ESDCS 2024 Perceptron Accelerator Project

24414 Gurjot Singh[1]
24580 Asutosh Ratha[2]
Dept. of Electronic Systems Engineering
Indian Institute of Science
Bangalore, India
gurjotsingh@iisc.ac.in[1]
asutoshratha@iisc.ac.in[2]

***Abstract*—In this project, the design for Deep Neural Network (DNN) accelerator for MNIST Digit Recognition Dataset is presented. It is a 784x10, fully-connected, feedforward neural network structure, which uses bfloat16 data type based approximate computations for inference [2]. The sub-modules are designed in Verilog, verified in Vivado, synthesized in Yosys, timing and power analysis is done in OpenSTA and Physical Design with OpenROAD using 45nm Standard Cell library. It has an operating frequency of 66MHz at TT corner, 31mW of Power Consumption and occupies 63000sq. μm of area after Synthesis. We achieve a throughput of 12.5 kSamples/s on the MNIST Data Set at 66 MHz.**

***Index Terms*—DNN, MNIST, bfloat16, Verilog, OpenSTA, YoSys, OpenROAD**

I. Introduction

Deep neural networks (DNNs) have the advantage that they can take into account a large number of parameters, which enables solving numerically complex tasks. They provide high accuracy in many Speech Recognition and Computer Vision applications [1]. This project is used to classify the MNIST digits data at RTL level. The computations are carried out at bfloat16 format, which is widely used for ML and DL applications. The bfloat16 format is less accurate than single precision 32 bit floating point representation, but provides huge advantage in speed, computational efficiency and power consumption, making it suitable for this application.

II. Model Training

Nowadays, one of the most important parameters for any ML algorithm is the data-type and the model architecture employed. The model for the DNN accelerator is decided based on training the available MNIST data in Python (Google Colab). The choice of architecture is based on the trade-off between speed, accuracy and energy per inference. The following sections describes the Neural Network Architecture and its training and testing accuracy.

A. 784-64-64-64-10 model

It uses 3 hidden layers with 64 neurons each. The Validation accuracy is as shown in Fig.2.

B. 784-64-64-10 model

It uses 2 hidden layers with 64 neurons each. The Validation accuracy is as shown in Fig.3.

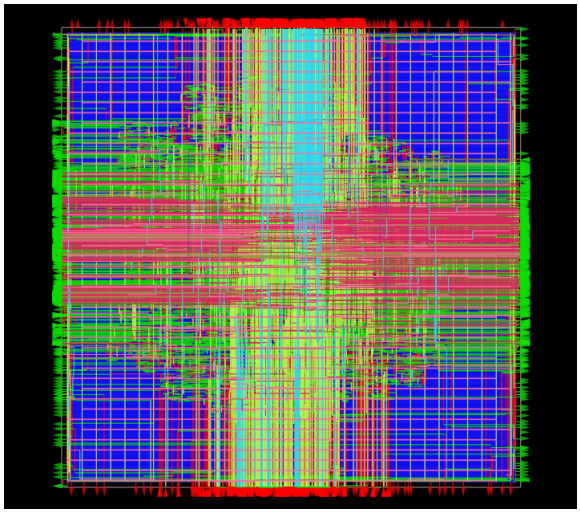

Fig. 1: Synthesized DNN Accelerator IP

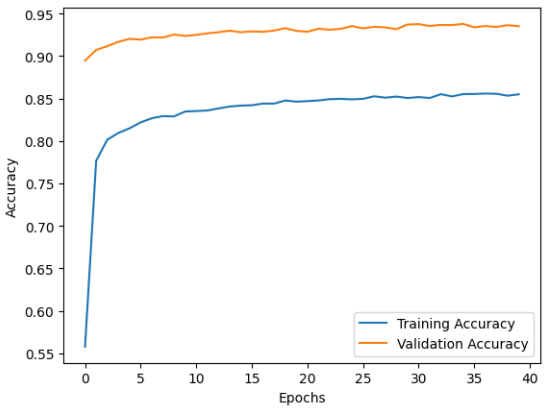

Fig. 2: 784-64-64-64-10 architecture with 93.52% accuracy

C. 784-32-10 model

It uses 1 hidden layer with 32 neurons. The Validation accuracy is as shown in Fig.4.

