Low-Power System Design

Final Exam (Take Home)
Report

Ali Negary
99721081

Question 1)

a) Design an 8x8 array multiplier using VHDL/Verilog.

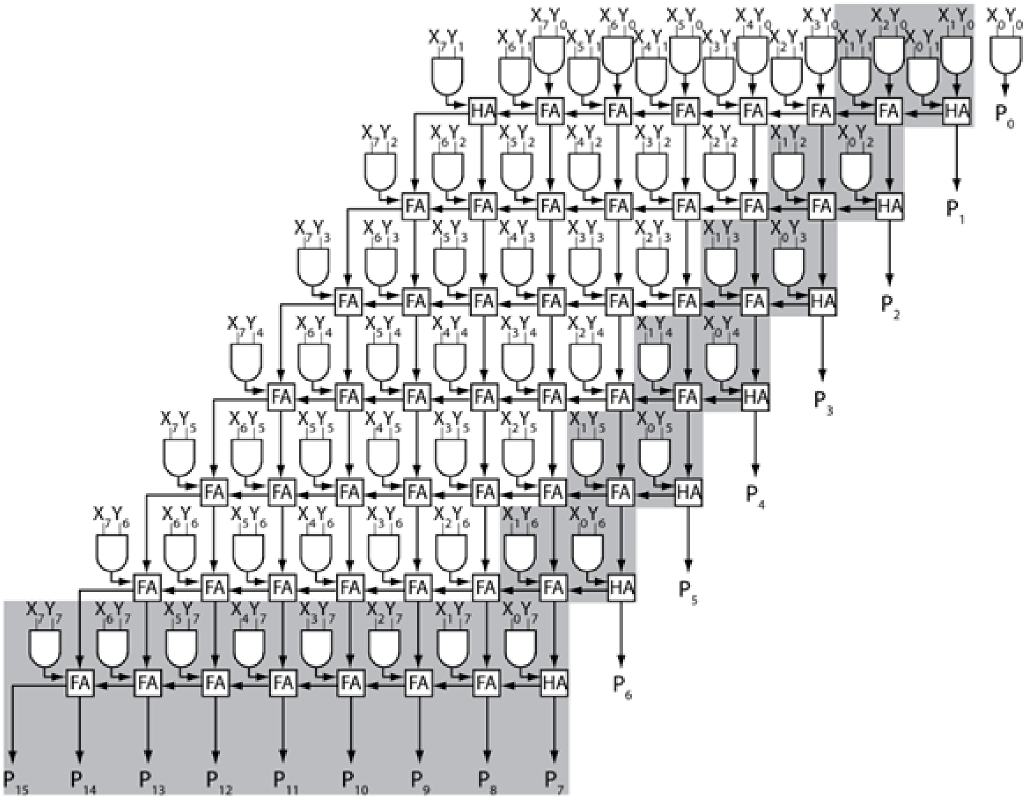
b) Synthesis your design using your desirable synthesis tool using 45nm technology.

c) Report occupation area, power consumption, and delay of the design.

d) Apply the pipelining technique to your design to reduce the power consumption and report the amount of power saving you have gained.

Answer:

For the implementation of a simple 8*8 array multiplier I utilized the following design.



