EE533_Lab10_Report

1. Verilog code for each unit

1.1 neurons_hidden

- Verilog

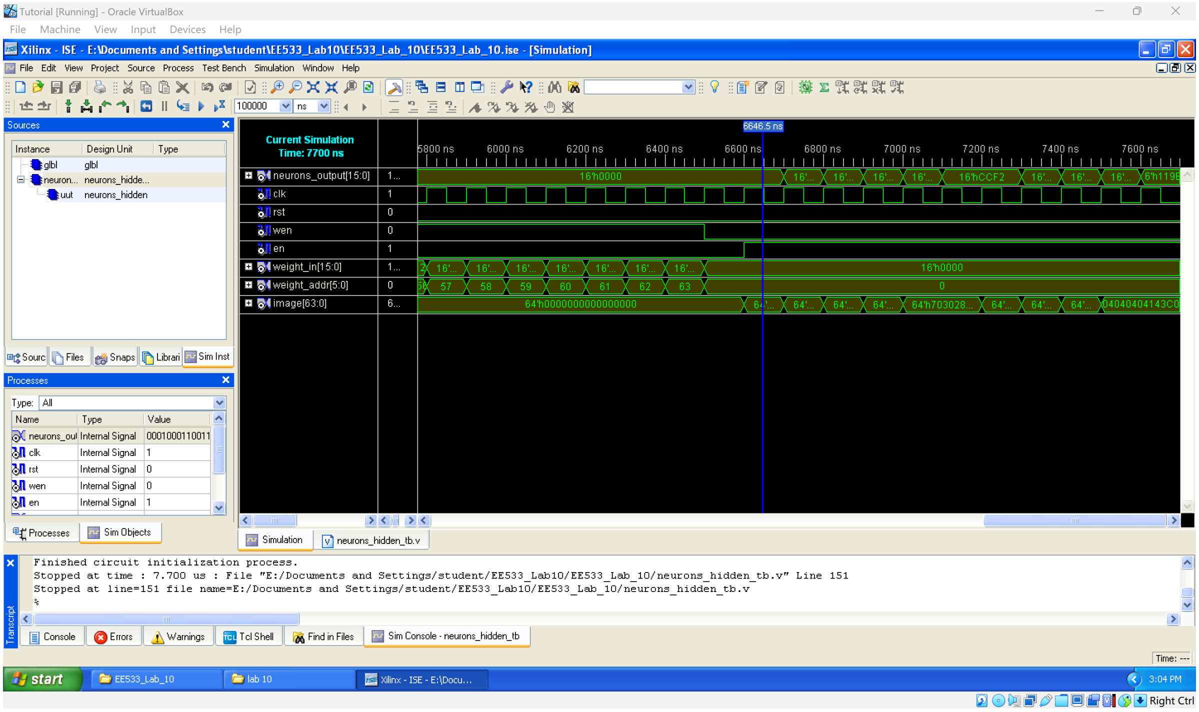
```
`timescale 1ns / 1ps

module neurons_hidden
(
    input clk,
    input rst,
    input wen,
    input en,
    input [15:0] weight_in,
    input [5:0] weight_addr,
    input [63:0] image,

    output reg [15:0] neurons_output
);

    reg [15:0] weight [63:0];
    reg [15:0] sum;
    integer i;

    always @(posedge clk) begin
        if (rst) begin
            sum <= 16'b0;
            neurons_output <= 16'b0;
            for (i = 0; i < 64; i = i + 1) begin
                weight[i] <= 16'b0;
            end
        end
        else if (wen) begin
            weight[weight_addr] <= weight_in;
        end
        else if (en) begin
            sum <= 16'b0;
            for (i = 0; i < 64; i = i + 1) begin
                sum = sum + ((image[i]) ? weight[i] : 16'b0);
            end
            neurons_output <= sum;
        end
    end

endmodule
```

- Testbench

```
`timescale 1ns / 1ps
```

```verilog
////////////////////////////////////////////////////////////////////////////////
// Company: 
// Engineer:
//
// Create Date:   14:59:51 03/21/2025
// Design Name:   neurons_hidden
// Module Name:   E:/Documents and Settings/student/EE533_Lab10/neurons_hidden_tb.v
// Project Name:  EE533_Lab10
// Target Device:  
// Tool versions:  
// Description: 
//
// Verilog Test Fixture created by ISE for module: neurons_hidden
//
// Dependencies:
// 
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
// 
////////////////////////////////////////////////////////////////////////////////

module neurons_hidden_tb;

    // Inputs
    reg clk;
    reg rst;
    reg wen;
    reg en;
    reg [15:0] weight_in;
    reg [5:0] weight_addr;
    reg [63:0] image;

    // Outputs
    wire [15:0] neurons_output;

    // Instantiate the Unit Under Test (UUT)
    neurons_hidden uut (
        .clk(clk), 
        .rst(rst), 
        .wen(wen), 
        .en(en), 
        .weight_in(weight_in), 
        .weight_addr(weight_addr), 
        .image(image), 
        .neurons_output(neurons_output)
    );

    always #50 clk = ~clk;

    initial begin
        // Initialize Inputs
        clk = 1;
        rst = 1;
        wen = 0;
        en = 0;
        weight_in = 0;
        weight_addr = 0;
        image = 0;

        // Wait 100 ns for global reset to finish
        @(posedge clk);

        // Add stimulus here
        rst = 0;
        wen = 1;
        weight_addr = 6'd0;  weight_in = 16'h2c72; @(posedge clk);
        weight_addr = 6'd1;  weight_in = 16'h2932; @(posedge clk);
        weight_addr = 6'd2;  weight_in = 16'he4e3; @(posedge clk);
        weight_addr = 6'd3;  weight_in = 16'he49f; @(posedge clk);
        weight_addr = 6'd4;  weight_in = 16'hedb5; @(posedge clk);
        weight_addr = 6'd5;  weight_in = 16'he6a1; @(posedge clk);
        weight_addr = 6'd6;  weight_in = 16'hd3f4; @(posedge clk);
        weight_addr = 6'd7;  weight_in = 16'h3b55; @(posedge clk);
        weight_addr = 6'd8;  weight_in = 16'h21c4; @(posedge clk);
        weight_addr = 6'd9;  weight_in = 16'heccd; @(posedge clk);
        weight_addr = 6'd10; weight_in = 16'hd5a7; @(posedge clk);
        weight_addr = 6'd11; weight_in = 16'hd6c1; @(posedge clk);
        weight_addr = 6'd12; weight_in = 16'hdaac; @(posedge clk);
        weight_addr = 6'd13; weight_in = 16'hecc7; @(posedge clk);
        weight_addr = 6'd14; weight_in = 16'hd977; @(posedge clk);
        weight_addr = 6'd15; weight_in = 16'h3a2c; @(posedge clk);
        weight_addr = 6'd16; weight_in = 16'heef2; @(posedge clk);
        weight_addr = 6'd17; weight_in = 16'hda95; @(posedge clk);
        weight_addr = 6'd18; weight_in = 16'h3a50; @(posedge clk);
        weight_addr = 6'd19; weight_in = 16'h3a25; @(posedge clk);
        weight_addr = 6'd20; weight_in = 16'h3a10; @(posedge clk);
        weight_addr = 6'd21; weight_in = 16'hd73b; @(posedge clk);
        weight_addr = 6'd22; weight_in = 16'h3a3e; @(posedge clk);
        weight_addr = 6'd23; weight_in = 16'h21f0; @(posedge clk);
        weight_addr = 6'd24; weight_in = 16'h3a11; @(posedge clk);
        weight_addr = 6'd25; weight_in = 16'h2a80; @(posedge clk);
        weight_addr = 6'd26; weight_in = 16'h3a15; @(posedge clk);
        weight_addr = 6'd27; weight_in = 16'h2b4d; @(posedge clk);
        weight_addr = 6'd28; weight_in = 16'he4c2; @(posedge clk);
        weight_addr = 6'd29; weight_in = 16'heea3; @(posedge clk);
        weight_addr = 6'd30; weight_in = 16'h2d35; @(posedge clk);
        weight_addr = 6'd31; weight_in = 16'h3b80; @(posedge clk);
        weight_addr = 6'd32; weight_in = 16'h3b15; @(posedge clk);
        weight_addr = 6'd33; weight_in = 16'h3b40; @(posedge clk);
        weight_addr = 6'd34; weight_in = 16'hdaad; @(posedge clk);
        weight_addr = 6'd35; weight_in = 16'h3a80; @(posedge clk);
        weight_addr = 6'd36; weight_in = 16'h3a92; @(posedge clk);
        weight_addr = 6'd37; weight_in = 16'h3ab0; @(posedge clk);
        weight_addr = 6'd38; weight_in = 16'hd9f2; @(posedge clk);
        weight_addr = 6'd39; weight_in = 16'hdc34; @(posedge clk);
        weight_addr = 6'd40; weight_in = 16'h3a50; @(posedge clk);
        weight_addr = 6'd41; weight_in = 16'hd5d0; @(posedge clk);
        weight_addr = 6'd42; weight_in = 16'hd8b0; @(posedge clk);
        weight_addr = 6'd43; weight_in = 16'h3a1f; @(posedge clk);
        weight_addr = 6'd44; weight_in = 16'hd8d5; @(posedge clk);
        weight_addr = 6'd45; weight_in = 16'hd7b4; @(posedge clk);
        weight_addr = 6'd46; weight_in = 16'hd7e0; @(posedge clk);
        weight_addr = 6'd47; weight_in = 16'h3f90; @(posedge clk);
        weight_addr = 6'd48; weight_in = 16'h3c80; @(posedge clk);
        weight_addr = 6'd49; weight_in = 16'h3a3d; @(posedge clk);
        weight_addr = 6'd50; weight_in = 16'h3aa0; @(posedge clk);
        weight_addr = 6'd51; weight_in = 16'h3b02; @(posedge clk);
        weight_addr = 6'd52; weight_in = 16'h3ac0; @(posedge clk);
        weight_addr = 6'd53; weight_in = 16'h3c50; @(posedge clk);
        weight_addr = 6'd54; weight_in = 16'h3a4d; @(posedge clk);
        weight_addr = 6'd55; weight_in = 16'h3b65; @(posedge clk);
        weight_addr = 6'd56; weight_in = 16'h2d20; @(posedge clk);
        weight_addr = 6'd57; weight_in = 16'h0000; @(posedge clk);
        weight_addr = 6'd58; weight_in = 16'hedbf; @(posedge clk);
        weight_addr = 6'd59; weight_in = 16'heed8; @(posedge clk);
        weight_addr = 6'd60; weight_in = 16'h3c10; @(posedge clk);
        weight_addr = 6'd61; weight_in = 16'h3b05; @(posedge clk);
        weight_addr = 6'd62; weight_in = 16'h3a9f; @(posedge clk);
        weight_addr = 6'd63; weight_in = 16'h3a80; @(posedge clk);

        wen = 0;
        weight_addr = 6'd0; weight_in = 16'h0000;
        @(posedge clk);

        en = 1;
        image = 64'h1830403030303c1c; @(posedge clk);
        image = 64'h201000e080c0f0f0; @(posedge clk);
        image = 64'h103010306030f0f0; @(posedge clk);
        image = 64'h20203c5020606060; @(posedge clk);
        image = 64'h7030281078d87060; @(posedge clk);
        image = 64'h7030281078d87060; @(posedge clk);
        image = 64'h70a0908010d0f010; @(posedge clk);
        image = 64'h3038383e30f01060; @(posedge clk);
        image = 64'h303e3e6c6c58b030; @(posedge clk);
        image = 64'h04040404143c0c04; @(posedge clk);

        @(posedge clk);
        $stop;

    end

endmodule
```