D. 784-10 model

It uses no hidden layers. The Validation accuracy is as shown in Fig.5. This architecture is very compact is terms of throughput, energy per inference and provides a reasonable

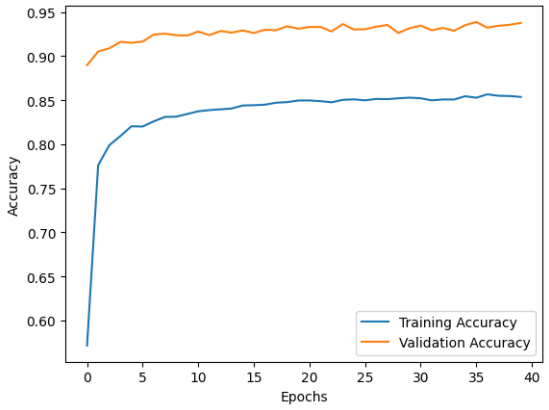

Fig. 3: 784-64-64-10 architecture with 93.19% accuracy

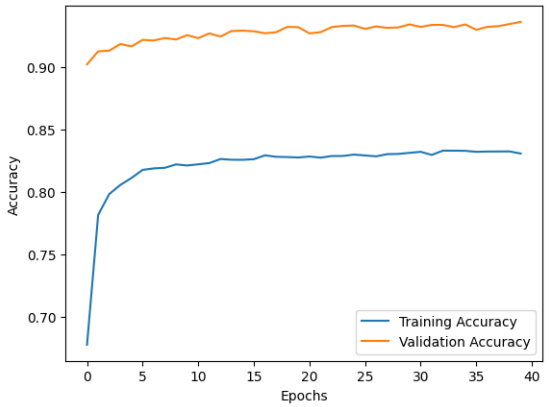

Fig. 4: 784-32-10 architecture with 93.19% accuracy

accuracy. This compactness will reduce energy consumption, making it ideal for energy-scarce environments.

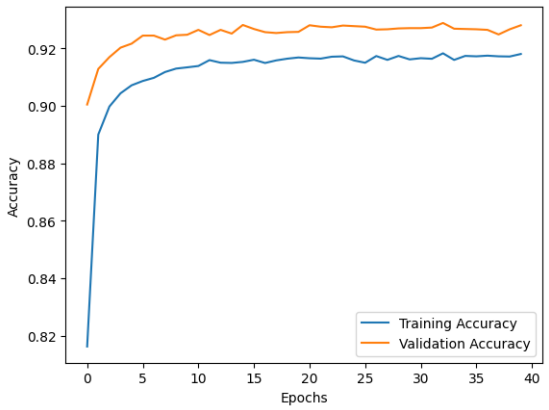

Fig. 5: 784-10 architecture with 92.81% accuracy

III. Architecture and Block Diagrams

The compact 784x10 neural network provides good speed, efficiency and accuracy and but we design and implement a flexible DNN Accelerator Core which can be configured to run any implementation of the Algorithm. This is done during the ”init” phase, where we can initialize the model architecture and layer sizes. The architecture block diagram is shown in Fig.6. We have 32 bfloat16 MACs, which are our Processing Elements PE's.

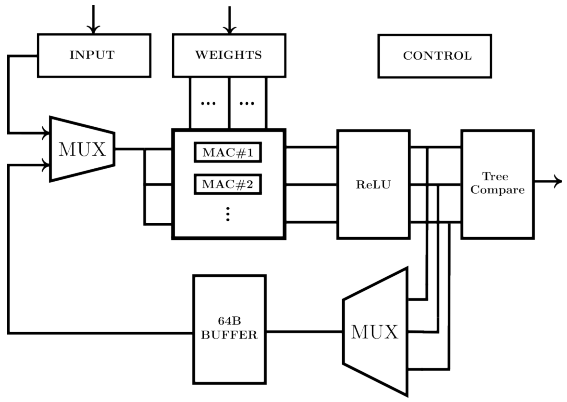

Fig. 6: Hardware Architecture

The control block is very important for this type of Cores. We design the following FSM and verify it's behaviour using multiple layer DNN Algorithms. At the last layer, size 10 tree comparator is used to extract the largest value from the results.

