Timer1 Capture Interrupts

NDSU ECE 376

Lecture #23

Inst: Jake Glower

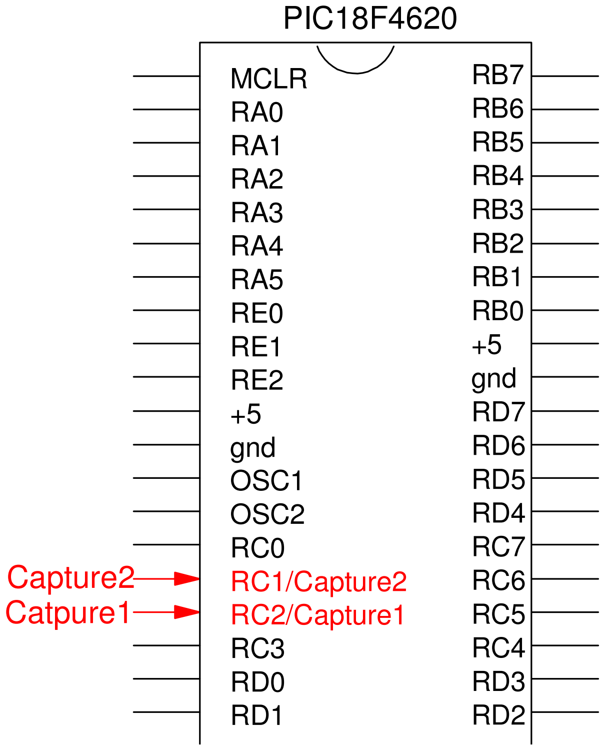
Please visit Bison Academy for corresponding lecture notes, homework sets, and solutions

Timer1 Capture Mode:

A PIC processor is able to measure time to 100ns. If you want to record the time of an event with time known to the clock, use Timer1 Capture Mode.

There are two Timer1 Capture interrupts:

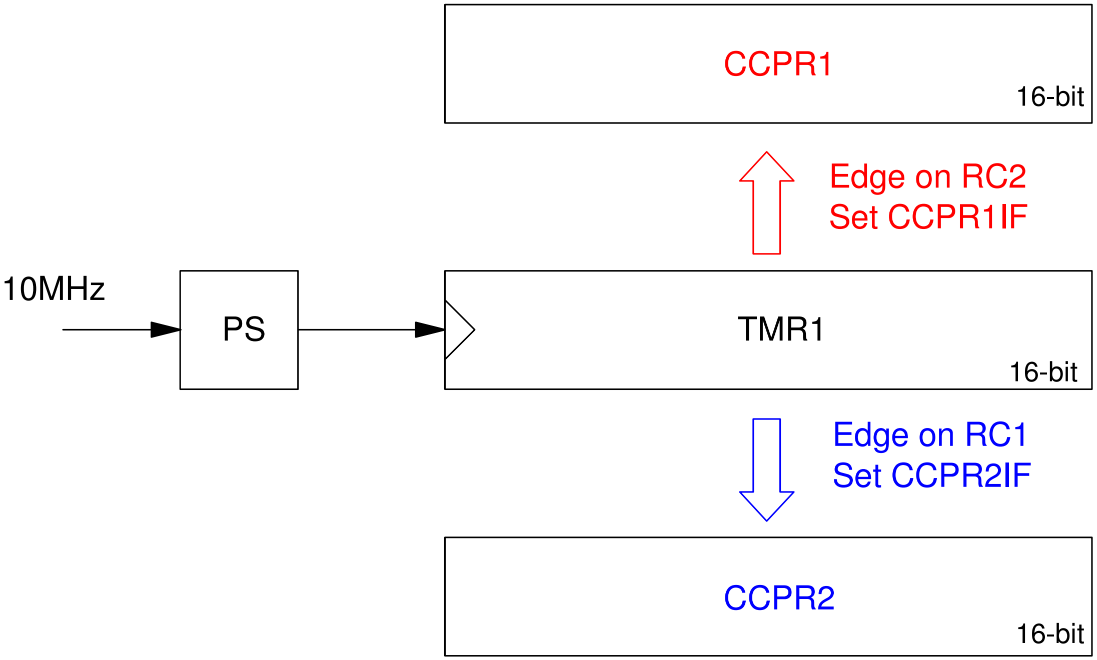
- Capture Mode 1 records the time of an event on pin RC2
- Capture Mode 2 records the time of an event on pin RC1 (yes, it's backwards)