- Waveform



Xilinx - ISE - E:\Documents and Settings\student\EE533_Lab10\EE533_Lab_10\EE533_Lab_10.ise - [Simulation]

Name	Type	Value
neurons_out	Internal Signal	0001000110011
clk	Internal Signal	1
rst	Internal Signal	0
wen	Internal Signal	0
en	Internal Signal	1

```
Finished circuit initialization process.
Stopped at time : 7.700 us : File "E:/Documents and Settings/student/EE533_Lab10/EE533_Lab_10/neurons_hidden_tb.v" Line 151
Stopped at line=151 file name=E:/Documents and Settings/student/EE533_Lab10/EE533_Lab_10/neurons_hidden_tb.v
%
```

- Image Input

```
64'h1830403030303c1c;
64'h201000e080c0f0f0;
64'h103010306030f0f0;
64'h20203c5020606060;
64'h7030281078d87060;
64'h7030281078d87060;
64'h70a0908010d0f010;
64'h3038383e30f01060;
64'h303e3e6c6c58b030;
64'h04040404143c0c04;
```

- Weights in neurons_hidden_0

```
    16'h2c72, 16'h2932, 16'he4e3, 16'he49f, 16'hedb5, 16'he6a1, 16'hd3f4, 
16'h3b55,
    16'h21c4, 16'heccd, 16'hd5a7, 16'hd6c1, 16'hdaac, 16'hecc7, 16'hd977, 
16'h3a2c,
    16'heef2, 16'hda95, 16'h3a50, 16'h3a25, 16'h3a10, 16'hd73b, 16'h3a3e, 
16'h21f0,
    16'h3a11, 16'h2a80, 16'h3a15, 16'h2b4d, 16'he4c2, 16'heea3, 16'h2d35, 
16'h3b80,
    16'h3b15, 16'h3b40, 16'hdaad, 16'h3a80, 16'h3a92, 16'h3ab0, 16'hd9f2, 
16'hdc34,
    16'h3a50, 16'hd5d0, 16'hd8b0, 16'h3a1f, 16'hd8d5, 16'hd7b4, 16'hd7e0, 
16'h3f90,
    16'h3c80, 16'h3a3d, 16'h3aa0, 16'h3b02, 16'h3ac0, 16'h3c50, 16'h3a4d, 
16'h3b65,
    16'h2d20, 16'h0000, 16'hedbf, 16'heed8, 16'h3c10, 16'h3b05, 16'h3a9f, 
16'h3a80
```

- Weights in neurons_hidden_1

```
    16'hf99e, 16'hf7fe, 16'hd1aa, 16'h3af8, 16'hf9d9, 16'hf89f, 16'h3a90,
16'h3a40,
    16'hf9a9, 16'h3a60, 16'h3a04, 16'h3a10, 16'h3a80, 16'hf89d, 16'h3a20,
16'hf8f0,
    16'hf8f4, 16'hd1a2, 16'hf95e, 16'hf9fe, 16'hc1c5, 16'hf9f1, 16'hf8d4,
16'h3a40,
    16'hf9e2, 16'hf7e1, 16'h3a10, 16'hf8d2, 16'hf9a1, 16'hf901, 16'h3b90,
16'hf9fd,
    16'hf9f9, 16'h3b50, 16'h3a90, 16'h3ab4, 16'hf8c8, 16'h3a30, 16'hf8d6,
16'h3a50,
    16'h3a20, 16'h3a80, 16'h3a10, 16'h3a50, 16'h3a60, 16'hf9e4, 16'h3a70,
16'hf8e0,
    16'h3a90, 16'hf8d0, 16'h3a40, 16'h3aa0, 16'h3a90, 16'h3ab0, 16'h3a20,
16'h3a50,
    16'hf9f0, 16'hf9f0, 16'hf8d0, 16'hf8d0, 16'h3aa0, 16'h3ab0, 16'h3a50,
16'h3a80
```

- Weights in neurons_hidden_2

```
    16'hf9f2, 16'hf8e4, 16'h2b20, 16'h3a50, 16'h2b10, 16'hf8c0, 16'h3b10,
16'hf7f0,
    16'hf8d4, 16'h3a70, 16'h3a90, 16'hf8b0, 16'h3a20, 16'h3a10, 16'hf8d0,
16'h3a00,
    16'h3a40, 16'h3a10, 16'h3a20, 16'hd1a0, 16'hf9c0, 16'h3a10, 16'hf8d0,
16'h3a00,
    16'hf8e4, 16'hf7f8, 16'h3a10, 16'hf8c0, 16'hf9a8, 16'hf9d0, 16'h3a70,
16'hf8e0,
    16'h3a20, 16'h3a90, 16'hf9c8, 16'h3a10, 16'hf8e0, 16'h3a50, 16'h3a40,
16'hf8c0,
    16'h3a10, 16'h3a30, 16'h3a40, 16'h3a50, 16'h3a60, 16'h3a20, 16'h3a10,
16'h3a50,
    16'hf8e0, 16'h3a10, 16'hf8d0, 16'h3a20, 16'hf8c0, 16'h3a10, 16'h3a30,
16'h3a40,
    16'hf9e0, 16'h3a10, 16'h3a20, 16'h3a30, 16'h3a40, 16'h3a50, 16'h3a60,
16'hf9d0
```

1.2 ReLu

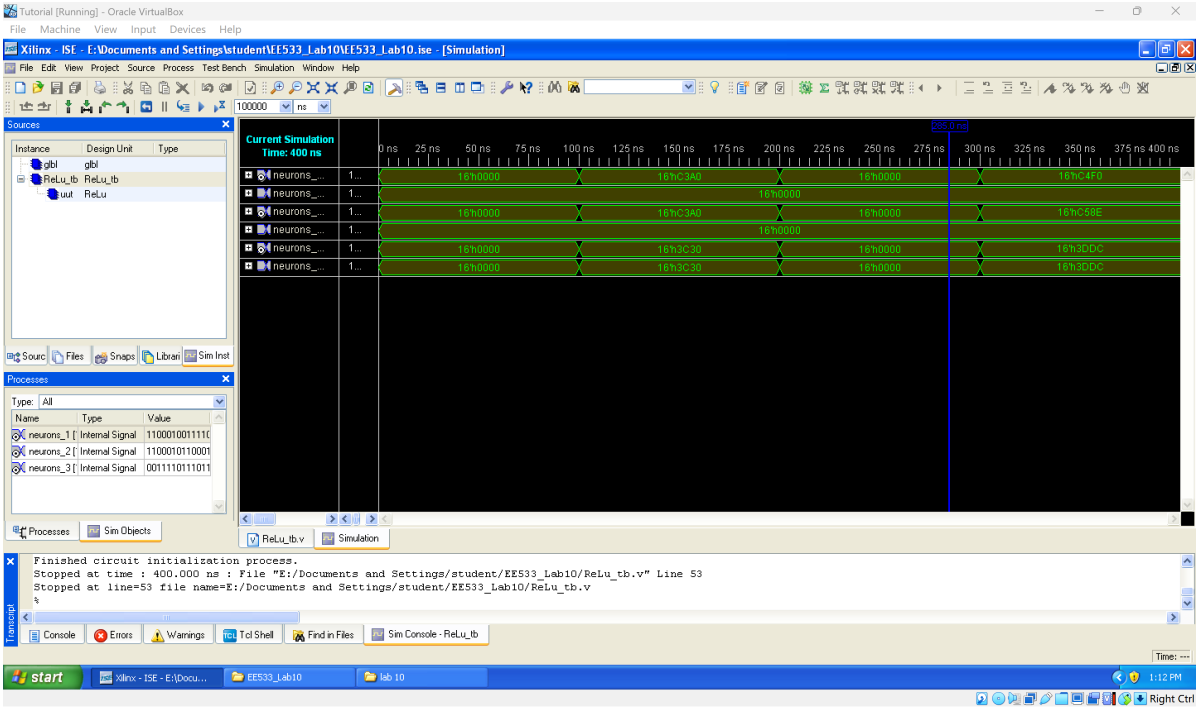
- Verilog

```
`timescale 1ns / 1ps

module ReLu
(
    input [15:0] neurons_1,
    input [15:0] neurons_2,
    input [15:0] neurons_3,

    output [15:0] neurons_1_out,
    output [15:0] neurons_2_out,
    output [15:0] neurons_3_out
);

    assign neurons_1_out = (neurons_1[15] == 1'b1) ? 16'b0 : neurons_1;
    assign neurons_2_out = (neurons_2[15] == 1'b1) ? 16'b0 : neurons_2;
    assign neurons_3_out = (neurons_3[15] == 1'b1) ? 16'b0 : neurons_3;

endmodule
```

- Testbench

```
`timescale 1ns / 1ps

////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:   14:50:19 03/21/2025
// Design Name:   ReLu
// Module Name:   E:/Documents and Settings/student/EE533_Lab10/ReLu_tb.v
// Project Name:  EE533_Lab10
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: ReLu
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////

module ReLu_tb;

    // Inputs
    reg [15:0] neurons_1;
    reg [15:0] neurons_2;
    reg [15:0] neurons_3;

    // Instantiate the Unit Under Test (UUT)
    ReLu uut (
        .neurons_1(neurons_1),
        .neurons_2(neurons_2),
        .neurons_3(neurons_3)
    );

    initial begin
        // Initialize Inputs
        neurons_1 = 0;
        neurons_2 = 0;
        neurons_3 = 0;

        // Wait 100 ns for global reset to finish
        #100;

        // Add stimulus here
        neurons_1 = 16'hC3A0; neurons_2 = 16'hC3A0; neurons_3 = 16'h3C30; #100;
        neurons_1 = 16'h0000; neurons_2 = 16'h0000; neurons_3 = 16'h0000; #100;
        neurons_1 = 16'hC4F0; neurons_2 = 16'hC58E; neurons_3 = 16'h3DDC; #100;

        $stop;

    end

endmodule
```

- Waveform


Tutorial [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Xilinx - ISE - E:\Documents and Settings\student\EE533_Lab10\EE533_Lab10.ise - [Simulation]
File Edit View Project Source Process Test Bench Simulation Window Help
100000 ns
Sources
Instance Design Unit Type
glbl glbl
ReLu_tb ReLu_tb
uut ReLu
Sourc Files Snaps Librari Sim Inst
Current Simulation Time: 400 ns
285.0 ns
0 ns 25 ns 50 ns 75 ns 100 ns 125 ns 150 ns 175 ns 200 ns 225 ns 250 ns 275 ns 300 ns 325 ns 350 ns 375 ns 400 ns
neurons_... 1... 16'h0000 16'hC3A0 16'h0000 16'hC4F0
neurons_... 1... 16'h0000
neurons_... 1... 16'h0000 16'hC3A0 16'h0000 16'hC58E
neurons_... 1... 16'h0000
neurons_... 1... 16'h0000 16'h3C30 16'h0000 16'h3DDC
neurons_... 1... 16'h0000 16'h3C30 16'h0000 16'h3DDC
Processes
Type: All
Processes Sim Objects
ReLu_tb.v Simulation
Transcript
Console Errors Warnings Tcl Shell Find in Files Sim Console - ReLu_tb
Time: ---
start Xilinx - ISE - E:\Docu... EE533_Lab10 lab 10
1:12 PM
Right Ctrl

