4840 Systolic Array Based on FPGA

Speaker: Hang Ye(hy2891), Yu Jia(yj2839), Siyuan Li(sl5590), Sicheng Hua（sh4605)

Contents

Block Diagram

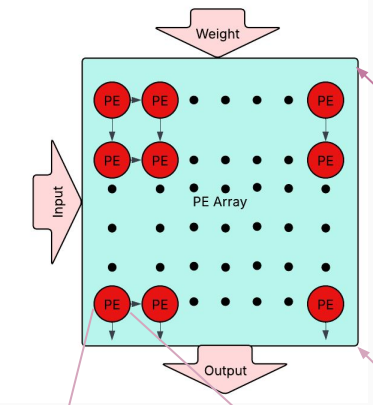


Figure 1: PE Array

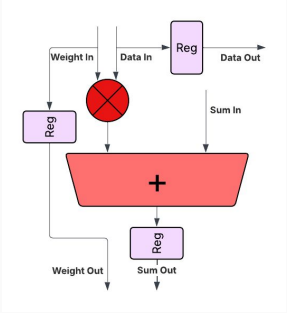


Figure 2: PE Unit

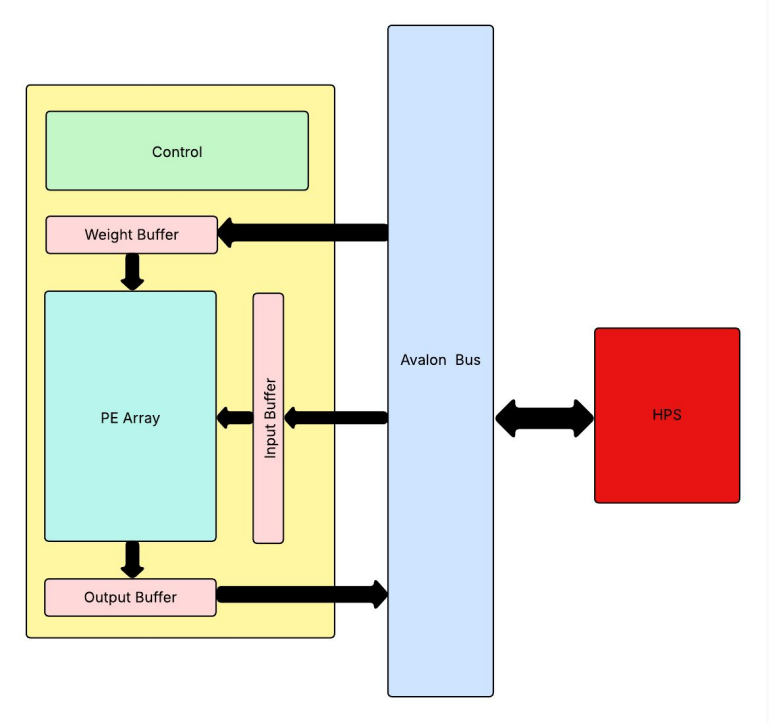
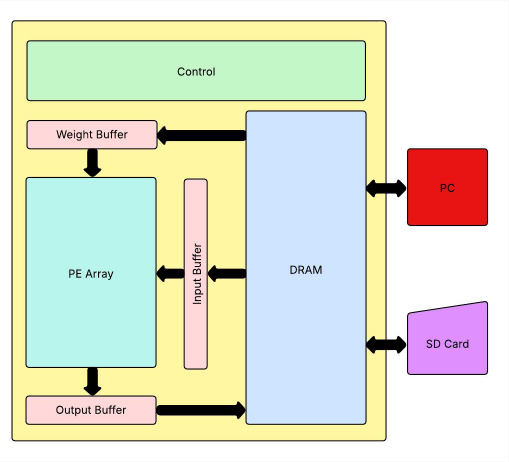


