TATER November Monthly Status Report
12-1-2017

1 Description

This status report covers the progress and research conducted so far for hardware, algorithm analysis, and antenna design. It also discusses the areas of focus for the future.

Contents

1 Description **1**

2 Hardware Capture **1**

3 Algorithm Analysis/Progress **2**

4 Antenna Design **5**

5 Future plans **6**

2 Hardware Capture

The main problem we are currently faced with is getting the data to the computer in a reasonable amount of time. Currently our oscilloscope can capture 8 million data points at 2GSPS, which only allows us to capture 8ms of data for the computer. While we can mitigate this issue by increasing the clock frequency and shortening the time of boot, the processor takes 80ms just to begin executing the boot code. However, we can also mitigate this by delaying the start time for capturing data.

Even though our main frequency of interest is 1Mhz because this is the frequency of the detectable instruction peaks, we need to sample at least 200MSPS based on our recorded waveform captures in order to detect modifications to particular instructions. The Nyquist sampling frequency allows sampling at a much lower rate, but in order to do proper peak detection, it entails a higher rate.

The following ADC FMC cards (mainly the AD9434) would be suitable for our design. However the speed requires a fairly fast FMC to PCIe carrier card.

- https://wiki.analog.com/resources/fpga/xilinx/fmc/ad9434
- https://wiki.analog.com/resources/fpga/xilinx/fmc/ad9467

Possibly paired with either of the following products would work for this implementation:

- https://www.vadatech.com/product.php?product=460catid_now=0catid_prev=0
- https://shop.trenz-electronic.de/en/TEF1001-01-Kintex-7-PCIe-FMC-Carrier-with-Xilinx-Kintex-7-160T-4-lane-PCIe-GEN2

Interestingly, the following PCI Express digitizers would fit the description perfectly, as they come in many different flavors and sampling rates. This would avoid any future complications with pairing and compatibility issues, as they support continuous capture modes and direct PCIe streaming with driver support.

- https://spectrum-instrumentation.com/en/ad-digitizer-transient-recorder-pci-express-pcie-5-gss

However, we feel that we can work around the limitations of the capture devices that we currently have. We may have to change the parameters of the project if this is the case.

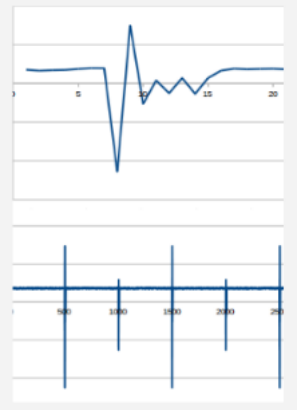
3 Algorithm Analysis/Progress

Due of the limits of our hardware analysis, we opted to simulate our waveform data in order to work in parallel.

A few simulation programs were constructed in order to ease implementation. The first exports a randomized block of data designed to model our boot code machine level instructions. This makes it possible to modify our simulation at the instruction level in order to inject, remove, and add instructions for testing, and to have a method that introduces configuration changes.

This data block is fed to the waveform simulation which takes each instruction and creates a corresponding waveform signature based on our previous measurements from the oscilloscope. This not only allows us to control variables such as sampling rate, but we can also introduce noise, amplitude variation, and other factors which have the potential to modify our signal. Shown below is a sample graph of a few data points of this simulation.

From there, the waveform data is fed into the front end of our algorithm which we are currently implementing, and is described below.



In signal processing, cross-correlation is a measure of similarity of two series as a function of the displacement of one relative to the other. We are beginning by using the following definition as it operates in O(n log n) time rather than O(n$\hat{2}$), based on the different methods of cross-correlation:

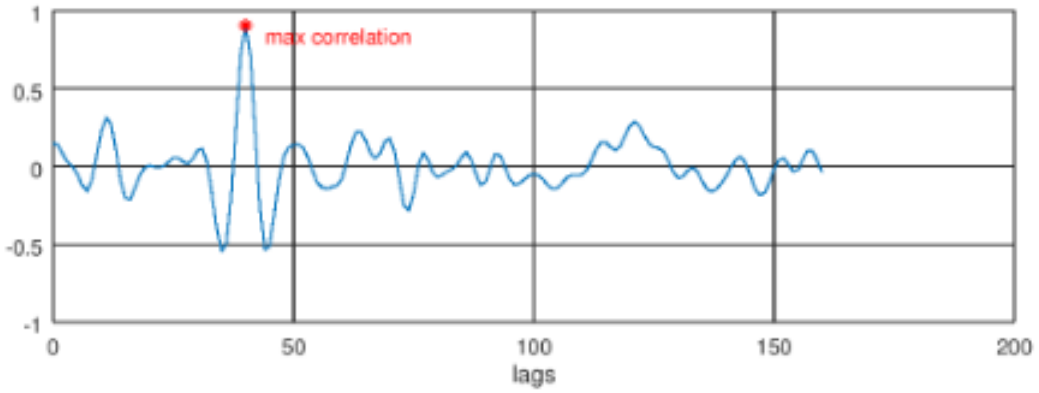
```
corr(a, b) = ifft(fft(a_and_zeros) * conj(fft(b_and_zeros)))
```

