

# Timers

*ELEC 330*

*Digital Systems Engineering*

*Dr. Ron Hayne*

*Images Courtesy of Ramesh Gaonkar and Delmar Learning*

# Hardware Counters and Timers

- ◆ Counter (register)
  - Can be decremented or incremented per clock cycle
- ◆ Time calculation
  - Find difference between beginning count and last count
  - Multiply the count difference by the clock period

# Hardware Counters and Timers

- ◆ Counter (register)
  - Replace the clock with a signal from an event
  - Count is incremented (or decremented)
  - Total number of events can be counted

# Types of Counters

- ◆ Up-counter
  - Counter is incremented at every clock cycle
  - When count reaches the maximum count, a flag is set
  - Counter can be reset to zero or to the initial value
- ◆ Down-counter
  - Counter is decremented at every clock cycle
  - When count reaches zero, a flag is set
  - Counter can be reset to the maximum or the initial value
- ◆ Free-running counter
  - Counter runs continuously and only readable
  - When it reaches the maximum count, a flag is set

# Timer Applications

- ◆ Time delay
- ◆ Pulse wave generation
- ◆ Pulse width or frequency measurement
- ◆ Timer as an event counter

# Capture, Compare, and PWM (CCP) Modules

- ◆ CCP modules are common in recent MCUs
  - Registers specially designed to perform the following functions (in conjunction with timers)
    - Capture: The CCP pin can be set as an input to record the arrival time of a pulse.
    - Compare: The CCP pin is set as an output, and at a given count, it can be driven low, high, or toggled.
    - Pulse width modulation (PWM): The CCP pin is set as an output and the duty cycle of a pulse can be varied.

# Pulse Width Modulation

- ◆ Duty cycle
  - Percentage ratio of on time of a pulse to its period
- ◆ Changing of the duty cycle is defined as PWM
  - CCP pin is set as an output
  - Count for period and duty cycle loaded into CCP registers
  - Varying the duty cycle generates PWM

