Hardware-Software Co-Design of a 4-bit Quantized MLP

From PyTorch Training to FPGA RTL Deployment

Presenter: Youssef Mohamed

Project Overview & Objectives

Goal

Design an FPGA-based accelerator for Handwritten Digit Recognition (MNIST).

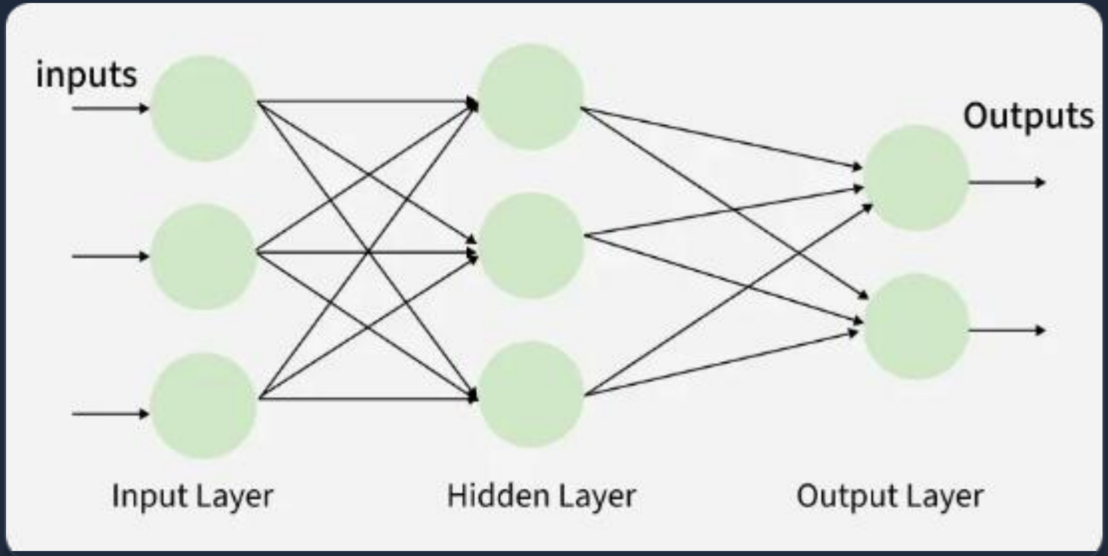
Key Achievement

Successfully verified a bit-exact match between the Python Quantized model and Verilog RTL simulation.

Project Scope

- **Training:** Develop a hardware-friendly Neural Network in PyTorch.
- **Quantization:** Compress model from Float32 to 4-bit Integers (Int4).
- **Hardware:** Implement the accelerator in Verilog RTL (simulation only).

Model Architecture (Software Design)



Network Structure: 3-Layer MLP

- **Input:** 784 neurons (28×28 image)
- **L1:** 64 Neurons + ReLU
- **L2:** 32 Neurons + ReLU
- **Output:** 10 Neurons (Digits 0-9)

Design Decision: Bias = False

Hardware forced: $y = Wx$

Hardware Benefit: Eliminates extra Adders and Muxing logic.

Impact: Negligible accuracy drop (<0.5%).

Quantization Methodology

Method

Post-Training Static Symmetric Quantization.

Bit-Width: 4-bit Weights and Activations.

Why Symmetric?

Maps Floating-point to Integer .

Equation: $Y = W * X$

Avoids expensive zero-point logic:

$$Y = (X - Z\cancel{X})(W - Z\cancel{W})$$

Data Ranges

Weights: Signed 4-bit (-8 to +7)