Timer1 Capture Description

What a Time1 capture interrupt does is

- When an event is observed on RC1 or RC2 (rising or falling edge),
  - The time stored in TMR1 is copied to a register, and
  - A Timer1 Capture interrupt is triggered.



To get Timer1 Capture to work

i) You have to turn on Timer1.

ii) You have to set the condition for Timer1 Capture interrupt:

- Capture every falling edge: CCP1CON = 0x04
- Capture every riding edge: CCP1CON = 0x05
- Capture every 4th rising edge: CCP1CON = 0x06
- Capture every 16th rising edge: CCP1CON = 0x07

iii) Pin RC1 or RC2 have to be input

iv) You have to enable the Timer1 Capture interrupt (various flags)

At that point, whenever the corresponding edge is detected on RC1 (Capture2) or RC2 (Capture1),

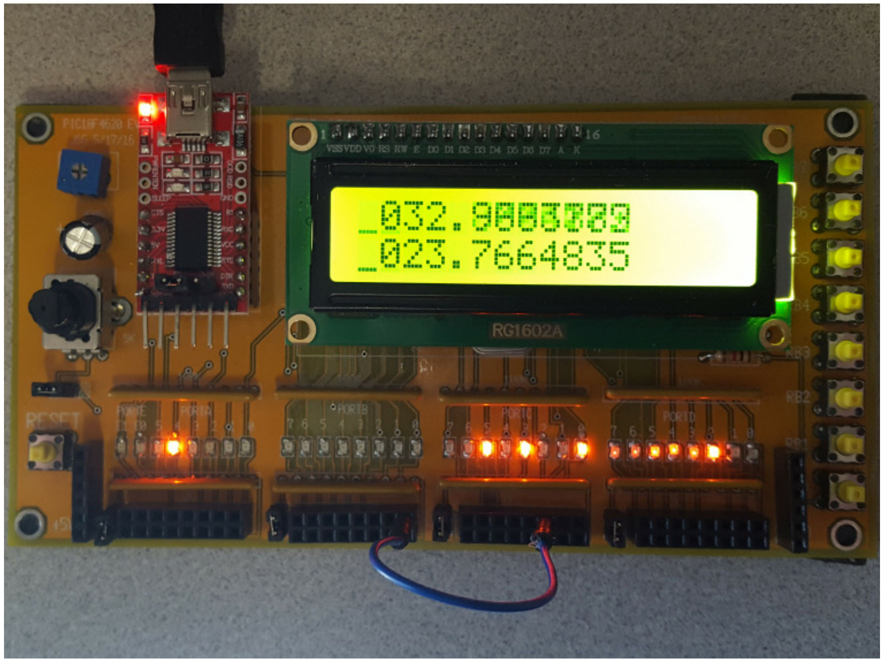
- The value of TMR1 is copied to CCPR1 or CCPR2
- A Timer1 Capture interrupt is triggered.

Example 1: Measure the time a button was pressed

INT interrupts are off by about 50 clocks

- The time it takes to trigger the interrupt

Capture1 interrupts is dead on.



