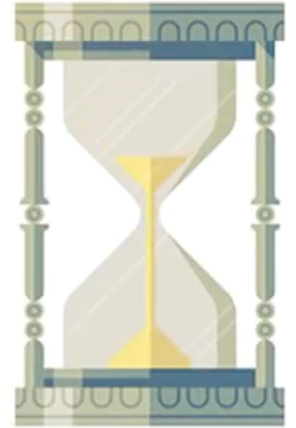
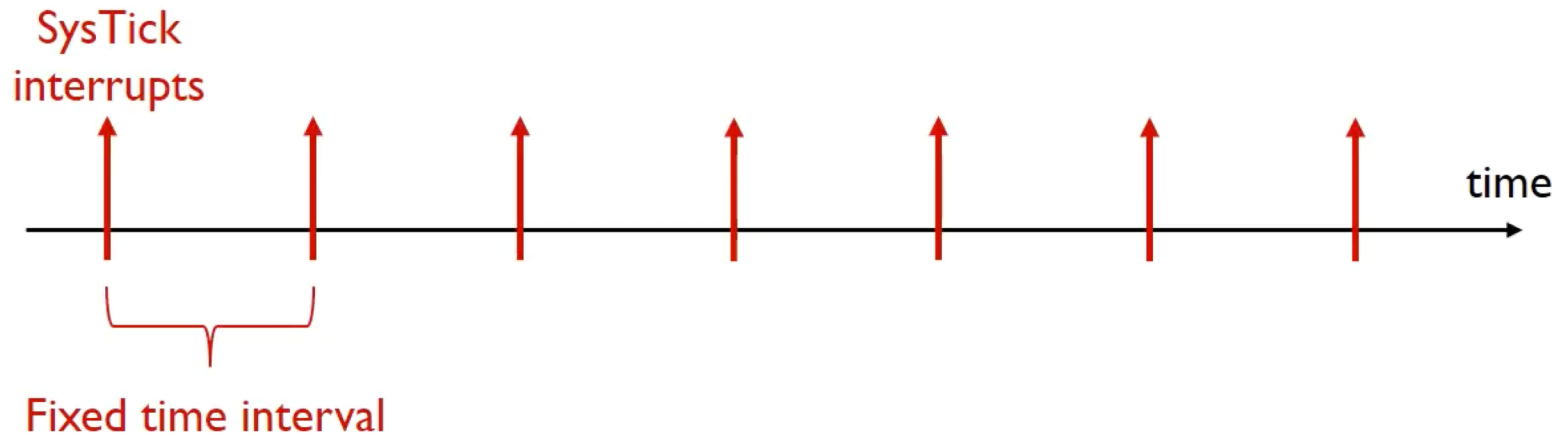


LAB - TIMERS

ARM CortexM

System Timer (SysTick)

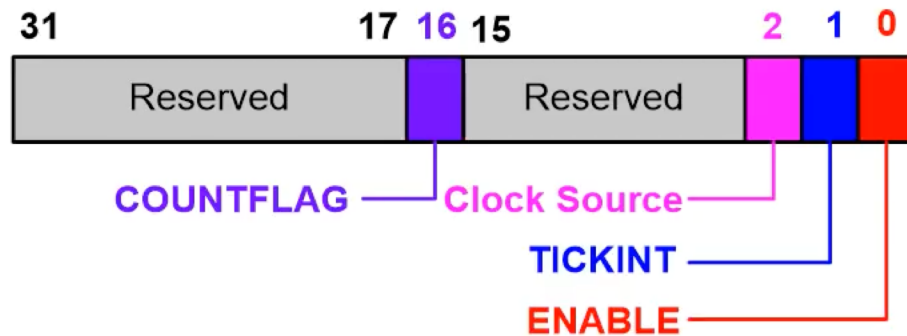
- ▶ Generate **SysTick interrupts** at a fixed time interval



- ▶ Example Usages:
 - ▶ Measuring time elapsed, such as time delay function
 - ▶ Executing tasks periodically, such as periodic polling, and OS CPU scheduling

Registers of System Timer

SysTick control and status register (SysTick_CTRL)