Activations: Unsigned 4-bit (0 to 15) via ReLU

Design Philosophy: Serial Reuse

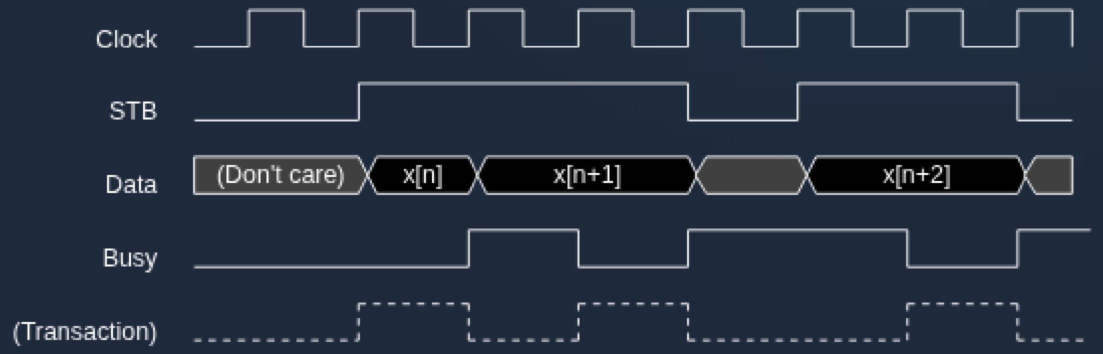
Instead of massive parallelism (area-expensive), we use a single Calculation Core reused sequentially.

Memory Hierarchy

- **ROM:** Stores Weights (linear addressing).
- **RAM:** Stores Input Image and Intermediate Layer Outputs.

Control Logic

- **Muxing:** Data Router switches inputs based on layer state.
- **FSM:** Orchestrates data movement.



Finite State Machine (FSM)

1. LAYER_X

Fetch data → Accumulate

(Repeat 784 times)

2. WRITE

Store result to RAM

(activates wen signal)

3. INC

Reset Accumulator

Increment Neuron Counter

4. ARGMAX

Layer 3 Only

Tracks max_logit for prediction

Challenge 1: Signed/Unsigned Arithmetic

⚠️The Issue

- Inputs are **Unsigned** (0..15).
- Weights are **Signed** (-8..7).
- Verilog standard multiplication can misinterpret the Unsigned input as a negative 2's complement number if the MSB is 1 (e.g., 1111 = 15 or -1?).

✅The Solution

Explicitly zero-padded the input to force positive interpretation.

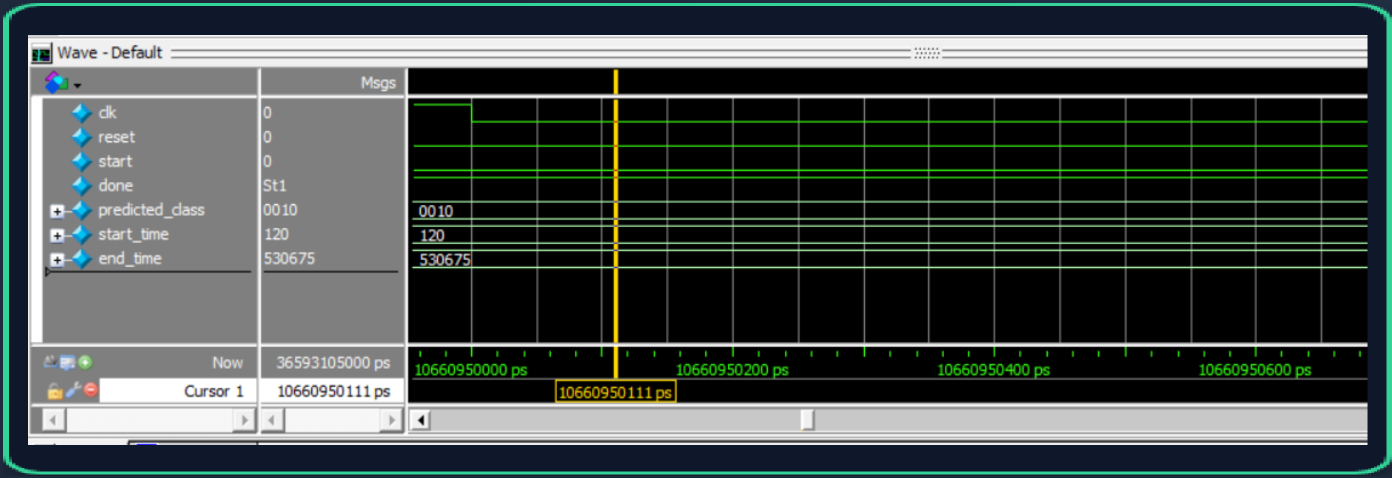
```
assign product = (current_input === 4'bx || current_weight === 4'bx)
     ? 0 : $signed({1'b0, current_input}) * $signed(current_weight);
```

Mathematically correct mixed-sign multiplication.

Challenge 2: Memory Latency & Pipelining

The Issue: BRAM Latency

Block RAM has a 1-cycle read latency. Accumulating data_out in the same cycle as addr request reads "garbage."



