# ECEN 5263 Final Report

Name: Xinyi Zhang

**Outline**:

1) Design architecture introduction
2) Code for modules and testbench
3) Simulation result
4) Synthesis result

## Design architecture introduction

As the number of multiplier is limited as 2, so the CNN architecture must be a pipeline architecture. I used the architecture which is shown in Fig.1.

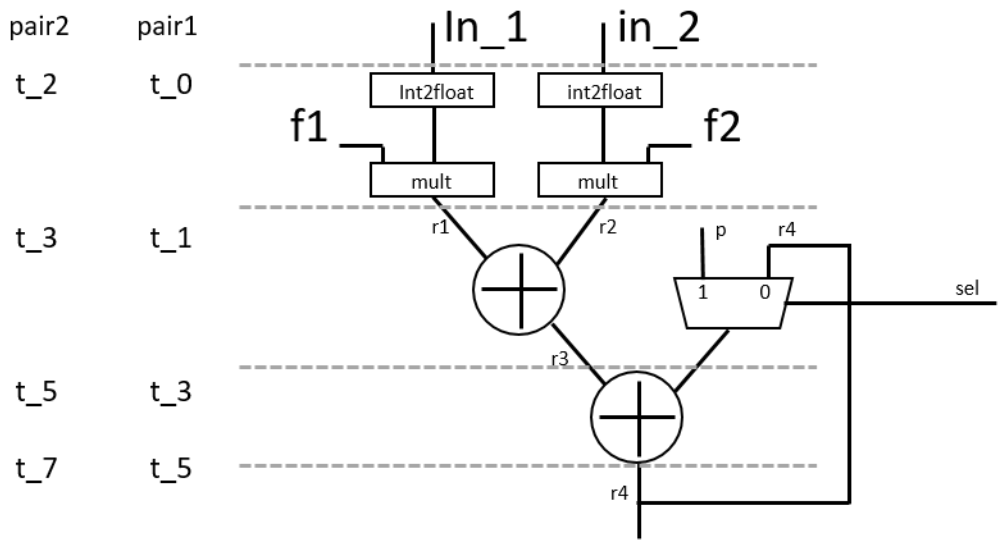



Fig. 1

During simulation, the int2float and mult modules have a total delay of 1 clock cycle; the adders have delay of 2 clock cycles. 8-bit integer number is in in_1 and in_2; weight number is in f1 and f2.

1) Inputs are 4 8-bit integer numbers and their weight every execution.
2) Working in pipeline, 4 inputs are divided into two pairs.
3) Two Integer number in each pair are converted to float16 first and then multiplied with its weight.
4) At t_0, the first pair is sent in; at t_2, the second pair is sent in (interval is 2 unit).
5) At t_1, r1 and r2 gets value; at t_3, r3 gets value.
6) Then r3 adds with the bias for CNN (bias is at port p).
7) At t_5, r4 gets value and goes back to the bottom adder.
8) R4 add with the newly come r3 at t_5 and then gets the final result at t_7.

As the adder's latency is two clock cycle and it's pipelined, the second pair of integer numbers must be sent in with a 2 clock cycles delay after the first pair. Only in this way, it can get correct result. The

selection node of MUX is connected to another clk (clk2 in the code). When the r3 from the first integer pair is available, it selects p (bias). When the r3 from the second integer pair is available, it selects r4. Thus, this architecture will get two outputs for every four inputs. The bias is added to the first pair and get r4. The second pair adds to r4. Then, the output is the final output.