Name	Type	Value
neurons_1 [	Internal Signal	110001001111(
neurons_2 [	Internal Signal	1100010110001
neurons_3 [	Internal Signal	0011110111011

```
Finished circuit initialization process.
Stopped at time : 400.000 ns : File "E:/Documents and Settings/student/EE533_Lab10/ReLu_tb.v" Line 53
Stopped at line=53 file name=E:/Documents and Settings/student/EE533_Lab10/ReLu_tb.v
%
```

1.3 neurons_output

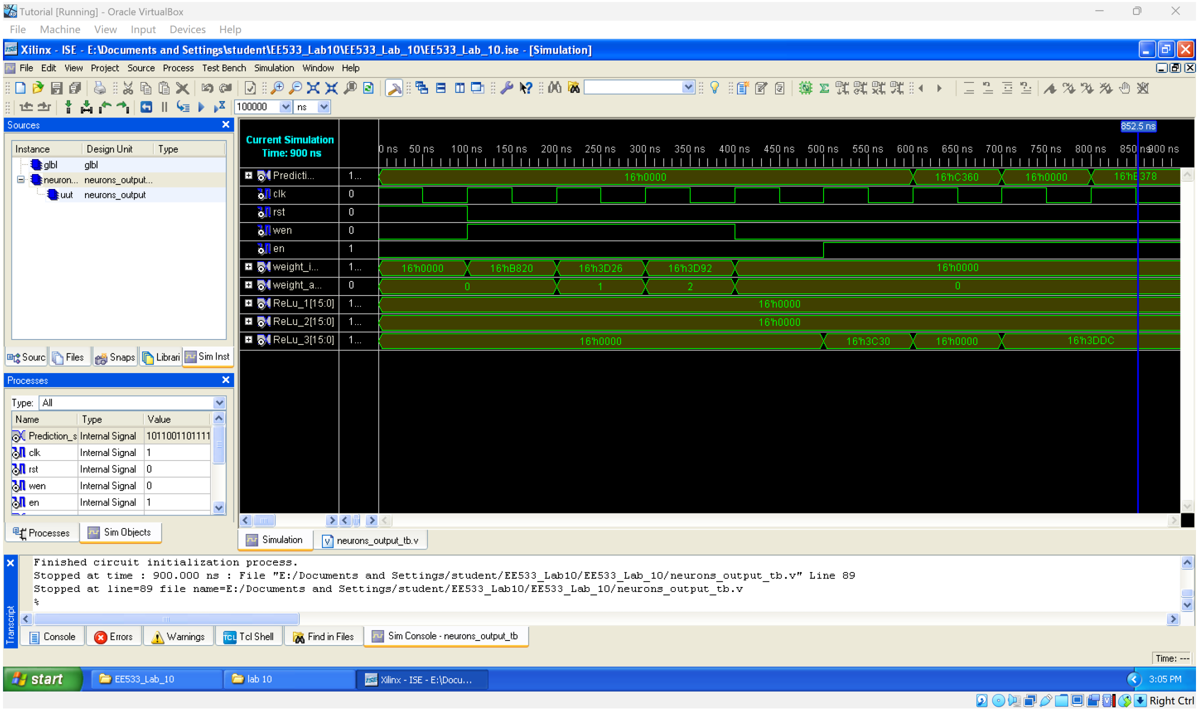
- Verilog

```
`timescale 1ns / 1ps

module neurons_output
(
    input clk,
    input rst,
    input wen,
    input en,
    input [15:0] weight_in,
    input [1:0] weight_addr,

    input [15:0] ReLu_1,
    input [15:0] ReLu_2,
    input [15:0] ReLu_3,

    output reg [15:0] Prediction_score
);

    reg [15:0] weight [2:0];
    integer i;
    always @(posedge clk) begin
        if (rst) begin
            Prediction_score <= 16'b0;
            for (i = 0; i < 3; i = i + 1) begin
                weight[i] <= 16'b0;
            end
        end
        else if (wen) begin
            weight[weight_addr] <= weight_in;
        end
        else if (en) begin
            Prediction_score <= ReLu_1 * weight[0] + ReLu_2 * weight[1] + ReLu_3
* weight[2];
        end
    end

endmodule
```

- Testbench

```
`timescale 1ns / 1ps

////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:   15:38:04 03/21/2025
// Design Name:   neurons_output
// Module Name:   E:/Documents and
Settings/student/EE533_Lab10/neurons_output_tb.v
// Project Name:  EE533_Lab10
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: neurons_output
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////

module neurons_output_tb;

    // Inputs
    reg clk;
    reg rst;
    reg wen;
    reg en;
    reg [15:0] weight_in;
    reg [1:0] weight_addr;
    reg [15:0] ReLu_1;
    reg [15:0] ReLu_2;
    reg [15:0] ReLu_3;

    // Outputs
    wire [15:0] Prediction_score;

    // Instantiate the Unit Under Test (UUT)
    neurons_output uut (
        .clk(clk),
        .rst(rst),
        .wen(wen),
        .en(en),
        .weight_in(weight_in),
        .weight_addr(weight_addr),
        .ReLu_1(ReLu_1),
        .ReLu_2(ReLu_2),
        .ReLu_3(ReLu_3),
        .Prediction_score(Prediction_score)
    );

    always #50 clk = ~clk;

    initial begin
        // Initialize Inputs
        clk = 1;
        rst = 1;
        wen = 0;
        en = 0;
        weight_in = 0;
        weight_addr = 0;
        ReLu_1 = 0;
        ReLu_2 = 0;
        ReLu_3 = 0;

        // Wait 100 ns for global reset to finish
        @(posedge clk);

        // Add stimulus here
        rst = 0;
        wen = 1;
        weight_addr = 2'd0; weight_in = 16'hb820; @(posedge clk);
        weight_addr = 2'd1; weight_in = 16'h3d26; @(posedge clk);
        weight_addr = 2'd2; weight_in = 16'h3d92; @(posedge clk);

        wen = 0;
        weight_addr = 2'd0; weight_in = 16'h0000; @(posedge clk);

        en = 1;
        weight_addr = 2'd0; weight_in = 16'h0000;
        ReLu_1 = 16'h0000; ReLu_2 = 16'h0000; ReLu_3 = 16'h3c30; @(posedge clk);
        ReLu_1 = 16'h0000; ReLu_2 = 16'h0000; ReLu_3 = 16'h0000; @(posedge clk);
        ReLu_1 = 16'h0000; ReLu_2 = 16'h0000; ReLu_3 = 16'h3ddc; @(posedge clk);

        @(posedge clk);
        $stop;
    end

endmodule
```

- Waveform



Tutorial [Running] - Oracle VirtualBox
Xilinx - ISE - E:\Documents and Settings\student\EE533_Lab10\EE533_Lab_10\EE533_Lab_10.ise - [Simulation]
Current Simulation Time: 900 ns
852.5 ns
Predicti... 1... 16'h0000 16'hC360 16'h0000 16'hE378
clk 0
rst 0
wen 0
en 1
weight_i... 1... 16'h0000 16'hB820 16'h3D26 16'h3D92 16'h0000
weight_a... 0 0 1 2 0
ReLu_1[15:0] 1... 16'h0000
ReLu_2[15:0] 1... 16'h0000
ReLu_3[15:0] 1... 16'h0000 16'h3C30 16'h0000 16'h3DDC
Simulation
neurons_output_tb.v
Processes
Sim Objects
Console
Errors
Warnings
Tcl Shell
Find in Files
Sim Console - neurons_output_tb

```
Finished circuit initialization process.
Stopped at time : 900.000 ns : File "E:/Documents and Settings/student/EE533_Lab10/EE533_Lab_10/neurons_output_tb.v" Line 89
Stopped at line=89 file name=E:/Documents and Settings/student/EE533_Lab10/EE533_Lab_10/neurons_output_tb.v
%
```

- Weights in ten neurons output unit, each contains 3 elements (in one row)

```
16'hb820, 16'h3d26, 16'h3d92;
16'h4240, 16'hbd6a, 16'hbc4a;
16'h3d90, 16'hbdc0, 16'h3b98;
16'hbb00, 16'hbcf2, 16'h3d98;
16'h3b80, 16'h3ce4, 16'hc5f2;
16'hbd2e, 16'h3c20, 16'h3d4a;
16'h3d88, 16'h3a00, 16'hac00;
16'hbd80, 16'h4280, 16'hbc9a;
16'h3da0, 16'hbc52, 16'h3280;
16'hbbd0, 16'h3380, 16'h3cfe;
```

