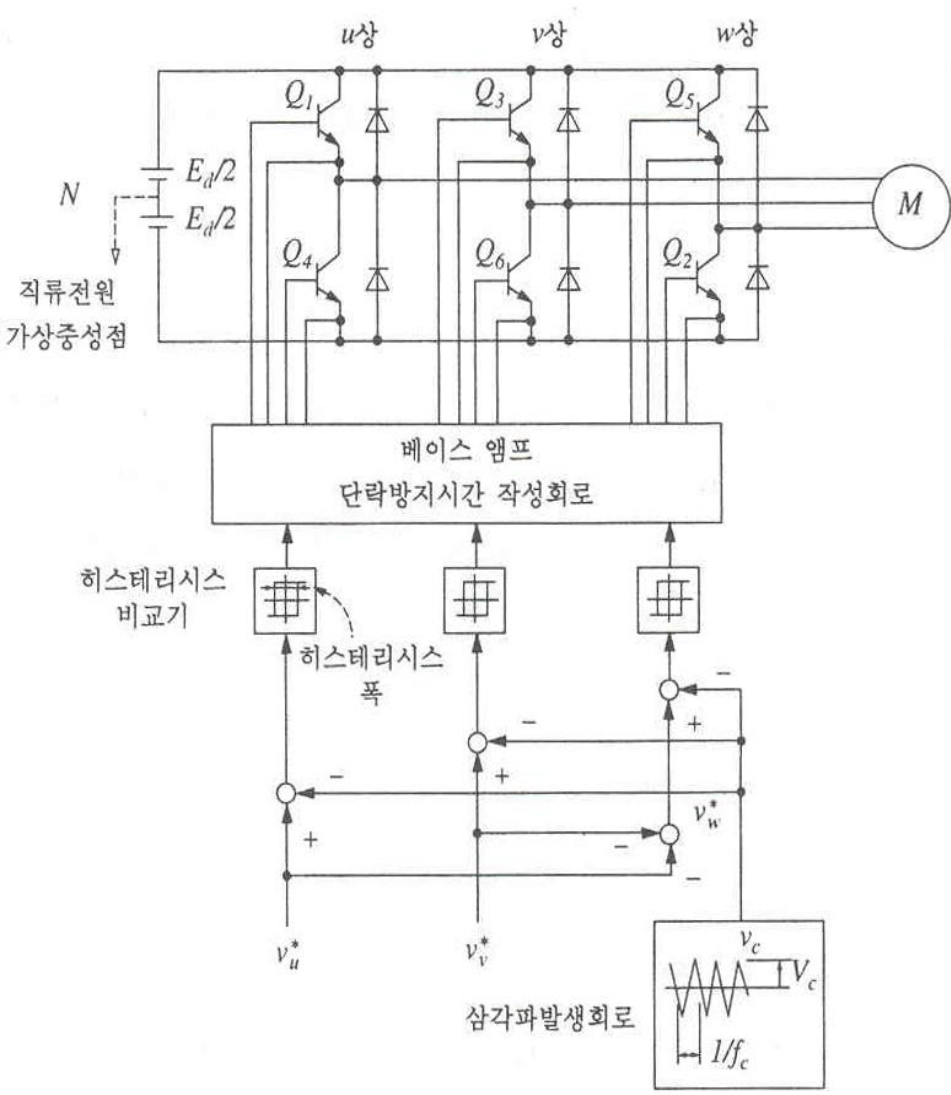
(b) $\phi = 60^\circ$, $\phi = 120^\circ$