### Code for modules and testbench

Top level:

```
module manage3(f1,f2,p,in_1,in_2,out,clk,clk2);
input [7:0]in_1, in_2;
input clk;
input clk2;
input [15:0]p;  //bias
input [15:0]f1,f2; //weight for every two numbers
output [15:0]out;
wire [15:0]a,b;
wire [15:0]r1,r2,r3,r4,r5;
convert inst_1(.float(a), .int8(in_1), .clk(clk));
convert inst_2(.float(b), .int8(in_2), .clk(clk));
mult inst_3(.B(f1), .C(r1), .A(a), .clk(clk));
mult inst_4(.B(f2), .C(r2), .A(b), .clk(clk));
main_adder inst_5 (.AA(r1), .BB(r2),.clk(clk), .R_add(r3));
mux inst_6(.in1(r4), .in2(p), .sel(clk2), .out(r5));
main_adder inst_7 (.AA(r3), .BB(r5),.clk(clk), .R_add(r4));
assign out=r4;
endmodule
```

The code for submodules are place in folder “code”

Testbench:

```
// `timesccale 1 ns/ 1 ps
module testbench51();
reg      clk;
reg      clk2;
reg    [15:0]f1,f2,p;
reg  [7:0]in_1;
reg  [7:0]in_2;
wire  [15:0]out;
reg [16:0] num; //num and num2 are used for shifting
reg [16:0] num2;
//reg  error;
reg [7:0] tv[100000000:0]; //define memory, input is 256*256
integer i,j, k;

manage3 i1 (

              .in_1(in_1),
              .in_2(in_2),
       .out(out),
        .clk(clk),
        .clk2(clk2),
.f1(f1),
.f2(f2),
.p(p)
              );

parameter weight1=16'b0011110000000000;
parameter weight2=16'b0011110000000000;
parameter weight3=16'b0011110000000000;
parameter weight4=16'b0011110000000000;
parameter bias=16'b0011110000000000;
```

```
//set clk for checking, T=20ps
always
  begin
  clk=1; #50; clk=0; #50; //set T=100ps
    end

initial
begin
  $readmemb("cnn.tv", tv); //matlab convert the matrix row by row, and then put all 8-bit inputs in one column
  #1;
  for (i=0; i< 254; i=i+1) //254=256-2, 256 is the image rows, determine which row
    for (j=0; j< 254; j=j+1)//254=256-2, 256 is the image colums, determine which column
  begin
  num=i*256; num2=(i+1)*256;
  in_1=tv[num+j];
  in_2=tv[num+j+1];
  f1=weight1;
  f2=weight2;
  p=bias;
  #200;   //200 is twice as 100, the speed of send input is 0.5x of main clk
  in_1=tv[num2+j];
  in_2=tv[num2+j+1];
  f1=weight3;
  f2=weight4;
  p=bias;
  #200;
  end
  end

initial
begin
#700;  //cannot get the forst result until 700ps
 for (k=0; k< 1000; k=k+1)
begin
$display("\nat current time", $time);
$display("This is result number %d", {k});
#200;  //add delay to show the output, this will avoid the first intermediate data
$display("%b", {out});
#200;
end
end
endmodule
```

The full function code is attached in folder “code”. Weights and bias can be easily changed in testbench.

As the main clk is 10MHz (100ns), the speed of sending inputs is every 200ns.
Here, I leave the clk2 which is for MUX selection float in testbench. This is because there is an offset before the clk2 starts and it starts with a falling edge. For main clk is 10MHz (T=100ps), I set clk2 ‘ T=400ps, the offset =105ps. Only in this way, the MUX will select correct value between bias and r4.  (I didn’t find a good way to add such offset in testbench, so I add it in waveform window.)  The clock setting is shown in Fig. 2.

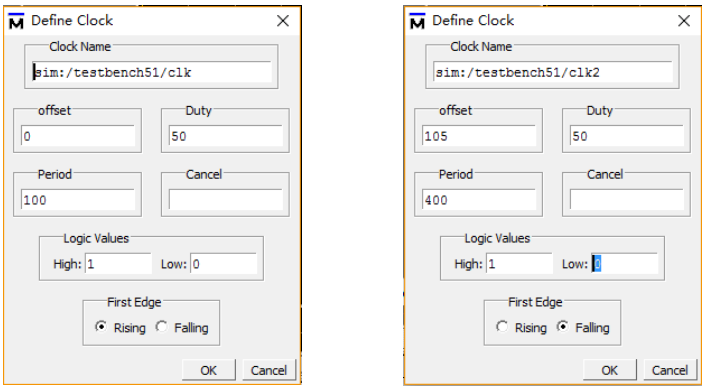



Fig. 2.

