# HARDWARE SECURITY FOR CLOUD BASED FPGAs

An Undergraduate Research Scholars Thesis

by

ROHITH RAMANUJAM KUMAR

Submitted to the LAUNCH: Undergraduate Research office at
Texas A&M University
in partial fulfillment of requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by
Faculty Research Advisor: Kevin Nowka
JV Rajendran

May 2021

Major: Electrical Engineering

# RESEARCH COMPLIANCE CERTIFICATION

Research activities involving the use of human subjects, vertebrate animals, and/or biohazards must be reviewed and approved by the appropriate Texas A&M University regulatory research committee (i.e., IRB, IACUC, IBC) before the activity can commence. This requirement applies to activities conducted at Texas A&M and to activities conducted at non-Texas A&M facilities or institutions. In both cases, students are responsible for working with the relevant Texas A&M research compliance program to ensure and document that all Texas A&M compliance obligations are met before the study begins.

I, Rohith Ramanajam Kumar, certify that all research compliance requirements related to this Undergraduate Research Scholars thesis have been addressed with my Research Faculty Advisors prior to the collection of any data used in this final thesis submission.

This project did not require approval from the Texas A&M University Research Compliance & Biosafety office.

# TABLE OF CONTENTS

## CHAPTERS

Page

# ABSTRACT

Cloud Based FPGA Hardware Security

Rohith Ramanajam Kumar
Department of Electrical and Computer Engineering
Texas A&M University

Research Faculty Advisor: Kevin Nowka
Department of Electrical and Computer Engineering
Texas A&M University

Research Faculty Advisor: JV Rajendran
Department of Electrical and Computer Engineering
Texas A&M University

Cloud compute is an opportunity that has been in the public conscious for some time now, however more recently we have tried to introduce FPGA's into cloud-based systems. This offers the advantage of parallelizing a workload and results in substantially higher output per voltage. However, FPGA's have several security vulnerabilities as they can be exploited to shut down systems or to gain unauthorized access to models of other clients running on the same platform.

This research is important in the cloud compute space as FPGA's have the potential to substantially speed up server performance per Watt but currently the security risks are vast. By removing these exploits, clients may be more willing to send sensitive tasks into the cloud thus reducing client hardware costs and hosts stand to increase revenue stream.

In this project we will attempt to replicate a cloud-based FPGA system using a Stratis V FPGA. We will use this system to run a ML model and then run an attack to extract the model or to disrupt the function of the FPGA itself. Finally, we will attempt to create a defense of the FPGA system.

# DEDICATION

*To Prem and Raji as well as my friends, families, instructors, and peers who supported me throughout the research process.*

# ACKNOWLEDGEMENTS


## Contributors

I would like to thank my faculty advisors, Dr. Nowka and Dr. Rajendran, for their guidance and support throughout the course of this research as well as Shankari G for her invaluable guidance through this project.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience.

The code analyzed/used for the research were provided by Shankari G and on open source resources.

All other work conducted for the thesis was completed by the student independently.

## Funding Sources

Undergraduate research was supported by the Electrical and Computer Engineering Department at Texas A&M University.

# NOMENCLATURE

| | |
|---|---|
| AI | Artificial Intelligence |
| ML | Machine Learning |
| FPGA | Field Programmable Gate Array |
| CNN | Convoluted Neural Network |
| V | Volts |
| MNIST | Modified National Institute of Standards and Technology database |
| IP | Intellectual Property |

# 1. INTRODUCTION

This Undergraduate Research Scholars capstone project sponsored by Kevin Nowka and JV Rajendran will explore the topic of hardware security in cloud FPGA systems. In machine learning and in design as a whole functionality is the main standard by which designs are often validated. But the security of the system is often neglected. Whether it be a simple coffee machine, or a ML program designed for national defense, security is often expected but not designed for. As we move machine learning and other computationally intensive tasks into the clous, this creates a new channel of vulnerabilities that many designers aren't considering during the design process. A tool like machine learning is susceptible to being stolen from a shared FPGA resulting in lost IP or can be tampered with which can have further reaching consequences.

When complex models are run on cloud FPGA farms, this opens a new and unique set of vulnerabilities. Because programs are being run on decentralized farms, users can attack each other to extract proprietary models/data or simply attempt to disruptive services. The attacks and defenses on cloud-based FPGA systems is a relatively unexplored area of research that will have important consequences in the future.

## 1.1 Background

The presence of cloud computing has become prevalent in industrial applications and has been pushed by major organizations such as Amazon (AWS) and Microsoft (Azure). This push towards cloud computing can be attributed to 3 main drivers: flexibility, efficiency, and strategic value. In terms of flexibility, companies can use as much hardware capability as they need at a given time on demand with the options to choose the best tools needed for the specific

application at hand. The efficiency of cloud technologies is driven in part by the same factors that drive flexibility. Because companies like Amazon can buy the infrastructure needed in much larger quantities than typical, the economies of scale bring down cost and allow for easy upgrades to the latest standards in use. The virtual nature of the cloud also allows for networked backups and higher accessibility. Ultimately the largest reason for organizations to adopt cloud computing servers is strategic value. Because cloud service providers handle the infrastructure, organizations can devote more resources to the core business and pivot more nimbly than in house server shops.

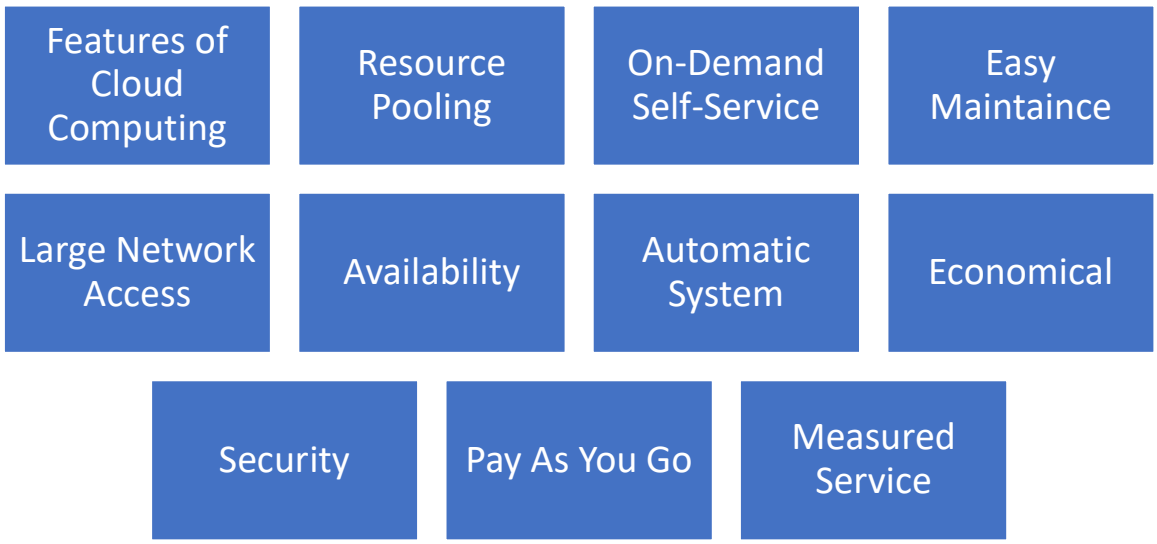


*Figure 1-1.* Advantages of cloud compute

Even though the implementation of FPGAs is being driven by ML, these same technologies hold the keys to exploiting vulnerabilities and mitigating these vulnerabilities as well. The authentication of FPGA's has been a traditional downside of the technology due to its lack of flash or other nonvolatile memory. To surmount this problem physical unclonable functions are used. These PUFs are an object that can take an input and conditions to produce a

unique hardware defined fingerprint as an output. However, ML possess a major threat to this security feature. Using pure ML attacks, the hardware mapping used to generate the unique fingerprint can be accurately predicated with a relatively small sample set. A way to further leverage ML may be to utilize side channel data in the model – for instance using capacitive crosstalk to glean data on encryption keys. From a defensive security standpoint ML tools can be used to analyze and hunt for latency, electromagnetic leakage etc. cause by hidden trojan circuits. Computer vision techniques using tools like meridian vision systems can also be used leveraging ML to identify security risks in ICs produced at outside fabs. However, the techniques may be prohibitively expensive for the scope of URS. Therefore, we would like to focus on modeling structural details and AI specific attacks by embedding trojans, changing model weights etc.

