AI디지털집적회로

Assignment #1

전기전자공학부

202110410

조민우

서론

1학기 AI가속기설계과목의 과제를 수행하며 Verilog 코드를 작성하고 simulation하며, 코드의 논리적인 동작을 검증하는 과정을 거쳐왔다. 2학기 과목인 AI디지털집적회로 과목은 behavior simulation을 넘어 logic synthesis 과정을 실습하며 RTL-to-Gate 합성 과정이 어떻게 이루어지는지 체험하고, TSMC의 License를 받아 실제 산업계에서 사용되는 EDA Tool의 작동 원리와 사용법을 습득하고자 한다.

이번 과제1의 목표는 다양한 Clk Frequency에서 반복적으로 RCA-Based 8 bit multiplier 합성을 수행하는 것이다. 구체적으로는 200MHz~3GHz까지 200MHz 단위로 Clk Frequency를 증가시키면서 Design Compiler를 이용해 합성을 진행해야 한다. 각 합성 과정을 거치고, report를 분석하여 timing constraint를 violate하지 않고 동작할 수 있는 maximum Frequency를 구해야한다. 또한, Frequency에 따른 area-power graph를 시각화하여 각 frequency에 따른 trade-off를 비교 및 평가해보려 한다.

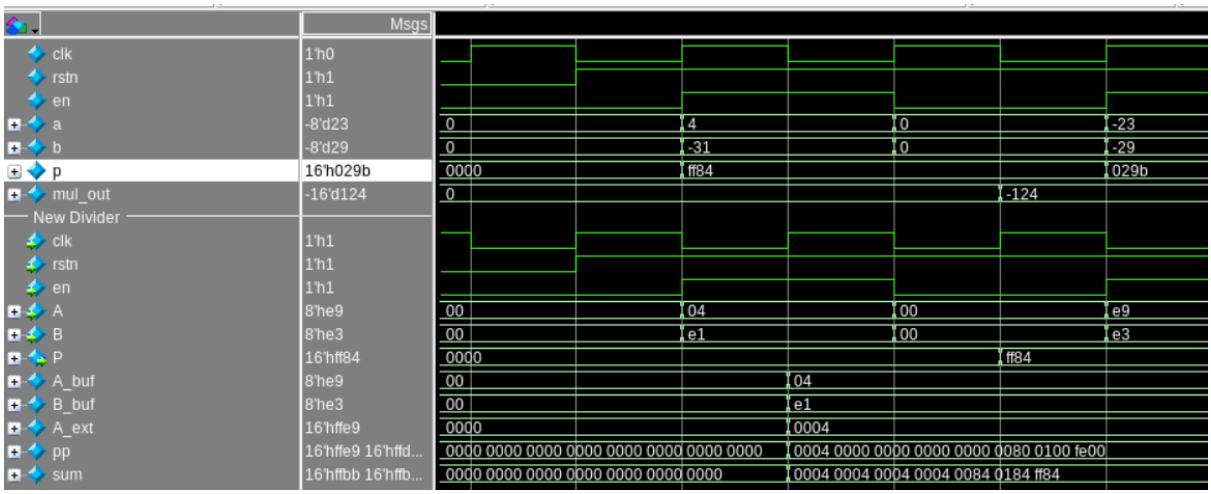
본론

1. Behavior simulation

```
wire [15:0] A_ext;
wire [15:0] pp [0:7];
    assign A_ext = {{8{A_buf[7]}}, A_buf};

    genvar i;
    for (i = 0; i < 7; i = i + 1) begin
            assign pp[i] = B_buf[i] ? (A_ext << i) : 16'd0;
    end
    assign pp[7] = B_buf[7] ? -(A_ext << 7) : 16'd0;

wire [15:0] sum [0:6];
generate
            for (i = 0; i < 7; i = i + 1) begin
                    rca #(16) U_rca ( .A(i == 0 ? pp[i] : sum[i-1]), .B(pp[i+1]), .S(sum[i]), .Cout() );
            end
    endgenerate

    always @ (posedge clk) begin
            P <= sum[6];
    end
endmodule
```

<Fig 1.1 mult.v>

rtl directory에 들어가보면 mult.v라는 파일이 존재한다. 우리의 곱셈기 mult8의 핵심 로직은 7개의 Ripple Carry Adder를 이용하여 8개의 pp(partial product)을 순차적으로 더하고 최종 합(sum[6])을 output P에 할당한다.