3상 전압형 PWM Inverter의 기본구성과 동작

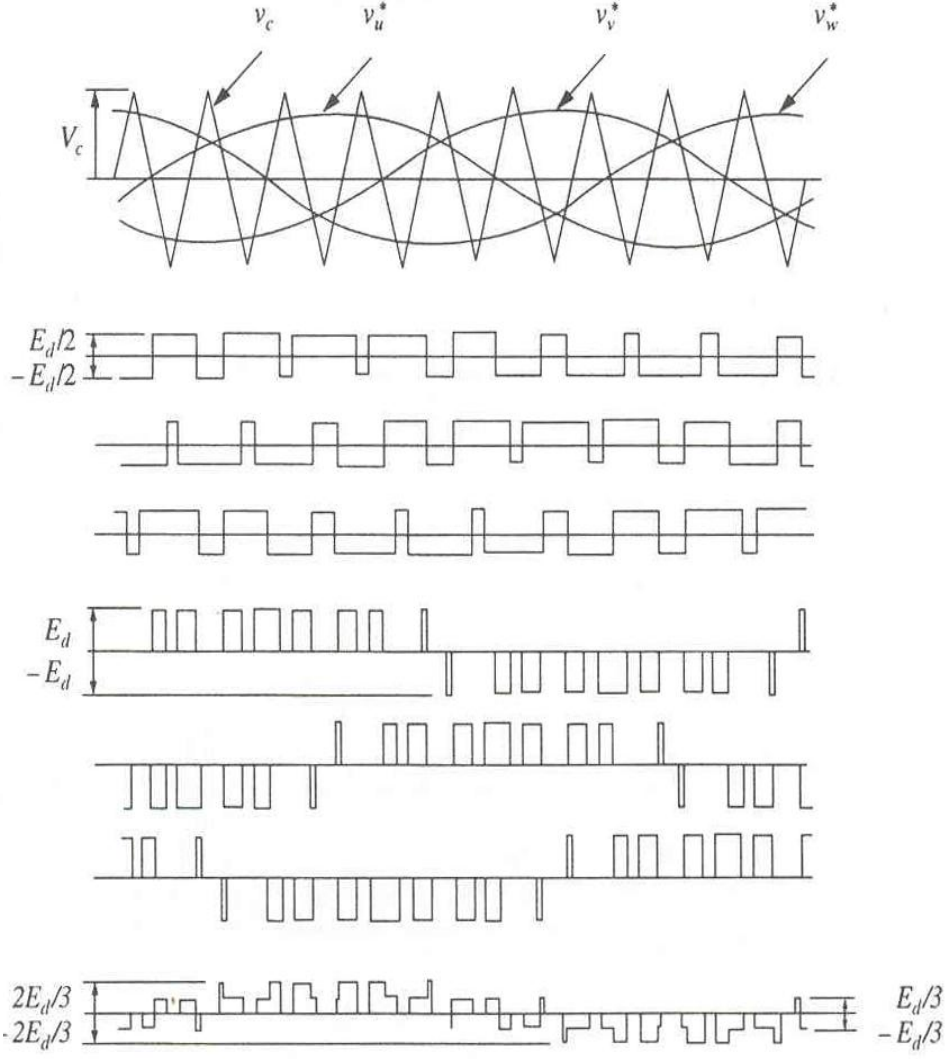


| Switching Mode | U | V | W | Mode |
|----------------|----|----|----|---------------|
| 0 | Q4 | Q6 | Q2 | Free-wheeling |
| 1 | Q1 | Q6 | Q2 | Motoring |
| 2 | Q1 | Q3 | Q2 | Motoring |
| 3 | Q4 | Q3 | Q2 | Motoring |
| 4 | Q4 | Q3 | Q5 | Motoring |
| 5 | Q4 | Q6 | Q5 | Motoring |
| 6 | Q1 | Q6 | Q5 | Motoring |
| 7 | Q1 | Q3 | Q5 | Free-wheeling |