## 1.2 Overview

The design component of this project will comprise of 4 stages. First, we will need to design and build a hardware & software framework that can emulate the infrastructure used in cloud computing applications – specifically the FPGA systems used. To this we will use available Intel Arria-10 FPGAs with a Quartus software layer. Next, a machine learning application must be constructed with valid datasets and the efficiency of the developed ML must be tested and validated. This model will then be sent to the FPGA and we will ensure that the model works more efficiently on a FPGA platform. Then a viable attack must be formulated that expose vulnerabilities in the hardware stack up or in the programming path. Finally, either hardening algorithms or other hardware strategies must be developed in parallel with methods by which vulnerabilities may be exploited. An important avenue to develop countermeasures will be for hardware specific vulnerabilities such as trojan circuits.

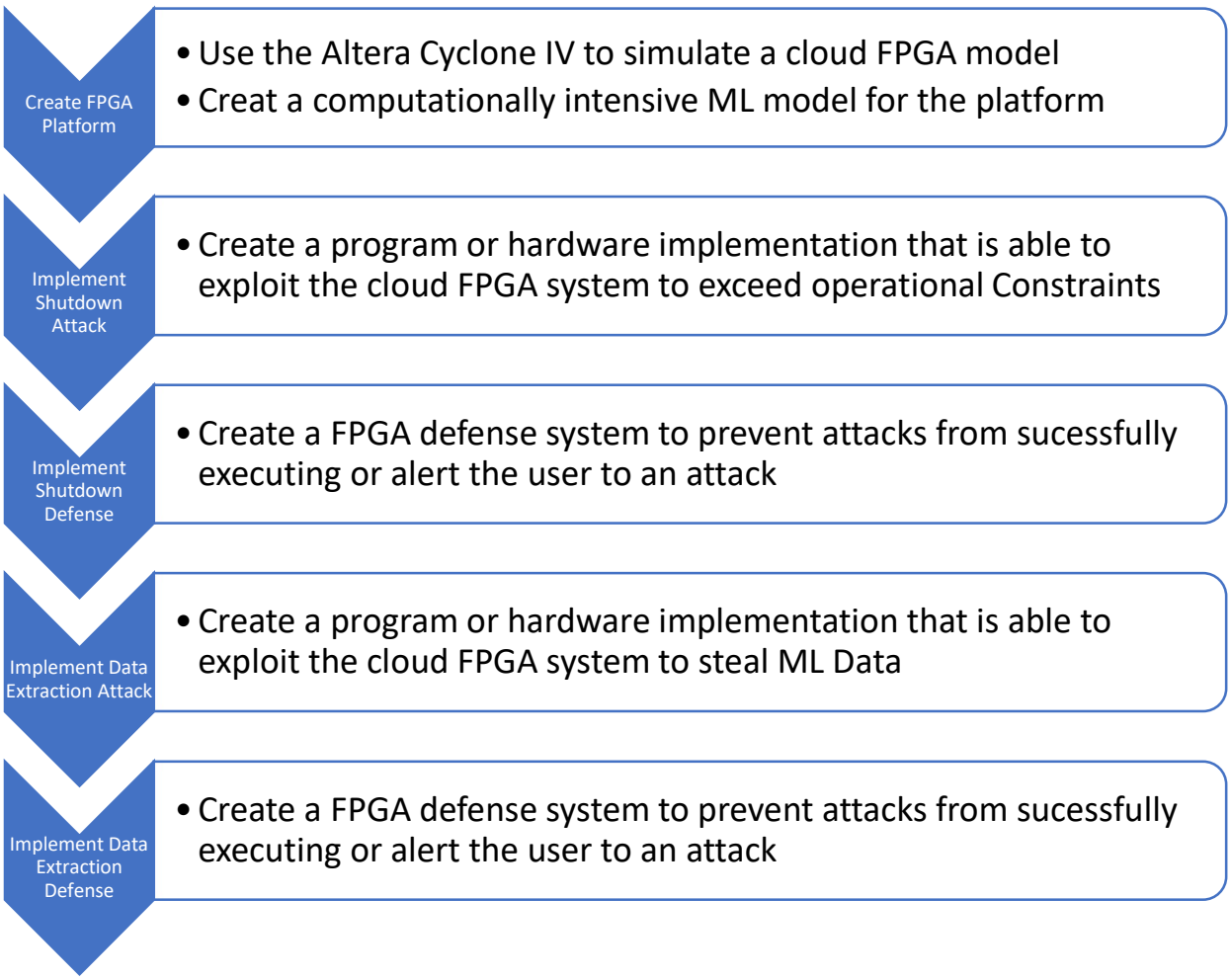


*Figure 1-2. Project steps*

# 2. METHODS

Initially the ML for this project was to be generated using the Intel OpenVino platform, however OpenVino only supports the higher end of Intel FPGAs and they are out of the operational budget of this project. Therefor a ML model must be developed in the lab with support from

## 2.1 Operational Description and Constraints

This project will be used as a platform to simulate cloud-based FPGA systems that are currently being introduced into the market. That is the project will be able to take at least one computational task and run it successfully on an FPGA. A true cloud FPGA platform may run several operations on the same FPGA, but this is not required to replicate the cloud FPGA environment for hardware security purposes.

We are constrained by cost as most FPGA's used in industrial cloud applications will perform at a very high level and thus be expensive. However, a cheaper FPGA board should be able to replicate the core functionality.

## 2.2 System Description

The system consists of three primary systems: The cloud FPGA system itself, an attack, and a defense. Each of these systems is described in further detail below.

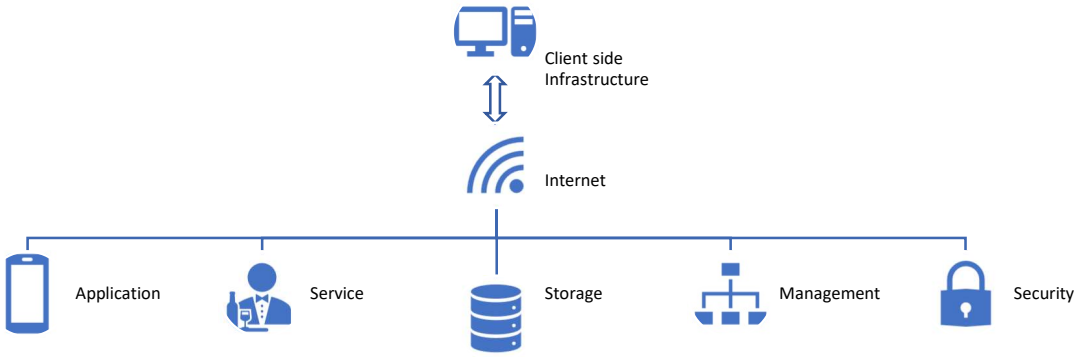


*Figure 2-1. Cloud compute system architecture*

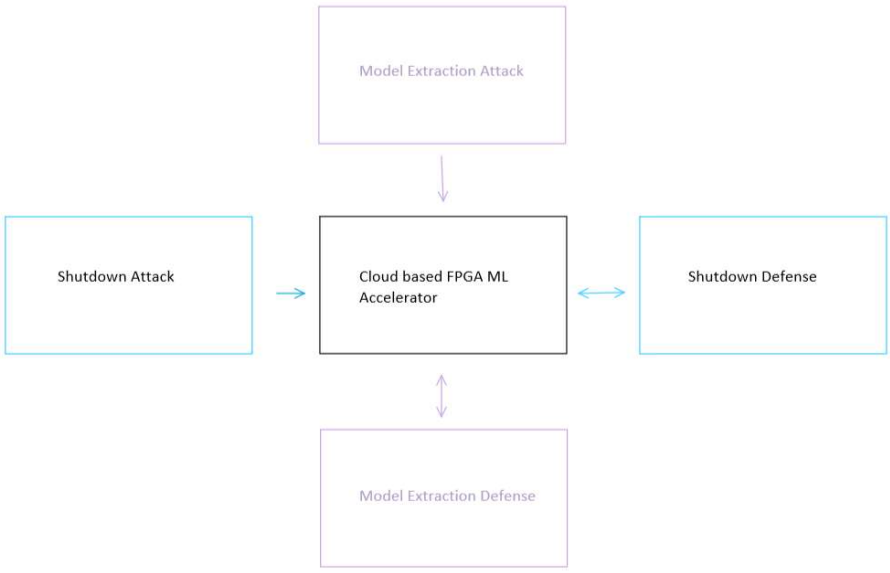


*Figure 2-2. Block Diagram of System. Black represents the core system. Blue represents core functional attack/defense and purple represents noncore functionality.*

### *2.2.1 FPGA Platform*

The FPGA platform is used to replicate a cloud-based FPGA server. On this we will have a computer-FPGA platform on which we can run ML or other models. This system will be comprised of an FPGA (potentially the Cyclone IV) and a ML model. The specific ML model is not necessarily important, but we will be creating one for the data extraction model. We may potentially run multiple models on the FPGA to better replicate a true cloud environment.