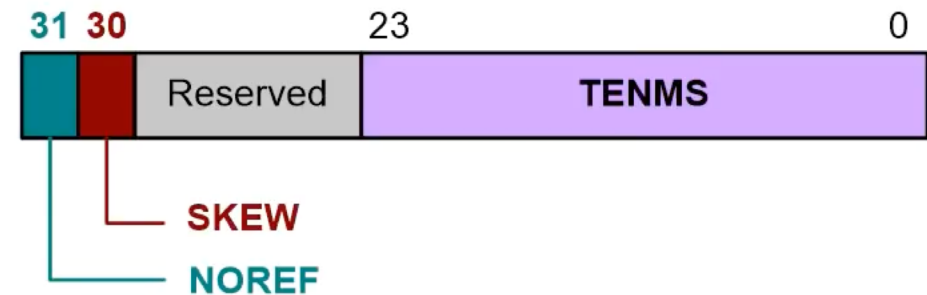
SysTick current value register (SysTick_VAL)



SysTick reload value register (SysTick_LOAD)



SysTick calibration register (SysTick_CALIB)

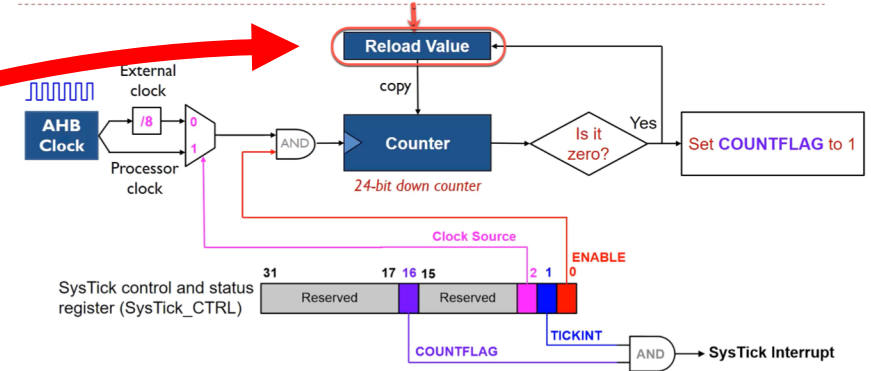


Registers of System Timer

SysTick reload value register (SysTick_LOAD)



Diagram of System Timer (SysTick)



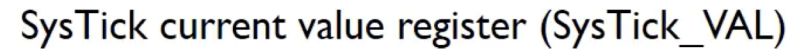
- ▶ 24 bits, maximum value 0x00FF.FFFF (16,777,215)
- ▶ Counter counts down from RELOAD value to 0.
- ▶ Writing RELOAD to 0 disables SysTick, independently of TICKINT
- ▶ Time interval between two SysTick interrupts

$$\text{Interval} = (\text{RELOAD} + 1) \times \text{Source_Clock_Period}$$

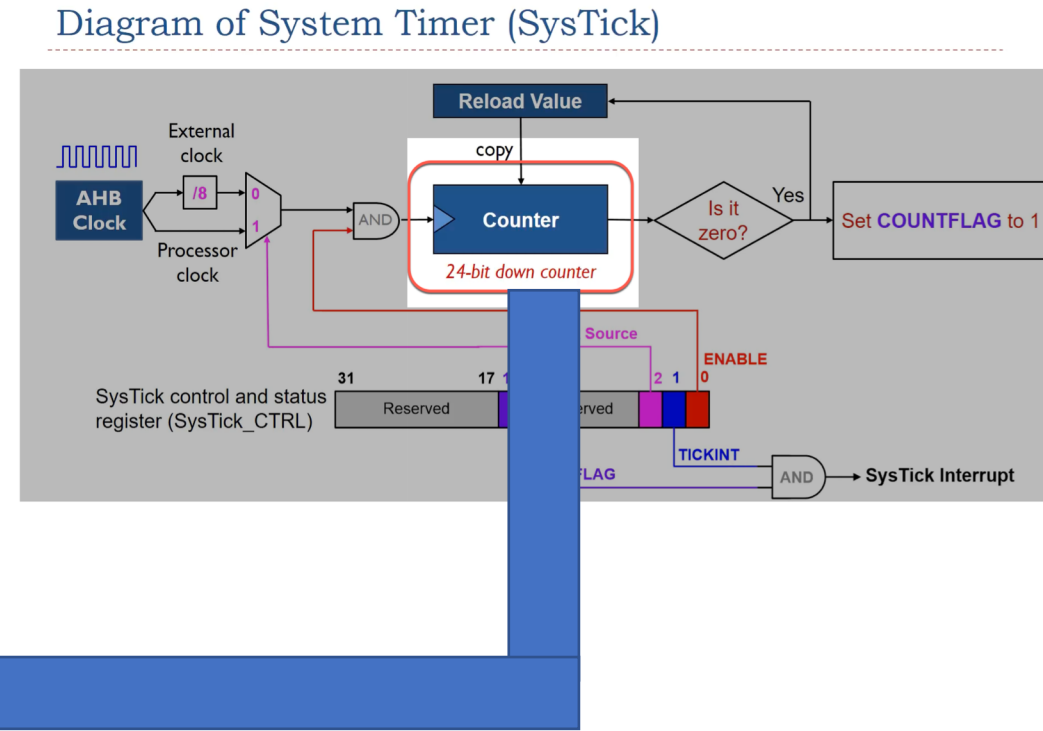
- ▶ If 100 clock periods between two SysTick interrupts