X_7Y_0 X_6Y_0 X_5Y_0 X_4Y_0 X_3Y_0 X_2Y_0 X_1Y_0 X_0Y_0
X_7Y_1 X_6Y_1 X_5Y_1 X_4Y_1 X_3Y_1 X_2Y_1 X_1Y_1 X_0Y_1
HA FA FA FA FA FA FA HA
P_0
X_7Y_2 X_6Y_2 X_5Y_2 X_4Y_2 X_3Y_2 X_2Y_2 X_1Y_2 X_0Y_2
FA FA FA FA FA FA FA HA
P_1
X_7Y_3 X_6Y_3 X_5Y_3 X_4Y_3 X_3Y_3 X_2Y_3 X_1Y_3 X_0Y_3
FA FA FA FA FA FA FA HA
P_2
X_7Y_4 X_6Y_4 X_5Y_4 X_4Y_4 X_3Y_4 X_2Y_4 X_1Y_4 X_0Y_4
FA FA FA FA FA FA FA HA
P_3
X_7Y_5 X_6Y_5 X_5Y_5 X_4Y_5 X_3Y_5 X_2Y_5 X_1Y_5 X_0Y_5
FA FA FA FA FA FA FA HA
P_4
X_7Y_6 X_6Y_6 X_5Y_6 X_4Y_6 X_3Y_6 X_2Y_6 X_1Y_6 X_0Y_6
FA FA FA FA FA FA FA HA
P_5
X_7Y_7 X_6Y_7 X_5Y_7 X_4Y_7 X_3Y_7 X_2Y_7 X_1Y_7 X_0Y_7
FA FA FA FA FA FA FA HA
P_6
P_{15} P_{14} P_{13} P_{12} P_{11} P_{10} P_9 P_8 P_7

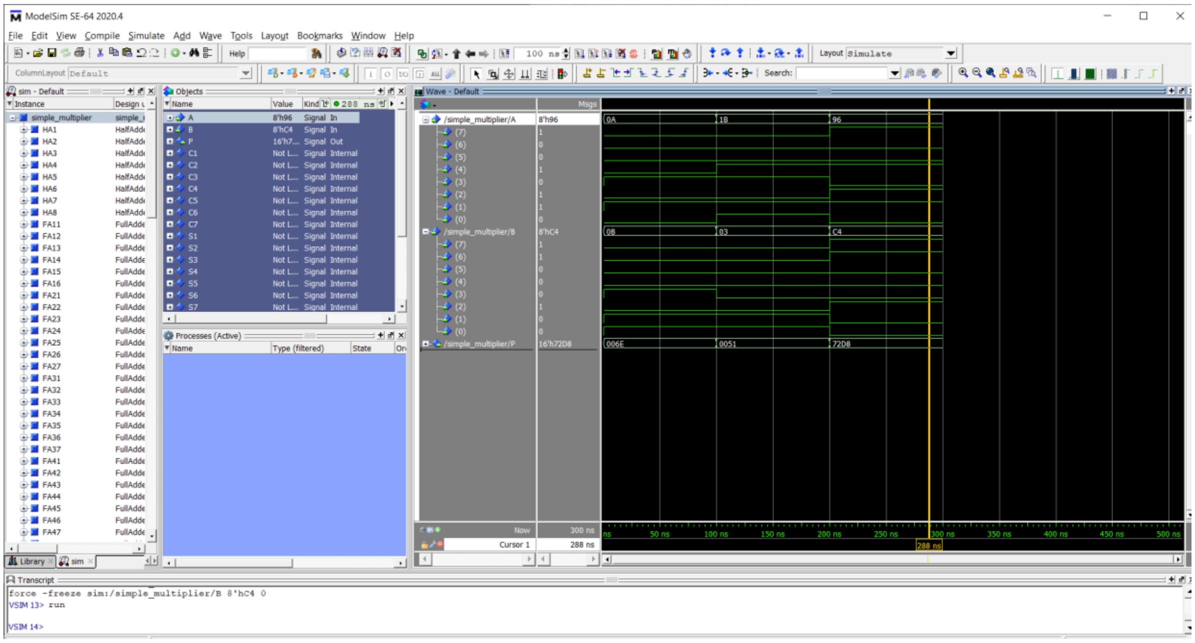
a)

You can find the corresponding VHDL code in the directory *Question1/Multiplier_8_8* under the name *simple_multiplier*.

b)

The code has been synthesized successfully using Xilinx ISE design tool on Xilinx Spartan-6 simulation setup which uses the 45nm technology.

Below, you can see the waveform of the output for 2 random inputs in the Modelsim simulator.

