# The University of Sydney

## Faculty of Engineering

## School of Electrical and Computer Engineering

## Year S1, 2024

| | |
|---|---|
| **Student Name** | **Prosper Su** |
| **SID** | **490447846** |
| **Unit of Study Code and Name** | **ELEC4712 Thesis A** |
| **Supervisor** | **Dr.David Boland** |
| **Title** | **ML on FPGA** |

# Introduction

The aim of the project is to explore the potential outcomes of integrating FPGA hardware with machine learning. The development of machine learning algorithms and FPGA hardware has expanded significantly in past decades, their integration still produces optimisation challenges. To address these challenges, various implementation methodologies such as pipeline systems and pre-classifiers will be employed to design and test the new results. The entire project consists of multiple systems that will generate various results which are compared to select the best performance at the end. The exploration of FPGA software-hardware implementation aims to provide better insights and opportunities for further development in the fields of FPGA technology and machine learning. Hence, this progress report will focus on providing a better view of current progress first.

# Literature Survey

## 1. Introduction

### 1.1 Machine Learning (ML)

Machine learning (ML) is one of the products of artificial intelligence (AI) that enables systems to learn and improve from previous experience without being programmed. To build an ML algorithms model, it requires sample data or training data to make predictions or decisions.

### 1.2 Hardware Acceleration

As machine learning models grow in complexity and size, their computational and memory demands also increase significantly. The exploration of hardware accelerators like GPUs, ASICs, and FPGAs

become more popular than CPUs due to its limitations on performance. FPGAs are this project’s focus because they provide high flexibility, parallel processing capabilities, and energy efficiency.

## 2. Background

### 2.1 Fundamentals of FPGAs

FPGAs are reconfigurable integrated circuits that can be customized for specific requirements to perform various tasks unlike fixed-function ASICs. They are made of an array of programmable logic blocks and these logic blocks can be wired together by a hierarchy of reconfigurable interconnects.

### 2.2 Importance of Quantization in ML

The purpose of quantization in ML is to reduce the precision of the numbers used to represent the model's parameters. This process is important for deploying ML models on hardware as it reduces the memory usage and computational complexity, making the models more suitable for hardware implementation with good performance.

## 3. Literature Review

### 3.1 Machine Learning on FPGAs [3]

The integration of ML on FPGAs has been developed due to the advantages FPGAs offer in terms of high flexibility and efficiency. A systematic literature review by Abu Talib et al. highlights various hardware accelerators such as CPUs, FPGAs, GPUs, and ASICs for ML. The paper also categorizes the application of this software-hardware, the two main applications are image classification and object detection. Several models with convolutional neural networks (CNNs) were trained and deployed on FPGAs. The performance was compared with CPU and GPU implementations, highlighting the advantages of using FPGAs. Object detection is another main

application. Same as previous applications, models are trained with relevant neural networks and deployed on FPGAs. The results comparison with CPU and GPU demonstrates significant improvements in latency and power efficiency.

### 3.2 Larq: Python library for training neural networks [2]

Larq is an open-source Python package designed for building and training Quantized Neural Networks (QNNs) and Binarized Neural Networks. BNNs are simple quantized networks where both weights and activations are restricted to 2 bits which significantly reduces memory size and energy consumption compared to traditional networks. Larq is based on TensorFlow Keras and provides an easy-to-use API for designing and training BNNs. However, during initial implementation, Larq was found to have compatibility issues especially the format and mapping with the Xilinx platform, limiting its utility for FPGA deployment.

### 3.3 Brevitas: building and training package for neural network [1]

Brevitas is also an open-source library designed for quantization training of neural networks in PyTorch. It supports a wide range of quantization schemes and allows users to customize their models. Brevitas enables the development of models that can be efficiently deployed on hardware accelerators, including FPGAs.

### 3.4 FINN: End-to-end FPGA Inference [4]

FINN, developed by Xilinx, is a framework for converting and integrating quantised neural networks into practical FPGA implementations. It provides a simple implementation flow starting from model quantization, hardware generation, and finally deployment. FINN is designed to provide high compatibility with models trained using Brevitas, enabling a flexible and simple end-to-end flow from training to deployment for users.

## 4. Current Progress

To understand the development of ML models into FPGAs, my initial research involved studying the fundamentals of ML and FPGAs. The first step was to understand the Binarized Neural Networks (BNNs). This led me to use Larq which provides details for beginners to test their neural network easily and flexibly. My test was successful, it demonstrated an executable simple binarized neural network and results were acceptable. However, it was previously mentioned that it produced compatibility issues with the target platform.

To address this challenge, I did the research again in terms of the Xilinx platform and discovered the FINN framework developed by Xilinx for converting quantized neural networks into FPGA implementations, which was based on prerequisite Brevitas, an open-source library that also supports the building and training neural networks. Brevitas can be seamlessly integrated with FINN. With the supports of these two complete and user-friendly techniques, users easily build ML models on FPGA hardware without any problems. It also contributes me to effectively implement and test my simple BNNs on FPGAs. The successful results from my experiment were demonstrated to professor during the meeting, further convincing its feasibility.

Once everything is complete, the outputs from FINN framework then will be employed onto practical Xilinx board through another technique called PYNQ. Further details will be explained in next report.

## 5. Conclusion

This literature survey highlights the development process of software-hardware implementations on FPGAs, focusing on relevant technologies in the field of machine learning algorithms. Conducting research initially helps the audience understand the fundamental concepts needed before diving into the critical parts of the project, leading to better insights and knowledge. Testing and building a relevant neural network support audience to know the role of functionality of machine learning. Consequently, the testing results

from the complete system using Brevitas and FINN provide a more comprehensive understanding of the software-hardware design flow and the necessary steps to complete the project.

## 6. References

[1] “Brevitas — Brevitas 0.10.2 documentation,” *xilinx.github.io*. https://xilinx.github.io/brevitas/# (accessed May 24, 2024).
[2] “larq/paper/paper.md at main · larq/larq,” *GitHub*. https://github.com/larq/larq/blob/main/paper/paper.md (accessed May 24, 2024).
[3] M. A. Talib, S. Majzoub, Q. Nasir, and D. Jamal, “A systematic literature review on hardware implementation of artificial intelligence algorithms,” *The Journal of Supercomputing*, May 2020, doi: https://doi.org/10.1007/s11227-020-03325-8.
[4] Y. Umuroglu *et al.*, “FINN,” *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, Feb. 2017, doi: https://doi.org/10.1145/3020078.3021744.