Figure 3: Block Diagram



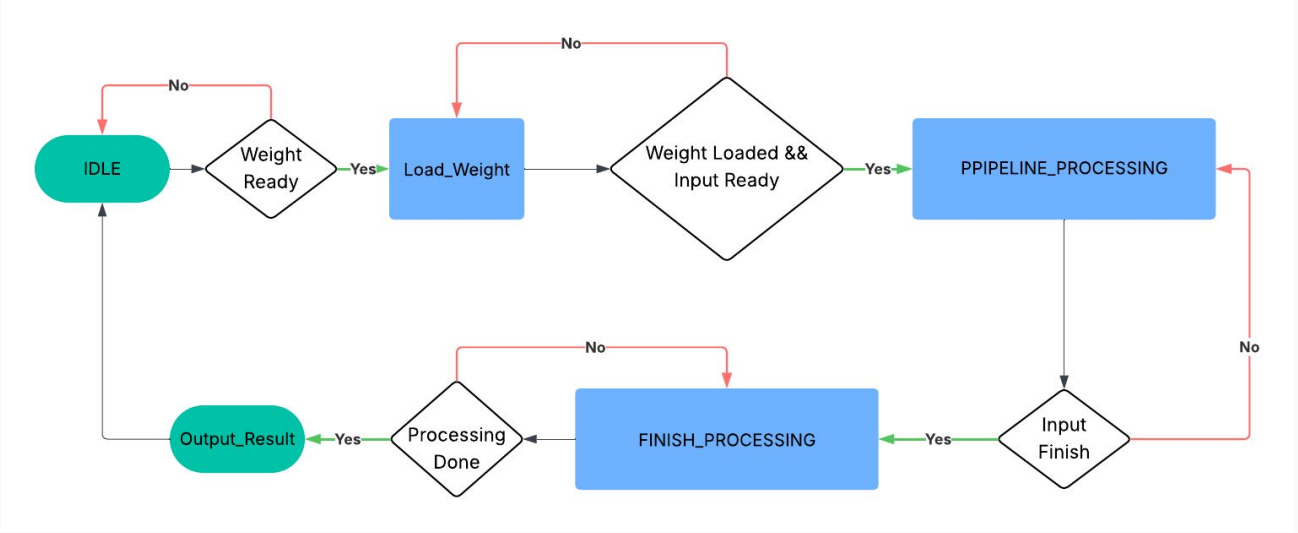


Figure 4 : FSM Diagram

Data Flow

