ECE 3120: Computer Systems
Chapter 8: Input Capture Function

Manjeera Jeedigunta
http://blogs.cae.tntech.edu/msjeedigun21
Email: msjeedigun21@tntech.edu
Tel: 931-372-6181, Prescott Hall 120

Input Capture Functions (1 of 2)

- Physical time is often represented by the contents of the main timer.
- The occurrence of an event is represented by a signal edge (rising or falling edge).
- The time when an event occurs can be recorded by latching the count of the main timer when a signal edge arrives as illustrated in Figure 8.4.
- The HCS12 has eight input capture channels. Each channel has a 16-bit capture register, an input pin, edge-detection logic, and interrupt generation logic.
- Input capture channels share most of the circuit with output compare functions. For this reason, they cannot be enabled simultaneously.

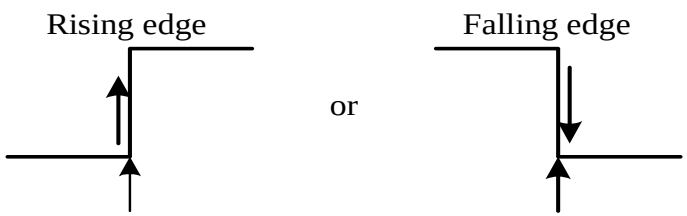


Figure 8.4 Events represented by signal edges

Input Capture Functions (2 of 2)

- The selection of input capture and output compare is done by programming the TIOS register.
- The contents of the TIOS register are shown in Figure 8.5. Setting a bit select the output compare function. Otherwise, the input capture function is selected.

	7	6	5	4	3	2	1	0
	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
value after reset	0	0	0	0	0	0	0	0

IOS[7:0] -- Input capture or output compare channel configuration bits
0 = The corresponding channel acts as an input capture
1 = The corresponding channel acts as an output compare

Figure 8.5 Timer input capture/output compare select register (TIOS)

- The following instruction will enable the output compare channels 7...4 and input capture channel 3…0:

```
movb    #$F0,TIOS
```

Timer Port Pins

- Each port pin can be used as a general I/O pin when timer function is not selected.
- Pin 7 can be used as input capture 7, output compare 7 action, and pulse accumulator input.
- When a timer port pin is used as a general I/O pin, its direction is configured by the DDRT register.

Timer Control Register 3 and 4

- The signal edge to be captured is selected by TCTL3 and TCTL4.
- The edge to be captured is selected by two bits. The user can choose to capture the rising edge, falling edge, or both edges.

	7	6	5	4	3	2	1	0
	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
value after reset	0	0	0	0	0	0	0	0

(a) Timer control register 3 (TCTL3)

	7	6	5	4	3	2	1	0
	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
	0	0	0	0	0	0	0	0

(b) Timer control register 4 (TCTL4)

EDGnB EDGnA -- Edge configuration

0	0 : Capture disabled
0	1 : Capture on rising edges only
1	0 : Capture on falling edges only
1	1 : Capture on both edges

Figure 8.5 Timer control register 3 and 4

Timer Interrupt Enable Register (TIE)

- The arrival of a signal edge may optionally generate an interrupt to the CPU.
- The enabling of the interrupt is controlled by the Timer Interrupt Enable Register.

	7	6	5	4	3	2	1	0
	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
reset:	0	0	0	0	0	0	0	0

C7I-C0I: input capture/output compare interrupt enable bits
0 = interrupt disabled
1 = interrupt enabled

Figure 8.7 Timer interrupt enable register (TIE)

Timer Interrupt Flag 1 Register (TFLG1)

- Whenever a signal edge arrives, the associated timer interrupt flag will be set to 1.

	7	6	5	4	3	2	1	0
	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
reset:	0	0	0	0	0	0	0	0

CnF: input capture/output compare interrupt flag bits
0 = interrupt condition has not occurred
1 = interrupt condition has occurred

Figure 8.8 Timer interrupt flag register 1 (TFLG1)

How to Clear a Timer Flag Bit

- In normal mode, write a 1 to the flag bit to be cleared.
- Method 1
  - Use the BCLR instruction with a 0 at the bit position (s) corresponding to the flag (s) to be cleared. For example,

