Busan Software Meister High School

MICROPROCESSOR

2309 양유빈

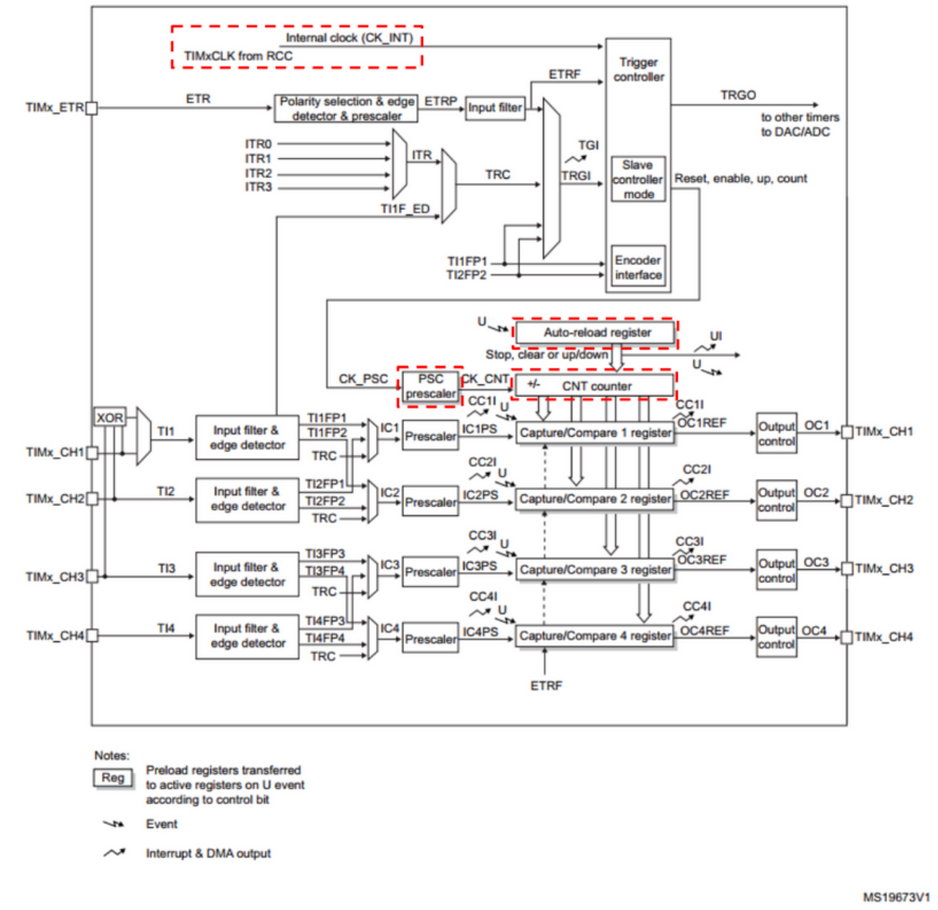
20230817 마이크로프로세서

STM32F103RB 범용 타이머 구조

STM32F103RB General Purpose Timer Structure

- 프리스케일러(PSC : Prescaler) : 16비트의 프리스케일러는 공급되는 클럭을 1 ~ 65,536(16bit)의 값으로 나누어 카운터의 동작 클럭을 만드는 분주기
- 카운터(CNT : Counter) : 16비트의 카운터는 타이머의 핵심적인 요소로 공급되는 클럭을 이용하여 업, 다운, 업/다운 카운팅 실시
- 오토 리로드 레지스터(ARR : Auto Reload Register) : 업 카운터는 'CNT 값 == ARR의 설정 값' 이 되면 CNT는 0부터 다시 카운팅되고, 다운 카운터는 'CNT 값 == 0' 이 되면 CNT는 ARR의 설정 값부터 다시 카운팅 시작
- 캡쳐/비교기(CCR : Capture/Compare Register) : 입력 신호가 주어질 때 카운터(CNT) 값을 캡쳐하거나 또는 CNT == CCR이 되면 인터럽트를 발생하거나 출력 채널로 0또는 1을 출력

