Report for ESE 507 Project 2

By

Aishwarya Gandhi (111424452)

&

Kewal Raul (111688087)

PART 1

4a.

Arithmetic operations required to multiply a 3×3 matrix with a vector of length 3:

Number of multiplications: 9

Number of additions: 6

Total operations: 9+6= **15**

Arithmetic operations required to multiply a k×k with a vector of length k: (**$2k^2$-k)**

4b.

Our control module provides write_enables, addresses and counts as control signals to the datapath. Write_enables and address signals are controlled according to states of s_valid and s_ready and counters are used to keep a count of written and read data. In our system, we have used s_ready to determine the state (reading or writing) of the system. S_ready is asserted in datapath module when it completes one matrix multiplication and ready to take new input values. Thus, it is taken as an input in the control module from the datapath module to control the control signals corresponding to the read and write states. As per synthesis report, there appears to be a latch in synthesis report in s_ready register. Even after multiple attempts, we were unable to get rid of the latch maybe due to lack of use of states.

An enable signal is take as an input to control module from datapath module to read synchronously. In datapath, enable remains asserted during a period of computation of a row of the matrix, until m_ready is high and the output is pushed to data_out. Thus, in control module new set of values are read only when enable is high.

4c.

For our verification strategy, we include a testbench which takes random inputs from a C file & the ouputs of both the C file & SystemVerilog code are compared. S_valid & M_ready are randomized similarly as your testbench.

4d.

Total cell area: **1597.06 μm^2**

Total dynamic power: **622.63µW**

Cell leakage power: **32.93 µW**

Minimum possible clock period: **1.58ns**

Maximum possible clock frequency: **630Mhz**

Critical path location: **data_out_reg to f_reg**

4e.

Clock cycles the system takes to load one set of inputs: **12 cycles**

Delay in switching from write to read state: 2 cycles

Clock cycles the system takes to compute one matrix-vector multiplication of size *k*=3: **16 cycles**

Clock cycles the system takes to output the result: **13 cycles**

Total count from first cycle of loading the input until the last cycle of outputting the result: **30 cycles**

Delay of the system: 30 * 1.58ns= **47.4ns**

4f.

Area delay product: 1597.06 μm^2 * 47.4ns

= **75700.64×10^{-21} $m^2 s$**

4g.

Energy consumed by system while computing one matrix-vector multiplication: 622.63µW * 16 * 1.58ns = **15.74pJ**

4h.

Operation count of this system: Multiplications+ Additions= 9+6= 15

Energy system consumes per arithmetic operation: = 15.74pJ/15 = **1.05pJ**

PART 2

4a.

Arithmetic operations required to multiply a 3×3 matrix with a vector of length 3:

Number of multiplications: 9

Number of additions: 19

Total operations: 9+9= **18**

Arithmetic operations required to multiply a k×k with a vector of length k*:* **$2k^2$**

4b.

As an additional vector is required for this system, so a new set of write_enables, address signals and counts are required to keep a track of its operation which are to be added to the control module.

For a computation of a row of the matrix, corresponding value of vector_b remains constant. So, accordingly the control module is modified to undergo this change.

4c.

For our verification strategy, we include a testbench which takes random inputs from a C file & the ouputs of both the C file & SystemVerilog code are compared. S_valid & M_ready are randomized similarly as your testbench.

4d.

Total cell area: **1983.3 μm^2**

Total dynamic power: **926.26 µW**

Cell leakage power: **38.02 µW**

Minimum possible clock period: **1.72 ns**

Maximum possible clock frequency: **581 Mhz**

Critical path location: **data_out_reg to f_reg**

4e.

Clock cycles the system takes to load one set of inputs: **15 cycles**

Clock cycles the system takes to compute one matrix-vector multiplication of size *k*=3: **16 cycles**

Delay in switching from write to read state: 2 cycles

Clock cycles the system takes to output the result: **13 cycles**

Total count from first cycle of loading the input until the last cycle of outputting the result: **33 cycles**

Delay of the system: 33 * 1.72ns= **56.76 ns**

4f.

