**Embedded Systems with ARM Cortex-M Microcontrollers in Assembly Language and C**

# Chapter 16
# General Purpose Timer and PWM
# Exercises ANS

Z. Gu

Fall 2025

Acknowledgement: Lecture slides based on Embedded Systems with ARM Cortex-M Microcontrollers in Assembly Language and C, University of Maine https://web.eece.maine.edu/~zhu/book/

Review

## Summary of Equations

- Timer clock frequency $f_{CK\_CNT}$ vs. CPU Clock Frequency $f_{SOURCE}$

$$f_{CK\_CNT} = \frac{f_{SOURCE}}{PSC + 1}$$

- Timer frequency $f_{Timer}$ with up-counting or down-counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{ARR + 1};\ Timer\ Period = \frac{ARR + 1}{f_{CK\_CNT}} = (ARR + 1) * Clock\ Period$$

- Timer frequency $f_{Timer}$ with center-aligned counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{2 * ARR};\ Timer\ Period = (2 * ARR) * Clock\ Period$$

- PWM duty cycle for Mode 1 (Low-True):

$$Duty\ Cycle = \frac{\mathrm{CCR}}{\mathrm{ARR} + 1}$$

- PWM duty cycle for Mode 2 (High-True):

$$Duty\ Cycle = 1 - \frac{\mathrm{CCR}}{\mathrm{ARR} + 1}$$

# Calculating ARR

- Suppose Timer Clock Frequency = 80MHz
- Goal: Timer Frequency = 100Hz
- What should the ARR value be for 1. up-counting mode; 2. down-counting mode; 3. center-aligned counting mode?

## Calculating ARR ANS

- Suppose Timer Clock Frequency = 80MHz
- Goal: Timer Frequency = 100Hz
- What should the ARR value be for 1. up-counting mode; 2. down-counting mode; 3. center-aligned counting mode?

For up-counting or down-counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{ARR + 1} = \frac{80MHz}{ARR + 1} = 100Hz$$

$$ARR = 799999$$

For center-aligned counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{2 * ARR} = \frac{80MHz}{2 * ARR} = 100Hz$$

$$ARR = 400000$$

But a 16-bit ARR register has value range of [0, 65535], so ARR value of 799999 or 400000 is out of range → cannot generate a 10ms timer from a 80MHz Timer Clock!
Solution: use prescaler (PSC) to reduce Timer Clock Frequency

## Calculating ARR with prescaler

- Suppose CPU Clock Frequency = `80MHz, PSC=79`
- Goal: Timer Frequency = 100Hz
- What should the ARR value be for 1. up-counting mode; 2. down-counting mode; 3. center-aligned counting mode?

# Calculating ARR with prescaler ANS

- Suppose CPU Clock Frequency = 80MHz， prescaler PSC=79
- Goal: Timer Frequency = 100Hz
- What should the ARR value be for 1. up-counting mode; 2. down-counting mode; 3. center-aligned counting mode?

$$Timer\ Clock\ Freq\ f_{CK\_CNT} = \frac{f_{CK_{PSC}}}{PSC+1} = \frac{80MHz}{80} = 1MHz$$

For up-counting or down-counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{ARR+1} = \frac{1MHz}{ARR+1} = 100Hz$$

$$ARR = 9999$$

For center-aligned counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{2*ARR} = \frac{1MHz}{2*ARR} = 100Hz$$

$$ARR = 5000$$

With prescaler, the ARR value of 9999 or 5000 is now within the range of 16-bit ARR register.

## Calculating the ARR

- Suppose a 16-bit timer has the following settings
  - CPU clock frequency is 4 MHz.
  - Prescaler PSC = 39
  - Counting direction: center-aligned counting
  - Desired timer frequency = 100 Hz
- Calculate the ARR.

## Calculating the ARR ANS

- Suppose a 16-bit timer has the following settings
  - CPU clock frequency is 4 MHz.
  - Prescaler PSC = 39
  - Counting direction: center-aligned counting
  - Desired timer frequency = 100 Hz
- Calculate the ARR.

$$Timer\ Clock\ Freq\ f_{CK\_CNT} = \frac{f_{CK\_PSC}}{PSC+1} = \frac{4\ \text{MHz}}{40} = 0.1\ \text{MHz}$$

$$Timer\ Freq\ f_{Timer} = \frac{f_{CK\_CNT}}{2 * ARR} = \frac{0.1\ MHz}{2 * \text{ARR}} = 100Hz$$

$$ARR = 500$$

## Clock Frequency and Timer Frequency

- Suppose all registers are 16 bits. CPU clock frequency is 400 MHz.
- What is the maximum and minimum Clock Frequency, and Timer Frequency, assuming up-counting mode?

