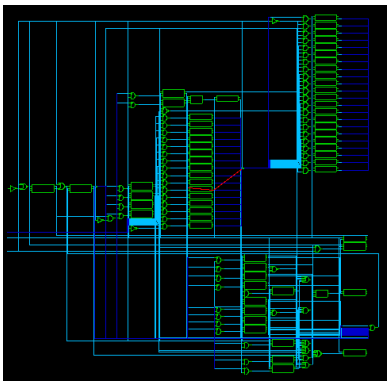
VLSI project_1 2017160111 안진모

(1) Experimental Goal

프로젝트는 기본적으로 Matrix multiplier를 design하는 것이다. 다양한 목적을 가지고 성능과 면적 등을 향상시켰다.

(2) Result

1. schematic.



- 보드에 합성하기 이전에, output 과 input을 정리하는 과정이 필요했다. 이에 내부에 존재하는 controller가 제대로 작동하기 위해서 A,B,done, rstn 등을 input으로, 결과를 output으로 설정했으며, NWRT, NCE, Counter 등 의 clk에 의하여 생성되는 모든 것들을 output으로 설정하였다. (S ram으로 apply되는) 특히 counter 부분을 output이나 input으로 할당하지 않는 경우, 보드에서 소실되므로 output이라 판단하였다.

```
input clk, rstn,start;
output reg done;
wire reset;
assign reset = ~rstn;

output reg  [22-1:0] DATA_in ;
wire [22-1:0]DATA_out;
reg [8-1:0] MEM_A_control,MEM_B_control;
input wire [8-1:0] MEM_A_out,MEM_B_out;
wire [22-1:0] MEM_C_control;
 wire [18:0] counter_in;
output reg [18:0] counter_in_ff;
wire [22-1:0] mat_c;
wire [11:0] counter_out;
output reg sel;
output reg NCE;
output reg NWRT_C;
output reg NWRT;
output reg NCE_C;
wire [11:0] counter_in_A,counter_in_B;
wire NWRT_nxt ,NCE_nxt;
```

< 참고 >

특히 주의를 기울였던 부분은 '최대한 reg를 적게 사용하는 것' 이었고, 목표에 따

라 A,B의 주소를 할당하는 part 등을 최대한 counter_in에서 assign으로 wire에 할당하는 방법을 사용하였다.

```
assign counter_in_A = {counter_in[17:12],counter_in[5:0]};
assign counter_in_B = {counter_in[5:0],counter_in[11:6]};
assign counter_out = counter_in[17:6]-1'b1;
```

(마지막 counter_out에서 1이 빠지는 이유는 MAC unit에서 1 stage를 더 사용하기 때문이다.)

Power consumption, Critical path delay, Area를 살펴보면,

```
Operating Conditions: nom_pvt   Library: lec25dscc25_SS
Wire Load Model Mode: top


Global Operating Voltage = 2.25
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 = 1pW


  Cell Internal Power  =   4.7267 mW   (73%)
  Net Switching Power  =   1.7742 mW   (27%)
                         ---------
Total Dynamic Power    =   6.5009 mW  (100%)

Cell Leakage Power     = 350.4072 uW
```

```
****************************************
Report : area
Design : Top_controller
Version: Z-2007.03-SP4
Date   : Fri May 12 02:45:05 2023
****************************************

Library(s) Used:

    lec25dscc25_SS (File: /home/admin/lib/lec25/lec25dscc25_SS.db)

Number of ports:              66
Number of nets:              218
Number of cells:             142
Number of references:         17

Combinational area:      41007.522861
Noncombinational area:   14730.865860
Net Interconnect area:     undefined  (No wire load specified)

Total cell area:         55738.390625
Total area:                undefined
1
design_vision-xg-t> report_power
```

