SysTick Timer

By Dr. Min He, David Mahakian

Outline

Clocks in TM4C123 Microcontroller

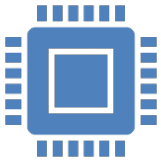
SysTick Timer

- SysTick Registers
- SysTick Initialization

Clocks in TM4C123 Microcontroller

- Execution speed of microcontroller is usually determined by an external crystal oscillator.
  - Internal are less accurate, use less power
  - External is more accurate, uses more power and requires an external component
- Most microcontrollers (including TM4C123) have a phase-lock-loop (PLL) that allows **software** to adjust the execution speed.
  - Higher frequency
    = faster execution speed
    = higher power usage

Clocks in TM4C123 Microcontroller



Clock Sources

Two internal oscillators

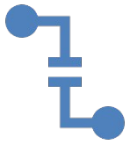
- Precision Internal Oscillator (PIOSC): 16 MHz ± 3%,recalibrate to 1%.
- Low Frequency Internal Oscillator (LFIOSC): 10 kHz to 90 kHz

Two external oscillators (on TM4C123 LaunchPad board)

- 16 MHz Crystal
- 32.768 kHz Crystal (for hibernation module clock source)

TM4C123 has max 80 MHz bus clock frequency

Bus clock frequency = speed at which system components interface to each other (internally component can go faster)



On TM4C, the PLL operates at 400 MHz

Multiple registers are used to divide that value to the intended value.

System Timer: SysTick Timer

- Part of NVIC (Nested Vectored Interrupt Controller)
- Timer/Counter operation
  - **24**-bit counter ***decrements*** at bus clock frequency
    - With 16 MHz bus clock, decrements every 62.5 ns
    - With 80 MHz bus clock, decrements every 12.5 ns
  - Counting is from n 0
    - For a period of m, load (m-1) into RELOAD
    - COUNT Flag set when counts from 1 to 0
    - Counter reset with RELOAD value

SysTick Registers

- **STCTRL** / NVIC_ST_CTRL_R
  [**S**ys**T**ick **C**on**tr**o**l**]
- Flags to configure SysTick timer
  - **ENABLE**: 1 = enabled, 0 = disabled
  - **CLK_SRC** (Clock Source): 0 = use external clock (not implemented on TM4C123), 1 = use system clock
  - **INTEN** (Interrupt Enable): 0 = SysTick interrupt disabled, 1 = SysTick interrupt enabled. SysTick interrupt triggers when

Address	31-24	23-17	16	15-3	2	1	0	Name
$E000E010	0	0	COUNT	0	CLK_SRC	INTEN	ENABLE	NVIC_ST_CTRL_R
$E000E014	0	24-bit RELOAD value						NVIC_ST_RELOAD_R
$E000E018	0	24-bit CURRENT value of SysTick counter						NVIC_ST_CURRENT_R

SysTick Registers

- **STRELOAD** / NVIC_ST_RELOAD_R
  [**S**ys**T**ick **Reload** Value]
  - Value to reload into CURRENT when 0 is reached
  - 24-bit value RELOAD. Timer will hit 0 in RELOAD + 1 clock cycles

When the **CURRENT** value counts down from 1 to 0, the **COUNT** flag is set. On the next clock, the **CURRENT** is loaded with the **RELOAD** value.

Address	31-24	23-17	16	15-3	2	1	0	Name
$E000E010	0	0	COUNT	0	CLK_SRC	INTEN	ENABLE	NVIC_ST_CTRL_R
$E000E014	0	24-bit RELOAD value						NVIC_ST_RELOAD_R
$E000E018	0	24-bit CURRENT value of SysTick counter						NVIC_ST_CURRENT_R

SysTick Registers

- **STCURRENT** / NVIC_ST_CURRENT_R
  [**S**ys**T**ick **Current** Value]
- Current value of timer
- 24-bit value CURRENT.
  - Write any value = clear the register (and clear COUNT bit)