    BCLR TFLG1, $FE

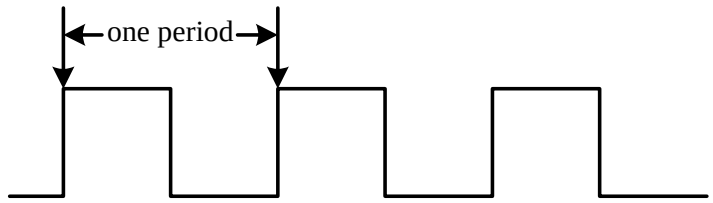
    will clear the C0F flag.
- Method 2
  - Use the movb instruction with a 1 at the bit position (s) corresponding to the flag (s) to be cleared. For example,

    movb #$01,TFLG1

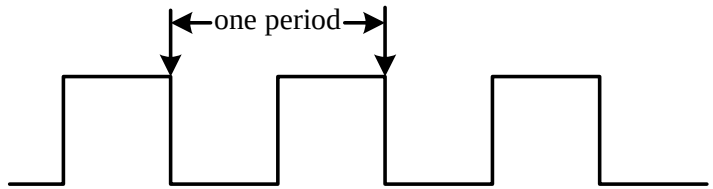
    will clear the C0F flag.
- When fast timer flag clear function is enabled, see Figure 8.1.

Applications of Input Capture Function

- Event arrival time recording
- Period measurement: need to capture the main timer values corresponding to two consecutive rising or falling edges



(a) Capture two rising edges



(b) Capture two falling edges

Figure 8.9 Period measurement by capturing two consecutive edges

- Pulse width measurement: need to capture the rising and falling edges

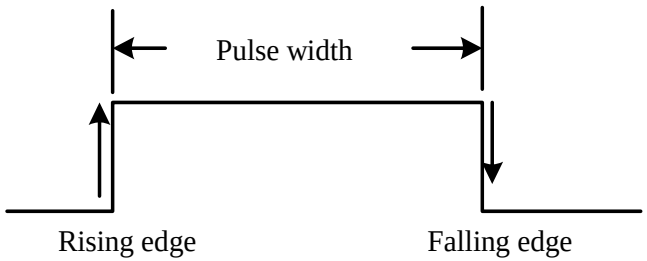


Figure 8.10 Pulse-width measurement using input capture

Input Capture

- Interrupt generation: Each input capture function can be used as an edge-sensitive interrupt source.
- Event counting: count the number of signal edges arrived during a period

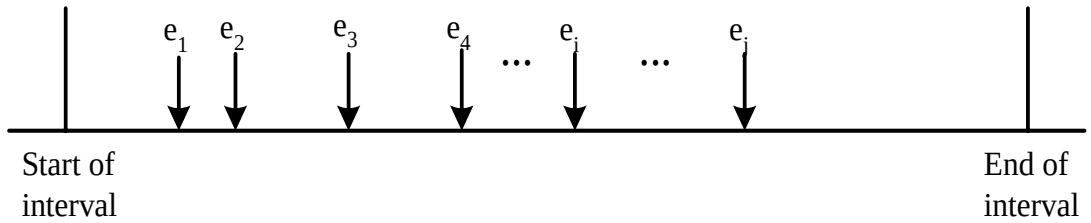


Figure 8.11 Using an input-capture function for event counting

- Time reference: often used in conjunction with an output compare function

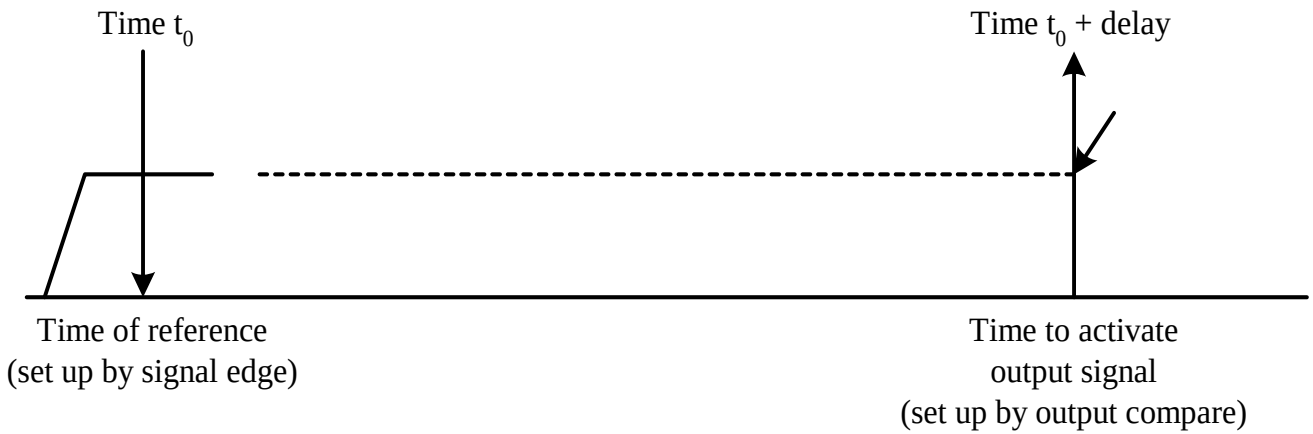
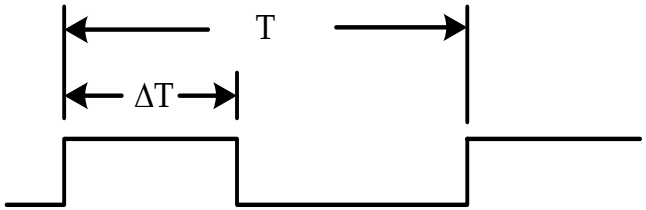