<Fig 1.2 waveform for run.sh>

Vivado에서 behavior simulation을 이용해 보았던 것과 같이 waveform을 분석할 수 있다. 간단하게 첫번째 연산 과정만 살펴보자. Clk이 high가 되면 (0x04) x (0xe1)의 pp들이 연산되고 pp들을 Ripple Carry Adder로 합하면 ff84라는 결과가 나오는 것을 볼 수 있다.

2. Synthesis(Design Compiler)

```
[team_342106@isa1 dc]$ ls
alib-52        mult8_0.2GHz  mult8_1.2GHz  mult8_2.2GHz  namingrules.tcl  WORK
clean.sh       mult8_0.4GHz  mult8_1.4GHz  mult8_2.4GHz  run.sh
command.log    mult8_0.6GHz  mult8_1.6GHz  mult8_2.6GHz  synth.log
dc_setup.tcl   mult8_0.8GHz  mult8_1.8GHz  mult8_2.8GHz  synth.tcl
mult8          mult8_1.0GHz  mult8_2.0GHz  mult8_3.0GHz  timing.tcl
```

<Fig 2.1 dc directory file tree>

dc directory에 들어가보면 교수님께서 미리 다양한 shell script를 미리 설정해두어 우리는 단순히 timing.tcl의 clk_period를 조정하고 run.sh script를 실행하면 된다.

```
# Variables for clock
set clk_period 5
set clk_uncertainty [expr $clk_period/20]
set clk_transition [expr $clk_period/80]
```

<Fig 2.2 Clk_freq : 200MHz>

처음에는 clk_period가 5ps로 설정하고, clk_freq = 200MHz인 상태로 synthesis를 진행하였다.

```
Point                              Fanout     Trans      Incr       Path       Attributes
-------------------------------------------------------------------------------------------
----
clock clk (rise edge)                                    0.00       0.00
clock network delay (ideal)                              0.00       0.00
input external delay                                     2.00       2.00 r
rstn (in)                                     0.01       0.01       2.01 r
rstn (net)                          2                    0.00       2.01 r
U110/Z (AN2D0BWP30P140)                       0.06       0.05       2.06 r
n51 (net)                          16                    0.00       2.06 r
A_buf_reg_0_/CN (DFKCNQD1BWP30P140)           0.06       0.00       2.06 r
data arrival time                                                   2.06

clock clk (rise edge)                                    5.00       5.00
clock network delay (ideal)                              0.00       5.00
A_buf_reg_0_/CP (DFKCNQD1BWP30P140)                      0.00       5.00 r
library setup time                                      -0.02       4.98
data required time                                                  4.98
-------------------------------------------------------------------------------------------
----
data required time                                                  4.98
data arrival time                                                  -2.06
-------------------------------------------------------------------------------------------
----
slack (MET)                                                         2.92
```

<Fig 2.3 200MHz timing.rpt>

timing report를 열어 slack을 확인해보자. slack은 (required time – arrival time)으로 구해지며, 이 slack 값이 양수여야 logic이 정상 동작한다. 주파수가 높아질수록 clk period가 짧아질 테니 이 slack도 점차 감소할 것이다. 이 slack이 음수가 된다면 timing을 위반한 것으로 주파수를 낮추거나 critical path의 standard cell의 크기를 조절하는 등의 조치를 취해야 한다.

```
Number of ports:                          935
Number of nets:                          1107
Number of cells:                          364
Number of combinational cells:            211
Number of sequential cells:                33
Number of macros/black boxes:               0
Number of buf/inv:                         44
Number of references:                      20

Combinational area:                215.711997
Buf/Inv area:                       14.616000
Noncombinational area:              66.402001
Macro/Black Box area:                0.000000
Net Interconnect area:        undefined  (Wire load has zero net area

Total cell area:                   282.113998
Total area:                   undefined
```

<Fig 2.4 200MHz area.rpt>

이번에는 area를 확인해보았다. 더욱더 높은 Frequency로 올리면 더 빠른 switching을 위해 더욱 넓은 width를 가진 큰 transistor(library에 존재하는)로 합성이 될 것이고, 이에 따라 Area도 커질 것으로 예상된다.

```
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 = 1nW


--------------------------------------------------------------------------------
                                   Switch    Int      Leak     Total
Hierarchy                          Power     Power    Power    Power    %
--------------------------------------------------------------------------------
mult8                              3.00e-03 2.20e-02 1.09e+03 2.61e-02 100.0
```

