Instruction Level Reverse Engineering through EM Side Channel

Client: Prof. Akhilesh Tyagi | **Advisor**: Varghese Vaidyan
Team Members: Matthew Campbell, Noah Berthusen, Cristian George, Jesse Knight, Jacob Vaughn, Evan McKinney
Spring 2021 Team 9 | **sdmay21-09@iastate.edu** | **https://sdmay21-09.sd.ece.iastate.edu**

Introduction

Problem: Reverse engineer the executing program of a processor through measurements of the electromagnetic radiation that it emits.
Solution: Collect EM data with a probe and send through a machine-learning algorithm to detect and classify code running on the processor.

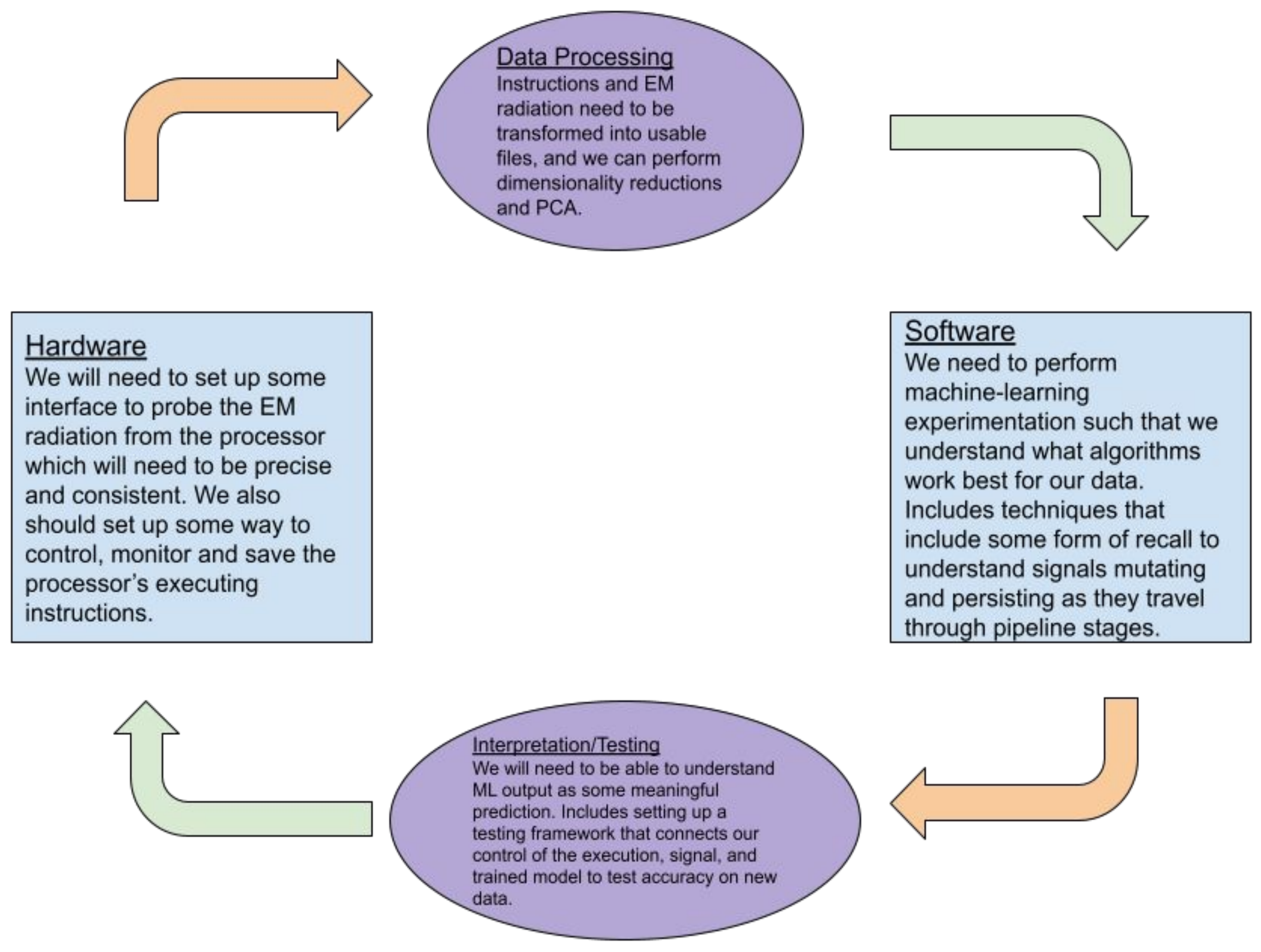
Design Requirements

- Collect EM data from a ARM-M7 processor with a 20+ Mhz frequency and 6-stage pipeline.
- Build an interface between the EM antenna and code to organize and filter relevant data.
- Large amount of data used to train the machine learning model
- Opcode detection with 90%+ accuracy
- Well-documented code
- Predictions formatted to be user-friendly