Figure 8.12 A time reference application

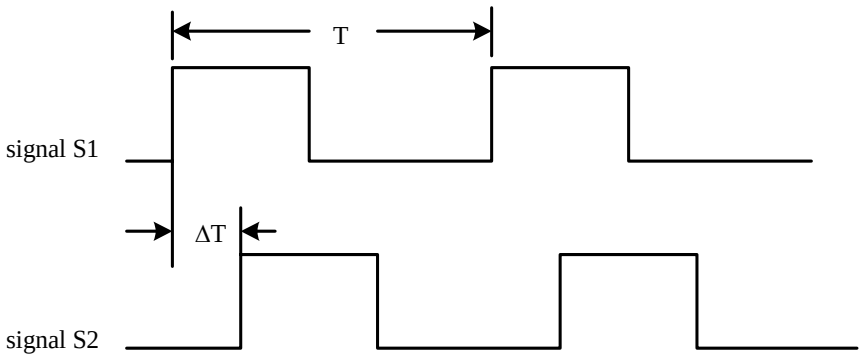
Duty Cycle Measurement



$$\text{duty cycle} = \frac{\Delta T}{T} * 100\%$$

Figure 8.13 Definition of duty cycle

Phase Difference Measurement



$$\text{phase difference} = \frac{\Delta T}{T} * 360^{o}$$

Figure 8.14 Phase difference definition for two signals

Period Measurement (1 of 2)

- Example 8.2 Use the IC0 to measure the period of an unknown signal. The period is known to be shorter than 128 ms. Assume that the E clock frequency is 24 MHz. Use the number of clock cycles as the unit of the period.
- Since the input-capture register is 16-bit, the longest period of the signal that can be measured with the prescaler to TCNT set to 1 is:

  $2^{16} \div 24$ MHz = 2.73 ms.

- To measure a period that is equal to 128 ms, we have two options:
  - Set the prescale factor to 1 and keep track of the number of times the timer counter overflows.
  - Set the prescale factor to 64 and do not keep track of the number of times the timer counter overflows.
- We will set the prescale factor to TCNT to 64. The logic flow for measuring the signal period is shown in Figure 8.16.

Period Measurement (2 of 2)

