dsPIC30F Peripheral Module

Input Capture Module

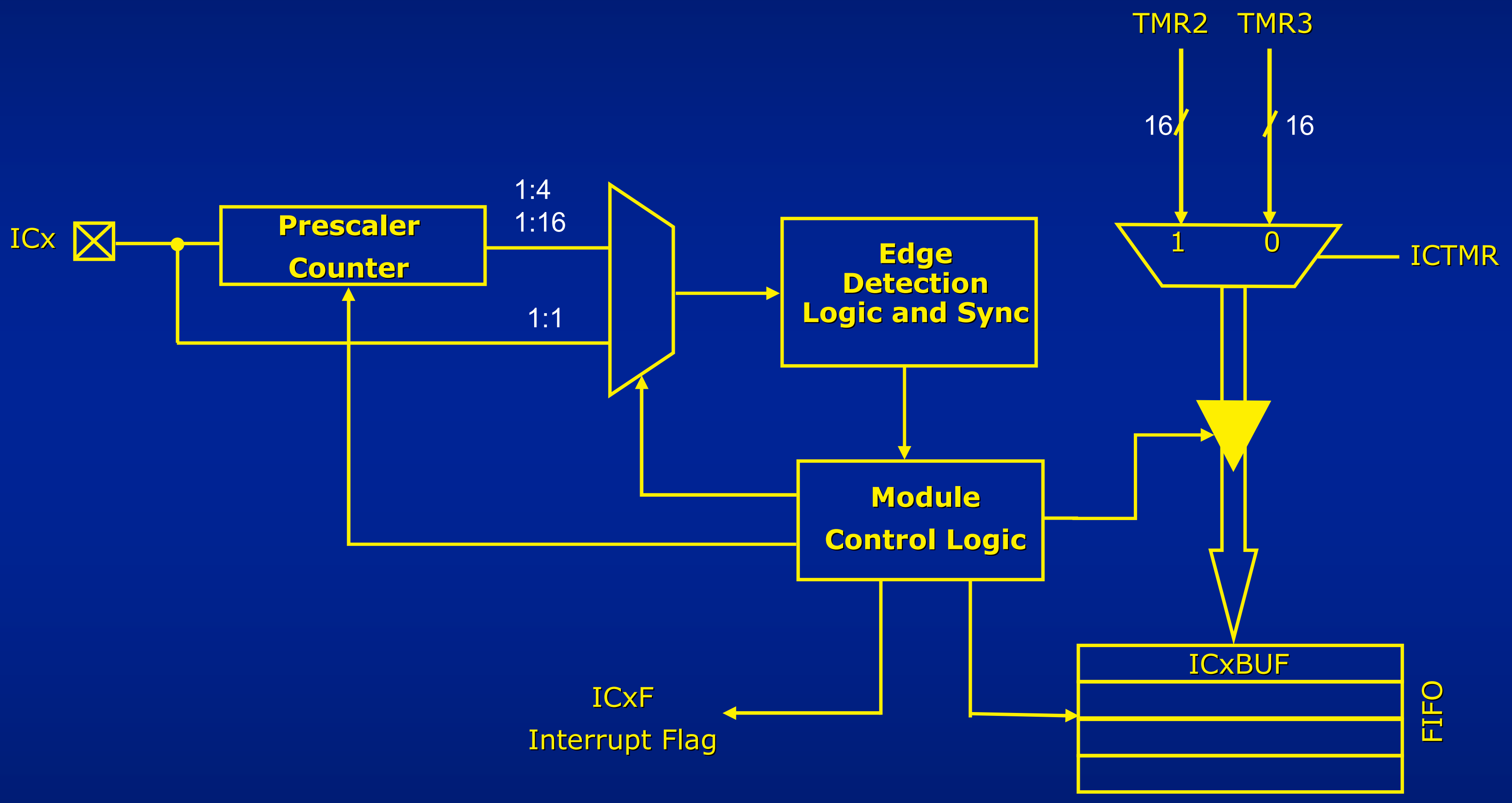
Input Capture

- Up to Eight Input Capture Channels
- Captures 16-bit timer value
  - At 30 MIPS resolution = 33 ns (Tcy)
  - At 30 MIPS with 16x pre-scale = 2.1 ns
- 4 deep buffer for each capture input
  - Interrupt on 1- 4 capture events
  - FIFO buffer overflow status
  - FIFO buffer empty status