2. Software Training Process

2.1 ann.py

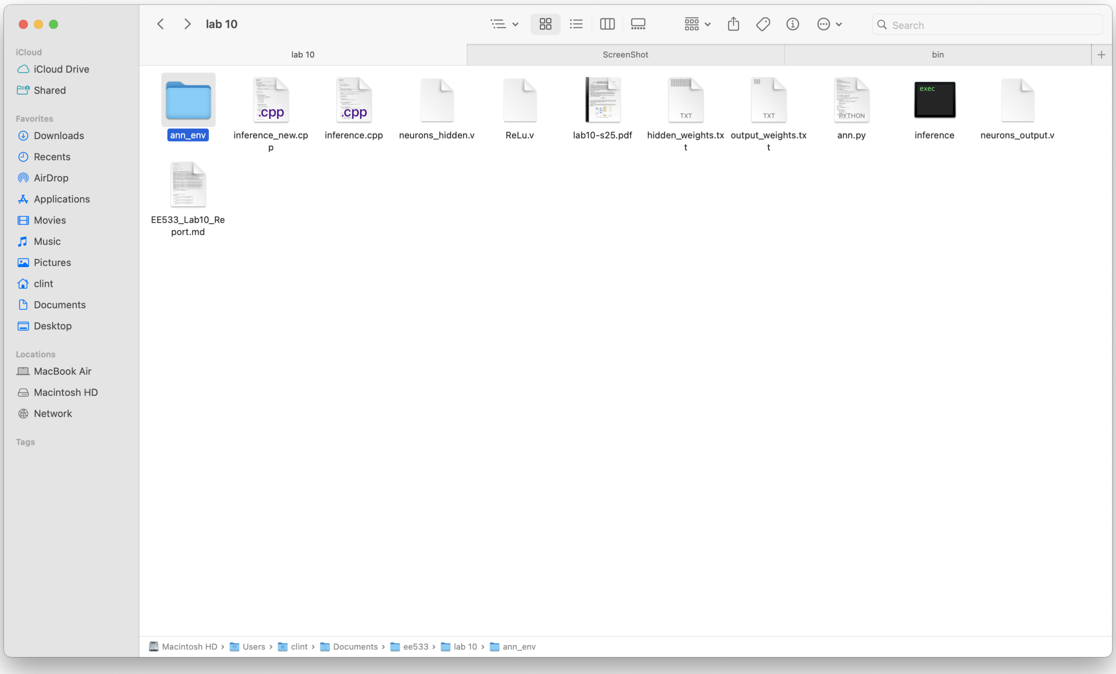
2.1.1 Creating Virtual Environment

- Terminal Window

```
clint@usc-secure-wireless-new256 lab 10 % python3 -m venv ann_env
clint@usc-secure-wireless-new256 lab 10 % source ann_env/bin/activate
(ann_env) clint@usc-secure-wireless-new256 lab 10 % python3 -m pip install torch
torchvision numpy scikit-learn matplotlib
Requirement already satisfied: torch in ./ann_env/lib/python3.13/site-packages
(2.6.0)
```

```
Requirement already satisfied: torchvision in ./ann_env/lib/python3.13/site-packages (0.21.0)
Requirement already satisfied: numpy in ./ann_env/lib/python3.13/site-packages (2.2.4)
Requirement already satisfied: scikit-learn in ./ann_env/lib/python3.13/site-packages (1.6.1)
Requirement already satisfied: matplotlib in ./ann_env/lib/python3.13/site-packages (3.10.1)
Requirement already satisfied: filelock in ./ann_env/lib/python3.13/site-packages (from torch) (3.18.0)
Requirement already satisfied: typing-extensions>=4.10.0 in ./ann_env/lib/python3.13/site-packages (from torch) (4.12.2)
Requirement already satisfied: networkx in ./ann_env/lib/python3.13/site-packages (from torch) (3.4.2)
Requirement already satisfied: jinja2 in ./ann_env/lib/python3.13/site-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in ./ann_env/lib/python3.13/site-packages (from torch) (2025.3.0)
Requirement already satisfied: setuptools in ./ann_env/lib/python3.13/site-packages (from torch) (77.0.1)
Requirement already satisfied: sympy==1.13.1 in ./ann_env/lib/python3.13/site-packages (from torch) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in ./ann_env/lib/python3.13/site-packages (from sympy==1.13.1->torch) (1.3.0)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in ./ann_env/lib/python3.13/site-packages (from torchvision) (11.1.0)
Requirement already satisfied: scipy>=1.6.0 in ./ann_env/lib/python3.13/site-packages (from scikit-learn) (1.15.2)
Requirement already satisfied: joblib>=1.2.0 in ./ann_env/lib/python3.13/site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in ./ann_env/lib/python3.13/site-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: contourpy>=1.0.1 in ./ann_env/lib/python3.13/site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in ./ann_env/lib/python3.13/site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in ./ann_env/lib/python3.13/site-packages (from matplotlib) (4.56.0)
Requirement already satisfied: kiwisolver>=1.3.1 in ./ann_env/lib/python3.13/site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in ./ann_env/lib/python3.13/site-packages (from matplotlib) (24.2)
Requirement already satisfied: pyparsing>=2.3.1 in ./ann_env/lib/python3.13/site-packages (from matplotlib) (3.2.1)
Requirement already satisfied: python-dateutil>=2.7 in ./ann_env/lib/python3.13/site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in ./ann_env/lib/python3.13/site-packages (from python-dateutil>=2.7->matplotlib) (1.17.0)
Requirement already satisfied: MarkupSafe>=2.0 in ./ann_env/lib/python3.13/site-packages (from jinja2->torch) (3.0.2)

[notice] A new release of pip is available: 25.0 -> 25.0.1
[notice] To update, run: pip install --upgrade pip
```

- Screenshot



2.1.2 Modified part for ann.py

- For printing the generated input images' 64-bit value in terminal window

```
    # Print out the image into 64-bit value
    print("64-bit input vector:\n")
    print(X_test[rand_idx])
```

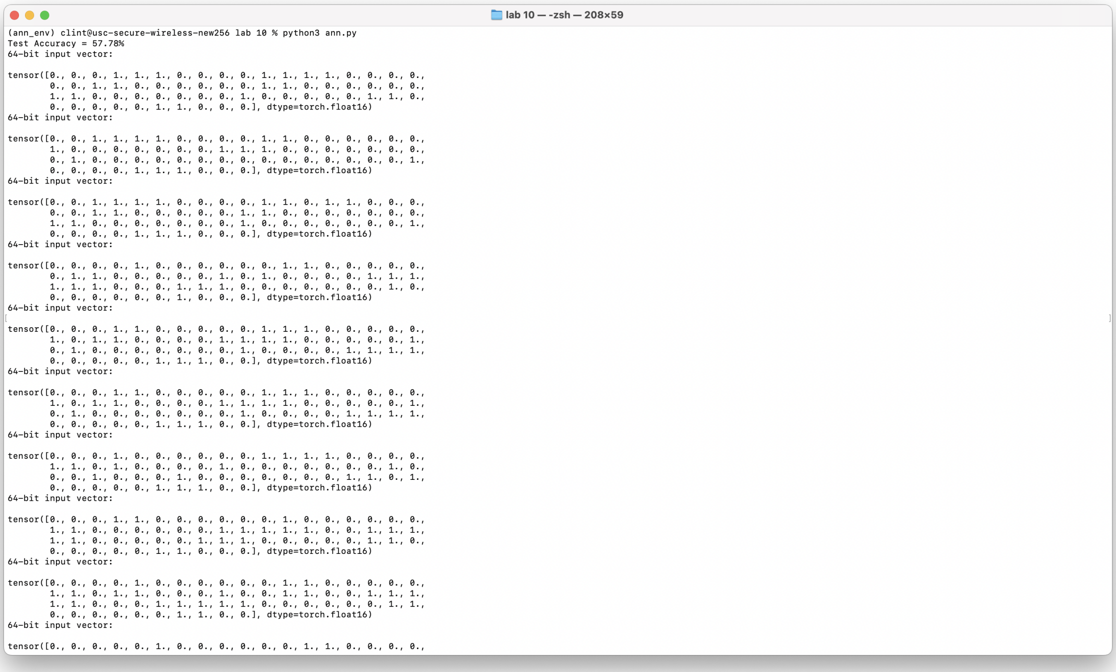
2.1.3 Software Training

- Terminal Window

```
(ann_env) clint@usc-secure-wireless-new256 lab 10 % python3 ann.py
Test Accuracy = 57.78%
64-bit input vector:

tensor([0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 0., 0., 0.,
        0., 0., 1., 1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.,
        1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 1., 1., 0.,
        0., 0., 0., 0., 0., 1., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 1., 1., 1., 1., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.,
        1., 0., 0., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0., 0., 0.,
        0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
        0., 0., 0., 0., 1., 1., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 1., 1., 1., 1., 0., 0., 0., 0., 1., 1., 0., 1., 1., 0., 0., 0.,
        0., 0., 1., 1., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0., 0.,
        1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 1.,
        0., 0., 0., 0., 1., 1., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:
```

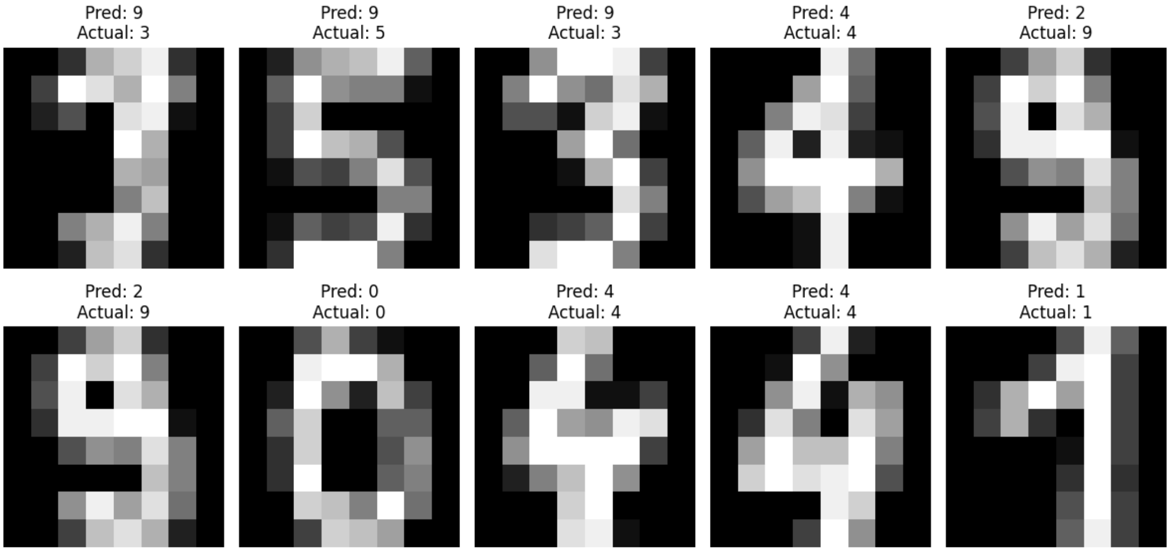
```
tensor([0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0.,
        0., 1., 1., 0., 0., 0., 0., 0., 1., 0., 1., 0., 0., 0., 0., 1., 1., 1.,
        1., 1., 1., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0.,
        0., 0., 0., 0., 0., 0., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0.,
        1., 0., 1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 0., 0., 0., 0., 1.,
        0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 1., 1., 1., 1.,
        0., 0., 0., 0., 0., 1., 1., 1., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0.,
        1., 0., 1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 0., 0., 0., 0., 1.,
        0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 1., 1., 1., 1.,
        0., 0., 0., 0., 0., 1., 1., 1., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 1., 1., 0., 0., 0., 0.,
        1., 1., 0., 1., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0.,
        0., 0., 1., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 1.,
        0., 0., 0., 0., 0., 1., 1., 1., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
        1., 1., 0., 0., 0., 0., 0., 0., 1., 1., 1., 1., 1., 0., 0., 1., 1., 1.,
        1., 1., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0., 1., 1., 0.,
        0., 0., 0., 0., 0., 1., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0.,
        1., 1., 0., 1., 1., 0., 0., 0., 1., 0., 0., 1., 1., 0., 0., 1., 1., 1.,
        1., 1., 0., 0., 0., 1., 1., 1., 1., 1., 0., 0., 0., 0., 0., 0., 1., 1.,
        0., 0., 0., 0., 0., 0., 1., 1., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0.,
        1., 1., 1., 1., 0., 0., 0., 0., 1., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
        0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 1.,
        0., 0., 0., 0., 0., 0., 0., 1., 0., 0.], dtype=torch.float16)
Hidden layer weights saved to hidden_weights.txt (float16 in text form).
Output layer weights saved to output_weights.txt (float16 in text form).
```

