ECE 4175
Project Eight
Frequency Counter

Complete by:
Wednesday March 1st for an A+

References:

Section 13.9	Frequency measurement
Section 13.8	Extended Timer3 operation
Figure 13-12	CCP2/Timer3 capture mode
Section 14.7	Floating-point subroutines
Section 14.8	Normalize subroutine

Preamble

For this project, you are going to use the CCP2/RC1 pin located at the bottom of the QwikFlash board. Make sure that the jumper located to the left of the IR sensor (just above the right side of the LCD) is removed as is the jumper to the left of the TMP04 temperature sensor located at the bottom of the QwikProto board. Both of these devices, with jumpers in place, will drive the CCP2 input, to the detriment of the measurement intended here.

After checking with the scope to ascertain that the red lead from the function generator available at each workstation is putting out a square wave that ranges between 0V and 5V, connect this output to the CCP2 pin and the function generator GND output (the black lead) to the GND pin next to the CCP2 pin.

Overview

For this project, you are to start over from Project One code, blinking the Alive LED but no longer stepping the stepper motor. Now, you will use the scheme of Figure 13-13 to measure the frequency of the input on the CCP2 input, collecting new data every second and using a (minimum) gate time of about 0.4 seconds. Use **CCP2CON** = 0b00000101 to capture every rising edge of the input.

Use Timer3 overflows to generate low-priority interrupts. Within the interrupt service routine, increment a RAM variable, **TMR3X**, to create a three-byte Timer3. Also clear the **TMR3IF** interrupt flag before returning from the low-priority interrupt.

To initiate a new measurement every second, clear **TMR3X, TMR3H** and **TMR3L** (in that order), clear the **CCP2IF** flag, clear **MX:MH:ML**, and set the **CCP2IE** local interrupt enable bit.

Use each rising edge of the input waveform to generate a CCP2 high-priority interrupt. Within the interrupt service routine do two things. For the very first interrupt of a measurement, copy the captured three-byte value of **TMR3X:CCPR2H:CCPR2L** into **STARTX:STARTH:STARTL** For each subsequent interrupt, increment **MX:MH:ML**. Also check whether **TMR3 - START** has exceeded 0.4 seconds. If so, then copy **TMR3X** to **NX** and **CCPR2H:CCPR2L** to **NH:NL** and disable further CCP2 interrupts.Now form the time between the start edge and the stop edge, **NX:NH:NL**, by subtracting **START** from **N** and return from the interrupt.

Calculation:

In the mainline loop, monitor the change of **CCP2IE** from one to zero (during successive loop times), indicating the completion of a measurement. When this occurs, use the equation at the bottom of page 182 to compute the frequency in units of Hz. However, what I want you to do is to present the resulting measurement as a seven digit result. For example, if the frequency is 13005.67 Hz, then the LCD should show

```
Freq  Hz
13005.67
```

To do this, first convert M, N and 2,500,000 to floating-point numbers and then carry out the computation using the floating point subroutines. At the completion, use the **Normalize** subroutine of Figure 14-9 and its table of Figure 14-8 to convert the result to the form exemplified by Example 14-4 on page 199. Next, convert this to a display string for the eight characters of the second line of the display. Finally, use the TMP0 output of the **Normalize** subroutine to move digits over and insert the decimal point in the correct position and then call the **DisplayV** subroutine to display the frequency on the second line. The first line of the LCD can be created in your **Initial** subroutine, once and for all.