Input Capture

- Timer 2 or Timer 3 as timebase
- Capture on:
  - ↑ edge at ICx pin
  - ↓ edge at ICx pin
  - Every 4th ↑ edge at ICx pin
  - Every 16th ↑ edge at ICx pin
  - ↑ edge and ↓ edge
    - Very useful for pulse and frequency measurement
    - Interface to hall sensors for rotor position feedback
    - Autobaud support for UART communications

Input Capture Block Diagram



Input Capture Control Register

- ICxCON
  - ❖ ICSIDL ⇒ Stop in Idle mode
  - ❖ ICTMR ⇒ Time Base Select for Input Capture
  - ❖ ICI<1:0> ⇒ Capture events per Interrupt select
  - ❖ ICOV ⇒ FIFO buffer overflow status
  - ❖ ICBNE ⇒ FIFO buffer Not Empty status
  - ❖ ICM<2:0> ⇒ Input Capture mode select

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
-	-	ICSIDL	-	-	-	-	-
bit15	14	13	12	11	10	9	bit8

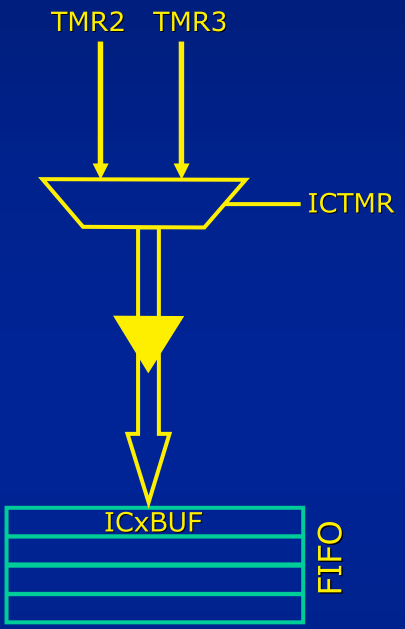
R/W-0	R/W-0	R/W-0	R-0, HC	R-0, HC	R/W-0	R/W-0	R/W-0
ICTMR	ICI<1:0>		ICOV	ICBNE	ICM<2:0>		
bit7	6	5	4	3	2	1	bit0

Input Capture Mode Select

- ICM<2:0> : Input Capture Mode Select bits
  - 000 : Input Capture turned off
  - 001 : Capture every edge change
  - 010 : Capture every falling edge is coming
  - 011 : Capture every rising edge is coming
  - 100 : Capture every 4th rising edge
  - 101 : Capture every 16th rising edge
  - 110 : Unused (Disable)
  - 111 : Input Capture function as interrupt only, when device is in the SLEEP or IDLE mode (rising edge detect only)

Input Capture FIFO

- 4 deep buffer for each capture input
- Interrupt on 1-4 capture events
- FIFO buffer overflow status
- FIFO buffer empty status