Converting a ML model into a Verilog based model that a ML system can process is a nontrivial task

### *2.2.2 Attack*

The system that is referred to as the attack for this project is a tool that we will use to disrupt the FPGA Platform system in any way. This system will strive to shut down the FPGA Platform or in an ideal scenario actively steal data on the FPGA Platform. This system will be code based and will be written in hardware description language.

After simulating the neural network and synthesizing it into RTL, it looks as pictured below. The neural network is by and far the largest portion of the schematic but there are still some challenges as the weights that need to be extracted still constitute a small number of registers when compared with the entire model. Thus, they will be difficult to isolate and launch an attack on. This is illustrated in the below image.

*Figure 2-2. RTL Schematic of Neural Network (bottom) and surrounding structures (top)*

The below image describes the resource allocation of the neural network. This is problematic in the attack process as it is not clear where each component is. However, in a shared resource setting like we wish to explore in this paper, the users are assigned rationed sections of the hardware which provides a conveniently packaged victim unit. The database of weights will be separately partitioned for the purposes of this attack.

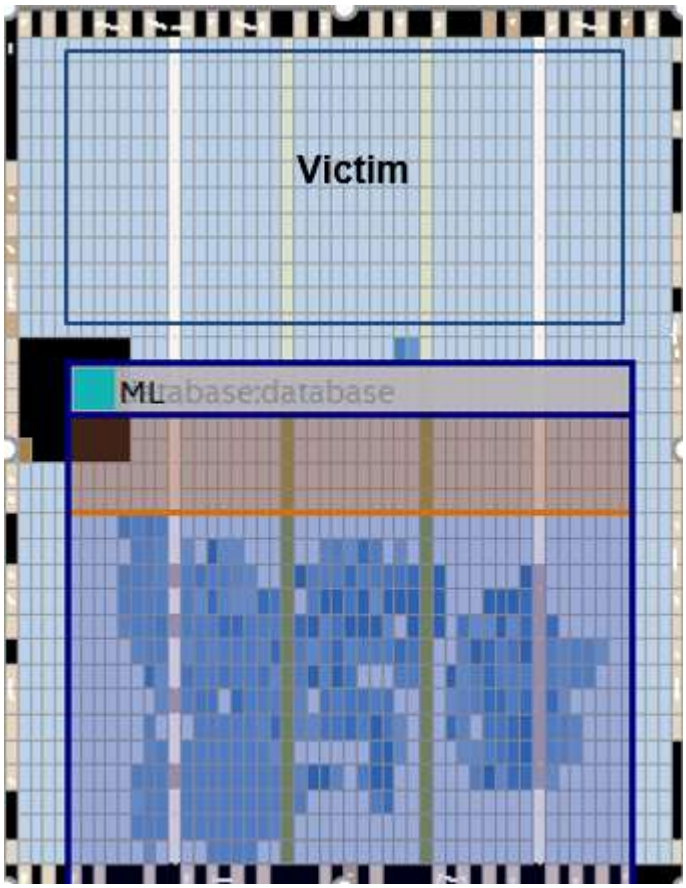


*Figure 2-3. Resource allocation of the Neural Network*

2.2.2.1 Shutdown Attack

In the shutdown attack the logical elements need to be rapidly triggered between 0 and 1 to fluctuate the power of the boards and thus produce heat which will damage the shared resource in question. This is difficult as synthesis tools inherently optimize away structures capable of creating this behavior and a large amount of overhead is needed to facilitate the heat producing structures. My approach to this problem is to create clocks using or gates and nots and instantiating these until all logical elements are used. To trigger these structures a shared register is used which pushes a constant 0 to the or gate which is then not'ed to be a 1. The resulting loop oscillates, and the output is or'd with the other n instances in the hardware.

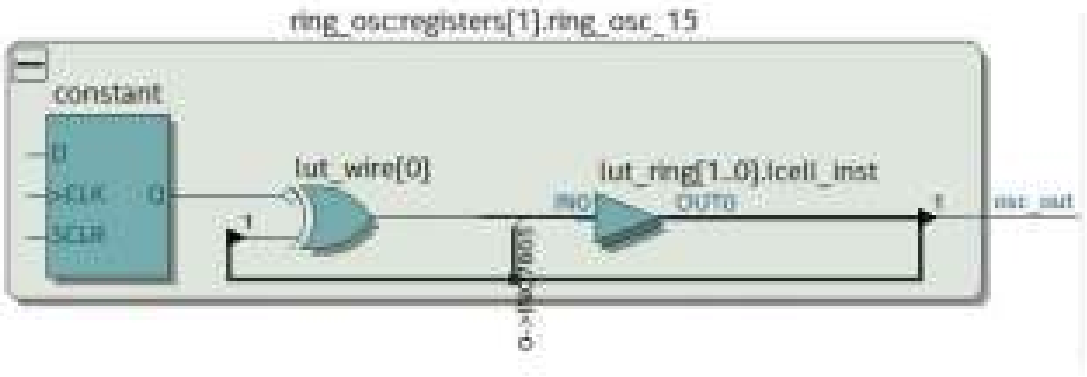


*Figure 2-4 The structure of the heat generating structure*

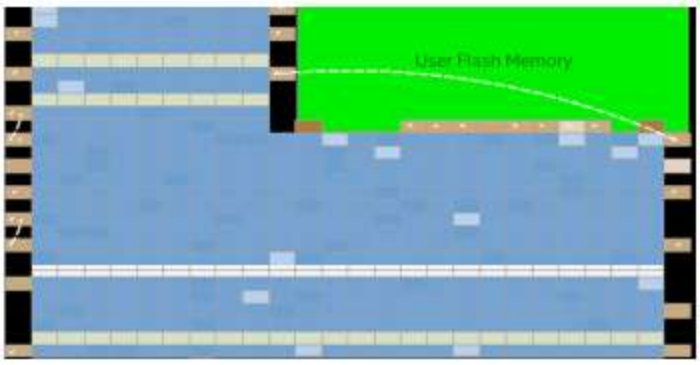


*Figure 2-5. the resource allocation of the structures (currently 77%)*

2.2.2.2 Extraction Attack

The extraction attack is based on a time to digital converter based on the carry primitive. While the weights and biases are being pulled out of RAM to be used in the convolutions a power-based side channel attack can be deployed to read the bitstream. By comparing the clock frequency of various sensors, we can determine if the RAM is outputting a 0 or a 1 at that instance. This data needs to be cascaded down through the carry logic so that we have enough resolution into the bitstream. Carry logic is by and far the fastest primitive on the FPGA fabric.

### *2.2.3 Defense*

The defense will be a layer built on top of the FPGA Platform. It will contain a defensive protocol that can complete two tasks. First it should be able to alert the host of the server that an attack has been launched. Secondly it should provide the host with some mechanism by which to deploy a defense that will prevent data from being stolen from the FPGA. The defense system may also have some sort of passive system of firewall that prevents an attack from successfully executing in the first place. This system will be code based and will be written in hardware description language.