<Fig 2.5 200MHz power.rpt>

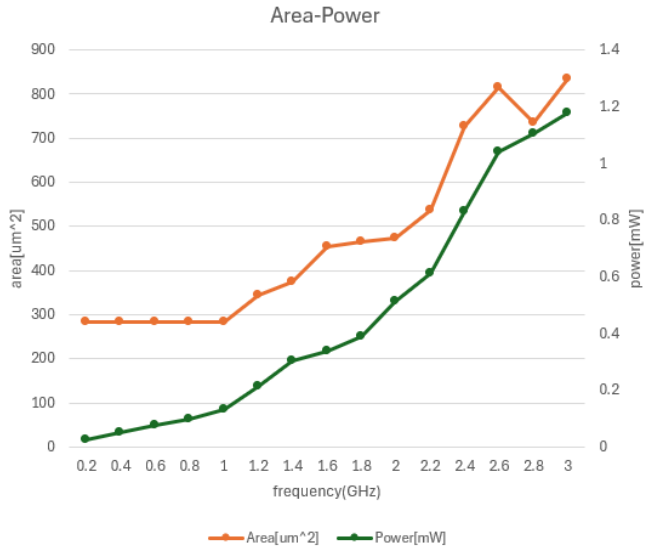
이번에는 power를 확인해보았다. 더욱더 높은 Frequency로 올리면 더 빠른 연산을 위해 design에 사용되는 transistor들이 고성능화 될 것이고, 결국 power역시 올라갈 것으로 보인다.

이와 같은 방식으로 freq를 200MHz~3GHz까지 조정하며 각 report들을 확인하였다.

3.1 Results

frequency[GHz]	Slack[ns]	Timing Met	Area[um^2]	Power[mW]
0.2	2.92	O	282.114	0.0261
0.4	1.05	O	282.114	0.0503
0.6	0.55	O	282.114	0.0751
0.8	0.16	O	282.114	0.1
1	0	O	283.878	0.131
1.2	0	O	344.358	0.213
1.4	0	O	373.716	0.304
1.6	0	O	454.986	0.339
1.8	0	O	466.074	0.391
2	0	O	473.886	0.512
2.2	0	O	535.752	0.612
2.4	0	O	726.39	0.832
2.6	-0.03	X	815.976	1.042
2.8	-0.04	X	734.58	1.105
3	-0.06	X	833.364	1.178

<Fig 3.1.1 Frequency Variation Analysis>



<Fig 3.1.2 Frequency Variation Graph(Area/Power)>

3.2 Results Analysis

Fig 3.1.1를 보면 timing violation이 발생하지 않는 **maximum frequency**는 **2.4GHz**임을 확인할 수 있다.

앞선 합성 과정속에서 예측한 바와 같이 Frequency가 올라갈수록 Area와 Power는 다음과 같은 변화를 보였다.

1. Power는 대체로 주파수에 비례하여 증가하는 모습을 보였다.

2. Area는 0.2GHz~1GHz까지는 증가하지 않다가 2.4GHz까지 증가하고, 2.8GHz에서 한 번 감소하였다가 다시 3GHz에서 증가하였다.

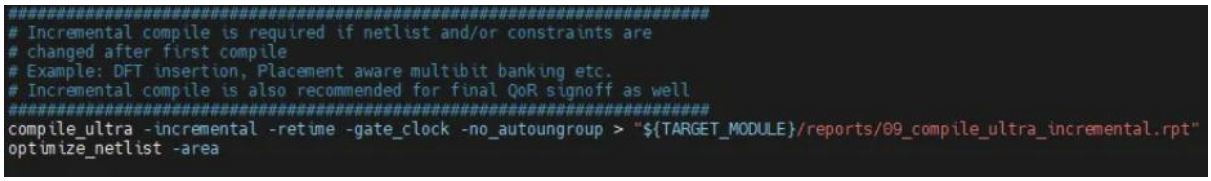
일반적으로 주파수가 올라가면 타이밍을 맞추기 위해 더 빠른 셀들을 사용해야 하고, 빠른 셀들은 보통 더 크기 때문에 Area가 증가해야 한다. 그런데 여기서는 2.8GHz에서 Area가 줄어들었다. 해당 원인을 분석하기 위해 2.6GHz, 2.8GHz, 3.0GHz에서의 Area report를 비교해보았다.