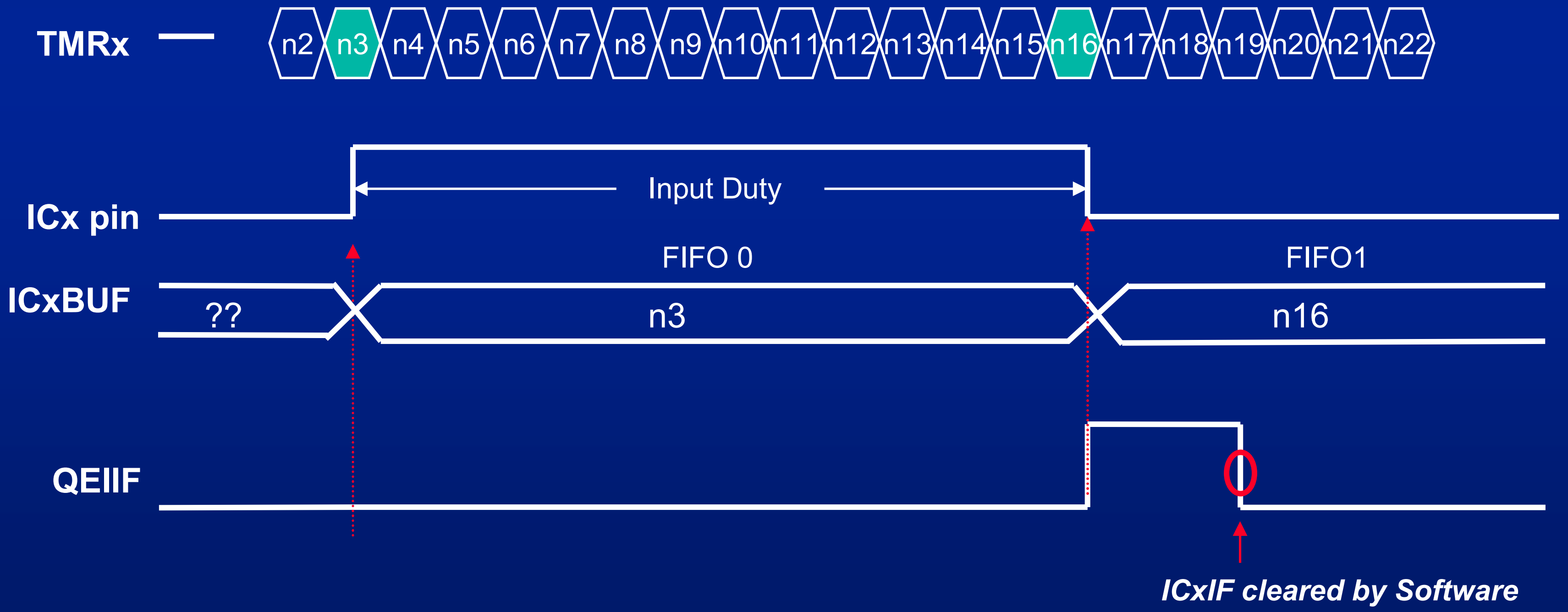
ICxCON SFR

-	-	ICSIDL	-	-	-	-	-
bit15	14	13	12	11	10	9	bit8

ICTMR	ICI<1:0>		ICOV	ICBNE	ICM<2:0>		
bit7	6	5	4	3	2	1	bit0

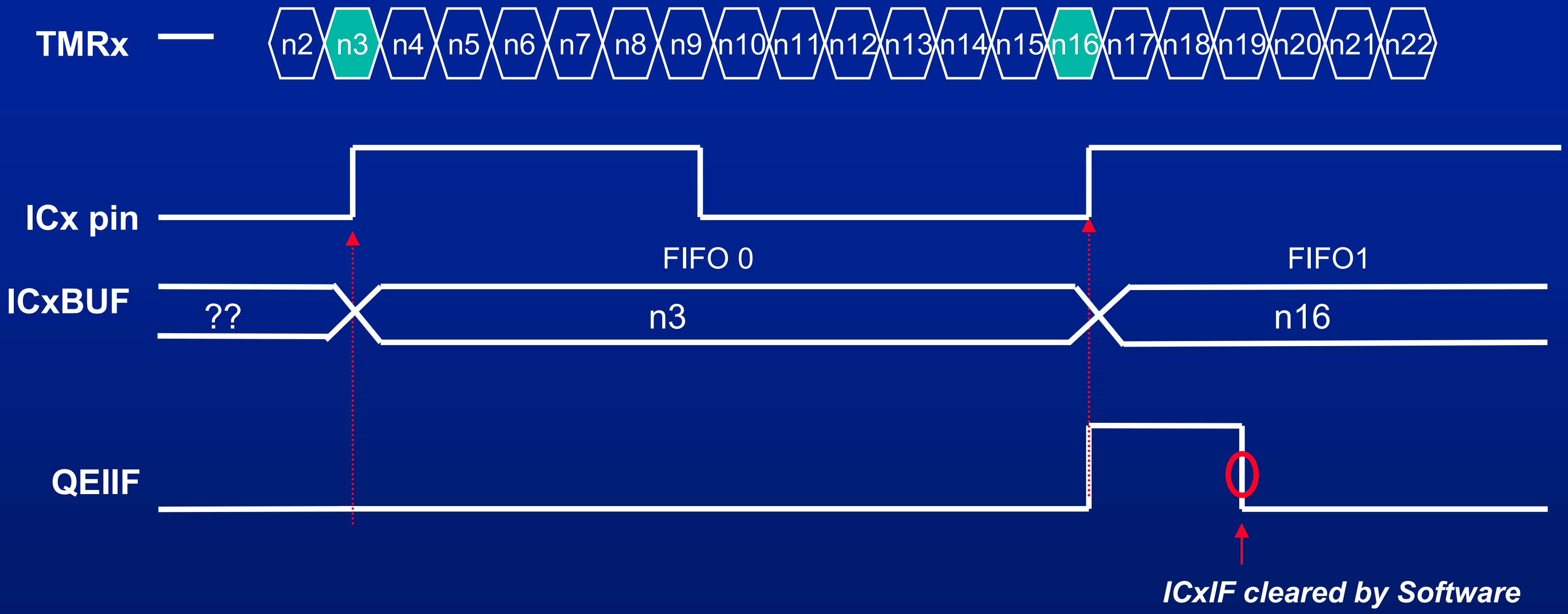
Edge Detection Mode
Calculate the Input Duty

