STM32 Timers

G V V Sharma*

Contents

Abstract—**This manual shows how to program timers in arm using STM32F103C8T6.**

1 Components

Component	Value	Quantity
Breadboard		1
Resistor	220 Ω	1
STM32F103C8T6		1
Seven Segment Display	Common Anode	1
Jumper Wires		20

TABLE 1.0

Problem 1.1. List all available clocks in the STM32F103C8T6 blue pill.

Solution: See Table 1.1.

Clock	Location	Type	Frequency
HSI	Internal	RC	8Mhz
LSI	Internal	RC	32.768 kHz
HSE	External	Crystal	8Mhz

TABLE 1.1

Problem 1.2. List all the available timers in the STM32F103C8T6 blue pill.

Solution: See Table 1.2

*The author is with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

Timer	Type	Counter Resolution
Systick	Default	24 bit
Independent	Watchdog	12 bit
Window	Watchdog	7 bit
TIM1	Advanced	16 bit
TIM2	General Purpose	
TIM3		
TIM4		

TABLE 1.2

2 Systick timer

The Systick timer is the default timer available on all ARM chips.

Problem 2.1. Make connections as shown in Table 2.1.

STM32	Seven Segment Display
3.3V	COM (through resistor)
PA1	DOT

TABLE 2.1

Problem 2.2. Execute the program in

```
https://github.com/gadepall/
    STM32F103C8T6/blob/master/
    examples/blink_systick.c
```

Problem 2.3. The default clock is the HSI 8MHz RC. Find the number of clock cycles required for a 1 s delay.

Solution: The time period is

$$T = \frac{1}{8}\mu s = 1 \text{ cycle} \tag{2.3.1}$$

Thus, the number of cycles required for 1 s delay is

$$1 \text{ second} = 8000000 \text{ cycles} \tag{2.3.2}$$

Problem 2.4. List the SysTick registers.

Solution: See Table 2.4.

Register	Command	Purpose
SysTick Control and Status	SysTick->CTRL	Timer control
SysTick Reload Value	SysTick->LOAD	Timer Count
SysTick Current Value	SysTick->VAL	Timer Initialize
SysTick Calibration Value		

TABLE 2.4

Problem 2.5. What do the following instructions do?

```
SysTick->LOAD = 4000000;
SysTick->VAL = 0;
```

Solution: See Table 2.4 for details. These two instructions ask the SysTick timer to count down from 4000000 to 0.

Problem 2.6. Explain the following instruction.

```
while(!(SysTick->CTRL & 0x00010000
    ));
```

Solution: Fig. 2.6 shows the SysTick CTRL register. 0x00010000 is used in the above command to mask all the bits except for bit 16, which is the COUNTFLAG. The **while** loop will stop once COUNTFLAG = 0. The while loop is used for the delay.

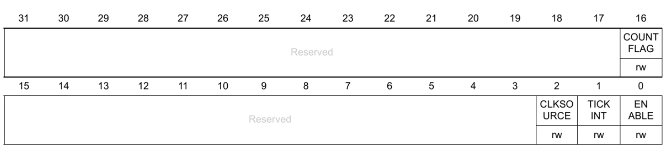

Fig. 2.6

Problem 2.7. What does the following instruciton do?

```
SysTick->CTRL = 0x00000005;      //
    8MHz clock
```

Solution: From Fig. 2.6, ENABLE = 1 enables the counter (for delay) and CLKSOURCE = 1 enables the 8 MHz internal RC clock.

Problem 2.8. Obtain a 1 MHz clock.

Solution: CLKSOURCE = 1 results in the $\frac{\text{Processor Clock}}{8}$ = 1 MHz clock.

```
SysTick->CTRL = 0x00000001;      //
    1MHz clock
```

Problem 2.9. Obtain a delay of 1 second using the 1 MHz clock.

3 TIM1

Problem 3.1. Make the connections according to Table 2.1. Execute the following program

```
https://github.com/gadepall/
  STM32F103C8T6/blob/master/
  examples/timer1_blink.c
```

Problem 3.2. Enable Timer1 through RCC.

Solution:

```
RCC->APB2ENR |= RCC_APB2ENR_TIM1EN
    ;
```

Problem 3.3. Select the HSI clock of 8 MHz as TIM1 clock.

Solution:

```
TIM1->SMCR  = 0;
```

Problem 3.4. Make TIM1 clock = 2 KHz.

Solution: Through the following command,

```
TIM1->PSC         = 3999;
```

$$TIM1\_CLK = \frac{HSI\_CLK}{TIM1->PSC+1} = \frac{8000000}{4000} \tag{3.4.1}$$

Problem 3.5. Make TIM1 count 1000 cycles of the 2 KHz TIM1 clock.

Solution:

```
TIM1->ARR         = 999;
```

Problem 3.6. What do the following instructions do?

```
if(TIM1->SR & 0x0001)//check if
  ARR count complete
{
        TIM1->SR &= ~0x0001; //
          clear status register SR
        GPIOA->ODR ^= (1 << 1); //
          blink LED through PA1
}
```

Solution: Once the TIM1 counter counts from 0 to TIM1− >ARR=999, it resets and starts counting again to 999. At the time of reset, the LSB of TIM1− >SR = 1. The **if** command checks this and when this condition is satisfied, TIM1− >SR is cleared and PA1 is toggled. This process keeps repeating. This results in a PA1 output of 1 and 0 with frequency

$$\frac{HSI\_CLK}{(TIM1->PSC+1)\,(TIM1->ARR+1)} = \frac{8000000}{4000 \times 1000} = 2 \text{ Hz} \quad (3.6.1)$$