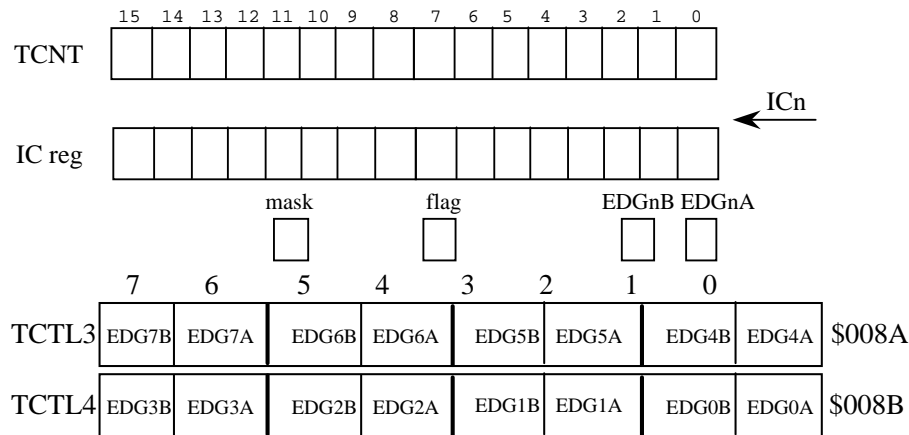


“When working on a software development team, everyone needs to be constantly reminded what is blindingly obvious.”

Tim Fields, UT EE grad, Lead Designer for Brute Force at Microsoft

16 bit Period Measurement with a 1µs resolution



When does an input capture happen?

on the active edge of the input **PTn**

EDGnB	EDGnA	active edge
0	0	none
0	1	capture on rising
1	0	capture on falling
1	1	capture on both rising and falling

What action occurs with an input capture

- sets a flag **CnF**
- the value of **TCNT** is latched into **TCn**
- optional (**CnI**): requests an interrupt

How to acknowledge interrupt

write a one to the **CnF** flag in **TFLG1**

Period measurement resolution

smallest change in period that can be detected
determined by the **TCNT** interval in **TSCR2**

in this example, Δp=1.33µs

Maximum period

determined by 16-bit size of counters and latches

does not work for DC inputs, $f=0$, period=infinite
 65535 TCNT cycles
in this example, $p_{max} = 65535 * 1.33\mu s = 87ms$

Minimum period

determined by the time to execute the ISR
 5 to 50 μs , depending on about much processing done
in this example, p_{min} less than $5\mu s$
 of course, at $p=5\mu s$, the foreground thread stops

Period measurement precision

determined by the TCNT and TCn precision
 16 bits or about 65000 alternatives

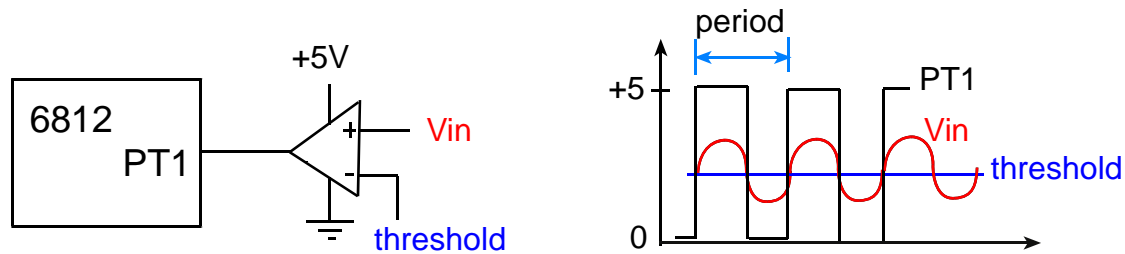


Figure 6.7. To measure period we connect the external signal.