We are planning to use cross correlation to first align our waveform sequence, which will also allow us to compare how similar the sequences are in regards to location of peaks and time in between. In addition to using cross correlation, we also intend to compare the amplitude of the corresponding peaks, which can be done by scaling the waveform data before applying the fft. This will allow us to detect if any small changes have been made to the hardware as well, and the antenna will pick up any trivial differences.

Approaches to dealing with multiple boot configurations:

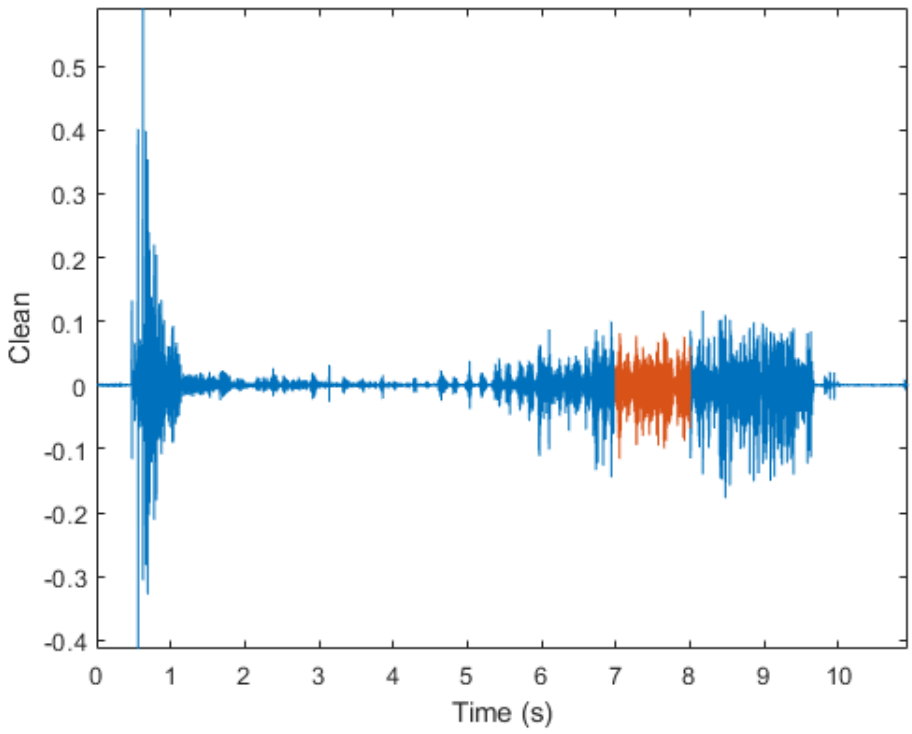
- Most simple and inefficient: have one sequence for every possible combination of configurations. Run the algorithm repeatedly to compare captured data with different allowable configurations.
- Identify regions of code that do not change (i.e. basic blocks of compiler code). Find and compare these regions. At the same time, compare blocks in between with allowable "configuration option" sequences.

For option (2), run cross correlation algorithm against basic baseline with no configuration options. If significant deviations are encountered, look at what configuration options may occur after that section. Shown below is a normalized output, where we can ensure that the wave forms have a close enough match. This will allow us to add support for sensitivity control as well in the future.

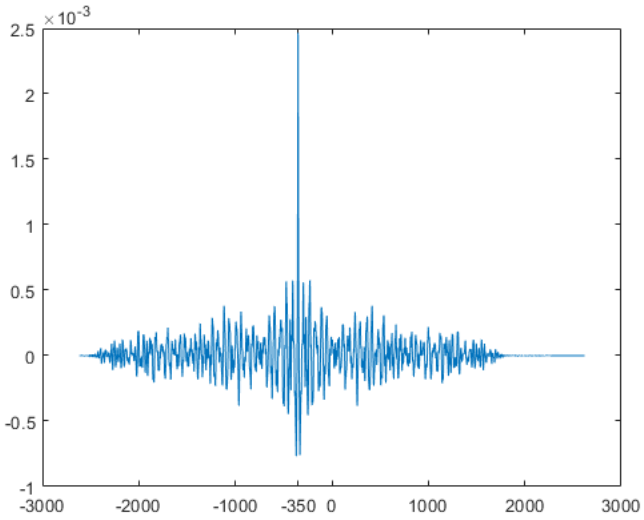


The fft will be performed across a sliding window to detect differences in each section of the captured waveform, which will be used to locate fragments of code that have allowable configuration changes. The main advantage of this would be faster detection, as we do not have to wait for the full boot code to be correlated before reporting to the user.