The weight extraction attack works on the premise that a user can utilize the shared hardware to read the power consumption of the device and extrapolate the victim's data. Thus, any sort of physical barrier to this communication will defend the victim. For instance, ring oscillators or any other high frequency structure around the victim side of the implementation will prevent an attacker from extracting data in the system by adding noise in between the sensors and the bitstream that we would like to sample.

## 2.3 Users

There are two fundamental types of users on a cloud-based FPGA system: clients and hosts. Ideally the client can use the system with a low bar to entry and is able to easily push computationally intensive tasks offsite. Although this is probabilistically not the most common scenario as a use case that is complex enough to warrant use of cloud compute services is probably of a certain level of complexity.

The host has a higher bar in terms of training as they are required to fully maintain, expand, and segment the hardware that is being used in the cloud application. The host also needs to have a higher level of training to monitor clients and ensure that no one is using the

platform to launch malicious attacks against each other. The host is responsible for making determinations on maintaining client data security in the event of an attack; Should the servers be taken down and process' terminated? The host also needs to design defenses and keep the systems updated to prevent successful attacks.

Both parties benefit from cloud compute FPGA hardware security as users will be able to send files more confidently with proprietary data to the host with confidence that unauthorized outside sources do not gain access to the data. The host benefits as enhanced security drives up confidence in the platform and thus usage and revenue.

The third notable type of user is the attacker. This person is trying to shut down the system or gain unauthorized access to other users' data. This system strives to make use of the system for these types of users more difficult.

# 3. FUNCTIONAL SYSTEM REQUIREMNTS

This section defines the minimum requirements that the development item(s) must meet. The requirements and constraints that apply to performance, design, interoperability, reliability, etc., of the system, are covered.

## 3.1 Functional / Performance Requirements for ML Acceleration

### *3.1.1 ML Classification*

The ML model should classify a test subset of the MNIST database and classify the input number with at least 80% accuracy.

*Rationale: Industry Norm*

### *3.1.2 Analysis Time*

The neural network should be able to classify an input within 5 seconds.

*Rationale: Industry Norm*

### *3.1.3 Stable Operating Temperature*

The temperature of the FPGA should not deviate more than 2 degree from the ambient temperature of the room during the normal test case.

*Rationale: Industry Norm*

### *3.1.4 FPGA Hardware*

The hardware accelerator in this use case shall an Intel FPGA and be designed around Intel primitives. The ML should not use more than 50% of the available resources on the FPGA so that there is space for a secondary user (i.e. An attacker)

*Rationale: Industry Norm & Specified by Sponsor*

## 3.2 Functional / Performance Requirements for Shutdown Attack

### *3.2.1 Resource exploitation*

The shutdown attack should exploit resources available to the system to force a shutdown of the system to prevent further damage. Some operational requirement should be violated.

*Rationale: Specified by Sponsor*

### *3.2.2 Permanent Damage to the FPGA Subsystem*

The attack should create permanent hotspots on the FPGA hardware that no longer function after it is run.

*Rationale: Specified by Sponsor*

### *3.2.3 Modular size*

The attack size should be modular so that it can be easily adjusted to target various FPGA families and fit in with various sizes of other users.

*Rationale: Specified by Sponsor*

### *3.2.4 Minimum size*

The primitive of the attack – smallest size of a single heater should not exceed 500 logical elements so that the size of the attack can be controlled with precision.

*Rationale: Specified by Sponsor*

## 3.3 Functional / Performance Requirements for Data Extraction Attack

### *3.3.1 Model Extraction*

The extraction attack should be able to read data from the victim side of the board.

*Rationale: Specified by Sponsor*

### *3.3.2 No Contact*

The extraction attack cannot contact or read any outputs from the system directly. The only shared signals should be system limited signals such as clocks.

*Rationale: Specified by Sponsor*

### *3.3.3 Minimum size*

The attack should not constitute more than 50% of the FPGA fabric so that a secondary user can operate.

*Rationale: Specified by Sponsor*

### *3.3.4 Detection avoidance*

The attack should not exploit a resource in any manner to trigger a shutdown. The attack is meant to be camouflaged.

*Rationale: Specified by Sponsor*

## 3.4 Functional / Performance Requirements for Defense

### *3.4.1 Voluntarily Power Down System*

The system shall voluntarily power down to prevent a model from being extracted or to prevent permanent damage.

*Rationale: Specified by Sponsor*

### *3.4.2 Alert User to Attacks*

The system shall alert the cloud host if an attack is being run on the board to either damage the board, slow down process' or extract a model

*Rationale: Specified by sponsor*

### *3.4.3 Attack Detection False Positive Rate*

The false positive rate of an attack alert should be less than 50%

*Rationale: Specified by sponsor*

### *3.4.4 Attack Detection False Negative Rate*

The false negative rate of an attack alert should be minimal for all cases and shall be less than 10% for extraction-based attacks

*Rationale: Specified by sponsor*

## 3.5 Software Requirements

### *3.5.1 Use of Quartus*

The Neural network acceleration in RTL and the attacks and defenses (code based) shall be done on the intel Quartus system.

*Rationale: Intel has a large market share amongst cloud compute applications*

## 3.6 Input/output

### *3.6.1 Hardware Voltage*

The hardware accelerator should be powered with an input voltage between -0.5V and 4.9V.

*Rationale: Specified in Intel Quartus documentation*

### *3.6.2 ML Output*

The training of the model on the FPGA should take the input and classify it into one of 10 possible numbers

*Rationale: Specified by sponsor*

# 4. RESULTS

## 4.1 Neural Network

In this research we successfully deployed an RTL based ML model that classifies with over 80% accuracy. The model uses multiple levels of 3x3 convolutions to achieve this. Because security is not a core focus of this paper, we will not delve further into the structure, but we will present results. The classifier takes images such as those found in figure one and converts them into images like those seen in image 2 so that it can be processed in RTL.


*Figure 3-1: Sample Input Data*

```
initial
begin
    storage[0] =  10'b1000110000; // [0.546875]
    storage[1] =  10'b1000111100; // [0.55859375]
    storage[2] =  10'b1001000100; // [0.56640625]
    storage[3] =  10'b1001001000; // [0.5703125]
    storage[4] =  10'b1001010000; // [0.578125]
    storage[5] =  10'b1001010000; // [0.578125]
    storage[6] =  10'b1001011000; // [0.5859375]
    storage[7] =  10'b1001011000; // [0.5859375]
    storage[8] =  10'b1001011000; // [0.5859375]
    storage[9] =  10'b1001011100; // [0.58984375]
    storage[10] =  10'b1001100100; // [0.59765625]
    storage[11] =  10'b1001101100; // [0.60546875]
    storage[12] =  10'b1001101100; // [0.60546875]
    storage[13] =  10'b1001111000; // [0.6171875]
    storage[14] =  10'b1001111000; // [0.6171875]
    storage[15] =  10'b1001111000; // [0.6171875]
    storage[16] =  10'b1001110100; // [0.61328125]
    storage[17] =  10'b1001110000; // [0.609375]
    storage[18] =  10'b1001101000; // [0.6015625]
    storage[19] =  10'b1001100100; // [0.59765625]
    storage[20] =  10'b1001100100; // [0.59765625]
    storage[21] =  10'b1001100000; // [0.59375]
    storage[22] =  10'b1001010100; // [0.58203125]
    storage[23] =  10'b1001010000; // [0.578125]
    storage[24] =  10'b1001001100; // [0.57421875]
    storage[25] =  10'b1001001000; // [0.5703125]
    storage[26] =  10'b1001000100; // [0.56640625]
    storage[27] =  10'b1000111000; // [0.5546875]
    storage[28] =  10'b1000111100; // [0.55859375]
    storage[29] =  10'b1001000100; // [0.56640625]
    storage[30] =  10'b1001001100; // [0.57421875]
    storage[31] =  10'b1001010000; // [0.578125]
    storage[32] =  10'b1001011000; // [0.5859375]
    storage[33] =  10'b1001011100; // [0.58984375]
    storage[34] =  10'b1001100100; // [0.59765625]
    storage[35] =  10'b1001100100; // [0.59765625]
    storage[36] =  10'b1001101000; // [0.6015625]
    storage[37] =  10'b1001101000; // [0.6015625]
```

*Figure 3-2: Sample testbench where each pixel is converted into a bitstream.*

This testbench is used with the Verilog ML model to simulate a cloud-based classifier. The code works in simulation and figure 1-3 demonstrates that the Verilog code compiled and can generate a bitstream, further in this document we will see the code running in simulation.