Freq: 2.6GHz	**Freq: 2.8GHz**	**Freq: 3.0GHz**
Number of ports: 967 Number of nets: 1553 Number of cells: 849 Number of combinational cells: 642 Number of sequential cells: 86 Number of macros/black boxes: 0 Number of buf/inv: 135 Number of references: 51 Combinational area: 636.678000 Buf/Inv area: 48.510000 Noncombinational area: 179.297995 Macro/Black Box area: 0.000000 Net Interconnect area: undefined (Wire l Total cell area: 815.975995	Number of ports: 979 Number of nets: 1531 Number of cells: 821 Number of combinational cells: 614 Number of sequential cells: 86 Number of macros/black boxes: 0 Number of buf/inv: 119 Number of references: 48 Combinational area: 553.140001 Buf/Inv area: 43.722000 Noncombinational area: 181.439994 Macro/Black Box area: 0.000000 Net Interconnect area: undefined (Wire l Total cell area: 734.579996 Total area: undefined	Number of ports: 974 Number of nets: 1517 Number of cells: 810 Number of combinational cells: 608 Number of sequential cells: 81 Number of macros/black boxes: 0 Number of buf/inv: 116 Number of references: 54 Combinational area: 658.727998 Buf/Inv area: 45.486000 Noncombinational area: 174.635994 Macro/Black Box area: 0.000000 Net Interconnect area: undefined (Wire l Total cell area: 833.363993

<Fig 3.2.1 Area Report[2.6GHz/2.8GHz/3.0GHz]>

Fig 3.2.1을 보면 2.6GHz에 비해 2.8GHz의 Area report에서 Combiantional Area가 13% 가량 감소(636.68->553.14)한 것을 확인할 수 있었다.

그 이유를 찾아보니 2.6GHz까지는 Design Compiler의 최적화 엔진이 timing을 맞추기 위해 더 빠른 셀과 복잡한 logic 구조를 사용하였지만 2.8GHz부터는 timing 맞추는 것을 포기하고 area 최적화에 집중하여 복잡한 logic 구조를 제거한 것으로 추측된다.



<Fig 3.2.2 synth.tcl 일부>

이에 대한 근거를 찾기위해 synth.tcl 코드를 읽어보니 optimize_netlist가 -area로 설정되어 있었음을 확인하였다.

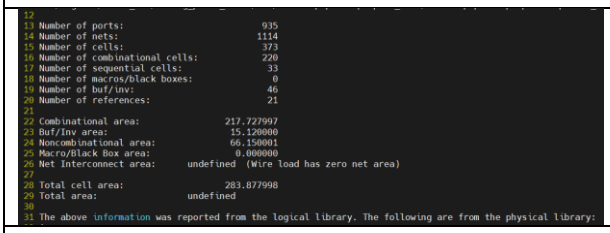
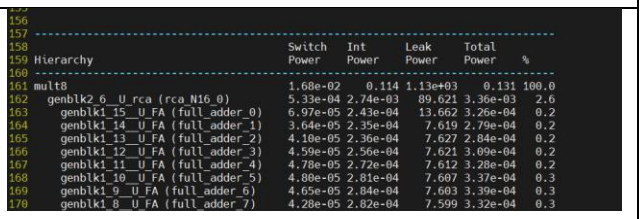
Optimal frequency는 power, performance, area 중 어느 환경이냐에 따라 상이하게 optimal frequency가 존재하겠지만, 여기서는 ppa 모두의 성능 고르게 반영하는 frequency를 찾아보려고한다.

Freq[GHz]	Freq	Area	Power	F/(A+P)	F/A + F/P	F/(A*P)	F^2/(A*P)
0.2	1.00	1.00	1.00	0.50	2.00	1.00	1.00
0.4	1.09	1.00	1.03	0.54	2.15	1.06	1.16
0.6	1.18	1.00	1.06	0.57	2.30	1.11	1.32
0.8	1.27	1.00	1.09	0.61	2.44	1.17	1.48
1	1.36	1.00	1.13	**0.64**	**2.56**	**1.20**	**1.64**
1.2	1.45	1.14	1.23	0.61	2.46	1.04	1.51
1.4	1.55	1.21	1.34	0.61	2.43	0.95	1.47
1.6	1.64	1.39	1.39	0.59	2.36	0.85	1.39
1.8	1.73	1.41	1.45	0.60	2.41	0.84	1.45
2	1.82	1.43	1.60	0.60	2.40	0.79	1.44
2.2	1.91	1.57	1.73	0.58	2.32	0.70	1.34
2.4	2.00	2.00	2.00	0.50	2.00	0.50	1.00

<Fig 3.2.2 Optimal Frequency Assessment>

Power, Performance, Area 모든 특성의 변화를 동일하게 반영하기위해 valid한 영역(0.2GHz~2.4GHz)내에서 min-max normalization을 진행하고 계산을 위하여 +1의 offset을 하여 data-preprocessing을 진행했다. 정규화된 값들을 바탕으로 총 4번의 서로 다른 efficiency 계산을 진행하였다. 그리고, 이 4번의 평가 방식에서 4번 모두 가장 우수한 score를 받은 주파수인 **1GHz**가 **optimal frequency**이다.