- Limited in data collection by number of available oscilloscopes
- Required sampling rate per processor speed
- Large amount of data needed to train model

- Followed software development and assembly engineering standards

Design Approach


Data Processing
Instructions and EM radiation need to be transformed into usable files, and we can perform dimensionality reductions and PCA.
Software
We need to perform machine-learning experimentation such that we understand what algorithms work best for our data. Includes techniques that include some form of recall to understand signals mutating and persisting as they travel through pipeline stages.
Interpretation/Testing
We will need to be able to understand ML output as some meaningful prediction. Includes setting up a testing framework that connects our control of the execution, signal, and trained model to test accuracy on new data.
Hardware
We will need to set up some interface to probe the EM radiation from the processor which will need to be precise and consistent. We also should set up some way to control, monitor and save the processor's executing instructions.

Resources

- Budget: ~ $100
  - EM Probes
  - Coax Cables
  - Nucleo Board

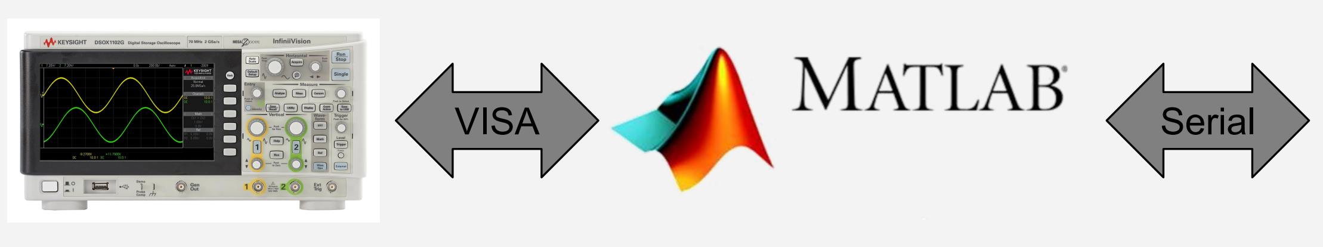
Security Concerns

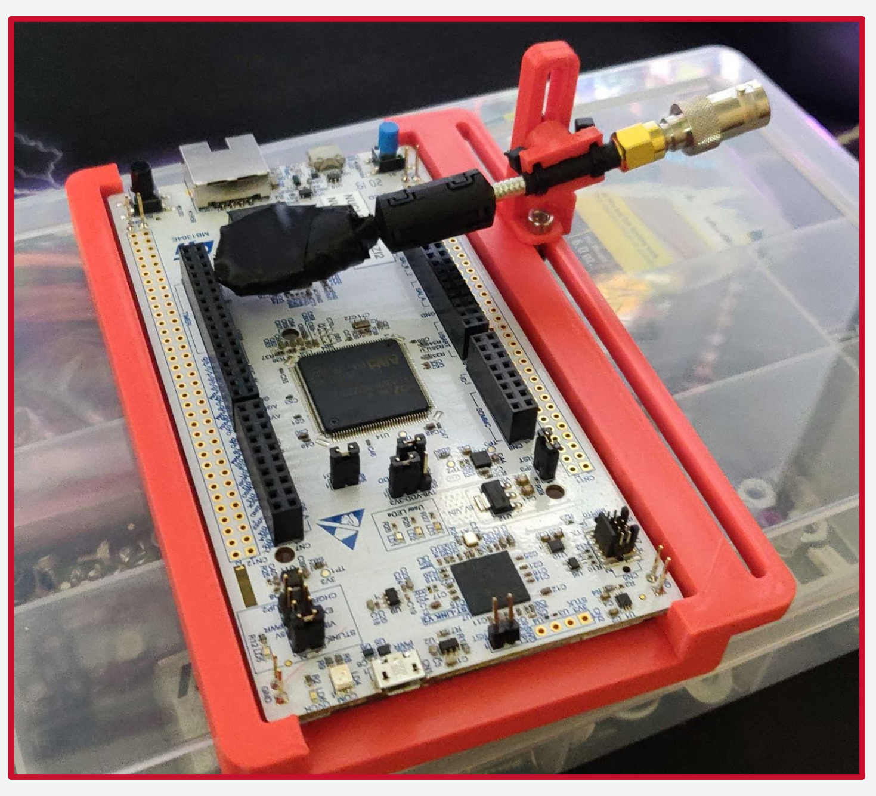
- Project is part of a larger research group:
  - Want to ensure that we do not release information before they are ready