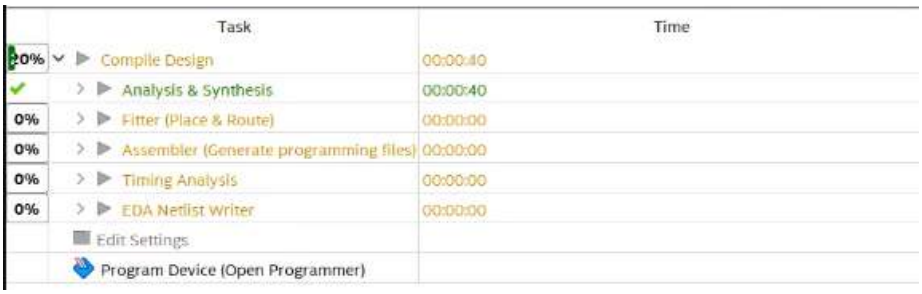


*Figure 3-3: ML model being successfully compiled in Quartus*

Figure 1-4 demonstartes simulation using modelSim Alterra. The ML model is being run with the produced testbench over top. This particular example is sucessfully classifying the number 5.

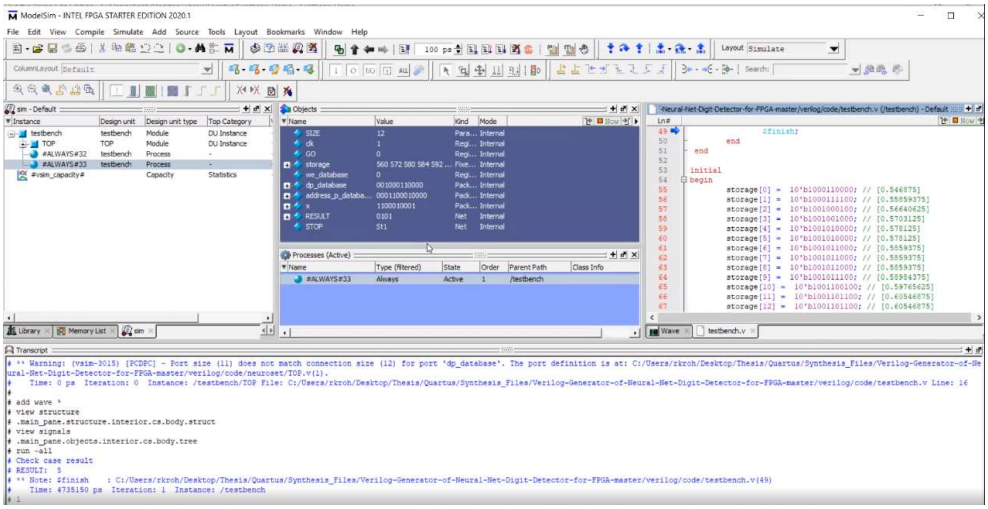

*Figure 3-4: Alterra ModelSim view of a testbench being processed.*

The process of simulation was used for 20 different inputs and we can see that the ML model in RTL works identically to a python model as expected with a high degree of accuracy.

Based on the test of the python model we had an accuracy of 97% which can be extrapolated to the RTL Model.

*Table 1: ML Results from Python ML and RTL ML*

| Input Value | Output Value - Python | Output Value - RTL |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 5 | 5 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 2 | 2 | 2 |
| 2 | 2 | 2 |
| 3 | 3 | 3 |
| 3 | 3 | 3 |
| 4 | 4 | 4 |
| 4 | 4 | 4 |
| 5 | 5 | 5 |
| 5 | 5 | 5 |
| 6 | 6 | 6 |
| 6 | 6 | 6 |
| 7 | 7 | 7 |
| 7 | 7 | 7 |
| 8 | 8 | 8 |
| 8 | 8 | 8 |
| 9 | 9 | 9 |
| 9 | 9 | 9 |
| 10 | 10 | 10 |

After converting the RTL to synthesizable code, we further tested the voltage characteristics of an Intel FPGA and tested the accuracy of the ML model in hardware using over 150 testbenches. This was a time consuming as the synthesizer took nearly 3 minutes to process each input. An interface to easily feed inputs to the system should be built for future testing. The chart below shows the results of our validation. The accuracy of the overall system was 85% and the primary source of error was the number 6 being classified as a letter. Please note that the

number 10 simply indicates NaN. The input and core voltages were constant during the implementation of the ML program.

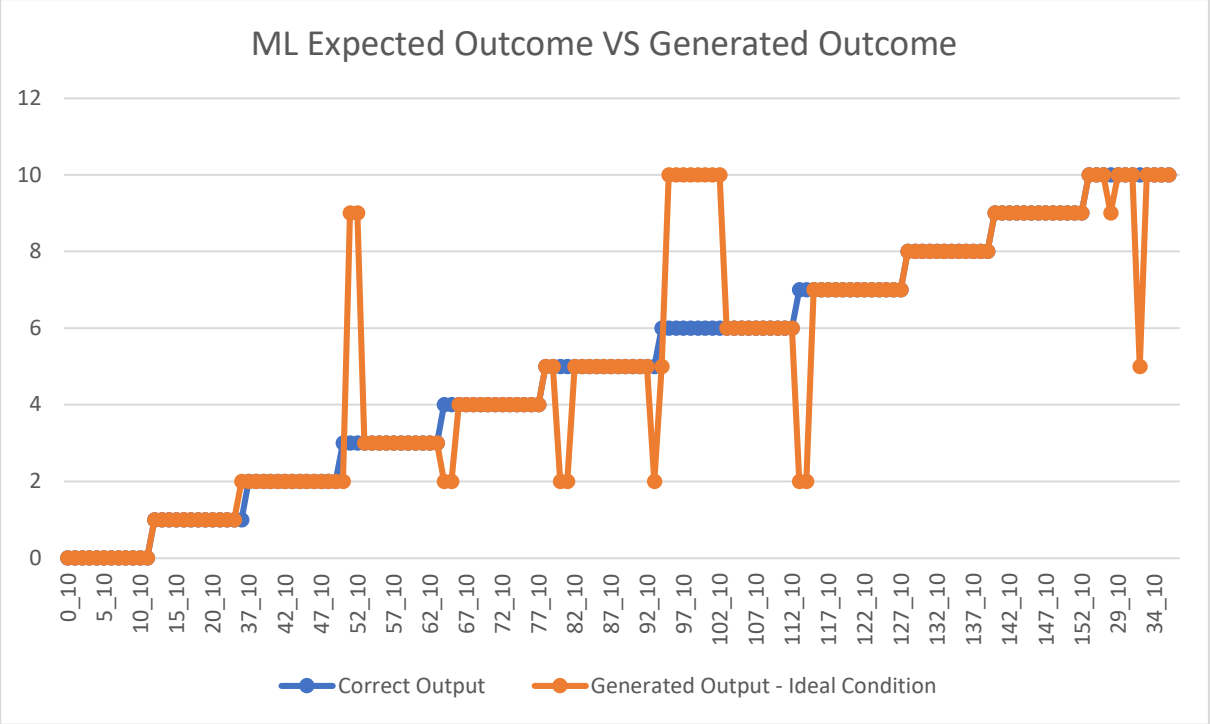


*Figure 3-5: ML model validation data comparing expected and actual results of the test bench.*