## Simulation result

Here is the north-west corner of the converted matrix

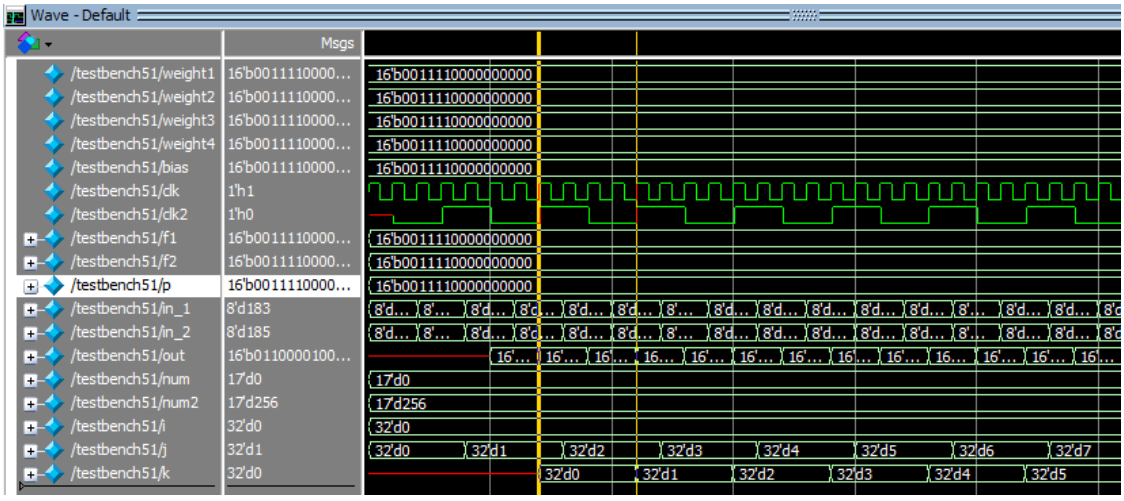
| | | | | | |
|---|---|---|---|---|---|
| 150 | 144 | 155 | 157 | 105 | 164 |
| 189 | 183 | 185 | 184 | 112 | 185 |
| 191 | 184 | 183 | 189 | 115 | 183 |
| 189 | 186 | 183 | 189 | 113 | 182 |
| 186 | 182 | 180 | 189 | 120 | 187 |
| 187 | 185 | 187 | 192 | 107 | 184 |

If we shift 2*2 block from left to right, row by row; with stride=1, weight=1, bias=1, we can get the following results:

667, 668, 682, 559….

Here is the simulation result for my code:

```
# at current time                700
# This is result number          0
# 0110000100110110
#
# at current time                1100
# This is result number          1
# 0110000100111000
#
# at current time                1500
# This is result number          2
# 0110000101010100
#
# at current time                1900
# This is result number          3
# 0110000001011110
#
# at current time                2300
# This is result number          4
# 0110000001101110
#
# at current time                2700
# This is result number          5
# 0110000110000010
```




The value from port “out” will display “intermediate data—result-- intermediate data—result-- intermediate data..”

Therefore, in my testbench, I add extra delay every time I display values for port “out” in transcript. The values in the transcript is the real results.

The result is correct.

For the 256*256 image, there will be 255*255 outputs.

As it counts from 0, he total time should be (255*255-1)*4T+7T=260103T.