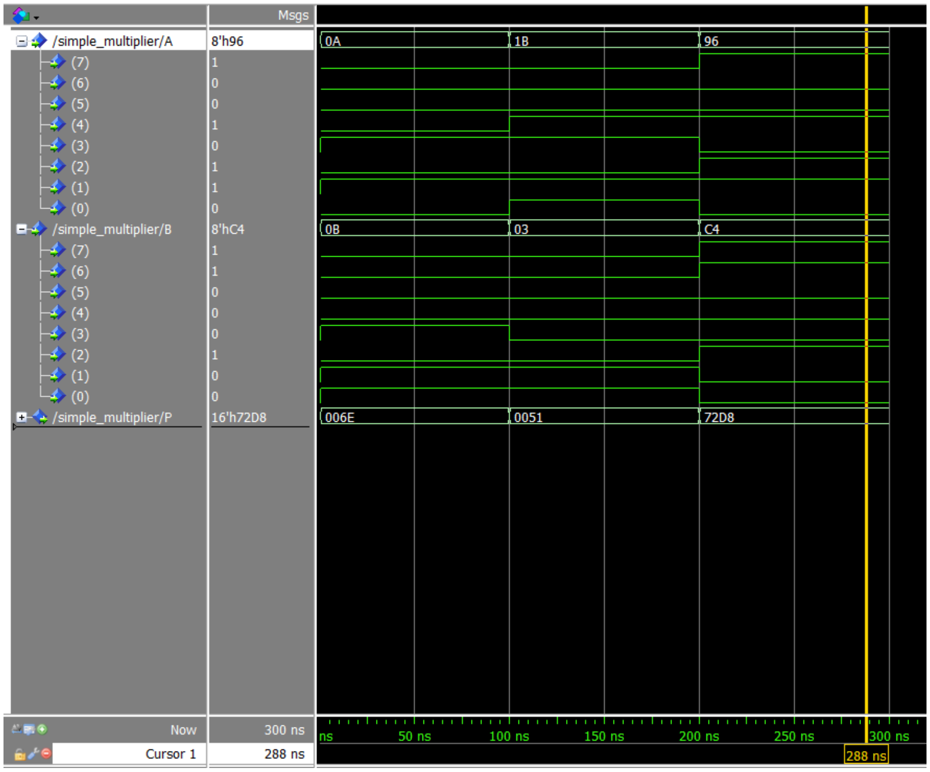


```
force -freeze sim:/simple_multiplier/B 8'hC4 0
VSIM 13> run

VSIM 14>
```

You can see the following test cases in the wave forms:

A	B	P
0A (10)	0B (11)	006E (110)
1B (27)	03 (3)	0051 (81)
96 (150)	C4 (196)	72D8 (29400)

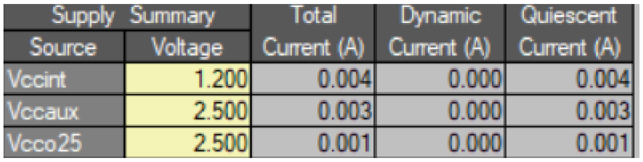


You can also find the ISE synthesis log from console in the same directory under the name *synthesis_log.txt*.

c)

According to the synthesis report (you can find the report in *Question1/C_Synthesis_Report.txt*):

- 83 LUTs has been used out of 2400 LUTs on FPGA.
- 32 IOBs has been used out of 102 IOBs on FPGA.
- Critical Path has started in A1 and Finished in P14. The path has 18 levels of logic.
- Delay of Critical path has been estimated 20.349ns (34.7% logic, 65.3% route).
- Average Fan-out of Non-Clock Nets was 4.68.
- For power analysis, it seems like a face a hidden error. Xilinx Xpower runs without any warnings or errors but it only shows me power leakage (I gave up after 4 days of searching for this issue). So, leakage power consumption has the average of 0.014 Watts and maximum of 0.023 Watts.
- The supply and current analysis is shown below:

