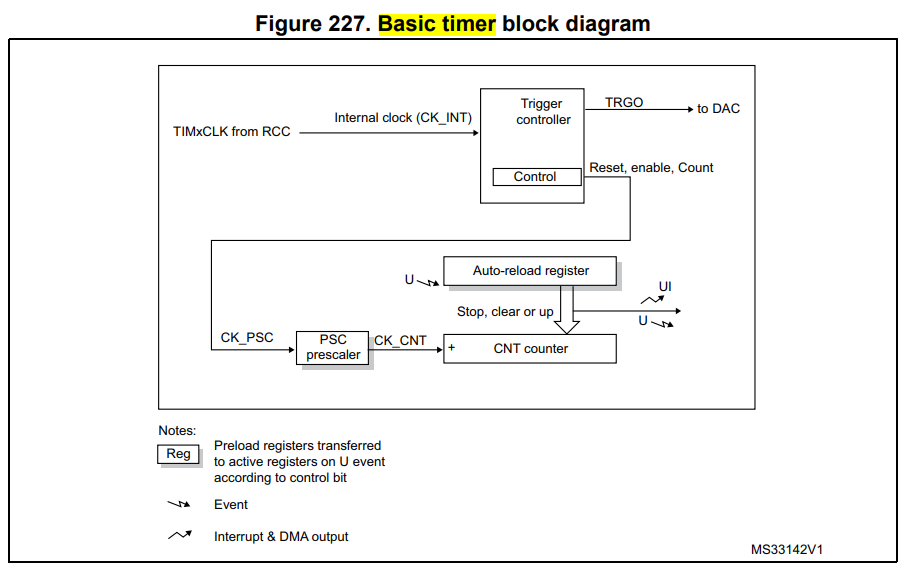
Timer tutorial polled mode

1. See application notes of timer page 7/39.
2. Overall we have 14 timers available for STM32F4 which is listed below
3. STM32F4 has 2 advanced timers Timer1 and timer 8
4. Tim2 and tim5 2 general purpose 32 bit timers available
5. Tim3 and Tim4 are general purpose 16 bit timers
6. Two basic timers tim6 and tim7
7. Tim10,11,13,14 general purpose timers are available but with single channel.
8. Tim9 and Tim12 general purpose timers are available but with dual channel.

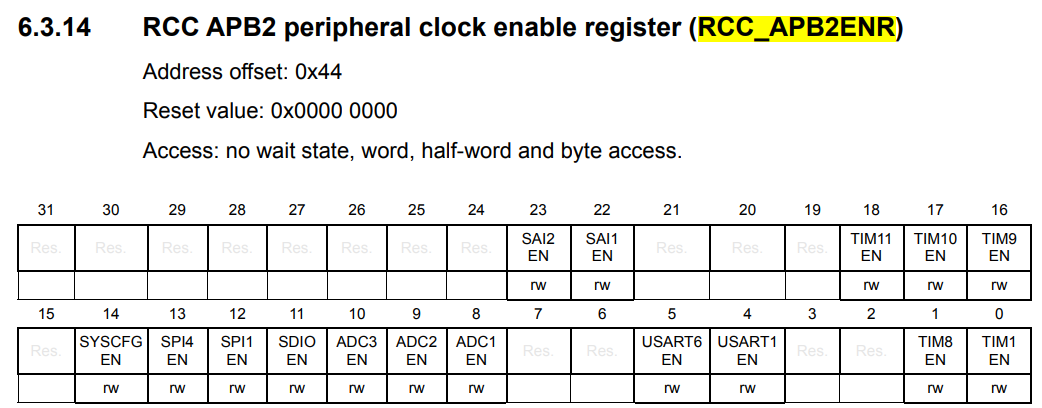
The timer can be used as a time base generator. Depending on the clock, prescaler and auto reload, repetition counter (if present) parameters, the 16-bit timer can generate an update event from a nanosecond to a few minutes. For the 32-bit timer, the range is larger.

The timer can be programmed in two ways to generate a hardware delay.

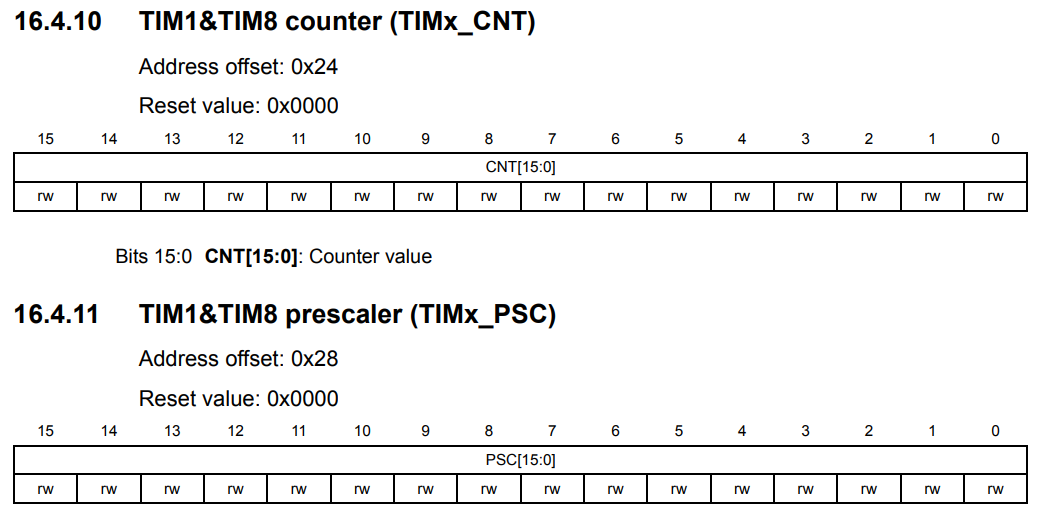
1. Timer in polled mode
2. Timer in interrupt mode.