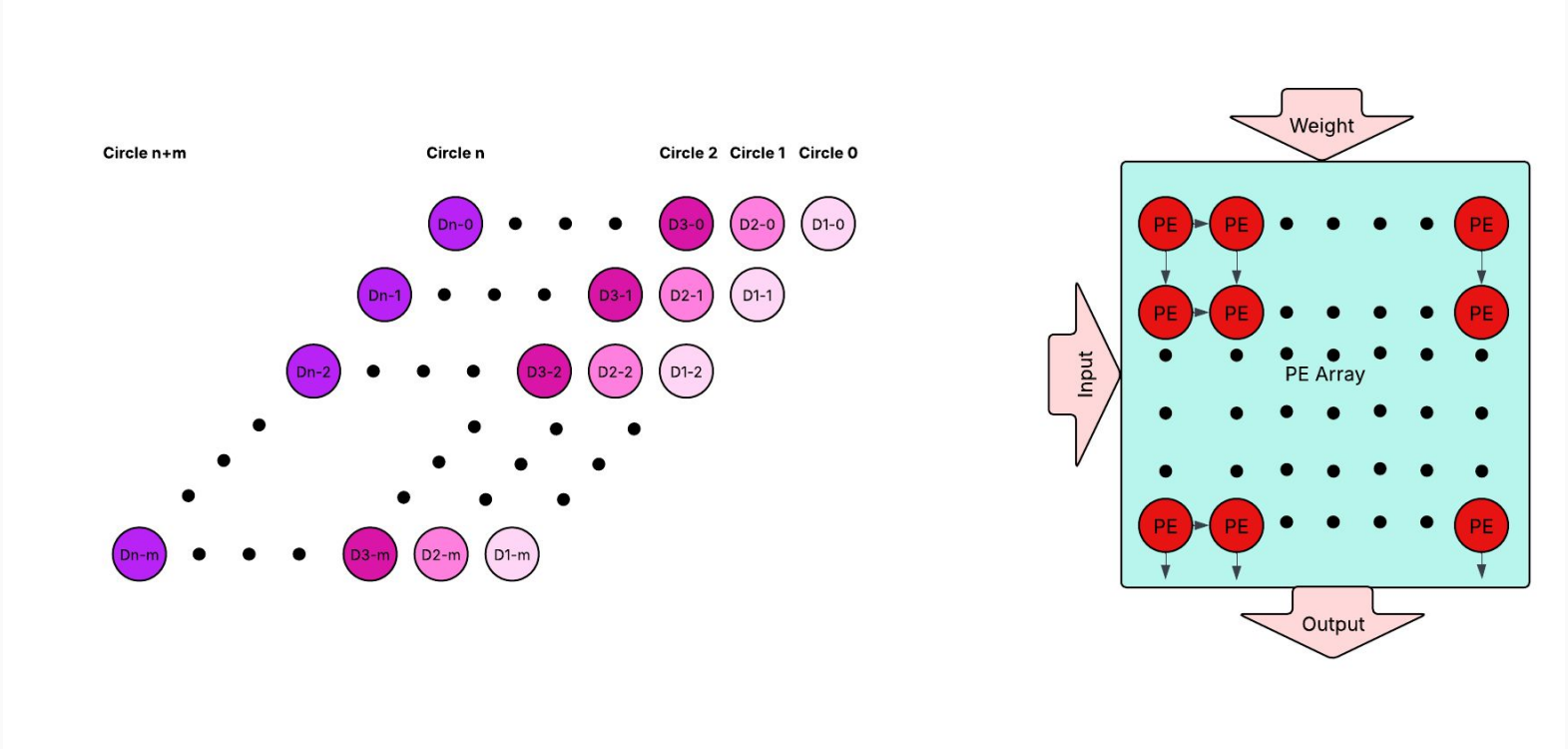
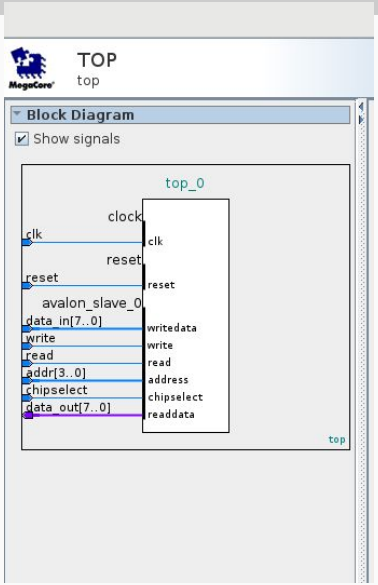
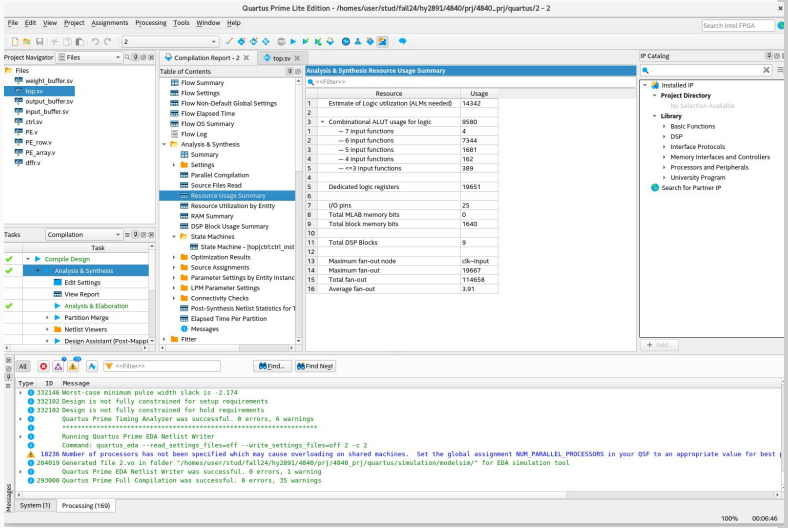