## Clock Frequency and Timer Frequency ANS

- Suppose all registers are 16 bits. CPU clock frequency is 400 MHz.
- What is the maximum and minimum Clock Frequency, and Timer Frequency, assuming up-counting mode?
- PSC, ARR are both in the range [0, $2^{16}$-1=65535].

$$f_{CK\_CNT} = \frac{f_{CK\_PSC}}{PSC+1} = \frac{400\text{ MHz}}{\text{PSC+1}} \in \left[\frac{400\text{MHz}}{65536}, \frac{400\text{MHz}}{1}\right] = [6.1\text{ KHz}, 400\text{MHz}]$$

$$f_{Timer} = \frac{f_{CK\_CNT}}{ARR+1} \in \left[\frac{6.1KHz}{65536}, \frac{400MHz}{1}\right] = [0.09Hz, 400\ MHz]$$

- (A realistic Timer Frequency should not be higher than 1 KHz. Hardware can generate high frequency timer interrupts, but software latency and Interrupt Service Routine (ISR) overhead are the real constraints.)

## PWM duty cycle

- Suppose a 16-bit timer has the following settings
  - CPU Clock frequency is 16 MHz.
  - Prescaler PSC = 159
  - ARR = 1999
  - CCR = 499
  - Counting direction: up-counting
  - Output is set as PWM Mode 2 (High True).
- Calculate the timer frequency and PWM duty cycle.

# PWM duty cycle ANS

- Suppose a 16-bit timer has the following setting
  - CPU clock frequency is 16 MHz.
  - Prescaler PSC = 159
  - ARR = 1999
  - CCR = 499
  - Counting direction: up-counting
  - Output is set as PWM Mode 2 (High True).
- Calculate the timer frequency and PWM duty cycle.
- ANS:

$$Timer\ Clock\ Freq\ f_{CK\_CNT} = \frac{f_{CK\_PSC}}{PSC+1} = \frac{16\ MHz}{159+1} = 0.1MHz$$

$$Timer\ Freq\ f_{Timer} = \frac{f_{CK\_CNT}}{ARR+1} = \frac{0.1\ MHz}{1999+1} = 50Hz$$

$$Duty\ Cycle = 1 - \frac{CCR}{ARR+1} = 1 - \frac{499}{1999+1} = \frac{1501}{2000} = 0.7505$$

PWM output is high on when the counter is 499, 500, 501, …, 1999, a total of 1501 cycles. (This statement is not required in the exam.)

## Interrupt (NOT COVERED)

- Suppose register *i* (*i* ≤ 12) is initialized to have a value of *i* (e.g. r0 = 0, r1= 1, r2 = 2, r3 = 3, etc.). Assume the main stack (MSP) is used. (Recall: in the interrupt handler, if LR = 0xFFFFFFF9, then the main stack (MSP) is used. If LR = 0xFFFFFFFD, then the process stack (PSP) is used. ) The program status register (PSR) = 0x00000020, PC = 0x08000020, and LR = 0x20008020, when the interrupt occurs.
- (1) Show the stack content immediately before the PUSH instruction runs. Suppose the stack pointer SP, *i.e.* MSP in this case, was 0x20000600 immediately before the system timer interrupt occurs
- (2) What are the values of these registers (R0-R12, LR, SP, PC, and PSR) immediately after the interrupt exits?

```
SysTick_Handler PROC
     EXPORT SysTick_Handler
     PUSH {lr, r0, r7}
     ADD  r0, r0, #1
     ADD  r1, r1, #1
     ADD  r2, r2, #1
     ADD  r3, r3, #1
     ADD  r4, r4, #1
     ADD  r5, r5, #1
     ADD  r6, r6, #1
     ADD  r7, r7, #1
     ADD  r8, r8, #1
     ADD  r9, r9, #1
     ADD  r10, r10, #1
     ADD  r11, r11, #1
     ADD  r12, r12, #1
     POP  {r0, r7, pc}
     ENDP
```

## Interrupt ANS (NOT COVERED)

- (1)

| | |
|---|---|
| 0x20000600 | |
| 0x200005FC | 0x00000020 (PSR) |
| 0x200005F8 | 0x08000020 (PC) |
| 0x200005F4 | 0x20008020 (LR) |
| 0x200005F0 | 12 (R12) |
| 0x200005EC | 3 (R3) |
| 0x200005E8 | 2 (R2) |
| 0x200005E4 | 1 (R1) |
| 0x200005E0 | 0 (R0) |
| 0x200005DC | |