REPORT

Name 1: Hrushikesh Patil **ID:** 112872294

Name 2: Nehul Oswal **ID:** 112820228

PART 1

Answer 1

The clock period of a system depends upon setup time of flip flop and Propagation delay of the concerned sequential and combinational circuits.

Shortest Possible Clock Period = (Setup time + Prop Delay) (Input flip flop) + Prop Delay (Comb logic) + (Setup Time + Prop Delay) (Middle Flip Flop) + Prop Delay (Comb logic) + (Setup time Time + Prop Delay) (Output flip flop)

Shortest Possible Clock Period = (4 + 3) ns + 4 ns + (4+3) ns + 3 ns + (4+3) ns

Shortest Possible Clock Period = 28 ns

Max Clock Frequency = (1 / Shortest Possible Clock Period)

Max Clock Frequency = (1 / 28 ns)

Max Clock Frequency = 35.71 MHz

Even if the hold time was 12 ns instead of 2 ns the Maximum Possible Clock Frequency will not change. If the hold time was 12 ns instead of 2 ns then the input data must remain stable for at least 12 ns after the clock event while in the previous case the input data must have remain stable for only 2 ns therefore if the hold time is increased there is a greater possibility of hold time violation.

Answer 2

The design contains an inferred latch. On each case of sel input the design needs to remember the previous value of g0, g1, g2 and g3. Hence each of these outputs get converted to latches. The solution to this is for each value of sel in case block all the outputs should be defined. Below is an example of synthesizable circuit.

```
module mod1(sel, g0, g1, g2, g3, a);
input [1:0] sel;
input a;
output logic g0, g1, g2, g3;
    always_comb begin
        case(sel)
            2'b00: begin
            g0 = a; g1 = 0; g2 = 0; g3 = 0;
            end
            2'b01: begin
            g1 = a;  g3 = 0; g2 = 0; g0 = 0;
```

```
            end
            2'b10: begin
             g2 = a; g1 = 0; g0 = 0; g3 = 0;
            end
            2'b11: begin
             g3 = a; g1 = 0; g2 = 0; g0 = 0;
            end
        endcase
    end
endmodule
```

Answer 3

Variable b is driven in two different always_comb blocks. To solve this problem b = e ^ a; should be removed or the value of e ^ a should be assigned to another output.

```
module mod2(a, b, c, d, e);
input a, c;
output logic b, d, e;
always_comb begin
if (c == 1) begin
      e = a;
      b = c;
end
else begin
      e = 0;
      b = a;
end
end
always_comb begin
      d = a | c;
end
endmodule
```

PART 2

Answer a)

Testing approach: The corner test cases were done manually. We did exhaustive testing using automated test cases. Test cases were generated by C program which wrote randomized value of a (input data) and valid_in (input). These values were stored in a text file which was used as an input for the testbench. The output was calculated using the C code and outputting it in a text file. The testbench also records the outputs in a text file. The two output files were compared for diff. The automated testbench was run for 100000 test cases. This covers a lot of scenarios like overflows, valid_in being 0 for multiples cycles etc. To match the output generated by testbench the output of automated C code was mod by 2^20 to protect from overflows.

Answer b)

In order to get the accumulator to overflow we gave largest valid input (i.e. in this case 255) for a clock cycle and therefore the result got so large that it was not possible to represent the output using a 20 bit register. In order to detect overflow, we can pad an extra bit to the output bits as MSB bit so that whenever the padded

bit is high it means that the output has overflown. Basically, we increase the output register size from 20 bits to 21 bits where the 20th bit (assuming smallest bit is the 0th bit) indicates whether the result has overflown.