- Screenshot



```
(ann_env) clint@usc-secure-wireless-new256 lab 10 % python3 ann.py
Test Accuracy = 57.78%
64-bit input vector:

tensor([0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 0., 0., 0.,
        0., 0., 1., 1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.,
        1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 1., 1., 0.,
        0., 0., 0., 0., 0., 1., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 1., 1., 1., 1., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.,
        1., 0., 0., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0., 0., 0.,
        0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
        0., 0., 0., 0., 1., 1., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 1., 1., 1., 1., 0., 0., 0., 0., 1., 1., 0., 1., 1., 0., 0., 0.,
        0., 0., 1., 1., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0., 0.,
        1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 1.,
        0., 0., 0., 0., 1., 1., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0.,
        0., 1., 1., 0., 0., 0., 0., 0., 1., 0., 1., 0., 0., 0., 0., 1., 1., 1.,
        1., 1., 1., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0.,
        0., 0., 0., 0., 0., 0., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0.,
        1., 0., 1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 0., 0., 0., 0., 1.,
        0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 1., 1., 1., 1.,
        0., 0., 0., 0., 0., 1., 1., 1., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0.,
        1., 0., 1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 0., 0., 0., 0., 1.,
        0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 1., 1., 1., 1.,
        0., 0., 0., 0., 0., 1., 1., 1., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 1., 1., 0., 0., 0., 0.,
        1., 1., 0., 1., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0.,
        0., 0., 1., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 1.,
        0., 0., 0., 0., 0., 1., 1., 1., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
        1., 1., 0., 0., 0., 0., 0., 0., 1., 1., 1., 1., 1., 0., 0., 1., 1., 1.,
        1., 1., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0., 1., 1., 0.,
        0., 0., 0., 0., 0., 1., 1., 0., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0.,
        1., 1., 0., 1., 1., 0., 0., 0., 1., 0., 0., 1., 1., 0., 0., 1., 1., 1.,
        1., 1., 0., 0., 0., 1., 1., 1., 1., 1., 0., 0., 0., 0., 0., 0., 1., 1.,
        0., 0., 0., 0., 0., 0., 1., 1., 0., 0.], dtype=torch.float16)
64-bit input vector:

tensor([0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0.,
```

- Output Figure



- Output txt file
    - hidden_weights.txt

```
0.11346 0.04938 -0.60938 -0.67139 -1.47070 -0.83740 -2.18164 0.96777 0.03436
-0.12866 -1.27441 -1.15332 -0.85254 -0.15381 -0.42358 0.86084 -0.00831
-0.57422 0.56592 -0.16846 -0.14197 -0.42163 0.52783 -0.10315 0.02483 0.38745
0.53027 -0.46826 -0.67676 -0.15417 1.15625 0.00417 -0.07196 0.30273 -0.04443
0.70557 0.99707 -0.53516 -0.19482 0.07904 -0.05725 -0.25830 2.31445 2.24414
0.68555 -0.74219 -0.55225 0.11664 -0.06622 0.50977 0.59473 1.30371 0.67627
1.64551 0.26636 0.44946 0.09113 -0.00078 -0.58643 -0.45459 2.57812 1.75684
1.30176 0.90918
-0.09937 -0.00195 -1.33789 0.69873 -0.06909 -0.55713 1.34961 0.86768 -0.09528
0.39941 0.03546 -0.02039 0.76709 -0.43970 0.05701 -0.02724 -0.06036 -1.36133
-0.17407 -0.01154 -2.10156 -0.24487 -0.75293 0.03452 -0.01903 -0.29321
0.18445 -2.43750 -2.14062 -0.16626 4.50781 -0.02739 -0.11761 1.13770 2.35352
-0.07422 0.35962 1.20215 1.32520 0.04865 0.04080 2.04883 2.44336 0.74023
0.32617 0.27734 0.74316 -0.01714 -0.01633 0.20081 -0.74902 1.74609 -0.42847
0.54395 -0.53271 -0.17676 0.06323 -0.21790 0.77344 -0.46558 -1.20117 -0.55713
-0.66699 -0.37085
-0.00962 -0.05368 0.26562 0.55566 0.04770 -0.48267 1.16797 -0.01275 -0.08411
0.54248 0.65771 -0.11035 0.34155 0.15369 -0.02936 0.02759 0.11584 0.00439
0.05743 -1.33691 -0.43457 0.64355 -0.12292 -0.08966 -0.02228 -0.79492
-0.13208 -1.50879 -1.01367 -0.65186 0.07251 -0.02141 -0.09106 -0.60059
-0.03363 -0.27563 -0.88721 -0.90918 -0.05042 -0.07495 0.05090 -1.27539
0.38574 -0.56250 -1.29980 0.07166 1.08008 -0.08344 0.11786 0.14221 -0.05258
0.94824 1.13770 1.05176 0.38379 -1.54395 0.00459 0.23779 1.02930 0.12915
2.55078 0.95459 1.05859 -1.97754
```

- output_weights.txt

```
-0.55615 1.22559 1.35645
2.02344 -1.91895 -1.44043
1.35156 -2.07812 0.91113
-1.10449 -1.57227 1.36621
1.06152 1.78516 -3.71094
-1.79199 0.98438 1.42383
1.38477 0.63818 -0.23511
-1.94336 2.14453 -1.70215
1.74316 -1.45703 0.14514
-1.40234 0.20984 1.47852
```

2.2 Inference.cpp

2.2.1 Modified Inference_new.cpp

```cpp
#include <iostream>
#include <fstream>
#include <vector>
#include <cmath>

using half = __fp16;

const int INPUT_SIZE = 64;
const int HIDDEN_SIZE = 3;
const int OUTPUT_SIZE = 10;
const int IMAGE_NUM = 10;
```

```
void load_weights(const std::string& filename, std::vector<half>& weights) {
    std::ifstream file(filename);
    if (!file) {
        std::cerr << "Error: Could not open " << filename << std::endl;
        exit(1);
    }
    float value;
    while (file >> value) {
        weights.push_back(static_cast<half>(value));
    }
    file.close();
}

inline half relu(half x) {
    return (x > 0) ? x : 0;
}

int main() {
    std::vector<half> hidden_weights;
    std::vector<half> output_weights;

    load_weights("hidden_weights.txt", hidden_weights);
    load_weights("output_weights.txt", output_weights);

    // half input[INPUT_SIZE] = {
    //     0, 1, 0, 0, 1, 1, 0, 1,
    //     1, 0, 1, 1, 0, 0, 1, 1,
    //     0, 1, 1, 0, 1, 1, 0, 1,
    //     0, 1, 0, 0, 1, 1, 0, 1,
    //     1, 0, 1, 1, 0, 0, 1, 1,
    //     0, 1, 1, 0, 1, 1, 0, 1,
    //     0, 1, 0, 0, 1, 1, 0, 1,
    //     1, 0, 1, 1, 0, 0, 1, 1
    // };
    half input[IMAGE_NUM][INPUT_SIZE] = {
        {
            0, 0, 0, 1, 1, 1, 0, 0,
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 0, 0, 1, 1, 0, 0,
            0, 0, 0, 0, 1, 1, 0, 0,
            0, 0, 0, 0, 1, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 0, 1, 1, 0, 0, 0
        },
        {
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 1, 1, 0, 0, 0, 0,
            0, 0, 1, 0, 0, 0, 0, 0,
            0, 0, 1, 1, 1, 0, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 1, 1, 1, 0, 0, 0
        },
        {
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 1, 1, 0, 1, 1, 0,
            0, 0, 0, 0, 1, 1, 0, 0,
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 0, 0, 1, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 1, 1, 1, 0, 0, 0
        },
        {
            0, 0, 0, 0, 1, 0, 0, 0,
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 1, 0, 1, 0, 0, 0,
            0, 1, 1, 1, 1, 1, 1, 0,
            0, 0, 1, 1, 1, 0, 0, 0,
            0, 0, 0, 0, 1, 0, 0, 0,
            0, 0, 0, 0, 1, 0, 0, 0
        },
        {
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 1, 1, 1, 0, 0, 0,
            0, 0, 1, 0, 1, 1, 0, 0,
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 0, 1, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 0, 1, 1, 1, 0, 0
        },
        {
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 1, 1, 1, 0, 0, 0,
            0, 0, 1, 0, 1, 1, 0, 0,
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 0, 1, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 0, 1, 1, 1, 0, 0
        },
        {
            0, 0, 0, 1, 0, 0, 0, 0,
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 1, 1, 0, 1, 0, 0,
            0, 0, 1, 0, 0, 0, 0, 0,
            0, 0, 1, 0, 0, 0, 1, 0,
            0, 0, 1, 0, 0, 0, 0, 0,
            0, 0, 1, 1, 0, 1, 0, 0,
            0, 0, 0, 1, 1, 1, 0, 0
        },
        {
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 0, 1, 0, 0, 0, 0,
            0, 0, 1, 1, 0, 0, 0, 0,
            0, 0, 1, 1, 1, 1, 1, 0,
            0, 1, 1, 1, 1, 1, 0, 0,
            0, 0, 0, 1, 1, 1, 0, 0,
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 0, 1, 1, 0, 0, 0
        },
        {
            0, 0, 0, 0, 1, 0, 0, 0,
            0, 0, 0, 1, 1, 0, 0, 0,
            0, 0, 1, 1, 0, 1, 1, 0,
            0, 0, 1, 0, 0, 1, 1, 0,
            0, 1, 1, 1, 1, 1, 0, 0,
            0, 1, 1, 1, 1, 1, 0, 0,
            0, 0, 0, 0, 1, 1, 0, 0,
            0, 0, 0, 0, 1, 1, 0, 0
        },
        {
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 1, 1, 0, 0,
            0, 0, 1, 1, 1, 1, 0, 0,
            0, 0, 1, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0
        }
    };
    // input[0][INPUT_SIZE] = {
    //     0, 0, 0, 1, 1, 1, 0, 0,
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 0, 0, 1, 1, 0, 0,
    //     0, 0, 0, 0, 1, 1, 0, 0,
    //     0, 0, 0, 0, 1, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 0, 1, 1, 0, 0, 0
    // };
    // input[1][INPUT_SIZE] = {
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 1, 1, 0, 0, 0, 0,
    //     0, 0, 1, 0, 0, 0, 0, 0,
    //     0, 0, 1, 1, 1, 0, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 0, 0, 0, 0, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 1, 1, 1, 0, 0, 0
    // };
    // input[2][INPUT_SIZE] = {
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 1, 1, 0, 1, 1, 0,
    //     0, 0, 0, 0, 1, 1, 0, 0,
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 0, 0, 1, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 1, 1, 1, 0, 0, 0
    // };
    // input[3][INPUT_SIZE] = {
    //     0, 0, 0, 0, 1, 0, 0, 0,
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 1, 0, 1, 0, 0, 0,
    //     0, 1, 1, 1, 1, 1, 1, 0,
    //     0, 0, 1, 1, 1, 0, 0, 0,
    //     0, 0, 0, 0, 1, 0, 0, 0,
    //     0, 0, 0, 0, 1, 0, 0, 0
    // };
    // input[4][INPUT_SIZE] = {
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 1, 1, 1, 0, 0, 0,
    //     0, 0, 1, 0, 1, 1, 0, 0,
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 0, 1, 0, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 0, 1, 1, 1, 0, 0
    // };
    // input[5][INPUT_SIZE] = {
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 1, 1, 1, 0, 0, 0,
    //     0, 0, 1, 0, 1, 1, 0, 0,
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 0, 1, 0, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 0, 1, 1, 1, 0, 0
    // };
    // input[6][INPUT_SIZE] = {
    //     0, 0, 0, 1, 0, 0, 0, 0,
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 1, 1, 0, 1, 0, 0,
    //     0, 0, 1, 0, 0, 0, 0, 0,
    //     0, 0, 1, 0, 0, 0, 1, 0,
    //     0, 0, 1, 0, 0, 0, 0, 0,
    //     0, 0, 1, 1, 0, 1, 0, 0,
    //     0, 0, 0, 1, 1, 1, 0, 0
    // };
    // input[7][INPUT_SIZE] = {
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 0, 1, 0, 0, 0, 0,
    //     0, 0, 1, 1, 0, 0, 0, 0,
    //     0, 0, 1, 1, 1, 1, 1, 0,
    //     0, 1, 1, 1, 1, 1, 0, 0,
    //     0, 0, 0, 1, 1, 1, 0, 0,
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 0, 1, 1, 0, 0, 0
    // };
    // input[8][INPUT_SIZE] = {
    //     0, 0, 0, 0, 1, 0, 0, 0,
    //     0, 0, 0, 1, 1, 0, 0, 0,
    //     0, 0, 1, 1, 0, 1, 1, 0,
    //     0, 0, 1, 0, 0, 1, 1, 0,
    //     0, 1, 1, 1, 1, 1, 0, 0,
    //     0, 1, 1, 1, 1, 1, 0, 0,
    //     0, 0, 0, 0, 1, 1, 0, 0,
    //     0, 0, 0, 0, 1, 1, 0, 0
    // };
    // input[9][INPUT_SIZE] = {
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 0, 0, 1, 1, 0, 0,
    //     0, 0, 1, 1, 1, 1, 0, 0,
    //     0, 0, 1, 0, 0, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0,
    //     0, 0, 0, 0, 0, 1, 0, 0
    // };

    for (int img = 0; img < IMAGE_NUM; img++) {
        half hidden[HIDDEN_SIZE] = {0, 0, 0};
        for (int h = 0; h < HIDDEN_SIZE; h++) {
            half sum = 0;
            for (int i = 0; i < INPUT_SIZE; i++) {
                sum += input[img][i] * hidden_weights[h * INPUT_SIZE + i];
            }
            hidden[h] = relu(sum);
        }

        half output[OUTPUT_SIZE] = {0};
        for (int o = 0; o < OUTPUT_SIZE; o++) {
            half sum = 0;
            for (int h = 0; h < HIDDEN_SIZE; h++) {
                sum += hidden[h] * output_weights[o * HIDDEN_SIZE + h];
            }
            output[o] = sum;
        }

        int predicted_class = 0;
        half max_value = output[0];
        for (int o = 1; o < OUTPUT_SIZE; o++) {
            if (output[o] > max_value) {
                max_value = output[o];
                predicted_class = o;
            }
        }

        std::cout << "Image " << img << " Output logits: ";
        for (int o = 0; o < OUTPUT_SIZE; o++) {
            std::cout << static_cast<float>(output[o]) << " ";
        }
        std::cout << "\nPredicted class: " << predicted_class << std::endl <<
std::endl;
    }

    return 0;
}
```