The Solution: 2-Stage Pipeline

- **Cycle 0:** `pipeline_valid <= 1 (Request)`
- **Cycle 1:** `pipeline_valid_d <= 1 (Accumulate)`

Result: Perfect synchronization; the accumulator only updates when valid data arrives.

Challenge 3: The "Epoch Trap"

🔍Observation

Extended training (30 epochs) caused Quantized Accuracy to crash to **73.32%**.

🐞Root Cause: Outlier Weights

As training continues, the model pushes some weights to extreme values (e.g., -3.5 or +4.0). Symmetric Quantization scales the entire 4-bit range based on the largest number, wiping out precision for small weights.

Solution: Used 6 EPOCHS ONLE

```
Epoch 25/30 complete. Loss: 0.0016
Epoch 26/30 complete. Loss: 0.0001
Epoch 27/30 complete. Loss: 0.0016
Epoch 28/30 complete. Loss: 0.0062
Epoch 29/30 complete. Loss: 0.0001
Epoch 30/30 complete. Loss: 0.0008

--- Calculating Original (Floating-Point) Model Accuracy ---
Original Model Accuracy: 97.41%

--- Starting Quantization ---
Saved w1.mem
Saved w2.mem
Saved w3.mem
Saved input1.mem

Expected Label for input.mem: 1

--- Golden Reference (Integer Simulation) ---
Layer 3 Output (logits): [-10.   8.  -5.  -6.   4.  -9. -10.   6.  -5.  -1.]
Predicted Digit: 1

--- Calculating Quantized Model Accuracy (Hardware Simulation) ---
Quantized Model Accuracy: 73.32%
```

Final Results & Verification

```
Training for 6 epochs (bias=False)...
Epoch 1/6 complete. Loss: 0.3354
Epoch 2/6 complete. Loss: 0.0656
Epoch 3/6 complete. Loss: 0.3705
Epoch 4/6 complete. Loss: 0.0265
Epoch 5/6 complete. Loss: 0.0149
Epoch 6/6 complete. Loss: 0.0371

--- Calculating Original (Floating-Point) Model Accuracy ---
Original Model Accuracy: 97.09%

--- Starting Quantization ---
Saved w1.mem
Saved w2.mem
Saved w3.mem
Saved input1.mem

Expected Label for input.mem: 1

--- Golden Reference (Integer Simulation) ---
Layer 3 Output (logits): [-17.  32. -17. -16. -13. -23. -15.   8. -10.  -5.]
Predicted Digit: 1

--- Calculating Quantized Model Accuracy (Hardware Simulation) ---
Quantized Model Accuracy: 95.56%
```

```
# DEBUG: L3 Neuron  0 | Logit:        1 | Current Max:  -262144
# DEBUG: L3 Neuron  1 | Logit:      -34 | Current Max:        1
# DEBUG: L3 Neuron  2 | Logit:       59 | Current Max:        1
# DEBUG: L3 Neuron  3 | Logit:       11 | Current Max:       59
# DEBUG: L3 Neuron  4 | Logit:      -89 | Current Max:       59
# DEBUG: L3 Neuron  5 | Logit:      -42 | Current Max:       59
# DEBUG: L3 Neuron  6 | Logit:      -57 | Current Max:       59
# DEBUG: L3 Neuron  7 | Logit:        6 | Current Max:       59
# DEBUG: L3 Neuron  8 | Logit:       17 | Current Max:       59
# DEBUG: L3 Neuron  9 | Logit:      -37 | Current Max:       59
# -------------------------
# Inference Complete.
# Predicted Class:  2
# Total Cycles:     53055
# Total Time:       530555 ns
```

```
Saved input1.mem
Saved input1.mem for Image Index 444 (True Label: 2)

--- Python Golden Prediction ---
Logits: [-10.   4.  53.  20. -59. -14. -50.  18.  18.  -7.]
Predicted Class: 2
```

Final Quantized Model (6 Epochs): 95.56% Accuracy

Hardware output exactly matches the Python Golden Reference.

Simulation Results & Verification

✓ Verification Successful

RTL Simulation achieved a **Bit-Exact Match** to the Quantized Python model.

Testbench: tb_mlp.v with w1.mem and input1.mem

Q & A

Thank you for your attention.