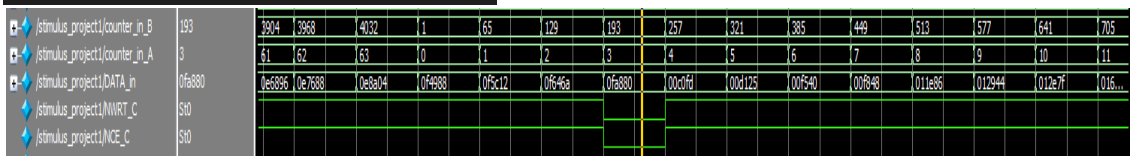
```
Startpoint: MEM_B_control_reg[2]
            (rising edge-triggered flip-flop clocked by clk)
Endpoint: ma1/add_ad_c_ff_reg[13]
            (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: max

Point                                                                  Incr       Path
-----------------------------------------------------------------------------------------
clock clk (rise edge)                                                  1.00       1.00
clock network delay (ideal)                                            0.00       1.00
MEM_B_control_reg[2]/CLK (dffs2)                                       0.00       1.00 r
MEM_B_control_reg[2]/Q (dffs2)                                         0.35       1.35 f
ma1/input_b[2] (matrix_multiplication_controller)                      0.00       1.35 f
ma1/mult_168/b[2] (matrix_multiplication_controller_DW_mult_uns_0)
                                                                       0.00       1.35 f
ma1/mult_168/U392/Q (nnd2s2)                                           0.17       1.53 r
ma1/mult_168/U313/Q (ib1s2)                                            0.10       1.62 f
ma1/mult_168/U505/Q (xor2s1)                                           0.41       2.03 r
ma1/mult_168/U366/Q (nor2s1)                                           0.19       2.23 f
ma1/mult_168/U354/Q (oai21s2)                                          0.22       2.44 r
ma1/mult_168/U353/Q (ib1s1)                                            0.18       2.62 f
ma1/mult_168/U492/Q (oai21s3)                                          0.31       2.94 r
ma1/mult_168/U395/Q (aoi21s3)                                          0.25       3.19 f
ma1/mult_168/U431/Q (oai21s2)                                          0.26       3.45 r
ma1/mult_168/U251/Q (aoi21s3)                                          0.24       3.68 f
ma1/mult_168/U565/Q (oai21s2)                                          0.29       3.98 r
ma1/mult_168/U270/Q (i1s2)                                             0.10       4.08 f
ma1/mult_168/U568/Q (oai21s2)                                          0.21       4.29 r
ma1/mult_168/U496/Q (xnr2s2)                                           0.37       4.66 f
ma1/mult_168/product[9] (matrix_multiplication_controller_DW_mult_uns_0)
                                                                       0.00       4.66 f
ma1/add_169/A[9] (matrix_multiplication_controller_DW01_add_0)
                                                                       0.00       4.66 f
ma1/add_169/U177/Q (nor2s1)                                            0.22       4.88 r
ma1/add_169/U298/Q (nor2s2)                                            0.22       5.10 f
ma1/add_169/U276/Q (aoi21s3)                                           0.27       5.36 r
ma1/add_169/U167/Q (ib1s1)                                             0.23       5.59 f
ma1/add_169/U186/Q (aoi21s2)                                           0.21       5.81 r
ma1/add_169/U242/Q (xnr2s1)                                            0.39       6.19 f
ma1/add_169/SUM[13] (matrix_multiplication_controller_DW01_add_0)
                                                                       0.00       6.19 f
ma1/add_ad_c_ff_reg[13]/CLRB (dffcs1)                                  0.00       6.19 f
data arrival time                                                                 6.19

clock clk (rise edge)                                                  6.70       6.70
clock network delay (ideal)                                            0.00       6.70
ma1/add_ad_c_ff_reg[13]/CLK (dffcs1)                                   0.00       6.70 r
library setup time                                                    -0.51       6.19
data required time                                                                6.19
-----------------------------------------------------------------------------------------
data required time                                                                6.19
data arrival time                                                                -6.19
-----------------------------------------------------------------------------------------
slack (MET)                                                                       0.00
```

위와 같다. 이를 표로 정리하면,

	MAT_multiplier
Min period [ns]	5.7
Clock speed [Hz]	1.754*10^8
Data arrival time [ns]	6.19
Area [μm^2]	55738
Total power [mW]	6.50

모든 정보들을 정리하면 위와 같다. 실제로, 모든 주소를 reg 로 세팅하였을 때, clock speed 는 차이가 무의미하지만, 92806 μm^2와 같았는데, 총 면적이 40.0%가 감소하는 것을 알 수 있었다. 사실 해당 Synthesis 에서는 차이가 나지 않지만, NCE 를 계속 0 으로 두는 경우, Energy efficiency 면에서 큰 차이를 갖는 다는 사실을 쉽게 유추할 수 있다. 따라서 해당 부분 역시 Memory 가 write 되는 경우에만 NCE_C 를 0 으로 둠으로, 다양한 부분을 체크 하기위해 노력하였다.

```
always@(posedge clk)begin
    if(~rstn|start) begin
        NCE_C<=1;
        NWRT_C<=1;
    end
    else begin
        if (sel & counter_in != 19'b00000000000000000010)begin
        NCE_C<=0;
        NWRT_C<=0;
        end
        else begin
        NCE_C<=1;
        NWRT_C<=1;
        end
    end
end
```