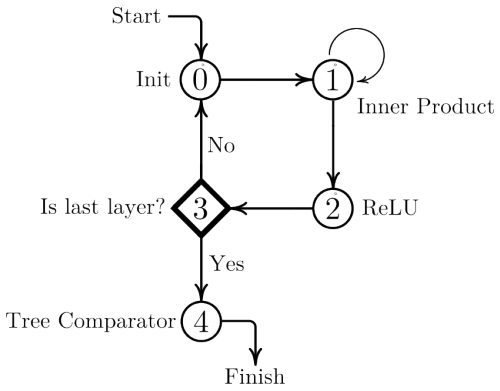

Fig. 7: Control Structure and FSM

IV. Testing of Inputs

Here, we include testing of certain random inputs from the MNIST datasets. We used Python scripts, which normalizes the contents of the input data vector and converts it into bfloat16 and feed to the memory of the testbench. Here, we demonstrate 2 inputs and the corresponding waveforms.

A. *Digit-5*

The MNIST data in PNG for digit 5 is shown in the Fig.8.

Fig. 8: Digit-5

The simulation waveforms are shown in Fig.10

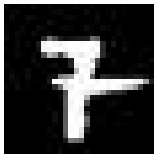

Fig. 9: Digit-7

B. Digit-7

The MNIST data in PNG for digit 7 is shown in the Fig.9. The simulation waveforms are shown in Fig.11

V. Design Trade-off

This project addresses an number of design trade-offs, which are as shown below.

A. Brain Float 16 Computation

This is a type of approximate computing, which is used for improving the computation speed, less hardware cost, area and power consumption, while sacrificing the computation accuracy. This number representation is widely used in ML and DL applications, where speed and efficient operations are important, since the model parameters are already trained, considering the Quantization aware training(QAT), which provides good accuracy during validation.

B. Compact Model of 784x10 Neural Network

This neural network, surprisingly provides reasonable accuracy during training but less accurate than other networks(less than 1%). This architecture saves computations, energy and thus is a viable option for implementation.

C. Flexible Accelerator Architecture

The design allows for implementing different architectures by configuring a few registors at start-up. During the ”init” phase, the no. of layers, no. of neurons in each layers are stored in internal registors and the generation of the input and weight addresses, MAC cycle counts and turning off the MACs not being used is taken care of by the control FSM.

VI. Synthesis and Metrics in YOSYS and OPENSTA

The screenshots of Timing Analysis and Power Consumption are as shown in Fig.12 and 13. We list all the performance metrics in the table below.

TABLE I: Design Metrics

Characteristics	Value
Operating Frequency	66 MHz
Synthesized Area	63000 squm
Power Consumption	31 mW
Throughput	12.5 kSamples/s
No. of Clock Cycles per Inference	5200
Energy per Inference	2.4 uJ

The power consumption of various parts is as shown in Table-II.

TABLE II: Power Consumption

Power type	Value
Leakage	1.28 mW
Switching	3.84 mW
Internal	26.80 mW

TABLE III: Design Contribution

SR No.	Name	Contribution
24414	Gurjot Singh	Design of MAC, Control FSM Structure, Testbench
24580	Asutosh Ratha	Training in Python, Activation Function, Testbench

VII. Summary and Conclusion

This project is a good way of understanding various design trade-offs between choosing efficient architectures and their complexity in design, both at architecture and algorithm level, speed, performance and energy consumption. We learned how to take up work as a team, conduct review studies of different architectures, discuss their implementation and other trade-offs. After having an high level simulation in Python, moving on to follow the bottom-up design flow, verifying the design functionally at various levels in the RTL stage and then integrating various blocks and a Master Control Block. This hierarchical design paradigm results in maximum throughput for given time frame.

References

[1] Learning on Hardware: A Tutorial on Neural Network Accelerators and Co-Processors Lukas Baischer and Matthias Wess and Nima TaheriNejad, Austria, 2021, https://arxiv.org/abs/2104.09252

[2] https://cloud.google.com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus

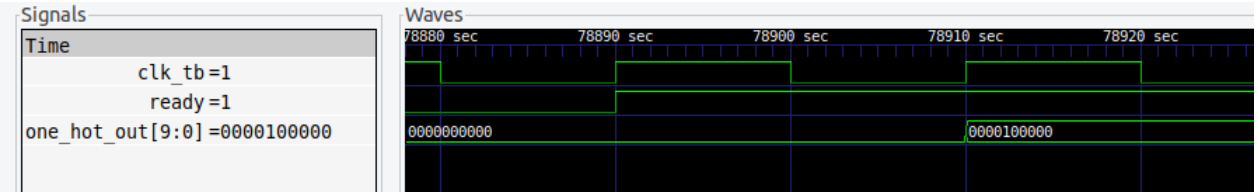

Fig. 10: Simulation for Digit-5

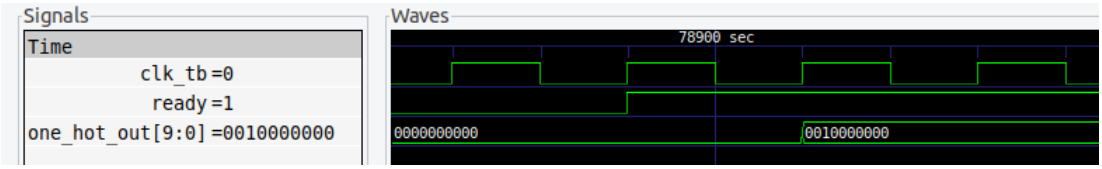

Fig. 11: Simulation for Digit-7

```
Number of wires:              52869
Number of wire bits:          52869
Number of public wires:       13878
Number of public wire bits:   13878
Number of memories:               0
Number of memory bits:            0
Number of processes:              0
Number of cells:              39901
  AND2_X1                      1488
  AND3_X1                       909
  AND4_X1                       321
  AOI211_X1                     553
  AOI21_X1                     3258
  AOI221_X1                     577
  AOI22_X1                      847
  DFFR_X1                      3275
  DFFS_X1                         1
  DFF_X1                       1088
  INV_X1                       2664
  MUX2_X1                      2418
  NAND2_X1                     4162
  NAND3_X1                     1318
  NAND4_X1                      547
  NOR2_X1                      4029
  NOR3_X1                       921
  NOR4_X1                       524
  OAI211_X1                     644
  OAI21_X1                     3545
  OAI221_X1                     127
  OAI222_X1                       1
  OAI22_X1                      497
  OAI33_X1                       97
  OR2_X1                        755
  OR3_X1                        677
  OR4_X1                        129
  XNOR2_X1                     2522
  XOR2_X1                      2007

Chip area for top module '\bfloat16_DNN_MASTER': 62977.628000
```

Fig. 12: Synthesis Results in YoSys

```
  Delay    Time   Description
---------------------------------------------------------
   0.00    0.00   clock CLK (rise edge)
   0.00    0.00   clock network delay (ideal)
   0.00    0.00 ^ _10345_/CK (DFFR_X1)
   0.25    0.25 ^ _10345_/Q (DFFR_X1)
   0.04    0.29 v _05695_/ZN (NAND3_X1)
   0.06    0.35 v _05696_/ZN (AND2_X1)
   0.04    0.39 ^ _05698_/ZN (NOR2_X1)
   0.03    0.42 v _05708_/ZN (NOR2_X1)
  14.03   14.45 ^ _05718_/ZN (NOR3_X1)
  -0.76   13.69 v _06606_/ZN (OAI21_X1)
   0.49   14.18 ^ _06607_/ZN (AOI21_X1)
   0.00   14.18 ^ _09262_/D (DFFR_X1)
          14.18   data arrival time

  15.00   15.00   clock CLK (rise edge)
   0.00   15.00   clock network delay (ideal)
   0.00   15.00   clock reconvergence pessimism
          15.00 ^ _09262_/CK (DFFR_X1)
  -0.06   14.94   library setup time
          14.94   data required time
---------------------------------------------------------
          14.94   data required time
         -14.18   data arrival time
---------------------------------------------------------
           0.75   slack (MET)


Group                  Internal  Switching    Leakage      Total
                          Power      Power      Power      Power
-----------------------------------------------------------------
Sequential             6.38e-03   6.37e-04   3.68e-04   7.39e-03  23.1%
Combinational          2.04e-02   3.20e-03   9.12e-04   2.45e-02  76.9%
Macro                  0.00e+00   0.00e+00   0.00e+00   0.00e+00   0.0%
Pad                    0.00e+00   0.00e+00   0.00e+00   0.00e+00   0.0%
-----------------------------------------------------------------
Total                  2.68e-02   3.84e-03   1.28e-03   3.19e-02 100.0%
                          84.0%      12.0%       4.0%
```

Fig. 13: Timing and Power Analysis in OpenSTA