Turn on Timer1 to record time to 100ns

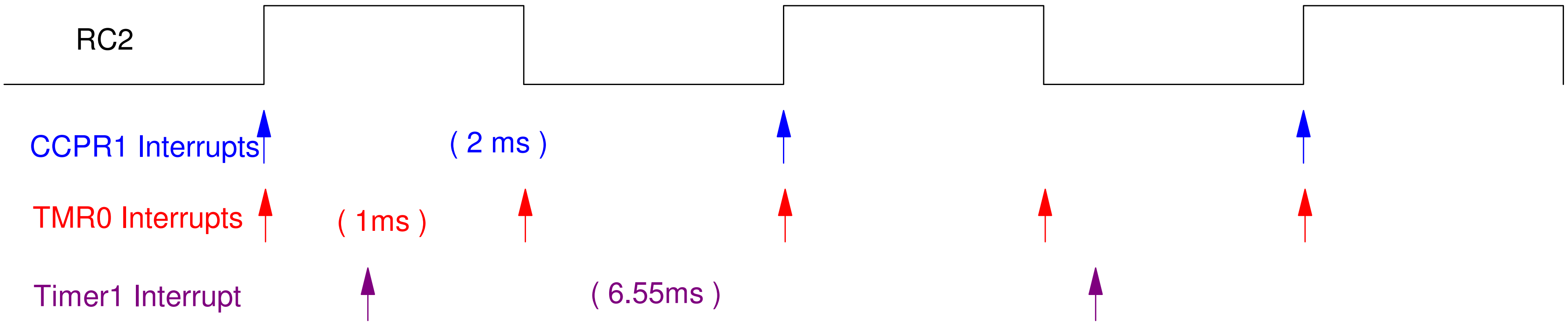
Turn on Capture1 to record the time that RC2 goes high

- Row 0: Running Time
- Row 1: Time RC2 goes high

Resulting time is dead on

Example 2: Measure the period of a square wave

- Record the time of each rising edge.
- The time between rising edges is the periond

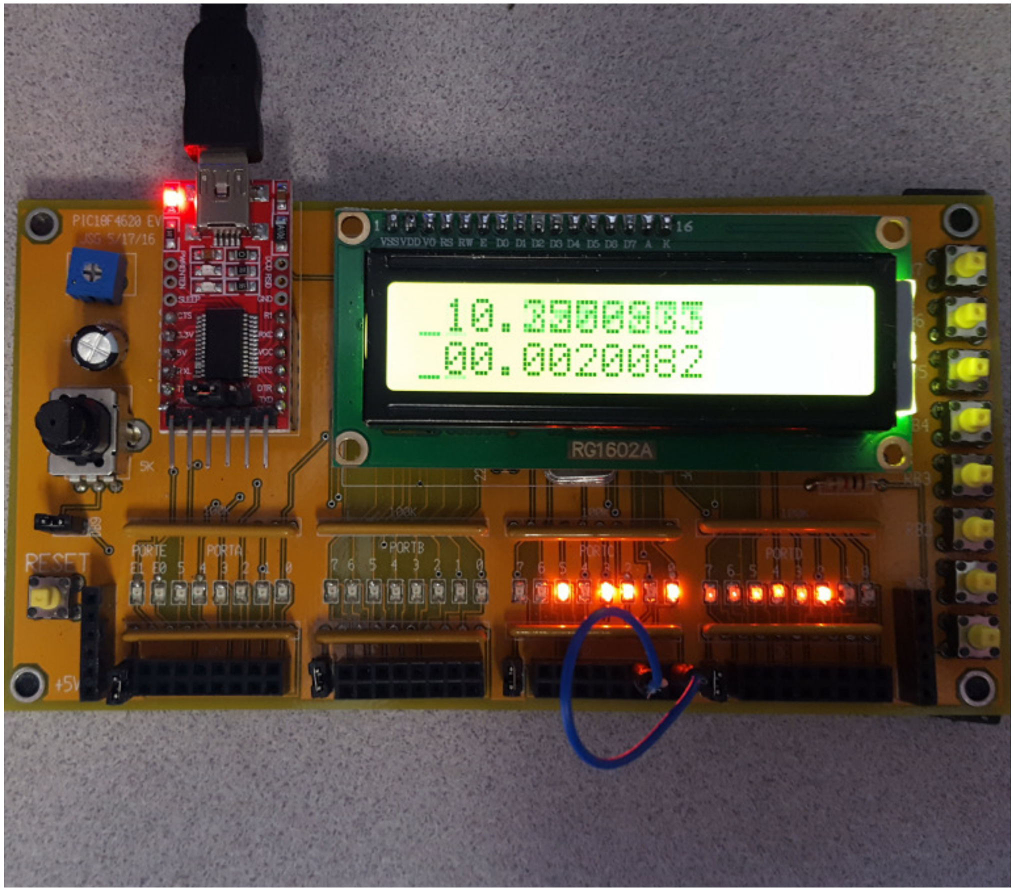


Three interrupts running in parallel:

Example:

- Timer0 toggles RC0 every 10,000 clocks
- Timer1 records time to 100ns
- Capture1 records the time that RC0 goes high

```
void interrupt IntServe(void)
{
   if (TMR0IF) {
      TMR0 = -10000;
      RC0 = !RC0;
      TMR0IF = 0;
      }
   if (TMR1IF) {
      TIME = TIME + 0x10000;
      TMR1IF = 0;
      }
   if (CCP1IF) {
      TIME0 = TIME1;
      TIME1 = TIME + CCPR1;
      CCP1IF = 0;
      }
   }
```

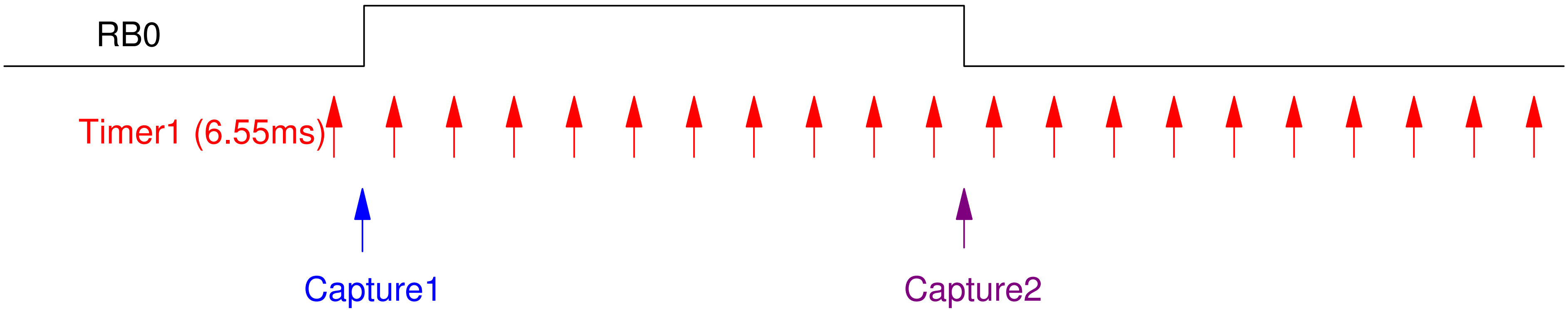


Example 3: Measure a pulse width (Capture2.C)

Tap button RB0. Measure how long that button was pressed (falling edge vs. rising edge).

To do this, use three interrupts:

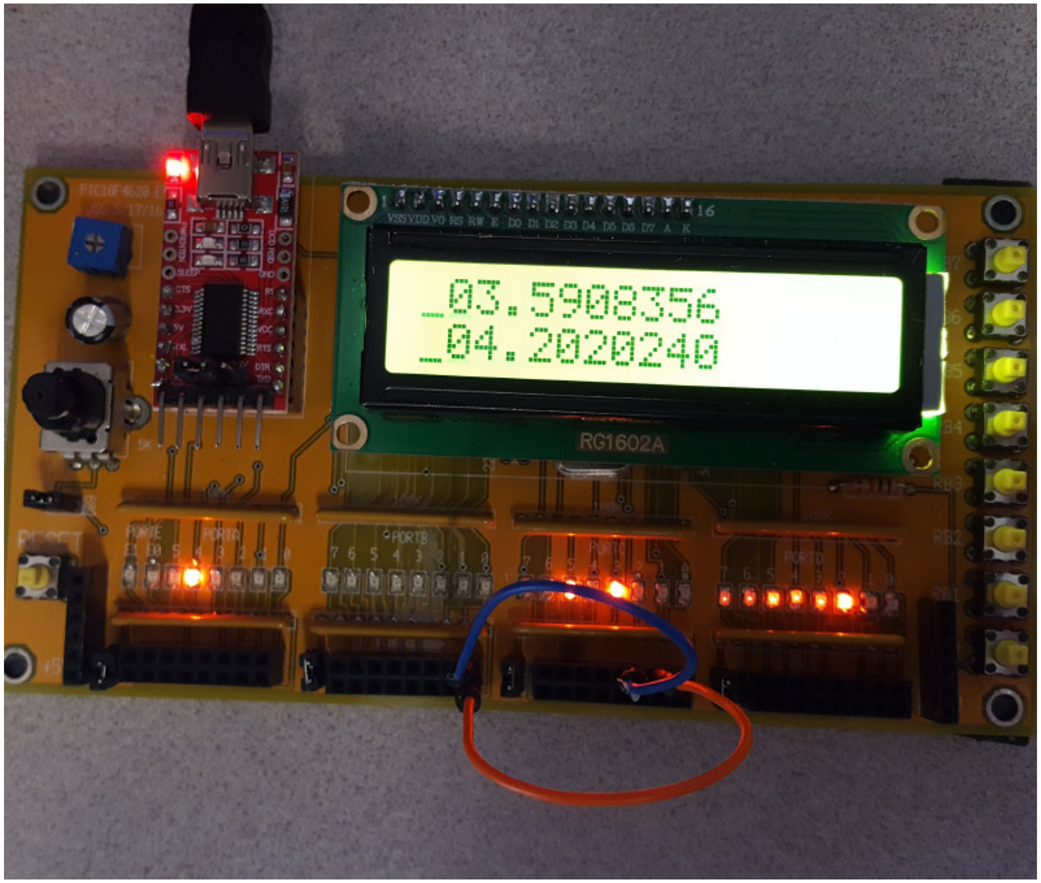
- Timer1 keeps track of time to 100ns
- Capture1 records the time of the rising edge
- Capture2 records the time of the falling edge



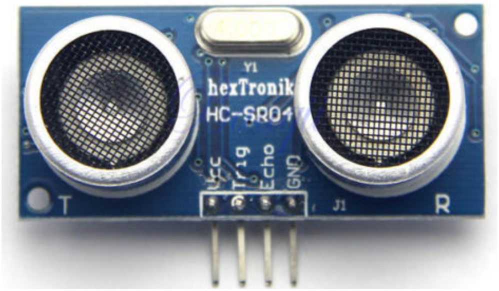
edges

Code:

```
void interrupt IntServe(void)
{
   if (TMR1IF) {     // 6.55ms
      TIME = TIME + 0x10000;
      TMR1IF = 0;
      }
   if (CCP1IF) {    // rising edge
      TIME0 = TIME + CCPR1;
      CCP1IF = 0;
      }
   if (CCP2IF) {    // falling edge
      TIME1 = TIME + CCPR2;
      CCP2IF = 0;
      }
   }
```



