CMPE 663 Project 1

Timers

Andrei Tumbar

Instructor: Wolfe
TA: Nitin Borhade

By submitting this report, you attest that you neither have given nor have received any assistance (including writing, collecting data, plotting figures, tables or graphs, or using previous student reports as a reference), and you further acknowledge that giving or receiving such assistance will result in a failing grade for this course.

Your Signature: _____

Analysis/Design

This project looked at implementing a timer for input capture. `TIM2` was set up for input capture on Channel 1. Microseconds between rising edges on the input pin were timed. A variable frequency sqaure wave generator was used to produce input signals.

The overall flow of the program was quite simple. A Power-on-self-test was created to test the signal of the input pin to verify a valid input wave.

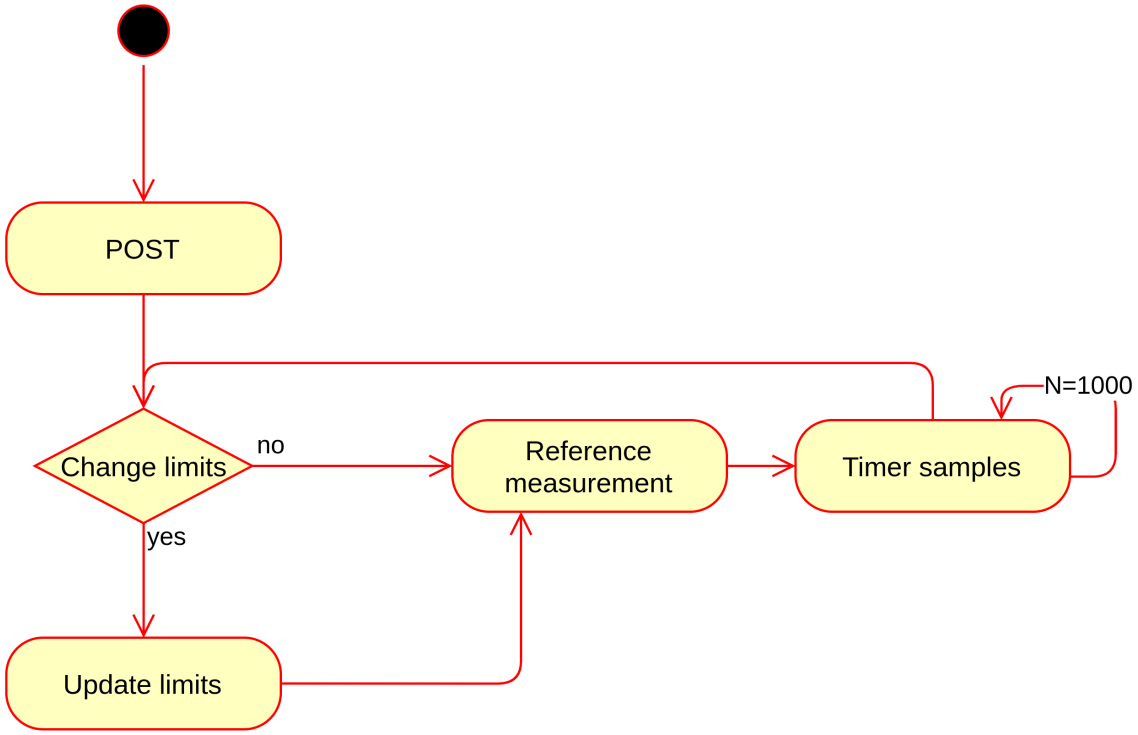

Figure 1: Design of input capture timer

Figure 1 shows the basic control flow of the project. The important thing to note here is that 1001 measurements are taken because the first tick needs to compare its counter to the event before. This is what the reference measurement does.

A UART interface was designed to utilize `printf`/`getline` features to the UART. Input and output via the uart was stardardized to more closely resemble a common C program.

Test plan

The POST routine was used to test the existence of a wave. To test the actual timer, various frequencies were tested to see if the buckets filled around the generated frequency. The sum

of all buckets was verified to equal 1000.

To test the UART portion, the limits of the input were tested as well as invalid inputs. The `DEL` ASCII code was tested to be handled properly during getline to allow the user to use backspace during inputs.

Project Results

When running the program with a 1 kHz wave connected, I'd expect the output to be grouped around 1000 ms.

```
999     36
1000    144
1001    146
1002    182
1003    195
1004    206
1005    79
1006    12
Enter lower limit period (us) or <ENTER> if no change (950 us): _
```

Figure 2: Input capture performed with a 1 kHz signal

When run again with a 2 kHz, I'd expect the period to be around 500 ms.

```
999     36
1000    144
1001    146
1002    182
1003    195
1004    206
1005    79
1006    12
Enter lower limit period (us) or <ENTER> if no change (950 us): _
```

Figure 3: Input capture performed with a 1 kHz signal

Both of the tests run and output expected results.

Lessons learned

I encountered two major issues while working on the project. The first issue was that the wave generator I was using initially was not configured to pulse to the correct voltage. The timer would never trip even though the wave looked correct on the oscilliscope. The second issue was due to my misunderstanding with how the timer works. I assumed that when an input event was triggered, the counter would be reset. Due to this assumption, the results

would become very skewed after re-running the program a couple of times. Once I realized how the timer works and the reference measurement was implemented, the program worked properly.