Figure 5 : Data Timing Diagram

Synthesize

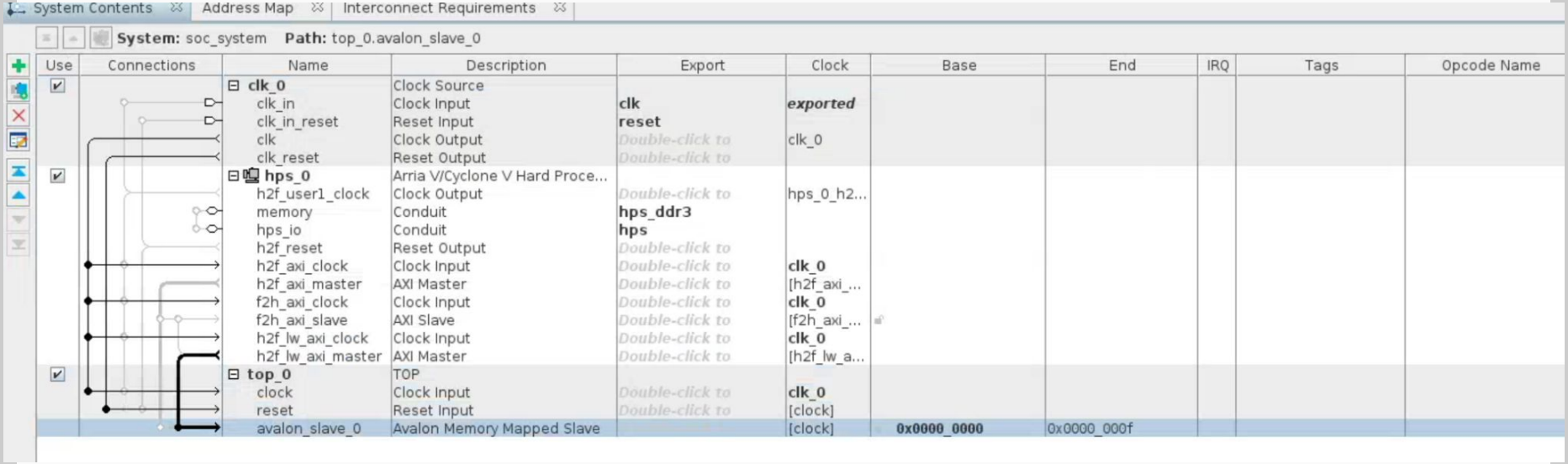
portname	Avalon Bus interface
addr	address
data_in	writedata 0 : imgsize 1 : weight_data 2 : input_data
data_out	readdata 3 : done 4 : output_data
write	write
read	read
clk	clock
reset	reset
chipselect	chipselect





Pictures for hardware synthesize

Final Connection



Simulation Result

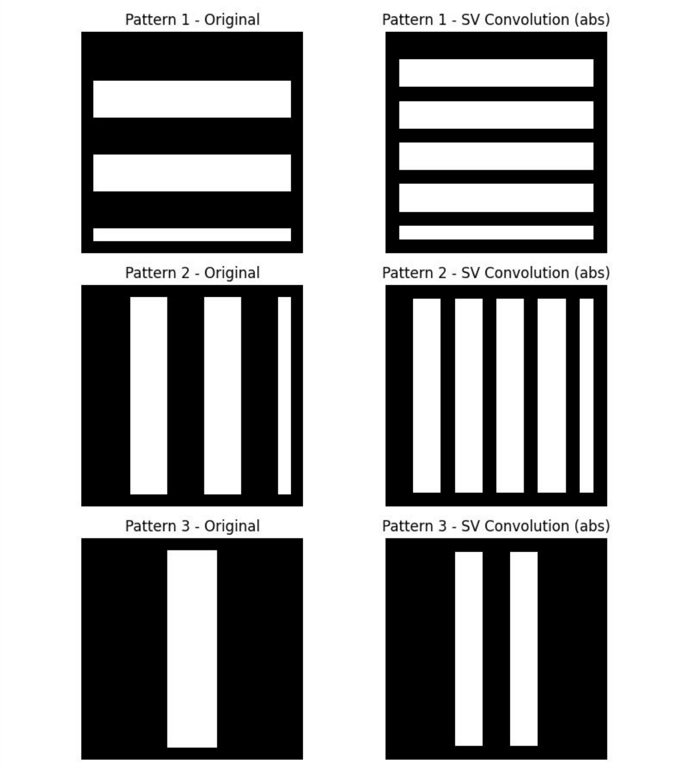
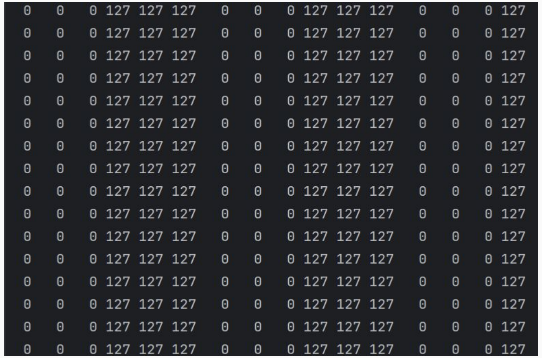
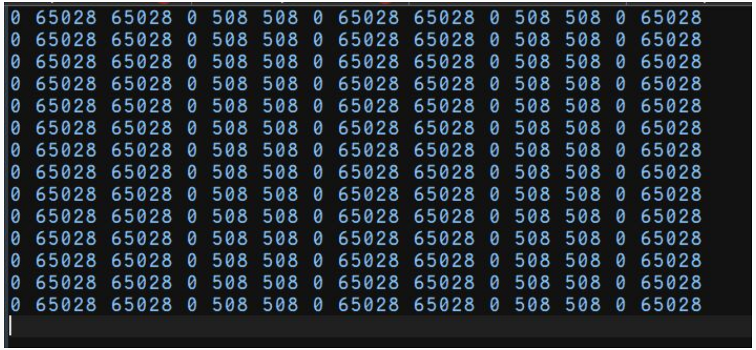


