DSP Processing of uFluidics Signal

Michael Nolan

I. DSP Portion

Because the uFluidics device generates an extremely weak signal, when the signal reaches the circuit to count the cells, it will have quite a lot of noise in it. In order to accurately count the cells, the signal will need to be processed in order to remove the noise and detect when a cell passes through the device.

A. DSP

The DSP for this project needed to

- Have a low cost
- Have an on board ADC
- Be more than able to process a 5ks/s signal (to allow for changes to processing algorithm)
- Have enough ram to process the above signal in batches of 1024 samples
- Be capable of driving a display or other peripherals to present results

After some searching, a dsPIC33f was determined to be able to fufill the above requirements, and the dsPIC33FJ128GP802 was selected as it had more than enough ram and processing power for our use. Additionally, a less powerful DSP from the same product line could be used in the final product after determining exactly how much ram and processing power are needed

B. Processing

According to [1], the optimal way to separate a known signal from white noise is to use correlation. Via COMSOL simulation and via a crude prototype involving dropping a straw through a coil, the signal of interest would have a shape similar to Figure 1

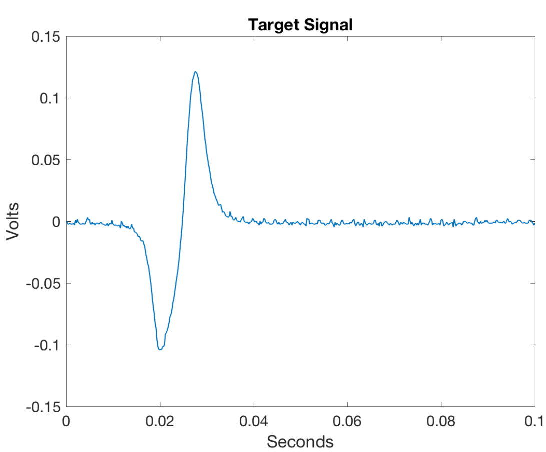

Fig. 1. Signal from prototype device

Therefore we should correlate our input signal with this signal to maximize the separation from noise. To make sure this technique would suit our needs, a script was created in MATLAB to illustrate the effects of the correlation.

First, the signal in Figure 1. was modified by adding some noise, and appears in Figure 2 Next, the signal was correlated

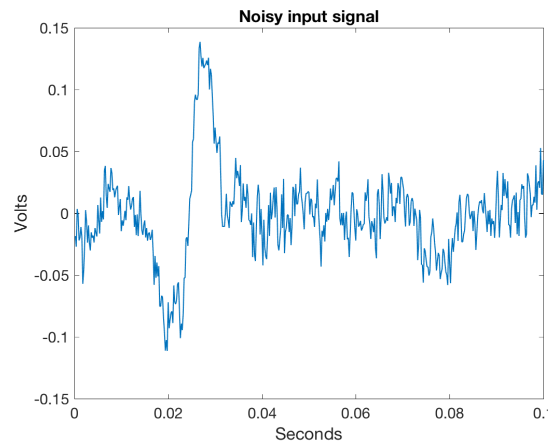

Fig. 2. Signal with simulated noise

with the signal in Figure 3, and the resulting signal appears in Figure 4.

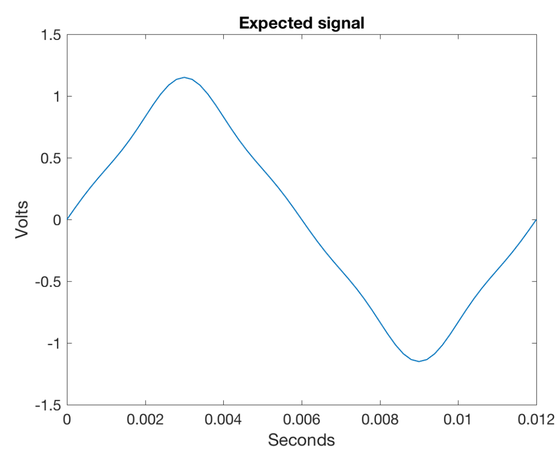

Fig. 3. Correlation target signal

Notice how Figure 4 has several small bumps where the target signal matched weakly with the noise, and also the very large negative spike from where the target signal was lined up with the pulse in the input signal. Unfortunately, the signal from the uFluidics device can be either positive or negative, so further processing is needed. To force both polarities to be positive, the signal in Figure 4 is squared, which also has

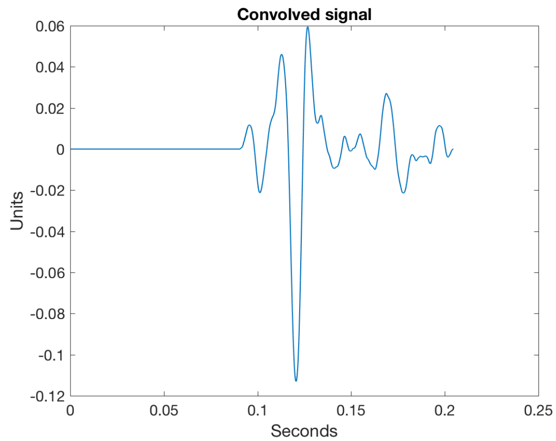

Fig. 4. Result of correlation

the effect of reducing the relative amplitude of the spikes from the noise compared with the desired spike. Finally, the number of spikes above a threshold value are counted, and the result appears in Figure 5.

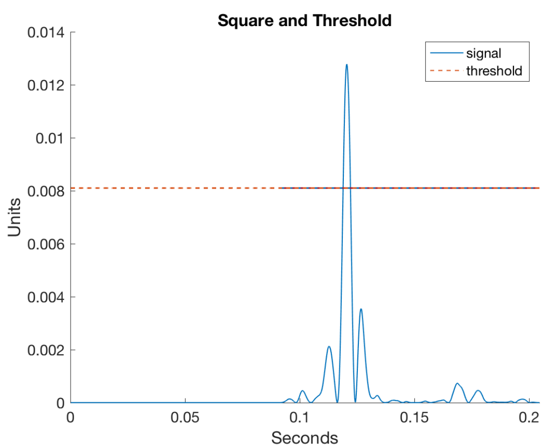

Fig. 5. Squared signal with threshold

C. Software

The DSP was programmed in C, with some functions, such as squaring and thresholding, written in assembly. The processing is as above, with some additional considerations for continuous processing. 256 samples are read from the ADC and fed into a buffer, along with the previous 256 samples. Then, those 512 samples are correlated with the signal from Fig. 3, and the resulting signal is placed into another buffer. Each sample in the buffer is squared, and then the 256 samples starting at sample number $256 - 61 = 195$ (256 - samples from the previous sample period, 61 - number of samples in the target signal) are fed to the threshold counting function, which counts the number of times the signal crosses the threshold value going positive. The threshold is only run on these samples so that a cell that generates a signal between the two aquisition periods will still get counted.

References

[1] S. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Pub, 1997.