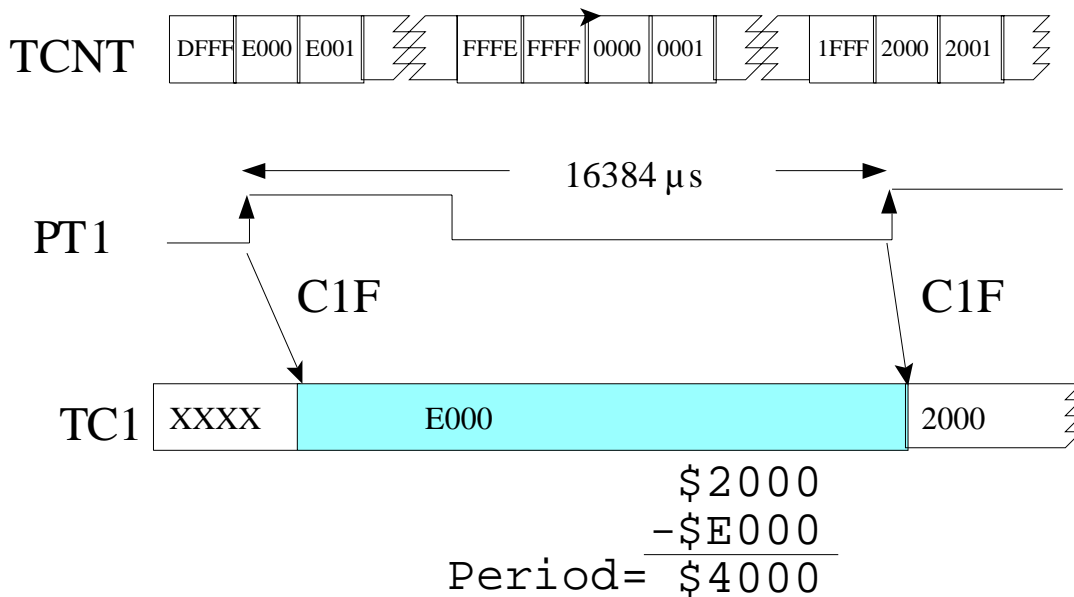


Figure 6.8. Timing example showing counter rollover during 16 bit period measurement.

```
unsigned short IR_Period[20]; // res = 1us
unsigned short IR_Count; // number of meas
unsigned short Last; // time of previous
```

```
// -----IR_Init-----
// input capture 2 on falling edge
// inputs: none
// outputs: none
// assumes 1us TCNT
void IR_Init(void){
asm sei // make atomic
  DDRT &= ~0x04; // PT2 input
  TCTL4 |= 0x20;
  TCTL4 &= ~0x10; // falling edge IC2
  TIE |= 0x04; // Arm IC2
  IR_Count = 0;
  Last = TCNT; // first wrong
  TIOS &= ~0x04; // PT2 input capture
asm cli
}
void interrupt 10 IC2Han(void){
  TFLG1 = 0x04; // clear C2F
  IR_Period[IR_Count] = TC2-Last;
  Last = TC2;
  if(IR_Count<19){
    IR_Count ++;
  }
}
```

Program 6.6. C language period measurement.

****Run with TExaS ****

6.3.2. Frequency Measurement with $\Delta f=100\text{Hz}$

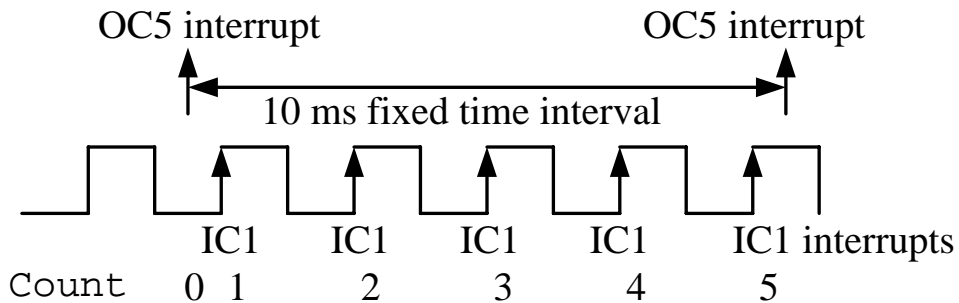


Figure 6.26. Basic timing involved in frequency measurement.

Frequency measurement resolution

smallest change in frequency that can be detected
determined by the fixed time interval, $\Delta f = 1/\text{time}$
 $\Delta f = 1/10\text{ms} = 100 \text{ Hz}$

Maximum frequency

determined by how fast the ISR can count pulses
 assuming ISR takes 5 μ s, $f_{max} = 1/5\mu s = 200$ kHz
 of course, at 200 kHz, the foreground thread stops

Minimum frequency

works without problem for $f=0$

Frequency measurement precision

maximum/resolution, $200kHz/100Hz = 2000$ or 11 bits
 counter precision

The rising edge will generate an input capture interrupt.

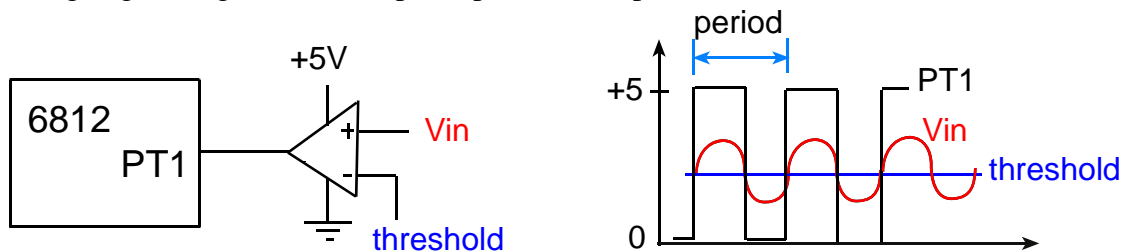


Figure 6.27. Frequency measurement.

```
unsigned short Freq; // units 100 Hz
unsigned char Done; // Set when done
```

```
There is a private global
unsigned short Count; // Number of edges
```