- ICM<2:0> = 001, for capture every edge change
- ICI<1:0> = 01, for Interrupt on every second event
- Input Duty = (n16 – n3) * TMRx input cycle



Edge Detection Mode
Calculate the Input Frequecncy

- ICM<2:0> = 011, for capture every rising edge
- ICI<1:0> = 01, for Interrupt on every second event
- Input Freq. = 1 / [(n16 – n3) * TMRx input cycle]



APP020 Plus DIP SW 的設定 (由左自右算起)

- DSW1 : 1 & 2 Closed，選 PGD/PGD 爲除錯腳位
- DSW3 : 1 & 2 Closed，選 VR 類比輸入(本練習未使用)
- DSW4 : 1 ~ 4 Closed，使用 QEI 信號輸出
- DSW2 : 1 & 2 Closed 選 UART1，3 & 4 選 UART2
- DSW5 : 1 & 2 Closed，LCD R_S & R_W 用 RF0 及 RF1；3 & 4 Closed LCD 選 RF4 & RF5 來推動。
- DSW6 : 1 ~ 6 都在 Off 位置。1 & 2 爲 I2C 腳位共用 PGC & PGD。(也就是說使用 I2C 時就無法使用 PGC & PGD 來除錯。)

Input Capture Lab1 (APP020 Plus)

- PIC16F684 的 QEA 的輸出直接接到 Input Capture 7 (IC7/RB4/AN4) 作爲頻率的測量的訊號源
- Input Capture 設定爲 2 FIFO 中斷模式來補抓輸入的訊號
- IC7 使用 Timer3 作爲計時的參考時脈
- Timer1 提供 200mS 的計時延遲作爲更新 LCD 的頻率與週期的顯示
- 改變 VR3 會變更 QEA 的輸出頻率，可以使用示波器相互比對所量到的頻率準確度；QEA 的頻率範圍 (0.8K ~ 3.8KHz)

Freq. = 1238 Hz
Period= 807 uS

LCD Module display items

Lab1 的提示
抓取輸入脈波的時間

- 因爲設定是兩次的下降緣輸入才會中斷，所以要讀取兩筆 FIFO 資料

- Input Capture 7 (IC7) Interrupt Function

```
void _ISR _IC7Interrupt(void)                          // Interrupt Function for the IC7
{
        ReadCapture7( &timer_edge[0]);                 // Read Timer count from FIFO 0
        ReadCapture7( &timer_edge[1]);                 // Read Timer count from FIFO 1
        Int_flag = 1;                                  // Set IC7 process Flag
        IFS1bits.IC7IF = 0;                            // Clear the IC7 interrupt flag
}
```

Lab1 的提示
計算輸入的頻率

- 需要先檢查兩的資料 FIFO 0 和 FIFO 1 之間是否有溢位的發生再做適度的差值計算

```
while (!Int_flag);                        // Get two input signal edge
DisableIntIC7;                            // Disable Interrupt of Capture 7
Int_flag = 0;

/* calculate time count between two capture events */

If ( timer_edge[1] >= timer_edge[0])      //  沒有溢位發生，新值減舊值即可
        period = timer_edge[1] - timer_edge[0];
else                                      // 有溢位發生，0xFFFF- 舊值後再加新值
        period = 65536 + timer_edge[1] - timer_edge[0];

frequency = FCY / period;      // 計算頻率，14745600Hz / Counter 數 (得到 Hz)
period = period / 14.745600 ;  // 計算週期 , 計數直乘以 一個計數單位的 Tcy
                               //  Tcy = 0.06782uS
```