2.2.2 Apply and Run ./inference

- Terminal Window

```
(ann_env) clint@usc-secure-wireless-new256 lab 10 % g++ -o inference
inference.cpp -std=c++11
(ann_env) clint@usc-secure-wireless-new256 lab 10 % ./inference
Output logits (printed as floats, stored in half):
0  0  0  0  0  0  0  0  0  0
Predicted class: 0
(ann_env) clint@usc-secure-wireless-new256 lab 10 % ./inference
Output logits (printed as floats, stored in half):
-0.0938721  -0.06604  0.210205  0.088562  -0.158203  0.0353088  -0.217041
-0.20874  0.189575  -0.153687
Predicted class: 2
(ann_env) clint@usc-secure-wireless-new256 lab 10 % g++ -o inference_new
inference_new.cpp -std=c++11
(ann_env) clint@usc-secure-wireless-new256 lab 10 % ./inference_new
Image 0 Output logits: 3.72852 -3.95898 2.50391 3.75391 -10.1953 3.91211
-0.645996 -4.67578 0.398926 4.0625
Predicted class: 9

Image 1 Output logits: 2.9707 -3.1543 1.99512 2.99219 -8.125 3.11719 -0.514648
-3.72656 0.317871 3.23633
Predicted class: 9

Image 2 Output logits: 2.41797 -2.56641 1.62402 2.43555 -6.61328 2.53711
-0.418945 -3.0332 0.258789 2.63477
Predicted class: 9

Image 3 Output logits: 2.64453 2.53711 -2.11719 -13.9297 14.7031 -5.67578 11.0781
-0.995117 3.12305 -7.12891
Predicted class: 4

Image 4 Output logits: 4.75781 -3.125 5.51562 4.05859 -13.6328 3.36914 0.908203
-9.52344 2.93359 4.11328
Predicted class: 2

Image 5 Output logits: 4.75781 -3.125 5.51562 4.05859 -13.6328 3.36914 0.908203
-9.52344 2.93359 4.11328
Predicted class: 2

Image 6 Output logits: 13.4531 -9.67188 1.14941 -7.14453 -7.51172 5.10156 10.5781
-9.53906 2.19336 2.94922
Predicted class: 0

Image 7 Output logits: 4.89062 -1.65723 -6.16797 -15.7344 16.8594 -3.06836
11.2578 3.51172 -0.184082 -5.96484
Predicted class: 4

Image 8 Output logits: 10.1016 -3.27539 -12.6875 -32.75 35.0938 -6.53516 23.5
7.10938 -0.230225 -12.5078
Predicted class: 4
```

```
Image 9 Output logits: -0.273682 0.996094 0.665039 -0.543457 0.522461 -0.881836
0.681641 -0.956543 0.85791 -0.69043
Predicted class: 1

(ann_env) clint@usc-secure-wireless-new256 lab 10 %
```

3. NPU

- Schematic

- Verilog

```
////////////////////////////////////////////////////////////////////////////////
// Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.
////////////////////////////////////////////////////////////////////////////////
//   ____  ____
//  /   /\/   /
// /___/  \  /    Vendor: Xilinx
// \   \   \/     Version : 10.1
//  \   \         Application : sch2verilog
//  /   /         Filename : NPU.vf
// /___/   /\     Timestamp : 03/22/2025 14:41:15
// \   \  /  \
//  \___\/\___\
//
//Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -
family virtex2p -w "E:/Documents and Settings/student/EE533_Lab10/NPU.sch" NPU.vf
//Design Name: NPU
//Device: virtex2p
//Purpose:
//    This verilog netlist is translated from an ECS schematic.It can be
//    synthesized and simulated, but it should not be modified.
//
`timescale 1ns / 1ps

module NPU(clk,
           en,
           image,
           neurons_hidden_weight_addr,
           neurons_hidden_wen,
           neurons_hidden_0_weight_in,
           neurons_hidden_1_weight_in,
           neurons_hidden_2_weight_in,
           neurons_output_weight_addr,
           neurons_output_wen,
           neurons_output_0_weight_in,
           neurons_output_1_weight_in,
           neurons_output_2_weight_in,
           neurons_output_3_weight_in,
           neurons_output_4_weight_in,
           neurons_output_5_weight_in,
           neurons_output_6_weight_in,
           neurons_output_7_weight_in,
           neurons_output_8_weight_in,
           neurons_output_9_weight_in,
           rst,
           prediction_class,
           prediction_score);

    input clk;
    input en;
    input [63:0] image;
    input [5:0] neurons_hidden_weight_addr;
    input neurons_hidden_wen;
    input [15:0] neurons_hidden_0_weight_in;
    input [15:0] neurons_hidden_1_weight_in;
    input [15:0] neurons_hidden_2_weight_in;
    input [1:0] neurons_output_weight_addr;
    input neurons_output_wen;
    input [15:0] neurons_output_0_weight_in;
    input [15:0] neurons_output_1_weight_in;
    input [15:0] neurons_output_2_weight_in;
    input [15:0] neurons_output_3_weight_in;
    input [15:0] neurons_output_4_weight_in;
    input [15:0] neurons_output_5_weight_in;
    input [15:0] neurons_output_6_weight_in;
    input [15:0] neurons_output_7_weight_in;
    input [15:0] neurons_output_8_weight_in;
    input [15:0] neurons_output_9_weight_in;
    input rst;
   output [3:0] prediction_class;
   output [15:0] prediction_score;

   wire [15:0] class_0_value;
   wire [15:0] class_1_value;
   wire [15:0] class_2_value;
   wire [15:0] class_3_value;
   wire [15:0] class_4_value;
   wire [15:0] class_5_value;
   wire [15:0] class_6_value;
   wire [15:0] class_7_value;
   wire [15:0] class_8_value;
   wire [15:0] class_9_value;
   wire [15:0] neurons_hidden_0_out;
   wire [15:0] neurons_hidden_1_out;
   wire [15:0] neurons_hidden_2_out;
   wire [15:0] ReLu_1;
   wire [15:0] ReLu_2;
   wire [15:0] ReLu_3;

   neurons_hidden neurons_hidden_0 (.clk(clk),
                                    .en(en),
                                    .image(image[63:0]),
                                    .rst(rst),

         .weight_addr(neurons_hidden_weight_addr[5:0]),

         .weight_in(neurons_hidden_0_weight_in[15:0]),
                                    .wen(neurons_hidden_wen),
         .neurons_output(neurons_hidden_0_out[15:0]));
   neurons_hidden neurons_hidden_1 (.clk(clk),
                                    .en(en),
                                    .image(image[63:0]),
                                    .rst(rst),

         .weight_addr(neurons_hidden_weight_addr[5:0]),

         .weight_in(neurons_hidden_1_weight_in[15:0]),
                                    .wen(neurons_hidden_wen),

         .neurons_output(neurons_hidden_1_out[15:0]));
   neurons_hidden neurons_hidden_2 (.clk(clk),
                                    .en(en),
                                    .image(image[63:0]),
                                    .rst(rst),

         .weight_addr(neurons_hidden_weight_addr[5:0]),

         .weight_in(neurons_hidden_2_weight_in[15:0]),
                                    .wen(neurons_hidden_wen),

         .neurons_output(neurons_hidden_2_out[15:0]));
   neurons_output neurons_output_0 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_0_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_0_value[15:0]));
   neurons_output neurons_output_1 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_1_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_1_value[15:0]));
   neurons_output neurons_output_2 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),
         .weight_in(neurons_output_2_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_2_value[15:0]));
   neurons_output neurons_output_3 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_3_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_3_value[15:0]));
   neurons_output neurons_output_4 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_4_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_4_value[15:0]));
   neurons_output neurons_output_5 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_5_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_5_value[15:0]));
   neurons_output neurons_output_6 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_6_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_6_value[15:0]));
   neurons_output neurons_output_7 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_7_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_7_value[15:0]));
   neurons_output neurons_output_8 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_8_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_8_value[15:0]));
   neurons_output neurons_output_9 (.clk(clk),
                                    .en(en),
                                    .ReLu_1(ReLu_1[15:0]),
                                    .ReLu_2(ReLu_2[15:0]),
                                    .ReLu_3(ReLu_3[15:0]),
                                    .rst(rst),

         .weight_addr(neurons_output_weight_addr[1:0]),

         .weight_in(neurons_output_9_weight_in[15:0]),
                                    .wen(neurons_output_wen),
                                    .Prediction_score(class_9_value[15:0]));
   prediction_output prediction_result (.neurons_output0(class_0_value[15:0]),
                                        .neurons_output1(class_1_value[15:0]),
                                        .neurons_output2(class_2_value[15:0]),
                                        .neurons_output3(class_3_value[15:0]),
                                        .neurons_output4(class_4_value[15:0]),
                                        .neurons_output5(class_5_value[15:0]),
                                        .neurons_output6(class_6_value[15:0]),
                                        .neurons_output7(class_7_value[15:0]),
                                        .neurons_output8(class_8_value[15:0]),
                                        .neurons_output9(class_9_value[15:0]),

         .prediction_class(prediction_class[3:0]),

         .prediction_score(prediction_score[15:0]));
   ReLu ReLU_MUX (.neurons_1(neurons_hidden_0_out[15:0]),
                  .neurons_2(neurons_hidden_1_out[15:0]),
                  .neurons_3(neurons_hidden_2_out[15:0]),
                  .neurons_1_out(ReLu_1[15:0]),
                  .neurons_2_out(ReLu_2[15:0]),
                  .neurons_3_out(ReLu_3[15:0]));
endmodule
```