Min-max normalization을 선택한 이유는 power/performance/area가 서로 비교할 수 없는 단위들을 가지고 trade-off 관계를 형성하기에 동일한 scale로 가공할 필요가 있다고 생각했기 때문이다.

Area	Power
	
282.878[um^2]	0.131[mW]

```
Number of ports:                          935
Number of nets:                          1114
Number of cells:                          373
Number of combinational cells:            220
Number of sequential cells:                33
Number of macros/black boxes:               0
Number of buf/inv:                         46
Number of references:                      21

Combinational area:             217.727997
Buf/Inv area:                    15.120000
Noncombinational area:           66.150001
Macro/Black Box area:             0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                283.877998
Total area:                 undefined

The above information was reported from the logical library. The following are from the physical library:
```

```
                                   Switch   Int      Leak     Total
Hierarchy                          Power    Power    Power    Power    %
-------------------------------------------------------------------------------
mult8                              1.68e-02   0.114 1.13e+03    0.131 100.0
  genblk2_6__U_rca (rca_N16_0)     5.33e-04 2.74e-03   89.621 3.36e-03   2.6
    genblk1_15__U_FA (full_adder_0)  6.97e-05 2.43e-04   13.662 3.26e-04   0.2
    genblk1_14__U_FA (full_adder_1)  3.64e-05 2.35e-04    7.619 2.79e-04   0.2
    genblk1_13__U_FA (full_adder_2)  4.10e-05 2.36e-04    7.627 2.84e-04   0.2
    genblk1_12__U_FA (full_adder_3)  4.59e-05 2.56e-04    7.621 3.09e-04   0.2
    genblk1_11__U_FA (full_adder_4)  4.78e-05 2.72e-04    7.612 3.28e-04   0.2
    genblk1_10__U_FA (full_adder_5)  4.80e-05 2.81e-04    7.607 3.37e-04   0.3
    genblk1_9__U_FA (full_adder_6)   4.65e-05 2.84e-04    7.603 3.39e-04   0.3
    genblk1_8__U_FA (full_adder_7)   4.28e-05 2.82e-04    7.599 3.32e-04   0.3
```

<Fig 3.2.3 Optimal Frequency[1GHz] Area/Power Report>

결론

1. Result Summary

Maximum frequency without violation : 2.4GHz

Optimal Frequency : 1GHz

2. Additional discussion for Maximum Frequency

Maximum frequency를 높여 성능을 올림과 동시에 회로가 정상적으로 동작하려면 timing violation이 있어서는 안된다. 우리의 설계에서는 2.6GHz부터 Slack이 음수로 나타나며 timing constraint를 위반한다. 따라서 2.6GHz 이상에서 정상 동작을 위해서는 critical path를 다음과 같은 방법들로 다시 설계할 필요가 있다.

1) Cell Upsizing

Critical path 상의 standard cell들을 더 큰 cell로 교체한다. 라이브러리에 있는 더욱 큰 크기의 cell로 교체를 하게 되면, 더욱 큰 transistor가 output load를 빠르게 구동하여

propagation delay를 줄일 수 있다.

2) Buffer Insertion

Critical path가 긴 경우, 중간에 buffer를 삽입하여 delay를 분산시킬 수 있다.

3) Logic Restructuring

동일한 논리 기능을 유지하면서 logic depth를 줄이는 방법이다. 순차적으로 연결된 논리 구조를 병렬로 재배치하여 critical path를 단축할 수 있다

3. Additional discussion for Optimal Frequency

이번 optimal Frequency를 찾는데에는 min-max normalization을 이용하여 attribution의 변화를 고려했다. 실제 설계에서는 설계 목적에 맞게 각 attribution의 변화에 weight를 추가해서 조정해가며 Efficiency를 평가하는 것이 좋을 것이다.

Assessments

한민준 – 200MHz부터 3GHz까지 직접 Synthesis를 진행하였음.

주형훈 – 얻어진 결과를 Excel로 정리해 체계적으로 공유해주었음.