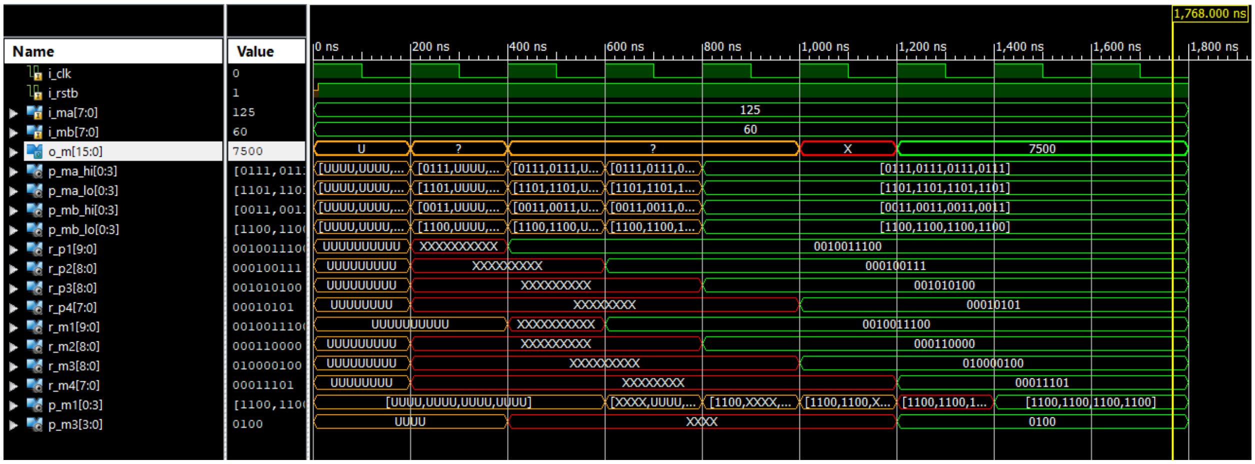


Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.200	0.004	0.000	0.004
Vccaux	2.500	0.003	0.000	0.003
Vcco25	2.500	0.001	0.000	0.001

d)

For applying the pipeline processing, I had to change the code completely. Because the whole process is different. One thing that is noticeable during the pipeline processing is the fact that output will be U or X for multiple consecutive periods (depending on the timing and number of stages to pass in the sequential circuit. I inherited the VHDL code from a schematic design. You can find the corresponding VHDL code in the directory *Question1/ Multiplier_8_8_Pipeline* under the name *pipeline_multiplier*.

Below, you can see the waveform of signal produced at each period of the VHDL code.



Name	Value
i_clk	0
i_rstb	1
i_ma[7:0]	125
i_mb[7:0]	60
o_m[15:0]	7500
p_ma_hi[0:3]	[0111,011
p_ma_lo[0:3]	[1101,110
p_mb_hi[0:3]	[0011,001
p_mb_lo[0:3]	[1100,110
r_p1[9:0]	001001110
r_p2[8:0]	000100111
r_p3[8:0]	001010100
r_p4[7:0]	00010101
r_m1[9:0]	001001110
r_m2[8:0]	000110000
r_m3[8:0]	010000100
r_m4[7:0]	00011101
p_m1[0:3]	[1100,110
p_m3[3:0]	0100

According to the synthesis report (you can find the report in *Question1/C_Synthesis_Report.txt*):

- 107 Registers has been used out of 4800 Registers on FPGA.
- 107 LUTs has been used out of 2400 LUTs on FPGA.
- 34 IOBs has been used out of 102 IOBs on FPGA.
- Critical Path has 10 levels of logic in aggregate.
- Delay of Critical path is unclear. But according to the report, offset before clock has been estimated 4.901ns (37.9% logic, 62.1% route) and offset after clock has been estimated 3.597ns (83.9% logic, 16.1% route).