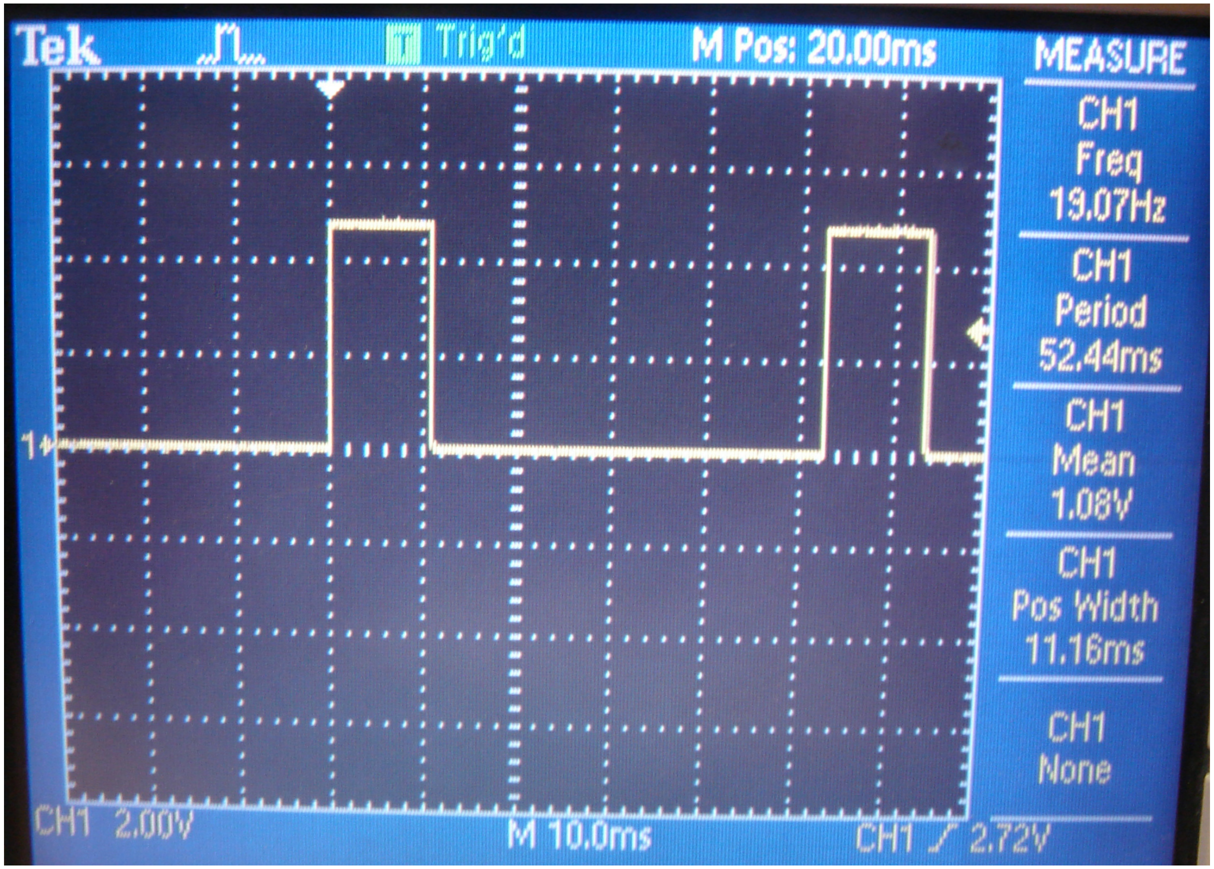
Example 4: Range Sensor (Range.C)



In your lab kit is an ultrasonic range sensor. This device has four pins:

- Vcc: input: +5V
- Trig: input: 0V/5V pulse from the PIC
- Echo: output: 0V/5V pulse to the PIC.
- Gnd: input: 0V

Each time you sent from the range sensor. The time it takes for the sound to return is the duration of the pulse on Echo. For example, if Trig is a 19Hz square wave, the signal on Echo might look like this:



M Pos: 20.00ms

MEASURE

CH1
Freq
19.07Hz

CH1
Period
52.44ms

CH1
Mean
1.08V

CH1
Pos Width
11.16ms

CH1
None

CH1 2.00V M 10.0ms CH1 ⁄ 2.72V

Distance = #clocks * 17.15 microns

```
void interrupt IntServe(void)
{
   if (TMR0IF) {
      RC0 = !RC0;
      TMR0IF = 0;
      }
   if (TMR1IF) {
      TIME = TIME + 0x10000;
      TMR1IF = 0;
      }
   if (CCP1IF) {
      if (CCP1CON == 0x05) {  // rising edge
         TIME0 = TIME + CCPR1;
         CCP1CON = 0x04;
         }
      else {
         TIME1 = TIME + CCPR1;
         dT = TIME1 - TIME0;
         CCP1CON = 0x05;
      }
      CCP1IF = 0;
      }
   }
```