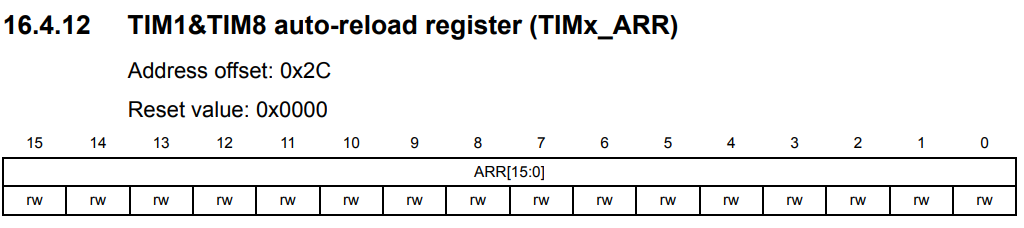


1. First enable the timer clock related to a timer (say timer 1) in relevant bits of RCC\_APB2ENR.

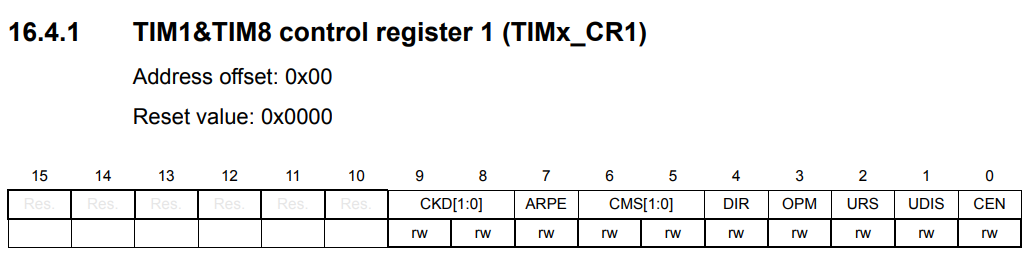


1. Compute PSC and ARR values for that respective timer(say timer 1)

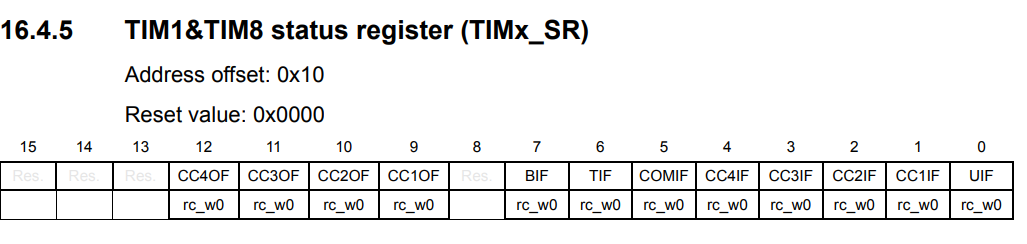




1. Start the Timer using its CR register by CEN bit.



1. Wait for the timer to overflow which is determined by its status register which is given below.



So all things put together can be as follows.

**void** **Timer1\_delay**(**void**)

{

RCC->APB2ENR |=(1U<<0);

/\*Set prescaler value\*/

TIM1->PSC = 16000-1;

/\*Set auto-reload value\*/

TIM1->ARR =500-1;

/\*Clear counter\*/

TIM1->CNT =0;

/\*Enable timer\*/

TIM1->CR1 = (1U<<0);

while(!(TIM1->SR & 0x01));

}

**int** **main**(**void**)

{

RCC->APB1ENR |=(1U<<0);

**while**(1)

{

**printf**("im in while\n");

GPIOA->ODR ^= (1<<5);

Timer1\_delay(1);

}

}

Note :- change the values of PSC and ARR value and observe the change in delay of blinking LED.

Timer tutorial in interrupt mode

1. Enable the clock for timer 2 on RCC\_APB1ENR
2. Compute the PSC and ARR values of timer 2.
3. Reset timer2 CNT register.
4. Enable DMA/Interrupt register of timer 2. ( DIER )
5. Enable the NVIC(IRQ\_Number) for the corresponding timer.
6. Start the timer
7. Call an infinite loop in main to observe a hardware handover to ISR from main.

Implement a timer 2 handler or ISR with the following contents in it .

1. Clear the timer status register
2. Toggle a LED port / print a message relevant to ISR
3. Return from ISR

Note : change the PSC and CNT register values and observe changes in duration of LED blinking.