In simulation, as the main clk’s period is 100ns, we can observe a total time of 26010300ns.

```
# at current time                 700      # at current time                 26009100
# This is result number           0       # This is result number           65021
# 0110000100110110                        # 0110001001110110
#                                         #
# at current time                 1100     # at current time                 26009500
# This is result number           1       # This is result number           65022
# 0110000100111000                        # 0110001001110110
#                                         #
# at current time                 1500     # at current time                 26009900
# This is result number           2       # This is result number           65023
# 0110000101010100                        # 0110001001110110
#                                         #
# at current time                 1900     # at current time                 26010300
# This is result number           3       # This is result number           65024
# 0110000001011110                        # 0110001001110110
#
```

## Synthesis result

The script used and reports in synthesis are attached in folder “rep”.

**Here is report for my design area:**

ssc_core_slow (File: /home/xinyiz/final/libs/core_slow.db)

```
Number of ports:                  847
Number of nets:                  2695
Number of cells:                 1813
Number of combinational cells:   1635
Number of sequential cells:       165
Number of macros/black boxes:       0
Number of buf/inv:                277
Number of references:               8

Combinational area:       24279.920019
Buf/Inv area:              1819.680014
Noncombinational area:     6447.420162
Macro/Black Box area:         0.000000
Net Interconnect area:    undefined  (Wire load has zero net area)

Total cell area:          30727.340181
```

**Here is report for my worst case path (f=50Mhz):**

```
  inst_4/n221 (net)          2    0.024122          0.000000   19.599531 r
  inst_4/U332/Y (oa1f3)                    0.270284  0.169050   19.768581 f
  inst_4/n222 (net)          1    0.015124          0.000000   19.768581 f
  inst_4/U333/Y (xor3b3)                   0.177798  0.266155   20.034737 f
  inst_4/n225 (net)          1    0.008124          0.000000   20.034737 f
  inst_4/U334/Y (or2c3)                    0.210728  0.097828   20.132565 r
```

```
inst_4/n25 (net)              1     0.007124          0.000000   20.132565 r
inst_4/C_reg[14]/D (fdf1c3)                0.210728  0.000034   20.132599 r
data arrival time                                    20.132599

clock clk (rise edge)                      20.000000  20.000000
clock network delay (propagated)                1.282780   21.282780
clock uncertainty                         -1.000000  20.282780
inst_4/C_reg[14]/CLK (fdf1c3)                   0.000000   20.282780 r
library setup time                        -0.141004  20.141775
data required time                              20.141775
--------------------------------------------------------------------------------
data required time                              20.141775
data arrival time                              -20.132599
--------------------------------------------------------------------------------
slack (MET)                                     0.009176
```

**Here is report for my power:**
I generate the VCD file with a main clk's period of T=100ns.

```
Global Operating Voltage = 1.62
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000pf
    Time Units = 1ns
    Dynamic Power Units = 1mW    (derived from V,C,T units)
    Leakage Power Units = Unitless


  Cell Internal Power  = 0.000000 mW    (0%)
  Net Switching Power  = 67.058174 uW  (100%)
                    ---------
Total Dynamic Power    = 67.058174 uW  (100%)

Cell Leakage Power     =   0.0000
```

Error: Either dynamic power or leakage power, in the library, is unitless. Unable to display complete power group summary. (PWR-799)

| Power Group | Internal Power | Switching Power | Leakage Power | Total Power ( % ) | Attrs |
|---|---|---|---|---|---|
| io_pad | 0.000000 | 0.000000 | 0.000000 | NA ( N/A) | |
| memory | 0.000000 | 0.000000 | 0.000000 | NA ( N/A) | |
| black_box | 0.000000 | 0.000000 | 0.000000 | NA ( N/A) | |
| clock_network | 0.000000 | 0.000000 | 0.000000 | NA ( N/A) | |
| register | 0.000000 | 1.512311e-02 | 0.000000 | NA ( N/A) | |
| sequential | 0.000000 | 1.158593e-03 | 0.000000 | NA ( N/A) | |
| combinational | 0.000000 | 5.077649e-02 | 0.000000 | NA ( N/A) | |
| Total | 0.000000 mW | 6.705819e-02 mW | 0.000000 | NA | |

I learned a lot this semester and I found I get much more familiar with Verilog code.
Thank you Dr. Stine!