# PIC18 Timers

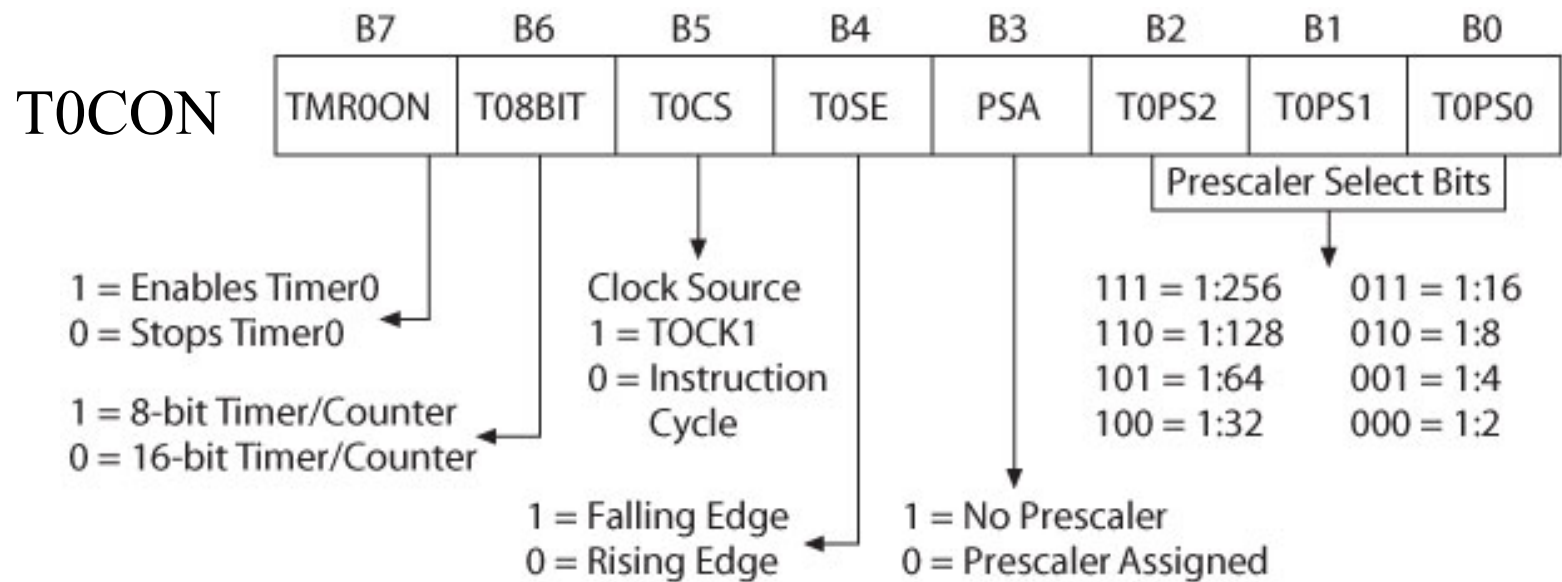
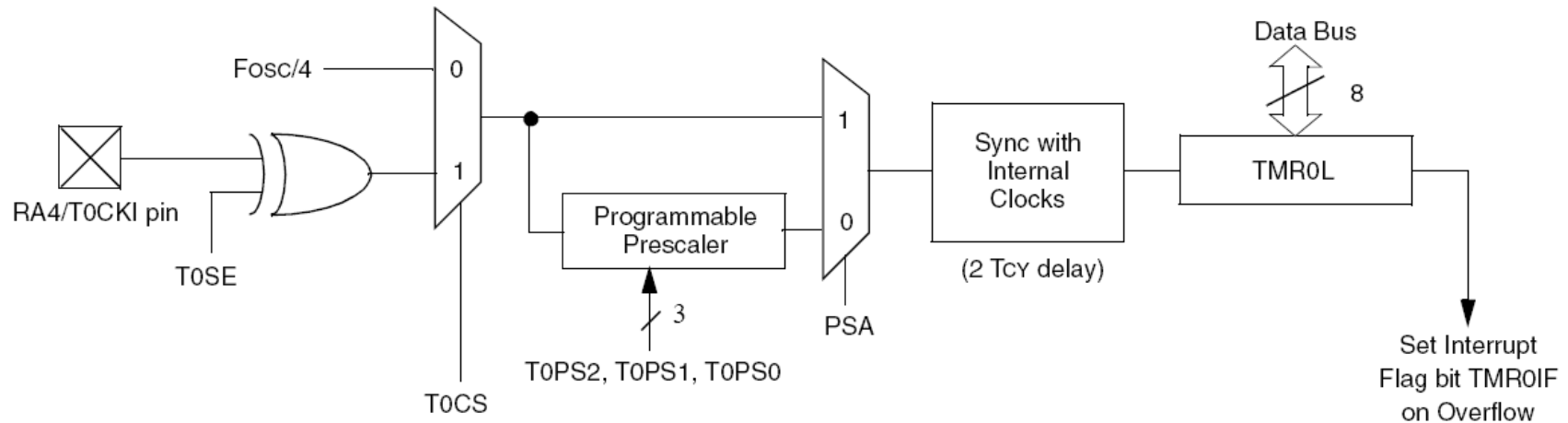
- ◆ PIC18 timers
  - All up-counters
  - 8-bit and 16-bit
- ◆ Timer0
  - 8-bit or 16-bit timer
- ◆ Timer1 (and Timer3)
  - 16-bit timers
- ◆ Timer2 (and Timer4)
  - 8-bit timer
- ◆ SFRs
  - T0CON-T2CON