- Testbench

```verilog
`timescale 1ns / 1ps

////////////////////////////////////////////////////////////////////////////////
// Company: 
// Engineer:
//
// Create Date:   14:46:56 03/22/2025
// Design Name:   NPU
// Module Name:   E:/Documents and 
Settings/student/EE533_Lab10/EE533_Lab_10/NPU_tb.v
// Project Name:  EE533_Lab_10
// Target Device:  
// Tool versions:  
// Description: 
//
// Verilog Test Fixture created by ISE for module: NPU
//
// Dependencies:
// 
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
// 
////////////////////////////////////////////////////////////////////////////////

module NPU_tb;

    // Inputs
    reg clk;
    reg en;
    reg [63:0] image;
    reg [5:0] neurons_hidden_weight_addr;
    reg neurons_hidden_wen;
    reg [15:0] neurons_hidden_0_weight_in;
    reg [15:0] neurons_hidden_1_weight_in;
    reg [15:0] neurons_hidden_2_weight_in;
    reg [1:0] neurons_output_weight_addr;
    reg neurons_output_wen;
    reg [15:0] neurons_output_0_weight_in;
    reg [15:0] neurons_output_1_weight_in;
    reg [15:0] neurons_output_2_weight_in;
    reg [15:0] neurons_output_3_weight_in;
    reg [15:0] neurons_output_4_weight_in;
    reg [15:0] neurons_output_5_weight_in;
    reg [15:0] neurons_output_6_weight_in;
    reg [15:0] neurons_output_7_weight_in;
    reg [15:0] neurons_output_8_weight_in;
    reg [15:0] neurons_output_9_weight_in;
    reg rst;

    // Outputs
    wire [3:0] prediction_class;
    wire [15:0] prediction_score;

    // Instantiate the Unit Under Test (UUT)
    NPU uut (
        .clk(clk),
        .en(en),
        .image(image),
        .neurons_hidden_weight_addr(neurons_hidden_weight_addr),
        .neurons_hidden_wen(neurons_hidden_wen),
        .neurons_hidden_0_weight_in(neurons_hidden_0_weight_in),
        .neurons_hidden_1_weight_in(neurons_hidden_1_weight_in),
        .neurons_hidden_2_weight_in(neurons_hidden_2_weight_in),
        .neurons_output_weight_addr(neurons_output_weight_addr),
        .neurons_output_wen(neurons_output_wen),
        .neurons_output_0_weight_in(neurons_output_0_weight_in),
        .neurons_output_1_weight_in(neurons_output_1_weight_in),
        .neurons_output_2_weight_in(neurons_output_2_weight_in),
        .neurons_output_3_weight_in(neurons_output_3_weight_in),
        .neurons_output_4_weight_in(neurons_output_4_weight_in),
        .neurons_output_5_weight_in(neurons_output_5_weight_in),
        .neurons_output_6_weight_in(neurons_output_6_weight_in),
        .neurons_output_7_weight_in(neurons_output_7_weight_in),
        .neurons_output_8_weight_in(neurons_output_8_weight_in),
        .neurons_output_9_weight_in(neurons_output_9_weight_in),
        .rst(rst),
        .prediction_class(prediction_class),
        .prediction_score(prediction_score)
    );

    always #50 clk = ~clk;

    initial begin
        // Initialize Inputs
        clk = 1;
        en = 0;
        image = 0;
        neurons_hidden_weight_addr = 0;
        neurons_hidden_wen = 0;
        neurons_hidden_0_weight_in = 0;
        neurons_hidden_1_weight_in = 0;
        neurons_hidden_2_weight_in = 0;
        neurons_output_weight_addr = 0;
        neurons_output_wen = 0;
        neurons_output_0_weight_in = 0;
        neurons_output_1_weight_in = 0;
        neurons_output_2_weight_in = 0;
        neurons_output_3_weight_in = 0;
        neurons_output_4_weight_in = 0;
        neurons_output_5_weight_in = 0;
        neurons_output_6_weight_in = 0;
        neurons_output_7_weight_in = 0;
        neurons_output_8_weight_in = 0;
        neurons_output_9_weight_in = 0;
        rst = 1;

        // Wait 100 ns for global reset to finish
        @(posedge clk);

        // Add stimulus here
        rst = 0;
        neurons_hidden_wen = 1;
        neurons_hidden_weight_addr = 6'd0; neurons_hidden_0_weight_in = 16'h2c72;
neurons_hidden_1_weight_in = 16'hf99e; neurons_hidden_2_weight_in = 16'hf9f2;

        neurons_output_wen = 1;
        neurons_output_weight_addr = 2'd0;
        neurons_output_0_weight_in = 16'hb820;
        neurons_output_1_weight_in = 16'h4240;
        neurons_output_2_weight_in = 16'h3d90;
        neurons_output_3_weight_in = 16'hbb00;
        neurons_output_4_weight_in = 16'h3b80;
        neurons_output_5_weight_in = 16'hbd2e;
        neurons_output_6_weight_in = 16'h3d88;
        neurons_output_7_weight_in = 16'hbd80;
        neurons_output_8_weight_in = 16'h3da0;
        neurons_output_9_weight_in = 16'hbbd0;
        @(posedge clk);

        neurons_hidden_weight_addr = 6'd1; neurons_hidden_0_weight_in = 16'h2932;
neurons_hidden_1_weight_in = 16'hf7fe; neurons_hidden_2_weight_in = 16'hf8e4;

        neurons_output_weight_addr = 2'd1;
        neurons_output_0_weight_in = 16'h3d26;
        neurons_output_1_weight_in = 16'hbd6a;
        neurons_output_2_weight_in = 16'hbdc0;
        neurons_output_3_weight_in = 16'hbcf2;
        neurons_output_4_weight_in = 16'h3ce4;
        neurons_output_5_weight_in = 16'h3c20;
        neurons_output_6_weight_in = 16'h3a00;
        neurons_output_7_weight_in = 16'h4280;
        neurons_output_8_weight_in = 16'hbc52;
        neurons_output_9_weight_in = 16'h3380;
        @(posedge clk);

        neurons_hidden_weight_addr = 6'd2; neurons_hidden_0_weight_in = 16'he4e3;
neurons_hidden_1_weight_in = 16'hd1aa; neurons_hidden_2_weight_in = 16'h2b20;

        neurons_output_weight_addr = 2'd2;
        neurons_output_0_weight_in = 16'h3d92;
        neurons_output_1_weight_in = 16'hbc4a;
        neurons_output_2_weight_in = 16'h3b98;
        neurons_output_3_weight_in = 16'h3d98;
        neurons_output_4_weight_in = 16'hc5f2;
        neurons_output_5_weight_in = 16'h3d4a;
        neurons_output_6_weight_in = 16'hac00;
        neurons_output_7_weight_in = 16'hbc9a;
        neurons_output_8_weight_in = 16'h3280;
        neurons_output_9_weight_in = 16'h3cfe;
        @(posedge clk);

        neurons_hidden_weight_addr = 6'd3; neurons_hidden_0_weight_in = 16'he49f;
neurons_hidden_1_weight_in = 16'h3af8; neurons_hidden_2_weight_in = 16'h3a50;

        neurons_output_wen = 0;
        neurons_output_weight_addr = 2'd0;
        neurons_output_0_weight_in = 0;
        neurons_output_1_weight_in = 0;
        neurons_output_2_weight_in = 0;
        neurons_output_3_weight_in = 0;
        neurons_output_4_weight_in = 0;
        neurons_output_5_weight_in = 0;
        neurons_output_6_weight_in = 0;
        neurons_output_7_weight_in = 0;
        neurons_output_8_weight_in = 0;
        neurons_output_9_weight_in = 0;
        @(posedge clk);

        neurons_hidden_weight_addr = 6'd4; neurons_hidden_0_weight_in = 16'hedb5;
neurons_hidden_1_weight_in = 16'hf9d9; neurons_hidden_2_weight_in = 16'h2b10;
@(posedge clk);
        neurons_hidden_weight_addr = 6'd5; neurons_hidden_0_weight_in = 16'he6a1;
neurons_hidden_1_weight_in = 16'hf89f; neurons_hidden_2_weight_in = 16'hf8c0;
@(posedge clk);
        neurons_hidden_weight_addr = 6'd6; neurons_hidden_0_weight_in = 16'hd3f4;
neurons_hidden_1_weight_in = 16'h3a90; neurons_hidden_2_weight_in = 16'h3b10;
@(posedge clk);
        neurons_hidden_weight_addr = 6'd7; neurons_hidden_0_weight_in = 16'h3b55;
neurons_hidden_1_weight_in = 16'h3a40; neurons_hidden_2_weight_in = 16'hf7f0;
@(posedge clk);
        neurons_hidden_weight_addr = 6'd8; neurons_hidden_0_weight_in = 16'h21c4;
neurons_hidden_1_weight_in = 16'hf9a9; neurons_hidden_2_weight_in = 16'hf8d4;
@(posedge clk);
        neurons_hidden_weight_addr = 6'd9; neurons_hidden_0_weight_in = 16'heccd;
neurons_hidden_1_weight_in = 16'h3a60; neurons_hidden_2_weight_in = 16'h3a70;
@(posedge clk);
        neurons_hidden_weight_addr = 6'd10; neurons_hidden_0_weight_in =
16'hd5a7; neurons_hidden_1_weight_in = 16'h3a04; neurons_hidden_2_weight_in =
16'h3a90; @(posedge clk);
        neurons_hidden_weight_addr = 6'd11; neurons_hidden_0_weight_in =
16'hd6c1; neurons_hidden_1_weight_in = 16'h3a10; neurons_hidden_2_weight_in =
16'hf8b0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd12; neurons_hidden_0_weight_in =
16'hdaac; neurons_hidden_1_weight_in = 16'h3a80; neurons_hidden_2_weight_in =
16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd13; neurons_hidden_0_weight_in =
16'hecc7; neurons_hidden_1_weight_in = 16'hf89d; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd14; neurons_hidden_0_weight_in =
16'hd977; neurons_hidden_1_weight_in = 16'h3a20; neurons_hidden_2_weight_in =
16'hf8d0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd15; neurons_hidden_0_weight_in =
16'h3a2c; neurons_hidden_1_weight_in = 16'hf8f0; neurons_hidden_2_weight_in =
16'h3a00; @(posedge clk);
        neurons_hidden_weight_addr = 6'd16; neurons_hidden_0_weight_in =
16'heef2; neurons_hidden_1_weight_in = 16'hf8f4; neurons_hidden_2_weight_in =
16'h3a40; @(posedge clk);
        neurons_hidden_weight_addr = 6'd17; neurons_hidden_0_weight_in =