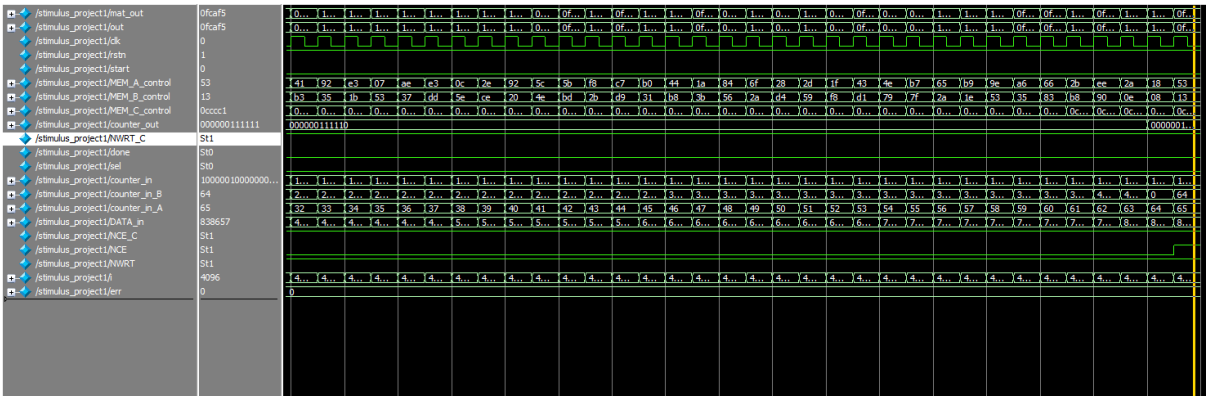


<참고>

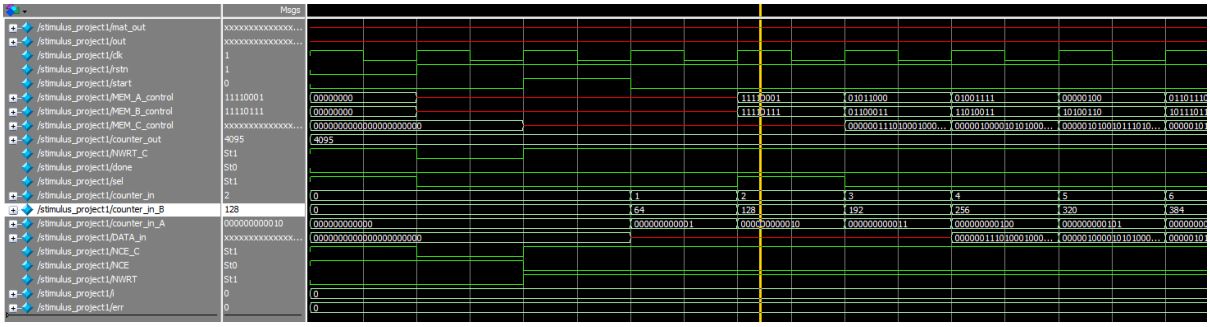
또한 모든 test 가 진행된 후, NCE_C,NCE 를 1 로 만들어 줌으로서, 에너지 낭비를 줄일 수 있었다. (테스트 종료 후 메모리를 끄는 과정을 synthesis 에서는 제거하였음)

```
        always @(posedge clk) begin
        if(counter_in == 19'b10000010000000000000) begin
        NCE <= 1'b1;
        NCE_C <= 1'b1;
        end
        end
```

이제 Timing Diagram 을 보자. 위와 같이 메모리에 저장하는 과정은 볼 수 있었다.



먼저 마지막 part 로 Error 가 0 인 모습을 볼 수 있다.