Figure 100. General-purpose timer block diagram



타이머

timer

- 클럭이 입력될 때마다 값이 1씩 증가 또는 감소하여 시간을 측정하거나 횟수를 세는 장치
- 보통 8비트나 16비트 또는 32비트의 카운터와 주변회로로 구성
- 타이머의 부가적 기능
  - 특정 값에 도달하면 인터럽트를 발생시키는 기능
  - 오버플로우가 생기면 인터럽트를 발생시키는 기능
  - 특정 포트 PWM(Pulse Width Modulation) 신호로 출력하는 기능 등
- STM32F103RB
  - 고기능 타이머 1개(TIM1)
  - 범용 타이머 3개(TIM2, TIM3, TIM4)
  - Watchdog 타이머 1개
  - SysTick 타이머 1개

Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/ compare channels	Complementary output	Max interface clock (MHz)	Max timer clock (MHz)
Advanced-control	TIM1, TIM8	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	Yes	84	168
General purpose	TIM2, TIM5	32-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM3, TIM4	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM9	16-bit	Up	Any integer between 1 and 65536	No	2	No	84	168
	TIM10, TIM11	16-bit	Up	Any integer between 1 and 65536	No	1	No	84	168
	TIM12	16-bit	Up	Any integer between 1 and 65536	No	2	No	42	84
	TIM13, TIM14	16-bit	Up	Any integer between 1 and 65536	No	1	No	42	84
Basic	TIM6, TIM7	16-bit	Up	Any integer between 1 and 65536	Yes	0	No	42	84

타이머 인터럽트 관련 HAL 라이브러리

HAL library related to timer interrupt

형태	HAL_StatusTypeDef HAL_TIM_Base_Start_IT(TIM_HandleTypeDef *htim)
설명	범용 타이머를 활성화 시키고, 초기에 세팅한 조건에 맞으면 인터럽트를 발생시킴
파라미터	htim : 활성화시킬 타이머 구조체 포인터 값(&htim1(timer1) ~ &htim3(timer3))
리턴값	실행 결과(HAL_OK, HAL_ERROR, HAL_BUSY, HAL_TIMEOUT)
예시	Timer1을 활성화 시키고 싶을 때의 코딩: HAL_TIM_Base_Start_IT(&htim1);

타이머 인터럽트 관련 HAL 라이브러리

HAL library related to timer interrupt

형태	void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
설명	타이머 인터럽트가 발생되었을 때, 최종적으로 실행되는 사용자 콜백 함수
파라미터	htim ：활성화시킬 타이머 구조체 포인터 값(&htim1(timer1) ~ &htim3(timer3))
리턴값	없음
예시	Timer1 타이머 인터럽트가 발생되었을 때의 코딩: HAL_TIM_PeriodElapsedCallback(&htim1) { … 사용자 작성 인터럽트 서비스 루틴 … }

printf() 사용을 위한 환경 설정

Preferences for using printf()

- main( ) 함수의 첫 부분에 “#include <stdio.h>” 삽입
- int __io_putchar(int ch) 함수 구현
  - printf( ) 라이브러리에서 최종적으로 사용하는 한 바이트 출력 함수
  - 1바이트를 최종적으로 출력하는 방법은 MCU마다 다르므로 사용자가 __io_putchar(int ch) 함수를 구현
  - __io_putchar(int ch) 함수의 기능은 HAL_UART_Transmit( ) 함수를 사용하여 구현

```
// printf() 함수 사용하기 위한 코드
int __io_putchar(int ch) {
  HAL_UART_Transmit(&huart2, (uint8_t *)&ch, 1, 0xFFFF);
  // 0xFFFF는 최대 대기 시간
  if (ch == '\n') {
    HAL_UART_Transmit(&huart2, (uint8_t *)"\r", 1, 0xFFFF);
  }
  return ch;
}
```

```
// scanf() 함수 사용하기 위한 코드
int __io_getchar(void) {
  uint8_t ch = 0;
  __HAL_UART_CLEAR_OREFLAG(&huart2);
  HAL_UART_Receive(&huart2, (uint8_t *)&ch, 1, 0xFFFF);
  return ch;
}
```