3상 전압형 PWM Inverter의 기본구성과 동작



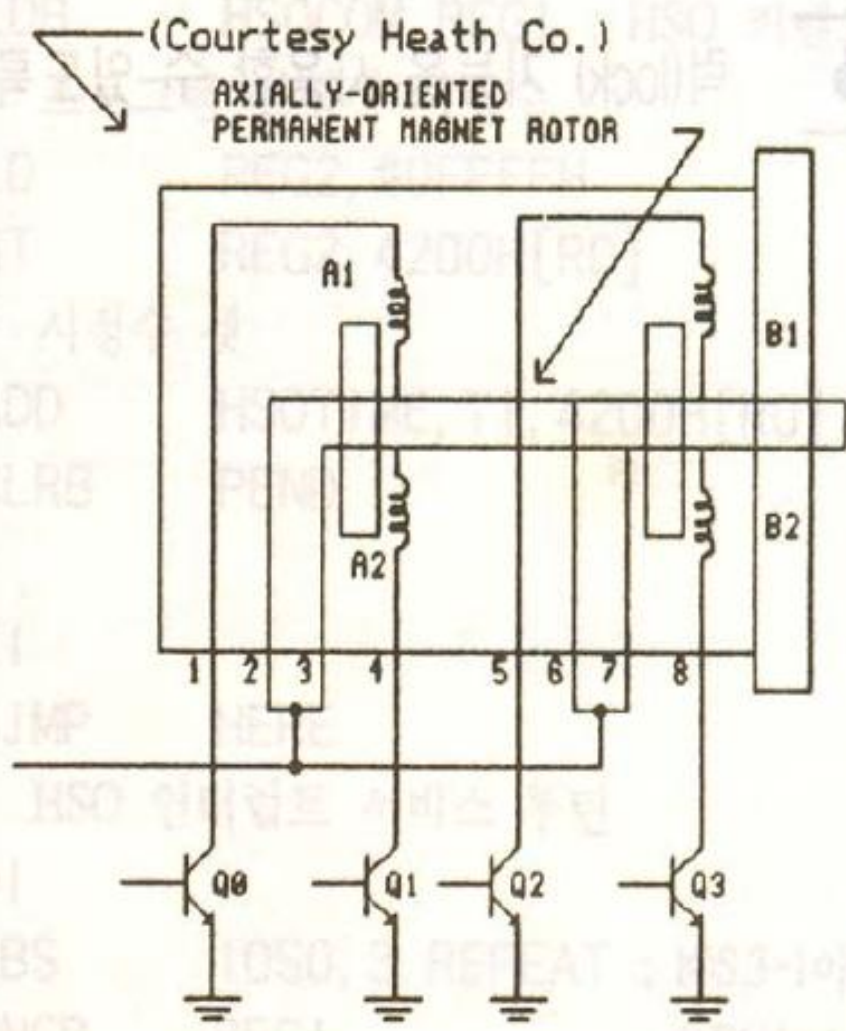
【그림 3.9】 3각파 비교법에 의한 3상 전압형 PWM 인버터



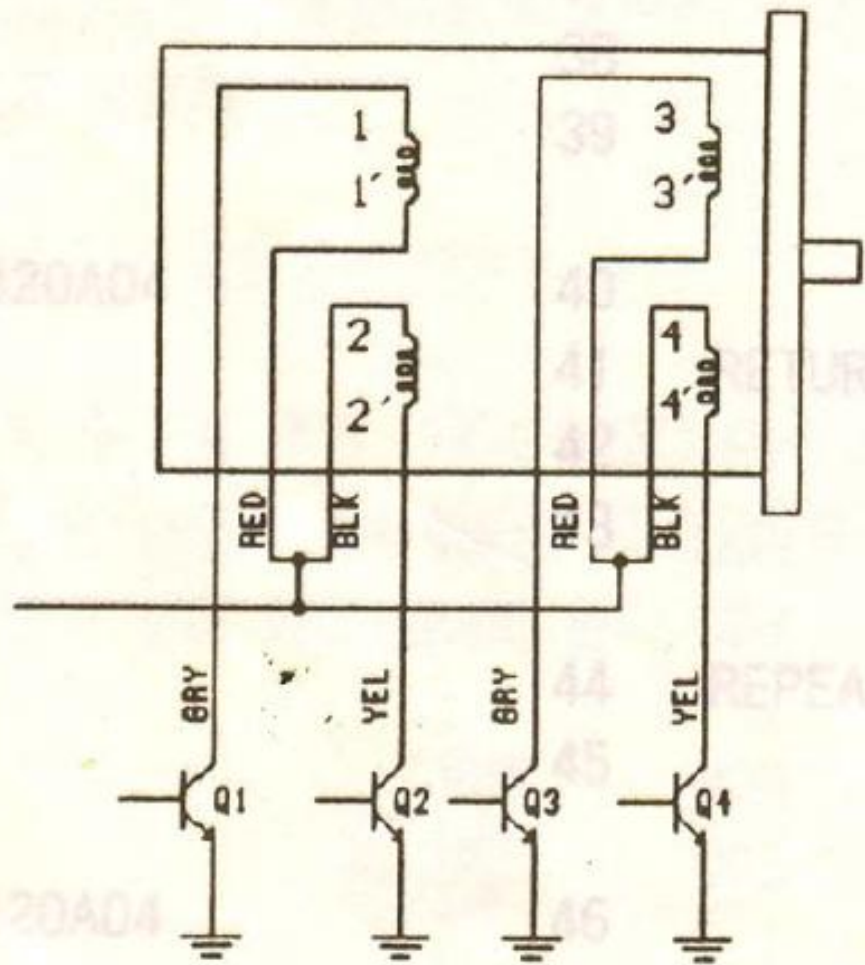
【그림 3.10】 삼각파 비교법에 의한 3상 전압형 PWM 인버터의 파형