# Timer0

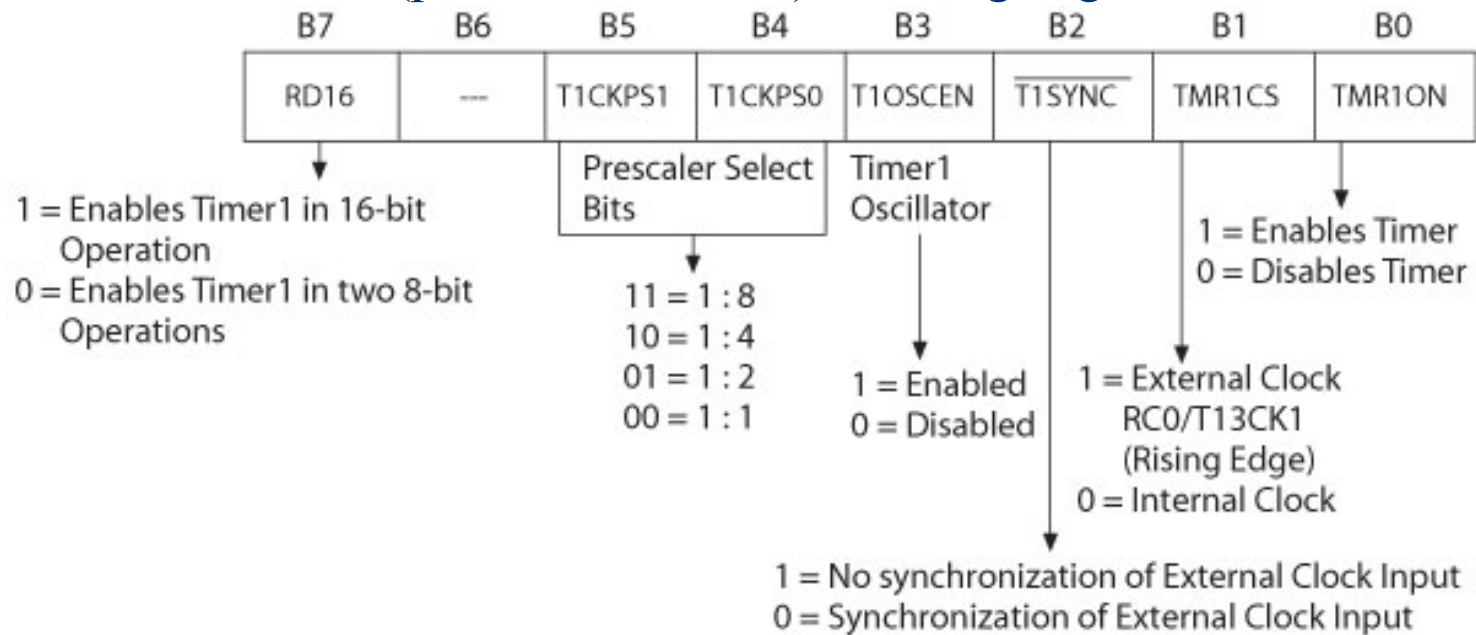
- ◆ 8-bit or 16-bit timer
  - Readable and writable
  - Parameters in T0CON register
  - Eight pre-scale values (Bit2-Bit0)
  - Clock source (Bit5)
    - Internal (instruction cycle)
    - External clock connected to pin RA4/T0CK1
      - ◆ Rising edge or falling edge (Bit4)
  - Generates an interrupt or sets a flag when it overflows
    - TMR0IF
    - Flag must be cleared to start the timer again

# Timer0



# Timer1

- ◆ 16-bit counter/timer
  - Two 8-bit registers (TMR1H and TMR1L)
  - Four prescale values (Bit5-Bit4)
  - Clock source (Bit1)
    - Internal (instruction cycle)
    - External (pin RC0/T13CK1) on rising edge