- Average Fan-out of Non-Clock Nets was 2.24.
- For power analysis, exact same issue with the other design. Code runs very well. It is compiled, synthesized, simulated, and the waveform is showing as it should. I thought maybe the issue is with the .vcd file that should be dumped by Xilinx Isim and I tried using Altera Modelsim. Unfortunately, the result is the same. Below are the commands I ran in Isim and the output file is the directory *Question1/Multiplier_8_8_Pipeline* under the name *isim_sim_result.vcd* and *modelsim_sim_results.vcd*:

```
•  ISim> vcd dumpfile isim_sim_result.vcd
•  ISim> vcd dumpvars -m /
•  ISim> vcd dumpon
•  ISim>
•  # isim force add {/pipeline_8x8/i_clk} 0 -radix bin -value 1 -radix bin -time 50
   ns -repeat 100 ns
•  ISim>
•  # isim force add {/pipeline_8x8/i_rstb} 1 -radix bin -cancel 20 ns
•  ISim>
•  # isim force add {/pipeline_8x8/i_ma} 01011011 -radix bin
•  ISim>
•  # isim force add {/pipeline_8x8/i_mb} 01110111 -radix bin
•  ISim> run 1500 ns
•  ISim> vcd dumpflush
•  ISim>
```

Comparing the results)

Comparing result of Array design and Pipeline design, although the number of registers, LUTs, and IO block has increased (0 -> 107, 83 -> 107, and 32 -> 34), delays has been improved significantly from 20.34 to 4.901ns. Also fan-out has decline to half the amount of array design.

So, with 2.5 more area (including the register which were not used in the array design), we achieved more than 4 times speed-up (315%).

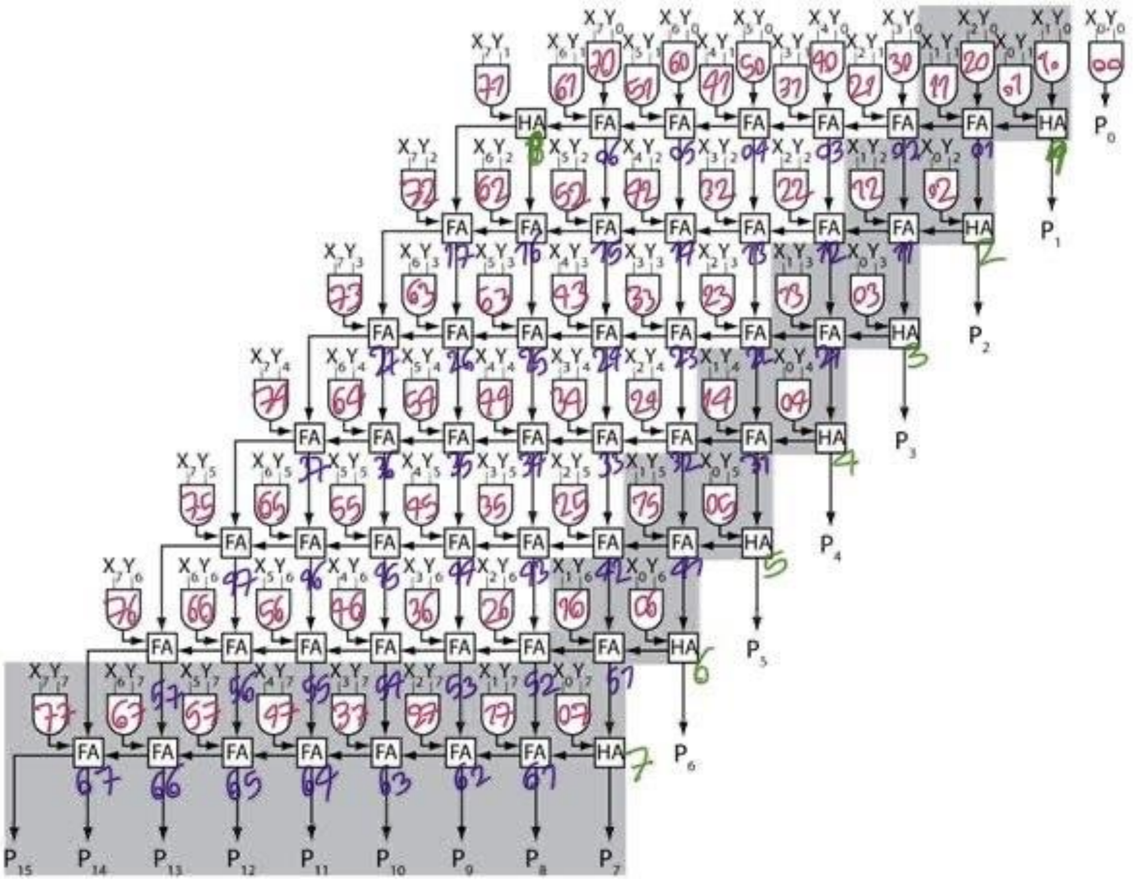
Question 2)