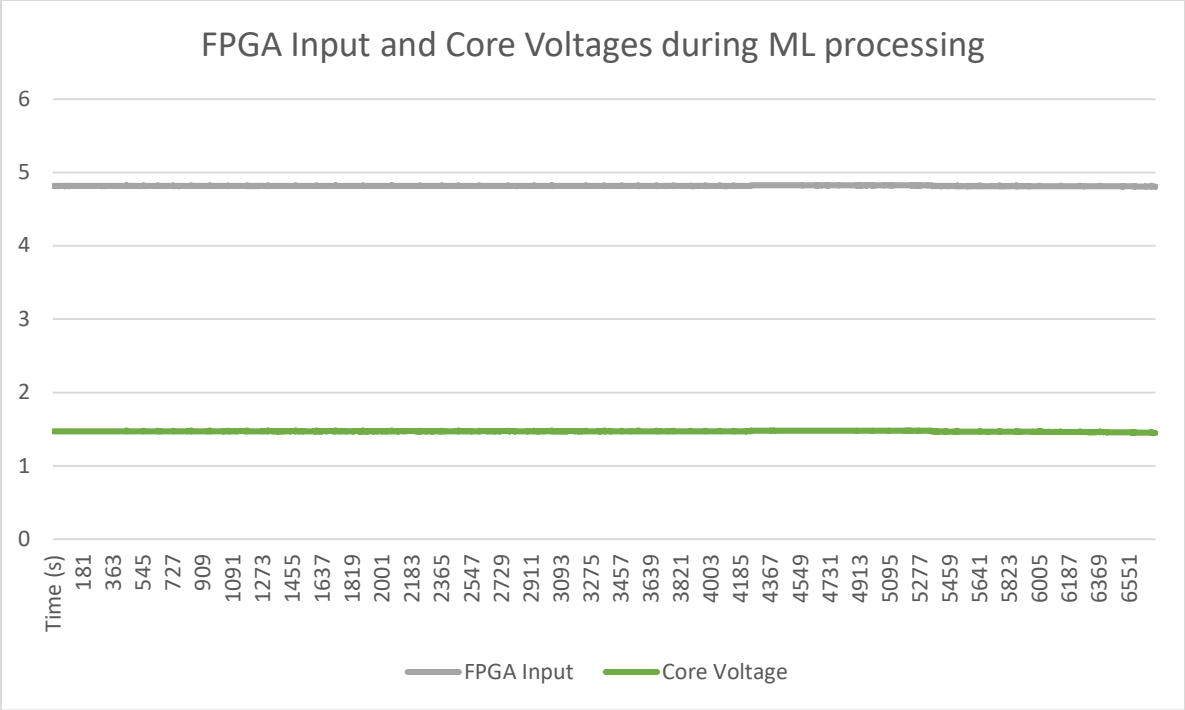


*Figure 3-6: ML model validation data comparing expected and actual results of the test bench.*

## 4.2 Shutdown Attack

The shutdown attack uses ring oscillator like heaters to draw power from the FPGA system and exceed the thermal operating limit of the FPGA thus forcing a cloud server to cease operation or risk damaging the hardware. The maximum temperature of the heaters at different numbers of oscillators was measured. The rows in green resulted in lowered temperatures after a sustained period of operation that could be indicative of LE's burning out.

*Table 2: Temperatures of heaters of various sizes*

| ***# inverters*** | ***Max Temperature (C)*** |
|---|---|
| *0* | *30.3* |
| *1* | *92.5* |
| *1 (full utilization)* | *96.5* |
| *2* | *68.5* |
| *7* | *44.2* |
| *8* | *42.1* |
| *97* | *31.8* |
| *98* | *30.3* |

Below we detail the voltage characteristics and temperatures of the FPGA at different numbers of ring oscillators. The 1 inverter attack is the successful attack that can reach temperatures of 96 degrees when measured using a thermocouple or 120 degrees when measured using the on-board temperature sensor. The successful attack impacts the input voltage of the system in an unexpected manner.