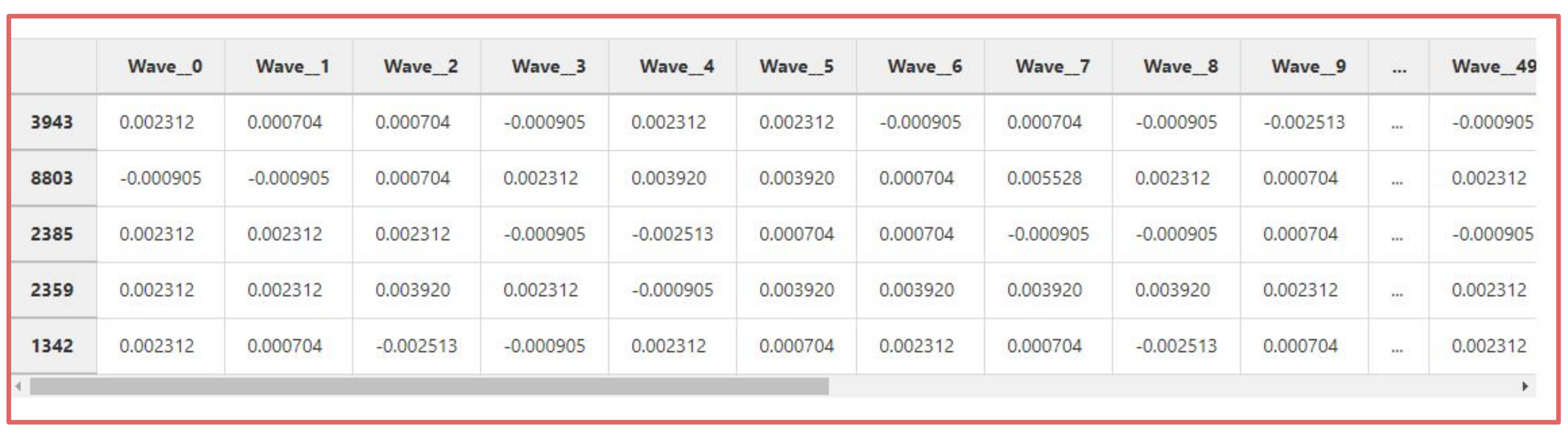
Technical Hardware Details

- Cortex ARM m7 processor - 6 stage pipeline at 20 Mhz
- Tektronix DPO3012 Digital Oscilloscope
- Windows computer running Matlab

- Visa Communication between Matlab and Oscilloscope
- Serial communication between Matlab and nucleo board







	Wave_0	Wave_1	Wave_2	Wave_3	Wave_4	Wave_5	Wave_6	Wave_7	Wave_8	Wave_9	...	Wave_49
3943	0.002312	0.000704	0.000704	-0.000905	0.002312	0.002312	-0.000905	0.000704	-0.000905	-0.002513	...	-0.000905
8803	-0.000905	-0.000905	0.000704	0.002312	0.003920	0.003920	0.000704	0.005528	0.002312	0.000704	...	0.002312
2385	0.002312	0.002312	0.002312	-0.000905	-0.002513	0.000704	0.000704	-0.000905	-0.000905	0.000704	...	-0.000905
2359	0.002312	0.002312	0.003920	0.002312	-0.000905	0.003920	0.003920	0.003920	0.003920	0.002312	...	0.002312
1342	0.002312	0.000704	-0.002513	-0.000905	0.002312	0.000704	0.002312	0.000704	-0.002513	0.000704	...	0.002312

Software

- Oscilloscope data is a time series with voltage on the y-axis
- Data is transformed using Fourier transform (1D) or continuous wavelet transform (2D)
- Scikit-Learn and Tensorflow, Numpy, etc. libraries for Python
- Train on GPUs

- Multiple models:
  - Single instruction predictions by instruction type
  - Multiple instruction multiclass/multilabel prediction. Trained on permutations of two instructions

Testing

- For both single instruction and multiple instruction classification, we successfully train a random forest classifier and a time series forest classifier using mean and standard deviation features extracted from random intervals of the time series data. Data is split 80/20 for training and testing.
- For each possible label in the classification task, 10,000 time series samples are created by polling our EM probe 5000 times over 12 of the processor's clock cycles. To limit the scope of required data collection, we focused on classification between different functional groups from the STM32 instruction set, i.e. Memory access instructions and General data processing instructions.
- Using a selection of instructions that vary most from another allows us to achieve up to 98% classification accuracy for both single and multi class tasks.