$$\text{RELOAD} = 99$$

Registers of System Timer

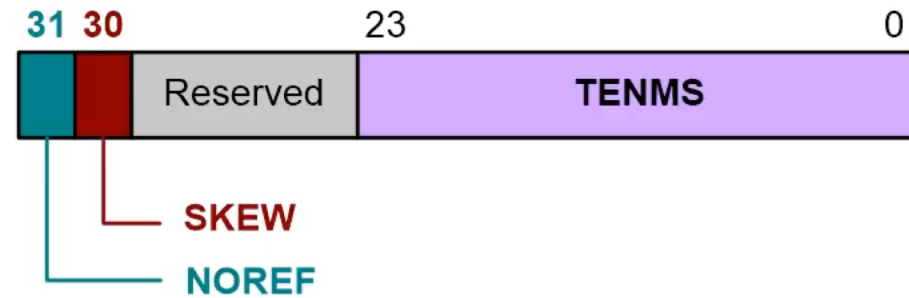


- ▶ Reading it returns the current value of the counter
- ▶ When it transits from 1 to 0, it generates an interrupt
- ▶ Writing to SysTick_VAL clears the counter and COUNTFLAG to zero
 - ▶ Cause the counter to reload on the next timer clock
 - ▶ But, does not trigger an SysTick interrupt
- ▶ It has random value on reset.
 - ▶ Always clear it before enabling the timer



Registers of System Timer

SysTick calibration register (SysTick_CALIB)



- ▶ A read-only register
- ▶ **TENMS (10 ms)** holds the reload value, which will yield a 10ms period
- ▶ May not be implemented or may be defined differently by chip designers

Example Code

```
void SysTick_Initialize (uint32_t ticks) {  
    SysTick->CTRL = 0;           // Disable SysTick  
    SysTick->LOAD = ticks - 1;   // Set reload register  
    // Set interrupt priority of SysTick to least urgency (i.e., largest priority value)  
    NVIC_SetPriority (SysTick_IRQn, (1<<__NVIC_PRIO_BITS) - 1);  
    SysTick->VAL = 0;           // Reset the SysTick counter value  
    // Select processor clock: 1 = processor clock; 0 = external clock  
    SysTick->CTRL |= SysTick_CTRL_CLKSOURCE;  
    // Enables SysTick interrupt, 1 = Enable, 0 = Disable  
    SysTick->CTRL |= SysTick_CTRL_TICKINT;  
    // Enable SysTick  
    SysTick->CTRL |= SysTick_CTRL_ENABLE;  
}
```

Implementing Delay Function

```
volatile int32_t TimeDelay;

int main (void {
    SysTick_Initialize(1000); // Interrupt period = 1000 cycles
    Delay(100);               // Delay 100 ticks
    ...
}

void SysTick_Handler (void) { // SysTick interrupt service routine
    if (TimeDelay > 0)         // Prevent it from being negative
        TimeDelay--;          // TimeDelay is a global volatile variable
}

void Delay (uint32_t nTime) {
    // nTime: specifies the delay time length
    TimeDelay = nTime;         // TimeDelay must be declared as volatile
    while(TimeDelay != 0);     // Busy wait
}
```

Calculating Reload Value

- ▶ Suppose clock source = 80MHz
- ▶ Goal: SysTick Interval = 10ms
- ▶ What is RELOAD value?

$$\begin{aligned} \text{Reload} &= \frac{10 \text{ ms}}{\text{Clock Period}} - 1 \\ &= 10\text{ms} \times \text{Clock Frequency} - 1 \\ &= 10\text{ms} \times 80\text{MHz} - 1 \\ &= 10 \times 10^{-3} \times 80 \times 10^6 - 1 \\ &= 800000 - 1 \\ &= 799999 \end{aligned}$$