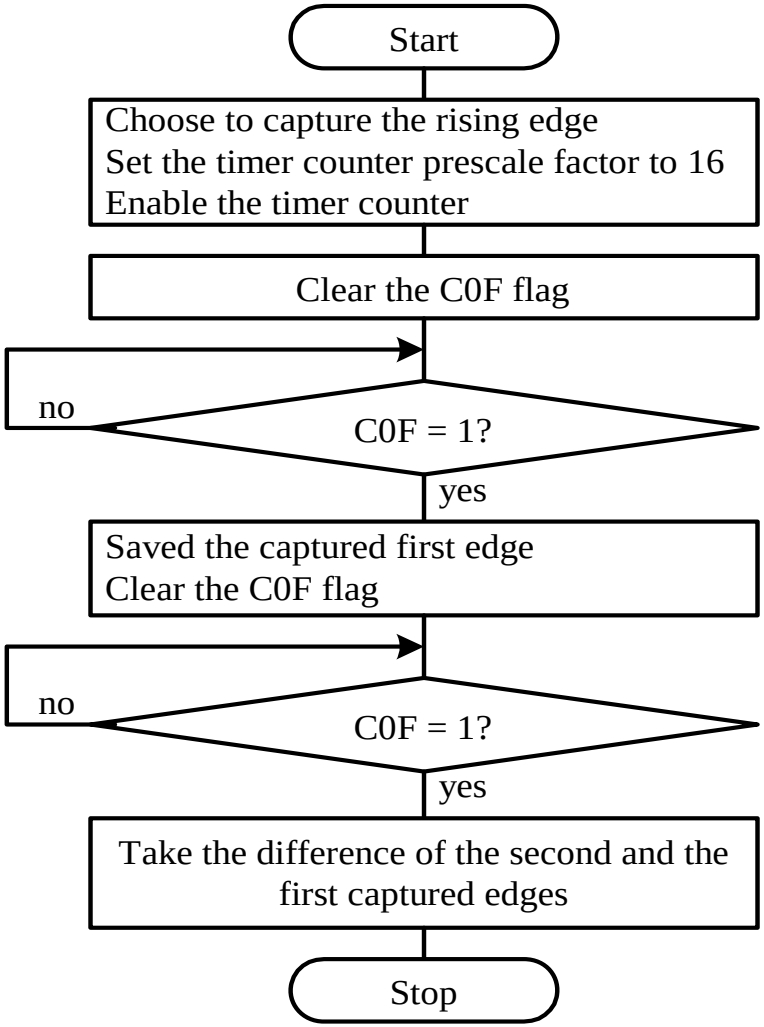


Figure 8.16 Logic flow of period measurement program

Solution: Period Measurement

```
#include  "c:\miniide\hcs12.inc"
          org     $1000
edge1     ds.b    2               ; memory to hold the first edge
period    ds.b    2               ; memory to store the period
          org     $1500
          movb    #$90,TSCR1      ; enable timer counter and enable fast timer flags clear
          bclr    TIOS,IOS0       ; enable input-capture 0
          movb    #$06,TSCR2      ; disable TCNT overflow interrupt, set prescaler to 64
          movb    #$01,TCTL4      ; capture the rising edge of PT0 signal
          movb    #$01,TFLG1      ; clear the C0F flag
          brclr   TFLG1,C0F,*     ; wait for the arrival of the first rising edge
          ldd     TC0             ; save the  first edge and clear the C0F flag
          std     edge1
          movb    #$01,TFLG1
          brclr   TFLG1,C0F,*     ; wait for the arrival of the second edge
          ldd     TC0
          subd    edge1           ; compute the period
          std     period
          swi
          end
```

- **Example 8.3 Write a program to measure the pulse width of a signal connected to the PT0 pin. The E clock frequency is 24 MHz.**

- **Solution**:
  - Set the prescale factor to TCNT to 32. Use clock cycle as the unit of measurement.
  - The pulse width may be longer than 216 clock cycles. We need to keep track of the number of times that the TCNT timer overflows. Let
    - *ovcnt* = TCNT counter overflow count
    - *diff* = the difference of two consecutive edges
    - *edge1* = the captured time of the first edge
    - *edge2* = the captured time of the second edge
  - The pulse width can be calculated by the following equations:

Case 1

edge2 ≥ edge1

$$\text{pulse width} = \text{ovcnt} \times 2^{16} + \text{diff}$$

Case 2

edge2 < edge 1

$$\text{pulse width} = (\text{ovcnt} - 1) \times 2^{16} + \text{diff}$$

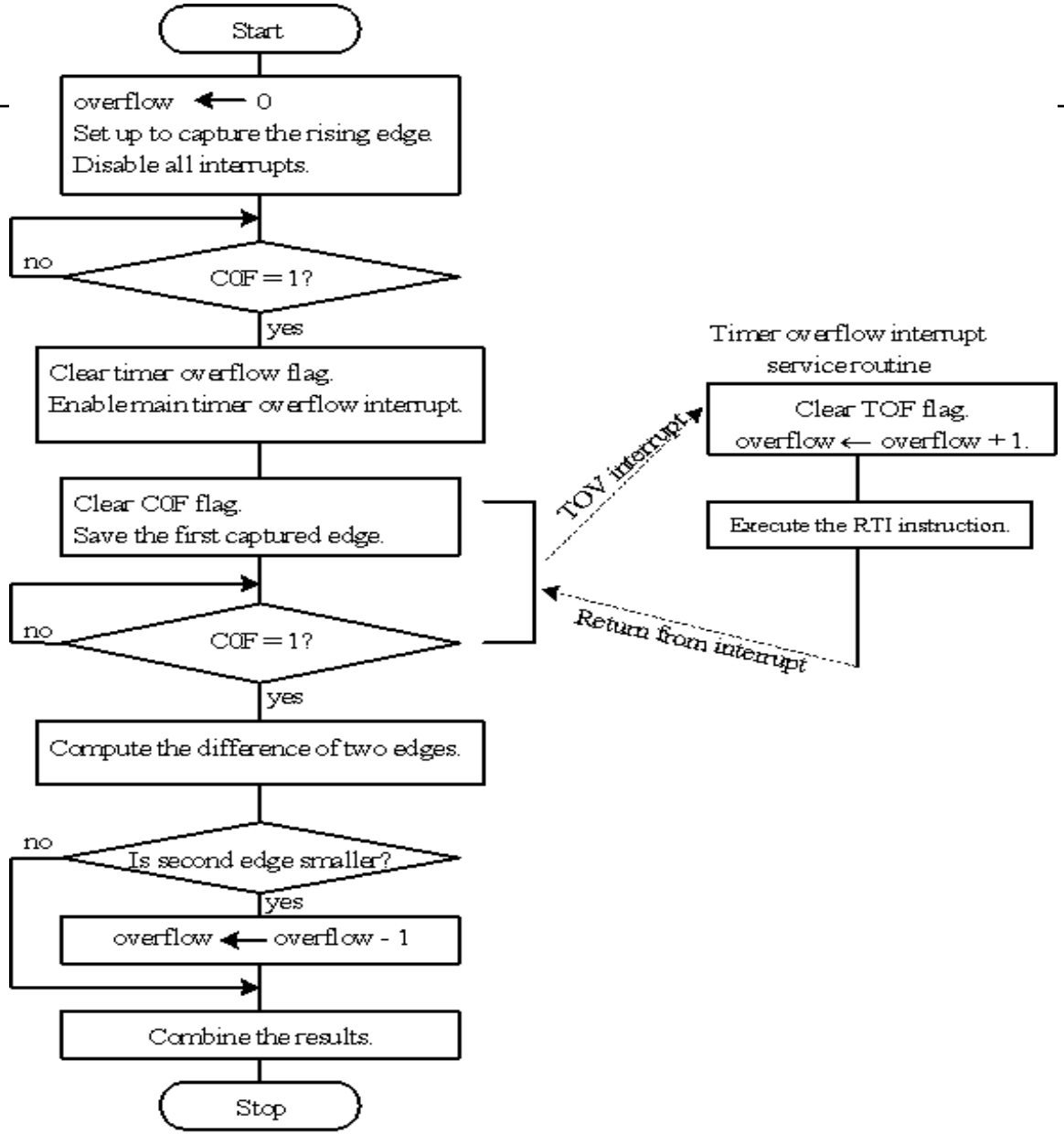


Figure 8.17 Logic flow for measuring pulse width of slow signals

```
#include    "c:\miniide\hcs12.inc"
            org     $1000
edge1       ds.b    2
overflow    ds.b    2
pulse_width ds.b    2
            org     $1500
            movw    #tov_isr,UserTimerOvf ; set up TCNT overflow interrupt vector
            lds     #$1500                ; set up stack pointer
            movw    #0,overflow
            movb    #$90,TSCR1            ; enable TCNT and fast timer flag clear
            movb    #$05,TSCR2            ; disable TCNT interrupt, set prescaler to 32
            bclr    TIOS,IOS0             ; select IC0
            movb    #$01,TCTL4            ; capture rising edge
            movb    #C0F,TFLG1            ; clear C0F flag
wait1       brclr   TFLG1,C0F,wait1       ; wait for the first rising edge
            movw    TC0,edge1             ; save the first edge & clear the C0F flag
            movb    #TOF,TFLG2            ; clear TOF flag
            bset    TSCR2,$80             ; enable TCNT overflow interrupt
            cli                           ;      "
            movb    #$02,TCTL4            ; capture the falling edge on PT0 pin
wait2       brclr   TFLG1,C0F,wait2       ; wait for the arrival of the falling edge
            ldd     TC0
            subd    edge1
            std     pulse_width
            bcc     next                  ; is the second edge smaller?
            ldx     overflow              ; second edge is smaller, so decrement
            dex                           ; overflow count by 1
            stx     overflow              ;      "
next        swi

tov_isr     movb    #TOF,TFLG2            ; clear TOF flag
            ldx     overflow
            inx
            stx     overflow
            rti
            end
```

Next…

- OC function