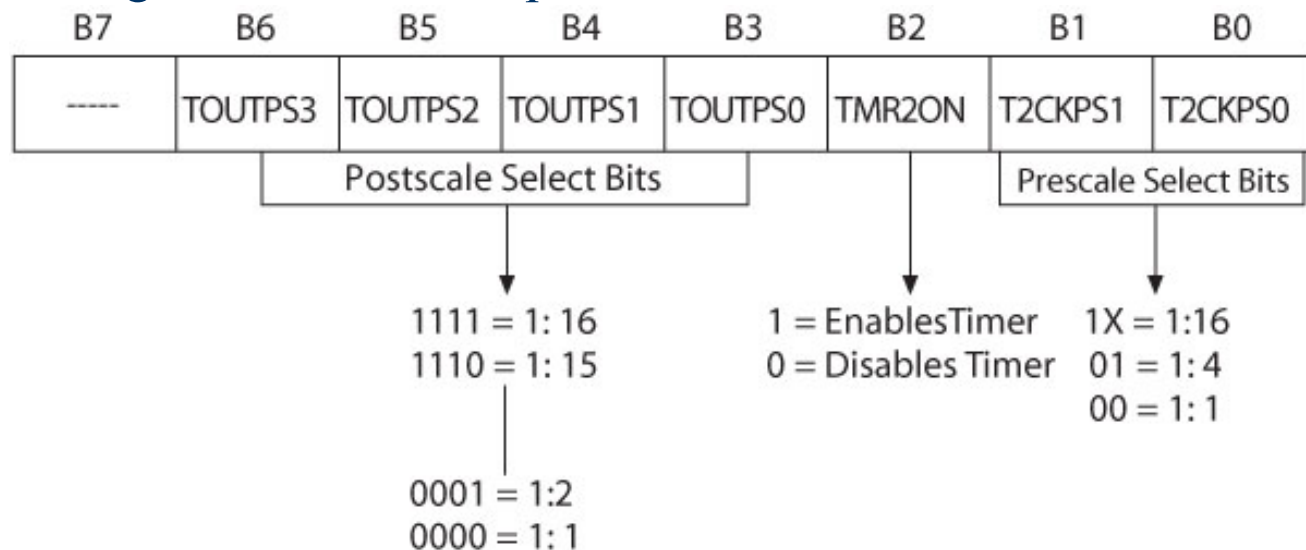


# Timer1

- ◆ Interrupt
  - Generates an interrupt or sets a flag when it overflows
    - TMR1IF
  - Flag must be cleared to start the timer again
- ◆ Resetting Timer1 using CCP module
  - CCP1 in the Compare mode
  - Timer1 and CCP1 compared at every cycle
  - When a match is found, Timer1 is reset