Figure 6: Pattern Comparison



```
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
0   0   0 127 127 127   0   0   0 127 127 127   0   0   0 127
```

Figure 7: Pattern 2 Input Data



```
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
0 65028 65028 0 508 508 0 65028 65028 0 508 508 0 65028
```

Figure 8: Pattern 2 SV Output Data

```
[Compare] Pattern 1
- Max Abs Error     : 0
- Mean Abs Error    : 0.00
- Match (zero diff) : 100.00%

[Compare] Pattern 2
- Max Abs Error     : 0
- Mean Abs Error    : 0.00
- Match (zero diff) : 100.00%

[Compare] Pattern 3
- Max Abs Error     : 0
- Mean Abs Error    : 0.00
- Match (zero diff) : 100.00%
```

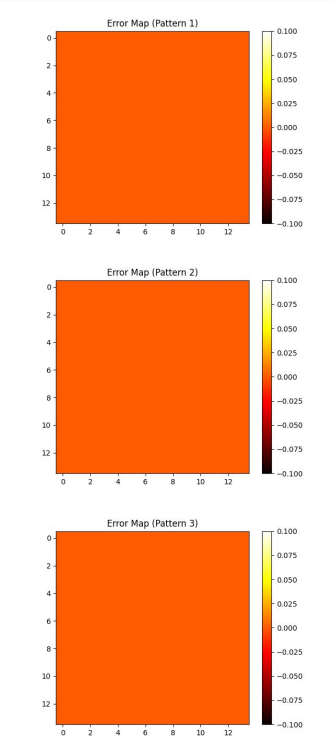


Figure 9: Verification Results

On board test

```
if (chipselect && read && !read_prev) begin
    case (addr)
        4'h3: begin
            output_sw_en <= 0;
            data_out <= done_1;
        end
        4'h4: begin
            output_sw_en <= 1;
            data_out <= output_data;
        end
        default: begin
            data_out <= 0;
            output_sw_en <= 0;
        end
    endcase
end
else begin
    output_sw_en <= 0;
end
```

IO read is a big problem, since after we tested, the read signal will last for more than 1 cycle.
So, we need to latch the counter even when the read is high.

```
printf("Reading output_data:\n");
for (i = 0; i < 14*14; i++) {
    output_data[i] = read_output_data();
    printf("  output_data[%d] = %d\n", i, output_data[i]);

    // Verify output_data
    if ((abs(output_data[i] * 512 - golden_data[i]) / golden_data[i]) > 0.5) {
        fprintf(stderr, "Error: output_data[%d] = %d (expected %d)\n",
                i, output_data[i], golden_data[i]);
    }
}
```

For the golden model, since the quantize scaling happens all over the computation, we need to scale back to do the comparison.