◆부록2 STEP Motor 구조 및 제어 신호

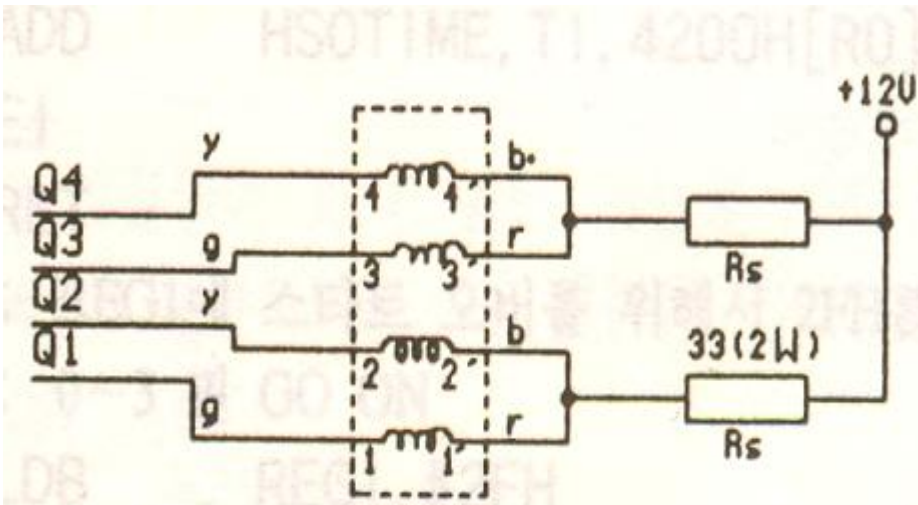
STEP Motor 제어



2,2'/4,4' reversed below



STEP Motor 제어



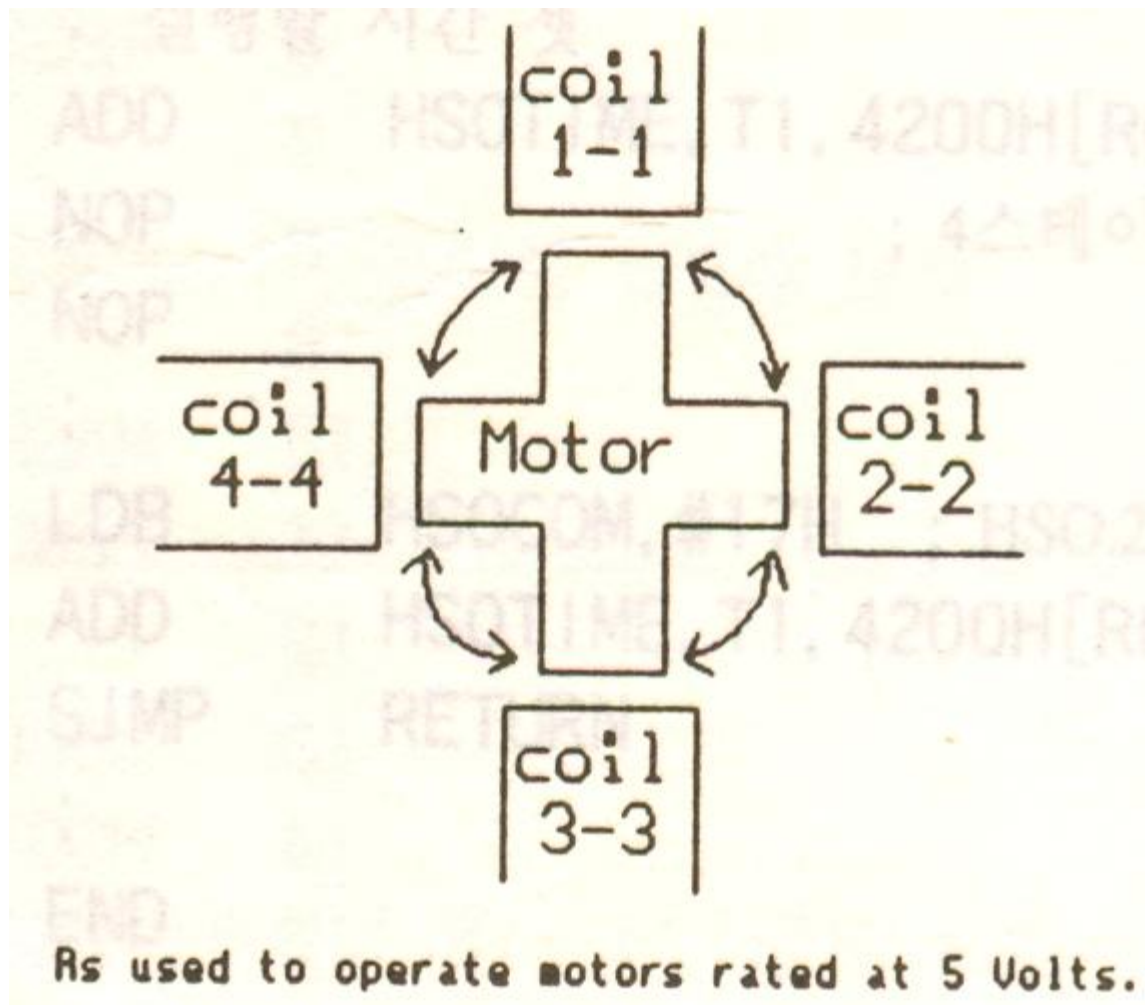
UNIPOLAR

| STEP | Q1 | Q2 | Q3 | Q4 |
|------|-----|-----|-----|-----|
| 1 | ON | OFF | ON | OFF |
| 2 | ON | OFF | OFF | ON |
| 3 | OFF | ON | OFF | ON |
| 4 | OFF | ON | ON | OFF |
| 1 | ON | OFF | ON | OFF |

CW ROTATION

CCW ROTATION


STEP Motor 제어




STEP Motor 제어

- 명령1: DIR + CK

DIR : High(CW), LOW(CCW)

CK : 

- 명령2: CW + CCW

CW : 

CCW : 

CCW : 

CW : 