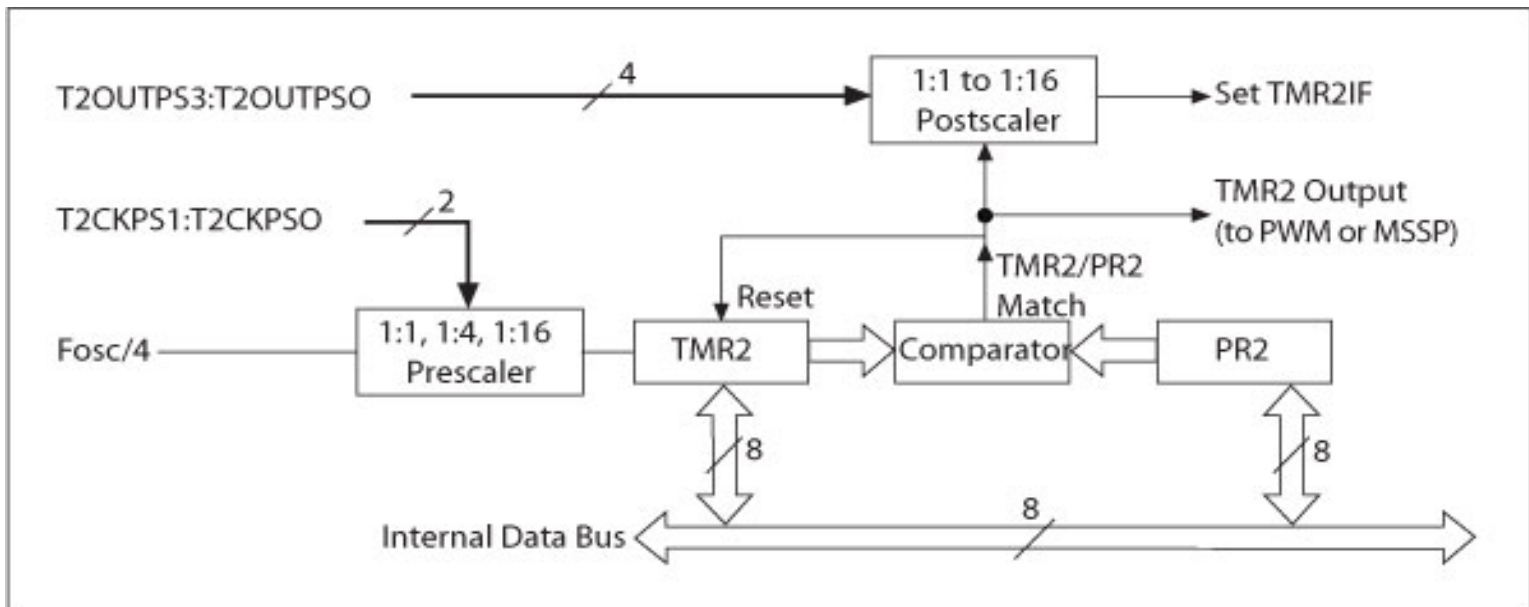
# Timer2

- ◆ 8-bit timer (TMR2)
  - 8-bit period register (PR2)
  - TMR2 and PR2 are readable and writable
  - Three prescale values (Bit1-Bit0)
  - 16 postscale values (Bit6-Bit3)
  - Flag (TMR2IF) is set when TMR2 matches PR2
    - Can generate an interrupt



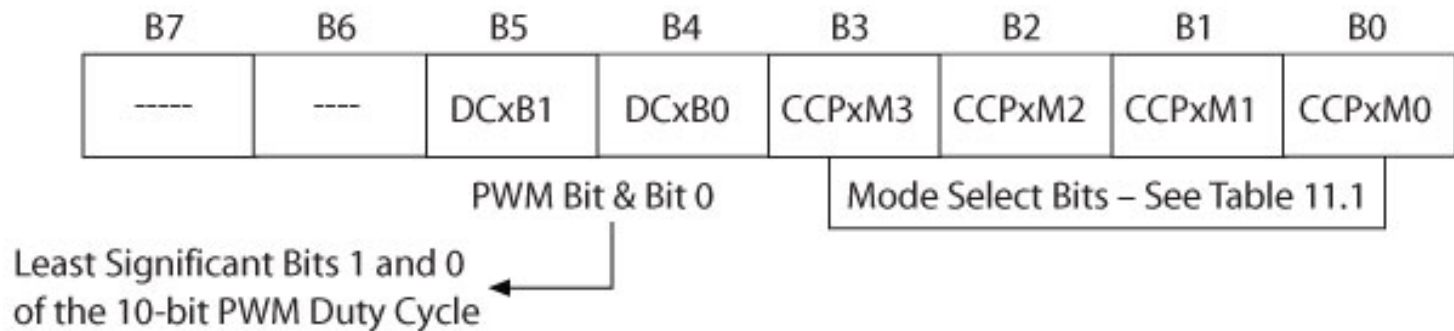
# Timer2

- ◆ Timer2 operation
  - 8-bit number is loaded in PR2
  - When TMR2 and PR2 match
    - Output pulse is generated and the timer is reset
  - Output pulse goes through postscaler
    - Sets the flag TMR2IF



# CCP Modules

- ◆ Capture, Compare, and Pulse Width Modulation (PWM)
- ◆ CCPR1H (high) and CCPR1L (low)
  - 16-bit Capture register
  - 16-bit Compare register
  - Duty-cycle PWM register
- ◆ Timer1 used as clock for Capture and Compare
- ◆ Timer2 used as clock for PWM



# CCP in the Capture Mode

- ◆ CCPR1 captures the 16-bit value of Timer1
  - When an event occurs on pin RC2/CCP1
- ◆ Interrupt request flag bit CCP1IF is set
  - Must be cleared for the next operation
- ◆ To capture an event
  - Set up pin RC2/CCP1 of PORTC as the input
  - Initialize Timer1
    - T1CON register
  - Initialize CCP1
    - CCP1CON register



# CCP in the Compare Mode

- ◆ CCPR1 constantly compared with TMR1
- ◆ When a match occurs
  - Pin RC2/CCP1 on PORTC
    - Driven high, low, or toggled
    - Interrupt flag bit CCP1IF is set
- ◆ To set up CCP1 in Compare mode
  - Set up pin RC2/CCP1 of PORTC as output
  - Initialize Timer1 and CCP1
  - Clear the flag CCP1IF

# PWM Mode

- ◆ CCP module with Timer2
  - Output a pulse wave form for a given frequency/duty cycle
- ◆ Duty cycle
  - CCPR1 register
- ◆ Period
  - PR2 register
- ◆ When TMR2 is equal to PR2
  - TMR2 is cleared
  - Pin RC2/CCP1 of PORTC is set high
  - PWM duty-cycle byte loaded into CCPR1