a) Design the 8x8 array multiplier in HSpice using 45nm technology (you can use Predictive Transistor Model: http://ptm.asu.edu/) and report the power consumption and the delay of your design.

b) Apply Dual-Vth technique to your design and report the power consumption and the delay. Please explain your observations and results in detail.

Answer:

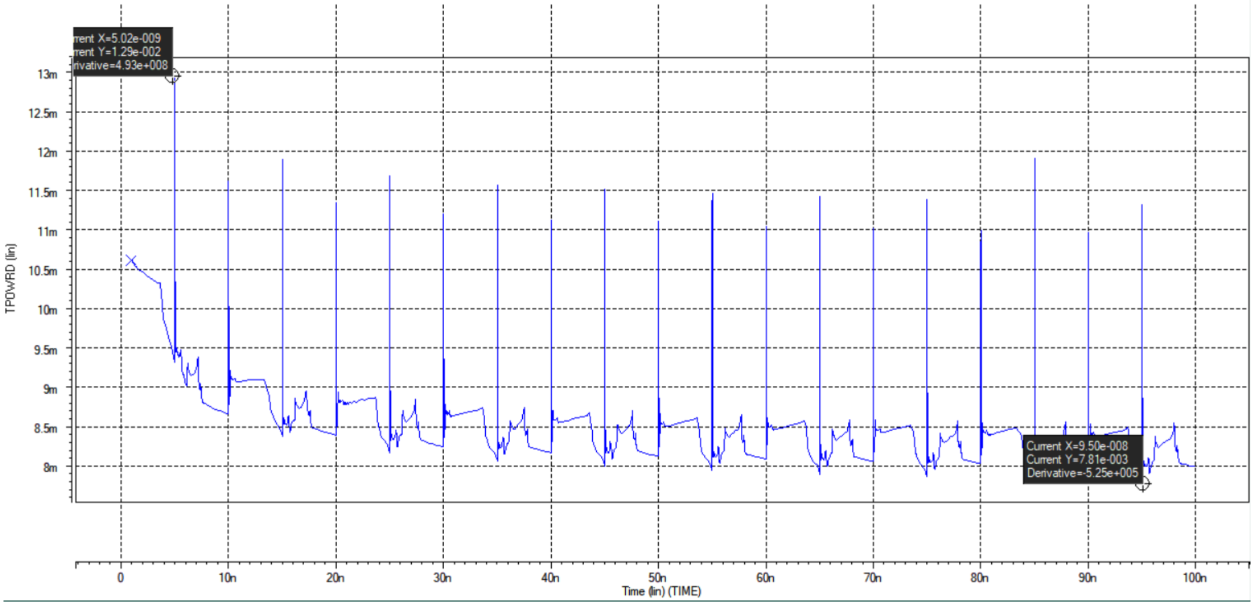
To implement Array Multiplier, I benefited from the architecture from Question1. I have named the component as follow which matches the name of components in the netlist:



To design the gates and logical components I have utilized PTM models in my simulations.