16'hda95; neurons_hidden_1_weight_in = 16'hd1a2; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
```

```
        neurons_hidden_weight_addr = 6'd18; neurons_hidden_0_weight_in = 16'h3a50; neurons_hidden_1_weight_in = 16'hf95e; neurons_hidden_2_weight_in = 16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd19; neurons_hidden_0_weight_in = 16'h3a25; neurons_hidden_1_weight_in = 16'hf9fe; neurons_hidden_2_weight_in = 16'hd1a0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd20; neurons_hidden_0_weight_in = 16'h3a10; neurons_hidden_1_weight_in = 16'hc1c5; neurons_hidden_2_weight_in = 16'hf9c0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd21; neurons_hidden_0_weight_in = 16'hd73b; neurons_hidden_1_weight_in = 16'hf9f1; neurons_hidden_2_weight_in = 16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd22; neurons_hidden_0_weight_in = 16'h3a3e; neurons_hidden_1_weight_in = 16'hf8d4; neurons_hidden_2_weight_in = 16'hf8d0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd23; neurons_hidden_0_weight_in = 16'h21f0; neurons_hidden_1_weight_in = 16'h3a40; neurons_hidden_2_weight_in = 16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd24; neurons_hidden_0_weight_in = 16'h3a11; neurons_hidden_1_weight_in = 16'hf9e2; neurons_hidden_2_weight_in = 16'hf8e4; @(posedge clk);
        neurons_hidden_weight_addr = 6'd25; neurons_hidden_0_weight_in = 16'h2a80; neurons_hidden_1_weight_in = 16'hf7e1; neurons_hidden_2_weight_in = 16'hf7f8; @(posedge clk);
        neurons_hidden_weight_addr = 6'd26; neurons_hidden_0_weight_in = 16'h3a15; neurons_hidden_1_weight_in = 16'h3a10; neurons_hidden_2_weight_in = 16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd27; neurons_hidden_0_weight_in = 16'h2b4d; neurons_hidden_1_weight_in = 16'hf8d2; neurons_hidden_2_weight_in = 16'hf8c0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd28; neurons_hidden_0_weight_in = 16'he4c2; neurons_hidden_1_weight_in = 16'hf9a1; neurons_hidden_2_weight_in = 16'hf9a8; @(posedge clk);
        neurons_hidden_weight_addr = 6'd29; neurons_hidden_0_weight_in = 16'heea3; neurons_hidden_1_weight_in = 16'hf901; neurons_hidden_2_weight_in = 16'hf9d0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd30; neurons_hidden_0_weight_in = 16'h2d35; neurons_hidden_1_weight_in = 16'h3b90; neurons_hidden_2_weight_in = 16'h3a70; @(posedge clk);
        neurons_hidden_weight_addr = 6'd31; neurons_hidden_0_weight_in = 16'h3b80; neurons_hidden_1_weight_in = 16'hf9fd; neurons_hidden_2_weight_in = 16'hf8e0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd32; neurons_hidden_0_weight_in = 16'h3b15; neurons_hidden_1_weight_in = 16'hf9f9; neurons_hidden_2_weight_in = 16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd33; neurons_hidden_0_weight_in = 16'h3b40; neurons_hidden_1_weight_in = 16'h3b50; neurons_hidden_2_weight_in = 16'h3a80; @(posedge clk);
        neurons_hidden_weight_addr = 6'd34; neurons_hidden_0_weight_in = 16'hdaad; neurons_hidden_1_weight_in = 16'h3a90; neurons_hidden_2_weight_in = 16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd35; neurons_hidden_0_weight_in = 16'h3a80; neurons_hidden_1_weight_in = 16'h3ab4; neurons_hidden_2_weight_in = 16'hf8e4; @(posedge clk);
        neurons_hidden_weight_addr = 6'd36; neurons_hidden_0_weight_in =
16'h3a92; neurons_hidden_1_weight_in = 16'hf8c8; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd37; neurons_hidden_0_weight_in =
16'h3ab0; neurons_hidden_1_weight_in = 16'h3a30; neurons_hidden_2_weight_in =
16'hf8c0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd38; neurons_hidden_0_weight_in =
16'hd9f2; neurons_hidden_1_weight_in = 16'hf8d6; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd39; neurons_hidden_0_weight_in =
16'hdc34; neurons_hidden_1_weight_in = 16'h3a20; neurons_hidden_2_weight_in =
16'h3a30; @(posedge clk);
        neurons_hidden_weight_addr = 6'd40; neurons_hidden_0_weight_in =
16'h3a50; neurons_hidden_1_weight_in = 16'h3a80; neurons_hidden_2_weight_in =
16'h3a40; @(posedge clk);
        neurons_hidden_weight_addr = 6'd41; neurons_hidden_0_weight_in =
16'hd5d0; neurons_hidden_1_weight_in = 16'h3a10; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd42; neurons_hidden_0_weight_in =
16'hd8b0; neurons_hidden_1_weight_in = 16'h3a50; neurons_hidden_2_weight_in =
16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd43; neurons_hidden_0_weight_in =
16'h3a1f; neurons_hidden_1_weight_in = 16'h3a80; neurons_hidden_2_weight_in =
16'hf8d0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd44; neurons_hidden_0_weight_in =
16'hd8d5; neurons_hidden_1_weight_in = 16'h3a10; neurons_hidden_2_weight_in =
16'h3a00; @(posedge clk);
        neurons_hidden_weight_addr = 6'd45; neurons_hidden_0_weight_in =
16'hd7b4; neurons_hidden_1_weight_in = 16'h3a80; neurons_hidden_2_weight_in =
16'hf8e4; @(posedge clk);
        neurons_hidden_weight_addr = 6'd46; neurons_hidden_0_weight_in =
16'hd7e0; neurons_hidden_1_weight_in = 16'h3a10; neurons_hidden_2_weight_in =
16'hf8d0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd47; neurons_hidden_0_weight_in =
16'h3f90; neurons_hidden_1_weight_in = 16'h3a50; neurons_hidden_2_weight_in =
16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd48; neurons_hidden_0_weight_in =
16'h3c80; neurons_hidden_1_weight_in = 16'h3a10; neurons_hidden_2_weight_in =
16'hf8e4; @(posedge clk);
        neurons_hidden_weight_addr = 6'd49; neurons_hidden_0_weight_in =
16'h3a3d; neurons_hidden_1_weight_in = 16'h3ab0; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd50; neurons_hidden_0_weight_in =
16'h3aa0; neurons_hidden_1_weight_in = 16'h3ac0; neurons_hidden_2_weight_in =
16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd51; neurons_hidden_0_weight_in =
16'h3b02; neurons_hidden_1_weight_in = 16'h3c50; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd52; neurons_hidden_0_weight_in =
16'h3ac0; neurons_hidden_1_weight_in = 16'h3a4d; neurons_hidden_2_weight_in =
16'h3a30; @(posedge clk);
        neurons_hidden_weight_addr = 6'd53; neurons_hidden_0_weight_in =
16'h3c50; neurons_hidden_1_weight_in = 16'h3b65; neurons_hidden_2_weight_in =
16'h3a40; @(posedge clk);
```

```
        neurons_hidden_weight_addr = 6'd54; neurons_hidden_0_weight_in =
16'h3a4d; neurons_hidden_1_weight_in = 16'h2d20; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd55; neurons_hidden_0_weight_in =
16'h3b65; neurons_hidden_1_weight_in = 16'h0000; neurons_hidden_2_weight_in =
16'h3a50; @(posedge clk);
        neurons_hidden_weight_addr = 6'd56; neurons_hidden_0_weight_in =
16'h2d20; neurons_hidden_1_weight_in = 16'hedbf; neurons_hidden_2_weight_in =
16'hf9e0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd57; neurons_hidden_0_weight_in =
16'h0000; neurons_hidden_1_weight_in = 16'heed8; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd58; neurons_hidden_0_weight_in =
16'hedbf; neurons_hidden_1_weight_in = 16'h3c10; neurons_hidden_2_weight_in =
16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd59; neurons_hidden_0_weight_in =
16'heed8; neurons_hidden_1_weight_in = 16'h3b05; neurons_hidden_2_weight_in =
16'hf8d0; @(posedge clk);
        neurons_hidden_weight_addr = 6'd60; neurons_hidden_0_weight_in =
16'h3c10; neurons_hidden_1_weight_in = 16'h3a9f; neurons_hidden_2_weight_in =
16'h3a20; @(posedge clk);
        neurons_hidden_weight_addr = 6'd61; neurons_hidden_0_weight_in =
16'h3b05; neurons_hidden_1_weight_in = 16'h3a80; neurons_hidden_2_weight_in =
16'h3a10; @(posedge clk);
        neurons_hidden_weight_addr = 6'd62; neurons_hidden_0_weight_in =
16'h3a9f; neurons_hidden_1_weight_in = 16'hf89d; neurons_hidden_2_weight_in =
16'h3a30; @(posedge clk);
        neurons_hidden_weight_addr = 6'd63; neurons_hidden_0_weight_in =
16'h3a80; neurons_hidden_1_weight_in = 16'hf8f0; neurons_hidden_2_weight_in =
16'hf9d0; @(posedge clk);

        neurons_hidden_wen = 0;
        neurons_hidden_weight_addr = 6'd0; neurons_hidden_0_weight_in = 16'h0000;
neurons_hidden_1_weight_in = 16'h0000; neurons_hidden_2_weight_in = 16'h0000;
@(posedge clk);

        en = 1;
        image = 64'h1C3C0C0C0C041818; @(posedge clk);
        image = 64'h3C30203804000438; @(posedge clk);
        image = 64'h3C360C180C040438; @(posedge clk);
        image = 64'h081818287E380808; @(posedge clk);
        image = 64'h18382C3C14043C1C; @(posedge clk);
        image = 64'h18382C3C14043C1C; @(posedge clk);
        image = 64'h18382C380C040438; @(posedge clk);
        image = 64'h18103C207E180808; @(posedge clk);
        image = 64'h081836267C7C0C0C; @(posedge clk);
        image = 64'h040C3C2404040404; @(posedge clk);

        @(posedge clk);
        @(posedge clk);
        @(posedge clk);
        @(posedge clk);
        $stop;


    end
```

```
endmodule
```

4. GitHub Link

- https://github.com/Fanzzw01/ee533.git