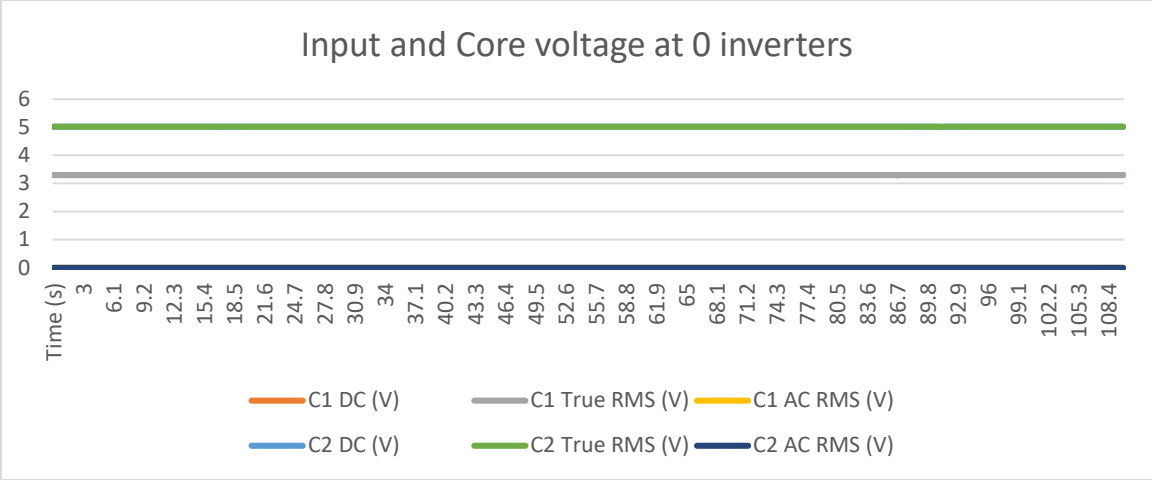


*Figure 3-7: Input and core voltage @0 inverters*

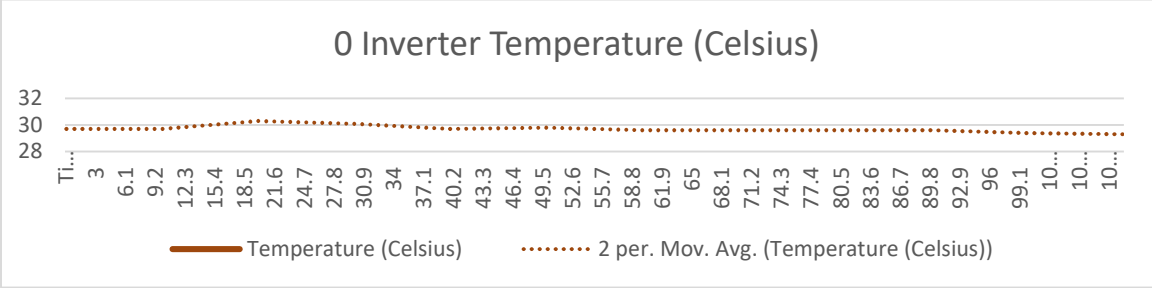


*Figure 3-8: FPGA system temperature while 0 inverter attack is running*

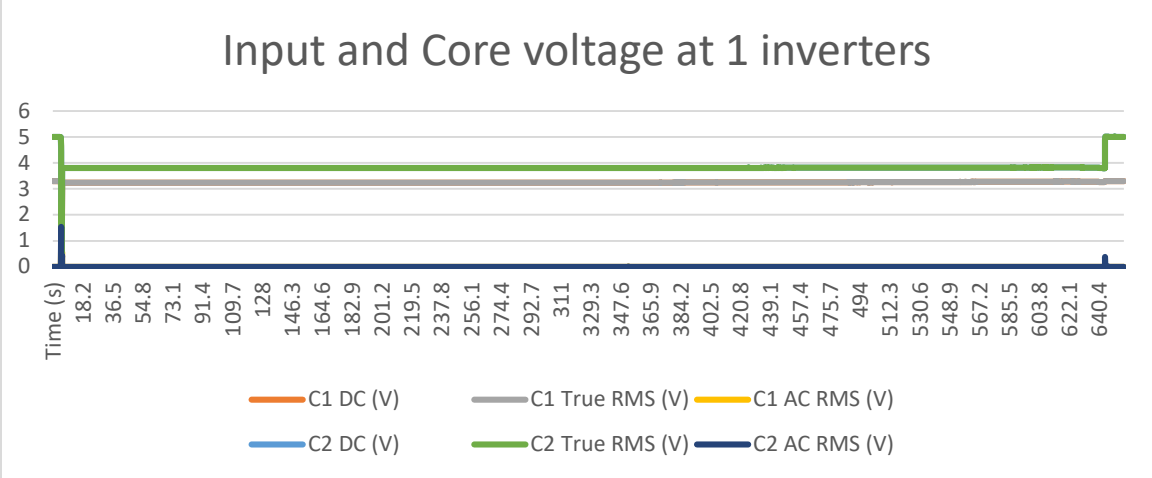


*Figure 3-9: Input and core voltage @1 inverters*

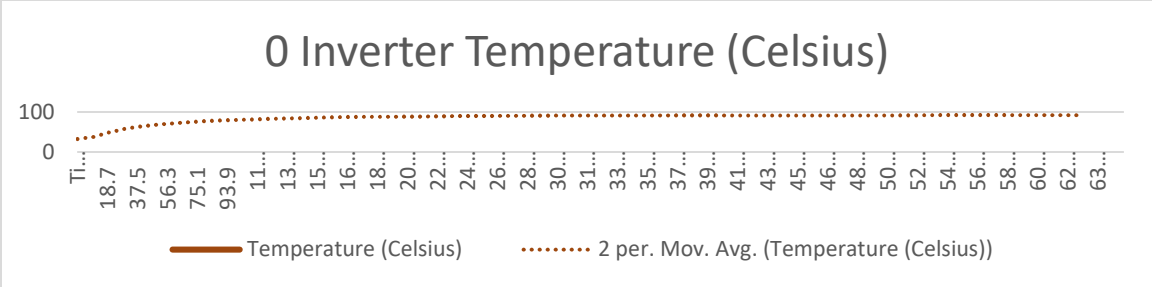


*Figure 3-10: FPGA system temperature while 0 inverter attack is running*

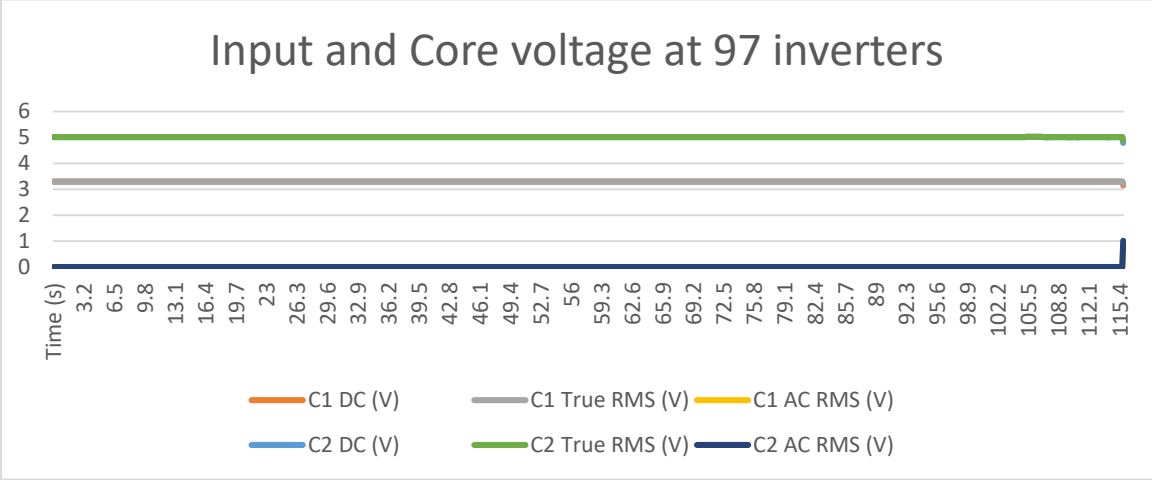


*Figure 3-11: Input and core voltage @97 inverters.*

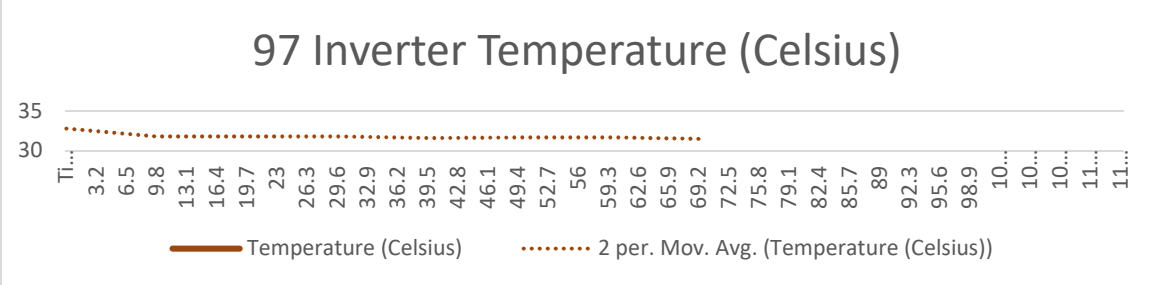


*Figure 3-12: FPGA system temperature while 0 inverter attack is running*

## 4.3 Shutdown Defense

On board diodes on the system were used to create a temperature sensor that alerted the user (cloud admin) that critical temperatures were being reached. The sensor uses the Nios soft processor platform to alert the user to an attack. The below image shows a temperature read out in progress. This can easily be connected to a relay or similar device to terminate power to a section of the datacenter or to monitor which code sample are creating high heat output.

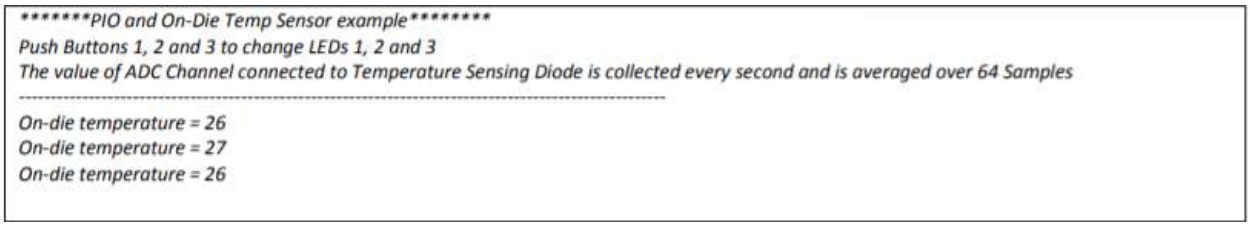


*Figure 3-13: On-board sensor providing user with temperature data*

## 4.4 Extraction Attack

The data extraction attack was intended to use a power side channel attack to read the data moving through the ML subsystem without authorization. Over the course of the project we found that the typical attack using time to digital converters is not possible to implement on an Intel FPGA for the following reasons:

- TDC's are built around the Xilinx 4 bit carry logic while intel only has a 1 bit carry primitive
- The carry itself has no declared inputs (the sensors are reading in other programs inputs) thus there is no fanout and Quartus does not permit the carry tree to be synthesized properly.

- Intel tools do not allow granular control of a PLL based clock divider in the same way Xilinx does.

We contacted Intel to trouble shoot these issues, but we have not been able to successfully synthesize the carry chain logic. Even if it were to have synthesized is it questionable if the synthesized module would have successfully replicated the function of the Xilinx primitive.