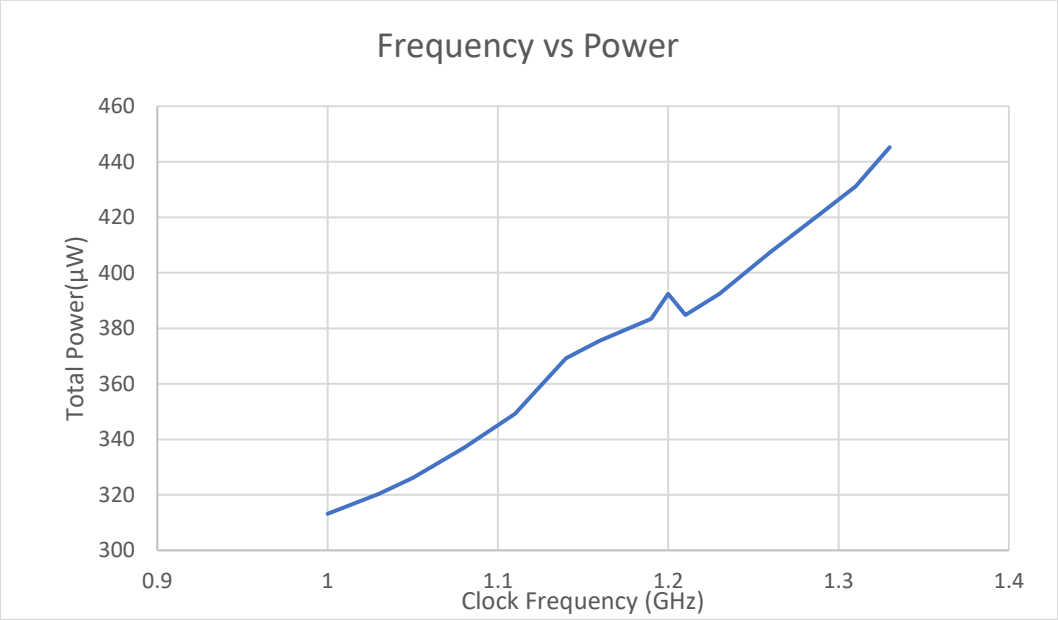
Answer c)

The fastest frequency reached was **1.33 GHz**.

Clock frequency(GHz)	Total Power(μW)	Area(μm^2)	Critical Path
1	313.17	453.26	The critical path goes from the 5th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 19th bit of the output flip flop (f).
1.03	320.4	446.61	The critical path goes from the 2nd bit of the input flip flop (dff_out) through the multiplier and adder and upto the 14th bit of the output flip flop (f).
1.05	326.07	446.61	The critical path goes from the 2nd bit of the input flip flop (dff_out) through the multiplier and adder and upto the 18th bit of the output flip flop (f).
1.08	336.99	452.19	The critical path goes from the 0th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 19th bit of the output flip flop (f).
1.11	349.22	462.57	The critical path goes from the 0th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 19th bit of the output flip flop (f).
1.14	369.29	480.39	The critical path goes from the 3rd bit of the input flip flop (dff_out) through the multiplier and adder and upto the 19th bit of the output flip flop (f).
1.16	375.52	487.84	The critical path goes from the 4th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 19th bit of the output flip flop (f).
1.19	383.47	475.87	The critical path goes from the 1st bit of the input flip flop (dff_out) through the multiplier and adder and upto the 14th bit of the output flip flop (f).
1.2	392.4	489.7	The critical path goes from the 0th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 15th bit of the output flip flop (f).
1.21	384.75	471.35	The critical path goes from the 2nd bit of the input flip flop (dff_out) through the multiplier and adder and upto the 11th bit of the output flip flop (f).
1.23	392.39	476.93	The critical path goes from the 0th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 11th bit of the output flip flop (f).
1.26	407.53	489.17	The critical path goes from the 6th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 18th bit of the output flip flop (f).