A frequency of 0 will result in no input capture interrupts and the system will properly report the frequency of 0.

```
// 9S12C32
interrupt 9 void TIC1handler(void){
    Count++; // number of rising edges
    TFLG1=0x02; // ack, clear C1F
}
#define Rate 10000 // 10 ms
interrupt 13 void TOC5handler(void){
    TFLG1= 0x20; // Acknowledge
    TC5 = TC5+Rate; // every 10 ms
    Freq = Count; // 100 Hz units
    Done = 0xff;
    Count = 0; // Setup for next
}
void ritual(void) {
```

```
asm sei                // make atomic
    TIOS &= ~0x02;    // enable IC1
    TIOS |= 0x20;     // enable OC5
    TSCR1 = 0x80;    // enable
    TSCR2 = 0x02;    // 1us clock
    PACTL = 0;       // no timer prescale
    TIE |= 0x22;     // Arm OC5 and IC1
    TC5 = TCNT+Rate; // First in 10 ms
    TCTL4 = (TCTL4&0xF3)|0x04;
/* C1F set on rising edges */
    Count = 0;       // Set up for first
    Done = 0;
/* Set on the subsequent measurements */
    TFLG1 = 0x22;    // clear C5F and C1F
asm cli
}
```

Program 6.30. C language frequency measurement.

6.5.2. Frequency Measurement with $\Delta f=0.1\text{Hz}$

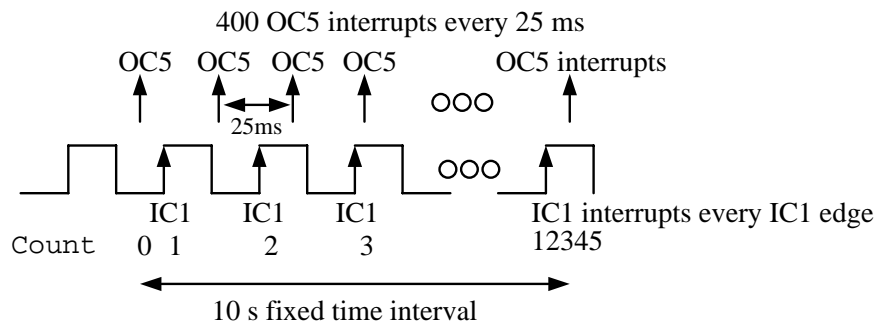


Figure 6.28. Basic timing involved in frequency measurement.

Frequency measurement resolution

$$\Delta f = 1/10\text{s} = 0.1 \text{ Hz}$$

Maximum frequency

assuming ISR takes $5 \mu\text{s}$, $f_{\text{max}} = 1/5\mu\text{s} = 200 \text{ kHz}$
 limited by counter precision, 16 bits
 $f_{\text{max}} = 65535 * \Delta f = 6553.5 \text{ Hz}$

Frequency measurement precision

$200\text{kHz}/0.1\text{Hz} = 2\text{E}6$ or 21 bits
counter precision, 16 bits

```
unsigned short Freq; // units 0.1 Hz
unsigned char Done; // Set when done
```

There are 2 private globals

```
unsigned short FourHundred; // 10sec
unsigned short Count;      // Number of edges
```

The input capture counts cycles

The output compare creates the 10-second fixed time interval.

```
// 9S12C32
interrupt 9 void TIC1handler(void){
    Count++;          // number of rising edges
    TFLG1 = 0x02;    // ack, clear C1F
}
#define Rate 25000 // 25 ms
interrupt 13 void TOC5handler(void){
    TFLG1 = 0x20;    // Acknowledge
    TC5 = TC5+Rate; // every 25 ms
    if(++FourHundred==400){
        Freq = Count; // 0.1 Hz units
        FourHundred = 0;
        Done = 0xff;
        Count = 0; // Setup for next
    }
}
void ritual(void) {
asm sei // make atomic
    TIOS| = 0x20; // enable OC5
    TSCR1 = 0x80; // enable
    TSCR2 = 0x02; // 1us clock
    TIE = 0x22; // Arm OC5 and IC1
    TCTL4 = (TCTL4&0xF3)|0x04;
/* C1F set on rising edges */
    TC5=TCNT+Rate; // First in 25 ms
    Count = 0; // Set up for first
    Done = 0; // Set every 10 seconds
    FourHundred = 0;
    TFLG1 = 0x22; // Clear C5F,C1F
asm cli
}
```

Program 6.36.Frequency measurement.