Below is an example of the fragment that may be extracted. The preprocessing algorithm can then store the time offsets and length of this signal, and then slide it against the captured signal when we are doing our analysis.



We can then correlate this to find the highest correlating peak:



This minimizes the amount of comparisons that must be made and improves efficiency. The model building algorithm will then compute a list of allowable waveform blocks that may differ according to different boot configurations. If the detected signals do not align with any of these listed sequences, then the algorithm will reject the code. Otherwise, it will be accepted.

The example output for our cross-correlation tests, where we implement the above algorithm:

```
orig a:  (di=8): 10.00, 0.00 -10.00, 0.00 10.00, 0.00  5.00, 0.00  0.00, 0.00  0.00, 0.00  0.00, 0.00  0.00, 0.00
orig b:  (di=8): 10.00, 0.00 -10.00, 0.00 10.00, 0.00  0.00, 0.00  0.00, 0.00  0.00, 0.00  0.00, 0.00  0.00, 0.00
fft a:   (di=8): 15.00, 0.00 -0.61,-6.46 -0.00,15.00 20.61,13.54 25.00, 0.00 20.61,-13.54  0.00,-15.00 -0.61, 6.46
fft b:   (di=8): 10.00, 0.00  2.93,-2.93 -0.00,10.00 17.07,17.07 30.00, 0.00 17.07,-17.07  0.00,-10.00  2.93, 2.93
conj:    (di=8): 10.00,-0.00  2.93, 2.93 -0.00,-10.00 17.07,-17.07 30.00,-0.00 17.07,17.07  0.00,10.00  2.93,-2.93
mult:    (di=8): 150.00, 0.00 17.16,-20.71 150.00, 0.00 582.84,-120.71 750.00, 0.00 582.84,120.71 150.00, 0.00 17.16,20.71
ifft:    (di=8): 2400.00, 0.00 -1200.00, 0.00 400.00, 0.00 400.00,-0.00  0.00, 0.00  0.00,-0.00 800.00,-0.00 -1600.00, 0.00
norm:    (di=8): 300.00, 0.00 -150.00, 0.00 50.00, 0.00 50.00,-0.00  0.00, 0.00  0.00,-0.00 100.00,-0.00 -200.00, 0.00
```

4 Antenna Design

This month, we brought the idea of possibly using a micro-strip antenna instead of the ferrite loop antenna. Using this type of antenna over the other would save space and allow for easier integration into our system. The downside to using the micro strip is that it has a small bandwidth. As it is very important to capture the entirety of our signal, the correct bandwidth to signal ratio needs to be determined. The two variables are inversely proportional, and finding the optimal ratio is a balancing act that still needs to be determined. If the optimal ratio is found and the bandwidth is still insufficient, then going back to the ferrite loop might be the best solution to find the correct values. The ferrite loop can also potentially be designed with a flat square sheet of ferrite instead of a cylinder, which would allow for easier integration into our system, as it would need less space to operate.

The frequency that the antenna operates at can be varied, so long as the scope of its operation encompasses our signal. The antenna was originally designed with a 1 MHz frequency, as that is the operating frequency of the device being received from. This is a typical starting point when designing antennas, and the frequency is then adjusted to fit the specific needs of the system. A frequency response graph has not been able to be drafted yet, as the design criteria has not quite been met to make one, though this will be remedied later on as a better understanding of the FEKO software is obtained.

5 Future plans

The plan of attack is to continue implementation of the algorithm based on our simulation wave forms as it will allow for faster testing. We will also work on testing and verifying the various antenna designs to ensure that they will provide the optimal response we need.

Another interesting idea suggested by the client involved characterizing instructions. Instead of cross correlating a series of single points representing individual instructions, it was suggested to look at a series of windows of 20-30 instructions around each peak. This would be accomplished by comparing that window against the entire capture of the boot up sequence by applying cross correlation. We plan to test this idea in our code as well.