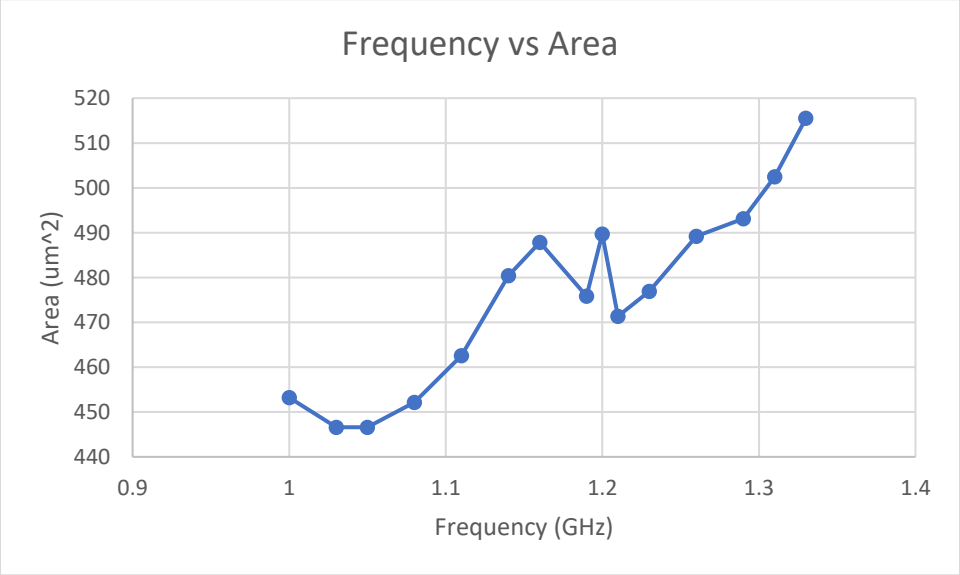
1.29	421.54	493.16	The critical path goes from the 3rd bit of the input flip flop (dff_out) through the multiplier and adder and upto the 11th bit of the output flip flop (f).
1.31	431.17	502.47	The critical path goes from the 6th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 17th bit of the output flip flop (f).
1.33	**445.24**	**515.5**	**The critical path goes from the 0th bit of the input flip flop (dff_out) through the multiplier and adder and upto the 15th bit of the output flip flop (f).**

Answer d) Frequency vs Power Graph



The Frequency vs Power graph is mostly linearly increasing. This is due to the fact while static power consumption is not dependent on frequency but dynamic power ($P_{dynamic} = ½ CV^2 F$) is directly proportional to frequency. Hence the Total Power consumption increases as the frequency is increased.

Frequency vs Area Graph



Frequency vs Area is also a generally linearly increasing with increase in frequency. However, there are few outliers in the graph as the synthesis tool uses different optimizations for different clock frequencies.

Answer f)

Energy will change if frequency of the system changes. This is because the total power consumed by the system will change as we change the frequency as dynamic power consumed by the system depends upon frequency and static power consumption may also change as the chip area may change if the frequency changes. Since energy is power consumed per second therefore as power is changing with frequency therefore energy will also change.

Answer g)

It is necessary to reset both the input and output registers as if they are not reset then the output of the registers will be Don't Care(X) and for every clock cycle X will be added to the input therefore the output will always be X for all the input cycles.

Part 3

Answer a)

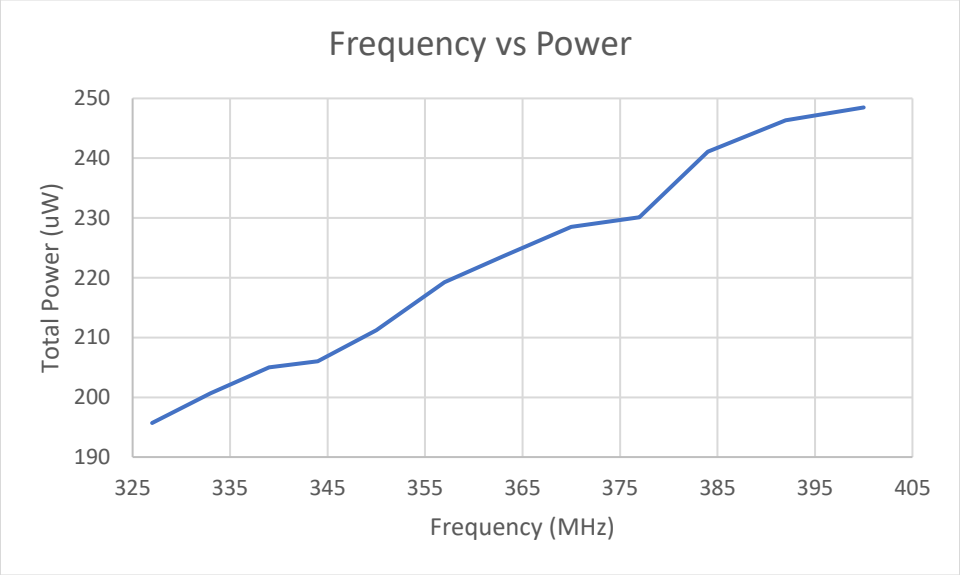
We needed to instantiate square root module and add one extra clock cycle delay and in the automated test cases we added the square root operation. All test cases from previous testbench were applicable with extra one clock cycle delay. In automated testcases the mod of the accumulator output was given as input to sqrt. The final output needs not to be detected for overflow as sqrt of 20 bit number needs max 10 bits to represent.