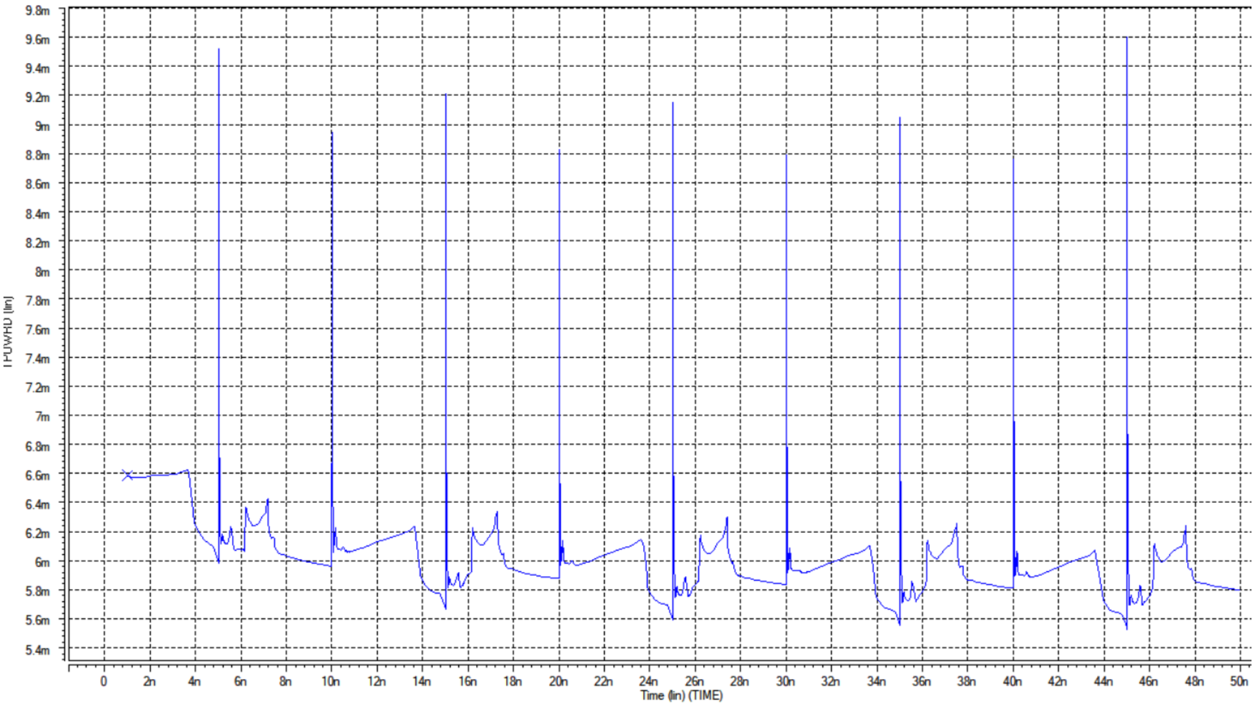
a)

You can find the netlist and simulation results for this part of the question in the directory *Question2/Main_Array_Multiplier*. The designed multiplier has been tested with multiple static and dynamic scenarios of multiplying different numbers. The average power consumption and the delay of the circuit reported in the HSpice's report was 8.31 mWatts and 0.51 nSeconds. Power consumption has peaked nearly 12.96 mWatts and 7.79 mWatts at both ends. After stabilizing, the power peaks at 11.8 and 7.9. You can see a part of the power consumption in the diagram below.



b)

You can find the netlist and simulation results for this part of the question in the directory *Question2/Main_Array_Multiplier_Dual_Vth*. For benefiting the Dual Vth method, I have used the same transistor with changing the Vth in NMOS and PMOS each by 0.2 Volts. Higher Vth has been applied to the component which are not in the critical path (highlighted components in part a). The designed multiplier has been tested with exact scenarios as part a. The average power consumption and the delay of the circuit reported in the HSpice's report was 6.1 mWatts and the delay was still in the same range as reported in part a (although in 2 executions it exceeded for about 2-3%). Power consumption has peaked nearly 9.5 mWatts and 5.58 mWatts at both ends. You can see a part of the power consumption in the diagram below.



It seems like the delay has not increased, therefore the critical path is same as the part a. However, the power consumption has improved by 36% which is good news.

Note1: For question 2, all the components have been implemented and tested separately. You can find the results in the directory *Question2*.

Note2: I wrote the python script *net_list_writer.py* in Question2, to help me write the netlist. However, it does not provide everything.