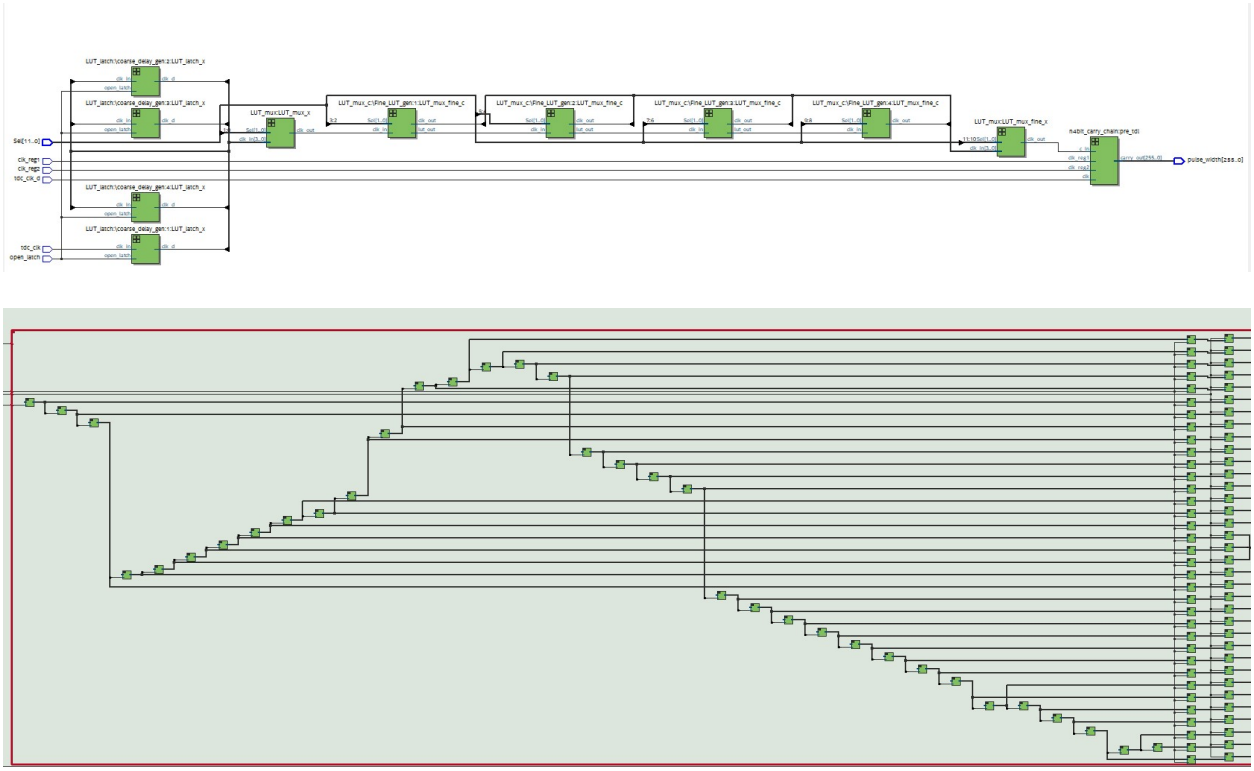

*Figure 3-14: TDC sensor structure (top) and un-synthesizable carry tree (bottom)*

If the extraction model was successfully executed, it could have been defeated by inserting a layer of noise due to ring oscillators or any other chain around the perimeter of the victim logic.

## 4.5 Overall Validation

The following table provides a comprehensive list of the tests run in this experimental process.

*Table 3:* Validation steps

| Test | Success Criteria | Methodology | Requirement | Status |
|---|---|---|---|---|
| **Functional/Performance Requirements for ML Acceleration** | | | | |
| ML Classification | Classify MNIST with 80% accuracy | Run a testbench on the ML | Critical | Passed |
| Analysis Time | The neural network should be able to classify an input within 5 seconds | Measure Processing time | Critical | Passed |
| Stable Operating Temperature | The neural network should not increase the FPGA temperature by more than 2 degrees Celsius | Measure the temperature of the FPGA during compute | Critical | Passed |
| FPGA Hardware | The hardware accelerator in this use case shall an Intel FPGA. | Verify source of chipsets | Critical | Passed |
| FPGA Hardware | The model should not use more than 50% of available hardware resources | Verify LE utilization post synthesis | Critical | Passed |
| **Functional/Performance Requirements for Shutdown Attack** | | | | |
| Resource Exploitation | The shutdown attack shall be able exceed the operational conditions of the hardware | Measure temp of fpga - if heat is outside of threshold damage is induced | Critical | Passed |
| Permanent Damage to the FPGA Subsystem | The attack should create permanent hotspots on the FPGA | measure temp of fpga - if temperature drops then damage is assumed | Critical | Passed |
| Modular size | The attack size should be easily configurable withing the attack | run a range of attacks with different numbers of primitives | Extra | Passed |
| Minimum size | A single attack primitive should not exceed 500 LE | Run attack RTL synthesis with one primitive | Extra | Passed |
| **Functional/Performance Requirements for Data Extraction Attack** | | | | |
| Model Extraction | The attack should be able to read from the victim side of the board | measure if system turns off before temperature parameters are breached | Critical | Failed |
| No Contact | The attack cannot contact the victim | Ensure the RTL schematics do not share non resource inputs | Critical | Passed |

| | | | | |
|---|---|---|---|---|
| Minimum size | the attack cannot be more than 50% of the FPGA fabric | Verify LE utilization post synthesis | Critical | Passed |
| Detection Avoidance | The attack should not trigger thermal or other warnings | Ensure that thermal characteristics and voltage are normal during run | Critical | Passed |
| **Functional/Performance Requirements for Defense** | | | | |
| Voluntary Power Down | The system shall voluntarily power down to prevent permanent damage. | measure if system turns off before temperature parameters are breached | Extra | Passed |
| Alert User | The system shall alert the cloud host if an attack is being run | measure if the system alerts if an attack is detected | Extra | Passed |
| False Positive | The false positive rate of an attack alert should be less than 50% | Measure the number of false positives | Extra | Passed |
| False Negative | The false negative rate of an attack alert should be less than 50% | Measure the number of false negatives | Extra | Passed |
| **Software Requirements** | | | | |
| Use of Quartus | The Neural network and the attacks/ defenses shall be done on the intel Quartus system. | Ensure required tools are used | Critical | Passed |
| **I/O Requirements** | | | | |
| Hardware Voltage | The hardware accelerator should be powered with a voltage between -0.5V and 4.9V. | measure voltage of hardware | Critical | Passed |
| ML Output | The FPGA should take the input and classify it into one of 9 possible digits | test the ML software with a testbench | Critical | Passed |

# 5. CONCLUSION

Over the curse of this project, we were able to show that a successful attack can be launched to force a system to come down using ring oscillators. There may be structures on an Intel FPGA that are subject to more operational constraints and a specific attack can be synthesized to target sections of an FPGA by an experienced user. The attack would also be more effective if the overhead of the structure was cut down. However, stopping this attack once launched is also relatively trivial using temperature diodes. The difficulty will lie in determining which users have legitimate computations that happen to heat the FPGA structure and which users are nefariously trying to manipulate voltage levels and temperatures of the system.

A data extraction attack is possible on a FPGA platform as shown by numerous previous researchers. However, we were not able to synthesize a TDC based data extraction attack on a ML algorithm on an Intel FPGA.

## 5.1 Next Steps

In the future it is necessary to launch a TDC based attack on a ML algorithm on the Xilinx platform. TDC based attacks have been documented on that chipset however the weights and bias of a ML model have not been extracted to the authors knowledge at this time.

Regarding the shutdown attack it is necessary to investigate vulnerabilities at the primitive level and conceptualize an attack that dies not alert the administrator that an attack is underway until the hardware has already been compromised.

# REFERENCES


[1] X. Xu and J. Zhang, “Rethinking FPGA Security in the New Era of Artificial Intelligence,” *2020 21st International Symposium on Quality Electronic Design (ISQED)*, 2020.

[2] J. Rajendran, *Security of Cloud FPGAs: A Survey*, 2020.

[3] “Benefits of cloud computing,” *IBM*. [Online].

[4] R. Solovyev, A. Kustov, D. Telpukhov, V. Rukhlov and A. Kalinin, "Fixed-Point Convolutional Neural Network for Real-Time Video Processing in FPGA," *2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, Saint Petersburg and Moscow, Russia, 2019, pp. 1605-1611, doi: 10.1109/EIConRus.2019.8656778.

[5] Dadouche, F. & Turko, Timothé & Uhring, Wilfried & Malass, Imane & Dumas, Norbert & Le, Jean-Pierre. (2015). New Design-methodology of High-performance TDC on a Low Cost FPGA Targets. Sensors and Transducers. 193. 123-134.

[6] Shayan Moini, Shanquan Tian, Jakub Szefer, Daniel Holcomb, and Russell Tessier. “Remote Power Side-Channel Attacks on CNN Accelerators in FPGAs.” *ArXiv*, 2020.