Answer b)

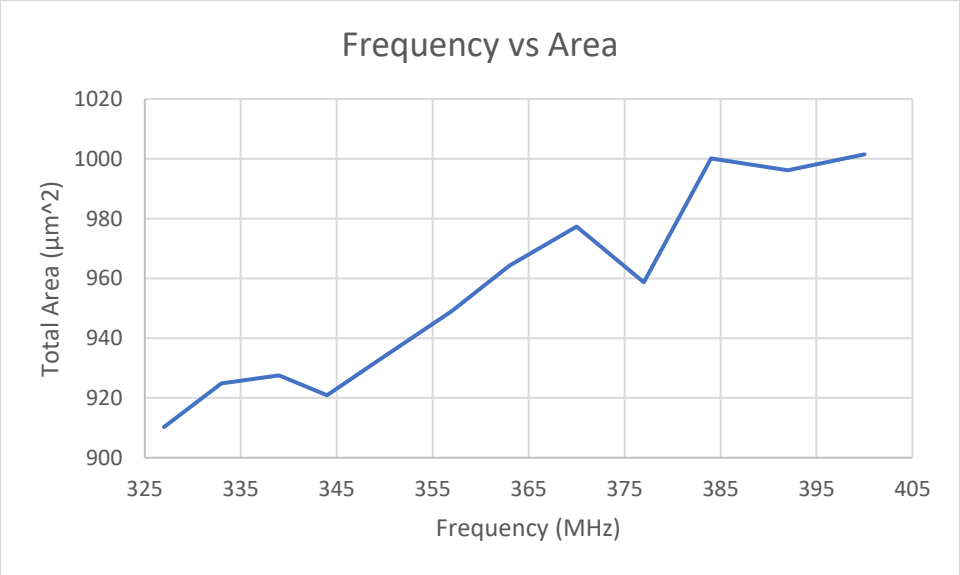
The maximum clock frequency is 400 MHz and critical path is from 18^{th} bit of the F (register after the accumulator) register through the square root module up to the 0^{th} bit of the final output register.

Frequency(MHz)	Total Power(µW)	Total Area(μm^2)
400	**248.4814**	**1001.48999**
392	246.3326	996.16999
384	241.1077	1000.15999
377	230.1278	958.66399
370	228.5162	977.28399
363	223.5824	964.24999
357	219.2325	949.087991
350	211.2115	933.925991
344	206.0436	920.891991
339	205.0093	927.541992
333	200.6829	924.881991
327	195.7033	910.251992

Frequency vs Power Graph



Frequency vs Area Graph



Upon comparing the results, we could infer that the frequency of operation of the part 3 module is much less than the part 2 module. The chip area for the part 3 module is much higher than the part 2 module. The increase in area is due to the increase in number of circuit elements in part3. The increase in circuit elements leads to increase in data required time resulting in lower max frequency compared to part2. However, the graphs for both part2 and part3 for both area and power are similar as they are mostly linearly increasing with a few outliers.