Area delay product: : 1983.3 μm^2 * 56.76 ns

112572.11 $\times$ $\mathbf{10^{-21}}$ $\mathbf{m^2 s}$

4g.

Energy consumed by system while computing one matrix-vector multiplication: 926.26 µW * 16 * 1.72ns = **25.49pJ**

4h.

Operation count of this system: Multiplications+ Additions= 9+6+3= 18

Energy system consumes per arithmetic operation: = 25.49pJ/18 = **1.42pJ**

PART 3

Our control module only controls write_enables, address signals and read/write counts, no substantial changes were required. As code written for part 2 was generalized using a parameter k which represents the size of the square matrix, changing k value to 4 was sufficient. Further changes

In datapath module only a small change in the enable signal conditions was required according to the number of computations.

As per our knowledge our design should handle larger values of k, only a few tweaks in the enable logic might be required. No major changes in the design logic would be required as, k increases, only changes in the bit size of control signals will be required.

4d.

Total cell area: **2776.8 μm^2**

Total dynamic power: **924.53 µW**

Cell leakage power: **57.5 µW**

Minimum possible clock period: **1.69 ns**

Maximum possible clock frequency: **591.7 Mhz**

Critical path location: **data_out_reg to f_reg**

4e.

Clock cycles the system takes to load one set of inputs: **24 cycles**

Clock cycles the system takes to compute one matrix-vector multiplication of size *k*=3: **26 cycles**

Delay in switching from write to read state: 2 cycles

Clock cycles the system takes to output the result: **22 cycles**

Total count from first cycle of loading the input until the last cycle of outputting the result: **52 cycles**

Delay of the system: 52 * 1.69ns= **87.88 ns**

4f.

Area delay product : 2776.8 μm^2 * 87.88ns

244025×10^{-21} m^2s

4g.

Energy consumed by system while computing one matrix-vector multiplication: 924.53 µW * 26 * 1.69ns = **40.62 pJ**

4h.

Operation count of this system: Multiplications+ Additions= 16+12+4=**32**

Energy system consumes per arithmetic operation: = 40.62 pJ/ 32 = **1.27pJ**

PART 4

A)

We added a pipeline register for the multiplier to boost the speed of the system. It provided us with a considerable increase of 98 Mhz from the previous design. We were in the process of trying to implement multiple adders, multipliers & memories but due to lack of time couldn't complete it.

B)

d.

Total cell area: **2903.39 μm^2**

Total dynamic power: **1140 µW**

Cell leakage power: 60.61 **µW**

Minimum possible clock period: **1.45 ns**

Maximum possible clock frequency: **689.66 Mhz**

Critical path location: **f_reg[0] to f_reg[15]**

e.

Clock cycles the system takes to load one set of inputs: **24 cycles**

Clock cycles the system takes to compute one matrix-vector multiplication of size *k*=3: **30 cycles**

Delay in switching from write to read state: 2 cycles

Clock cycles the system takes to output the result: **25 cycles**

Total count from first cycle of loading the input until the last cycle of outputting the result: **56 cycles**

Delay of the system: 56 * 1.45ns= **81.2 ns**

f.

Area delay product: 2903.39 μm^2 * 81.2ns

=235755.27 $\times$ 10^{-21} m^2s

g.

Energy consumed by system while computing one matrix-vector multiplication: 1140 µW * 30 * 1.45ns = **49.6 pJ**

h.

Operation count of this system: Multiplications+ Additions= 16+12+4=**32**

Energy system consumes per arithmetic operation: = 49.6 pJ/ 32 = **1.55pJ**

C).

For Part 3,

Area delay product**: 244025** $\times$ **10^{-21} m^2s**

Energy system consumes per arithmetic operation: 1.27pJ

For Part 4,

Area delay product**: 235755.27** $\times$ **10^{-21} m^2s**

Energy system consumes per arithmetic operation**: 1.55pJ**

Even though energy per operation increases, the area delay product decreases for the new design. Thus, it is a better design.

E)

If we had separate parallel input ports for matrix_m, vector_x and vector_b, it would have allowed us to simultaneous operations as u read them individually avoiding the delay of waiting for values for above 3 as per current constraints. Also memories with separate read and write addresses would have made the design even faster.