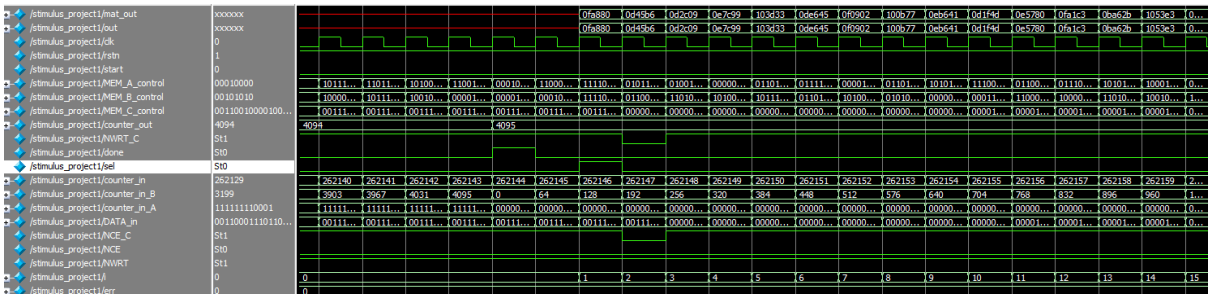


초기 상태이다. 잘 연결이 되고 있음을 알 수 있다.

이 때, 3cycle 이 밀려서 나오는 이유는 메모리에 데이터를 저장, 꺼낸 후, reg 에 넣은 후 computation 을 진행하기 때문이다.

위에 메모리 C 에 저장하는 과정을 설명이 되어있으므로, 생략한다.

마지막으로 메모리에서 값이 나오기 시작하는 과정을 보면,



제대로 된 clock 에서 error 가 0 가 되는 모습을 볼 수 있었다.

이는 회로의 본질적으로 중요한 점인 Verification을 이다. Test bench는 내가 verification하는 design의 input을 만들어 주어야 하며, Reference model과 동일하게 operation하는지 output을 check하는 부분으로 이루어져 있다. Operating 시나리오를 기반으로 test case의 constraint를 만들고 그 안에서 random한 input을 apply한다. 이를 마지막 score board에서 Wave form으로 check하며, RTL algorithm이 동일하게 구현되었는지 error를 check한다 IP의 모든 부분이 connected 되었는지 check하기 위하여 coverage를 check하는 것도 필요하다. 이제 Verification 결과를 간단하게 확인하자. 우리는 Matlab을 통하여 input과 해당 value 간의 convoltion을 이용한 output 결과를 만들었다. 즉, 앞서 언급한 operating 시나리오를 위한 constraint를 matlab을 통하여 txt로 만든 것이다. 이를 testbench file에 apply하고, 앞에서 본 결과를 살펴보면 먼저 a와 b의 convolution이 잘 되었는지 확인하기 위하여 시나리오 상의 output constraint와 실제 회로의 output과 비교를 진행했고, 이후 해당 값과 오차를 error로 출력하였다. 앞의 Testbench의 결과로 clk를 apply한 것의 유무에 관계없이, error가 0을 유지하는 것을 볼 수 있다.

마지막으로 Critical path를 보자. 사실 memory를 control 하는 부분을 떼어놓으니, 결국 MAC part만 Synthesis가 된다. 이 때, 나오는 delay는 실제 Total convolution연산의 delay가 아니라 1회의 MAC에 대한 delay가 나온다. 즉, 모든 연산을 하게 될 경우 주어진 Delay에 64*64배가 걸린다는 사실이다. 먼저 MAC연산의 code를 확인하면,

```
module matrix_controller (
    in_a, in_b, output_c, sel, clk, reset
);
input                       clk, reset;
input                       sel;
input [8-1:0]               in_a, in_b;
output wire [22-1:0]              output_c;

wire [16-1:0]               mul;    // A * B
wire [22-1:0]               add_ab_c;  // A * B + C
reg  [22-1:0]               add_ad_c_ff;
wire [22-1:0]                muxout;

assign mul                = in_a * in_b;
assign add_ab_c             = mul + muxout;
assign output_c             = add_ad_c_ff;
assign muxout = sel ? 0 : add_ad_c_ff;

always @(posedge clk) begin
    if (reset) begin
        add_ad_c_ff         <= 0;
    end
    else begin
        add_ad_c_ff         <= add_ab_c;
    end
end

endmodule
```

이와 같다. 이 때, muxout은 기존의 값을 쓰거나 혹은 sel이 1 즉, C

주소에 할당한 이후 reset되는 과정을 담당하는 것이다. Critical path는 간단하게, multiplier 이후 adder가 되는 부분으로 쉽게 알 수 있었다.

(3) Reference

Verilog HDL -joseph Cavanagh

Verilog HDL 이론 -한양대학교 전자전기공학부

Quora- CPU clock speed vs power vs area

ResearchGate – Circuit Area vs frequency

https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=laonple&logNo=220926179193

고려대학교 전자전기공학부 – VLSI design project