Address	31-24	23-17	16	15-3	2	1	0	Name
$E000E010	0	0	COUNT	0	CLK_SRC	INTEN	ENABLE	NVIC_ST_CTRL_R
$E000E014	0	24-bit RELOAD value						NVIC_ST_RELOAD_R
$E000E018	0	24-bit CURRENT value of SysTick counter						NVIC_ST_CURRENT_R

SysTick Timer Initialization

1. **Clear ENABLE** bit to stop counter
2. **Set RELOAD** to desired value
3. **Clear the counter**
   - Write *any* value to NVIC_ST_CURRENT_R
4. **Set CLK_SRC**
   - **CLK_SRC = 1 (use system clock) required for TM4C123**
5. Set INTEN to enable/disable interrupt
   - 0 = disable, 1 = enable (Set to 0 for now)
6. Set ENABLE bit to start counter

Address	31-24	23-17	16	15-3	2	1	0	Name
$E000E010	0	0	COUNT	0	CLK_SRC	INTEN	ENABLE	NVIC_ST_CTRL_R
$E000E014	0	24-bit RELOAD value						NVIC_ST_RELOAD_R
$E000E018	0	24-bit CURRENT value of SysTick counter						NVIC_ST_CURRENT_R

SysTick Initialization

```
10 #define NVIC_ST_CTRL_R          (*((volatile uint32_t *)0xE000E010))
11 #define NVIC_ST_RELOAD_R        (*((volatile uint32_t *)0xE000E014))
12 #define NVIC_ST_CURRENT_R       (*((volatile uint32_t *)0xE000E018))
13 #define NVIC_ST_CTRL_COUNT      0x00010000  // Count flag
14 #define NVIC_ST_CTRL_CLK_SRC    0x00000004  // Clock Source
15 #define NVIC_ST_CTRL_ENABLE     0x00000001  // Counter mode
16 #define TEN_MICRO_SEC           160000  // reload value for generating10ms time interval for 16 MHz system clock.
17
18 // Initialize SysTick with busy wait running at bus clock.
19 void SysTick_Init(void){
20   NVIC_ST_CTRL_R &= ~NVIC_ST_CTRL_ENABLE;// disable SysTick during setup
21   NVIC_ST_CTRL_R |= NVIC_ST_CTRL_CLK_SRC;
22 }
```

Address	31-24	23-17	16	15-3	2	1	0	Name
$E000E010	0	0	COUNT	0	CLK_SRC	INTEN	ENABLE	NVIC_ST_CTRL_R
$E000E014	0	24-bit RELOAD value						NVIC_ST_RELOAD_R
$E000E018	0	24-bit CURRENT value of SysTick counter						NVIC_ST_CURRENT_R

Generate Time Delay with Systick Busy Waiting

```
24  // Time delay using busy wait.
25  // This assumes 16 MHz system clock.
26  // This function generate a time delay that is multiple of 10ms
27  // Parameter:
28  // delay: specify how many 10ms will be generated.
29  void SysTick_Wait10ms(uint32_t delay){
30    NVIC_ST_RELOAD_R = delay*TEN_MICRO_SEC - 1; // number of counts to wait
31    NVIC_ST_CURRENT_R = 0;                      // any value written to CURRENT clears
32    NVIC_ST_CTRL_R |= NVIC_ST_CTRL_ENABLE;      // enable SysTick timer
33    while ((NVIC_ST_CTRL_R & NVIC_ST_CTRL_COUNT) == 0) { } // wait for COUNT flag to be raised
34    NVIC_ST_CTRL_R &= ~NVIC_ST_CTRL_ENABLE;     // disable SysTick timer
35  }
36
```

Address	31-24	23-17	16	15-3	2	1	0	Name
$E000E010	0	0	COUNT	0	CLK_SRC	INTEN	ENABLE	NVIC_ST_CTRL_R
$E000E014	0	24-bit RELOAD value						NVIC_ST_RELOAD_R
$E000E018	0	24-bit CURRENT value of SysTick counter						NVIC_ST_CURRENT_R

References

- Text book: Chapter 4, Section 4.4
- Example Project: SysTick