Adv Digital Design III

Dr. Michaela Amoo

Assigned: 02/25/2020

Due:4/2/2020

Hunter Bryant @02777748

Braxton Dula @02784592

Jordan Aley @02864549

Harena Yemane @02838537

Homework #5

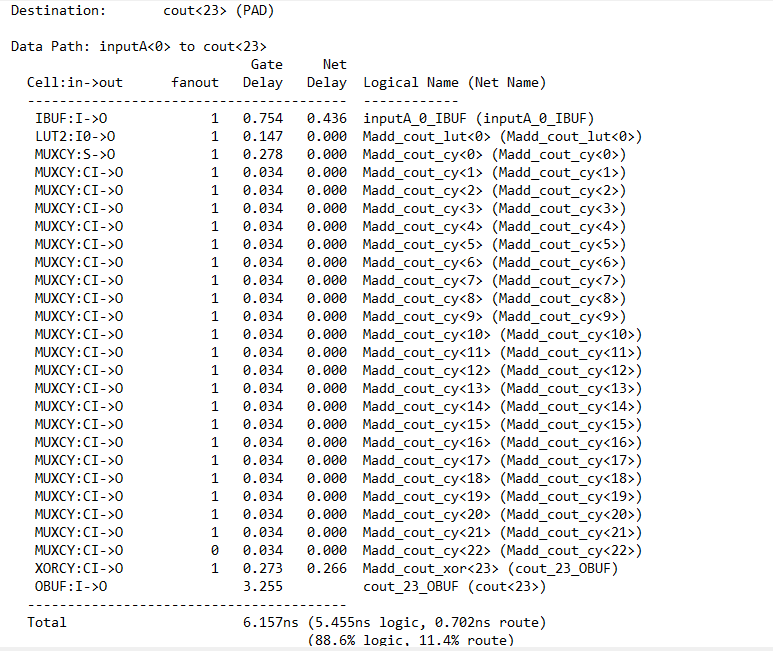
**Component Definition Critical Path Analysis:**

This section defines the components used by the synthesis tool to synthesize the five multipliers. This section also describes the critical paths of the multipliers in terms of their components. Components previously defined in the last report are omitted.

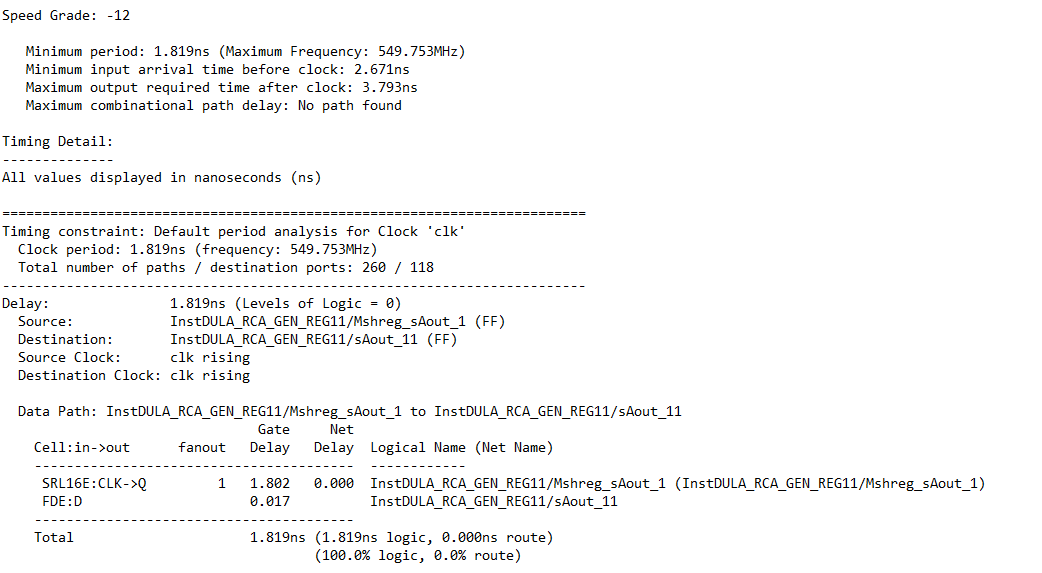
**MUXF5:**

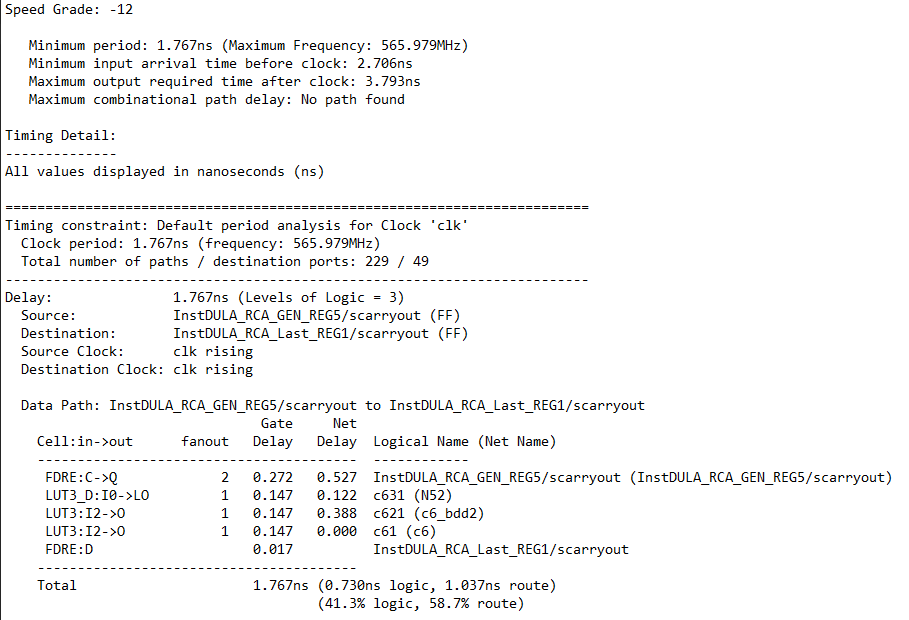
2-to-1 Look-Up Table Multiplexer with General Output. This design element is a two input multiplexer The local outputs (LO) from two LUT4 look-up tables are connected to the I0 and I1 inputs of the MUXF5. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1. The O output is a general interconnect.

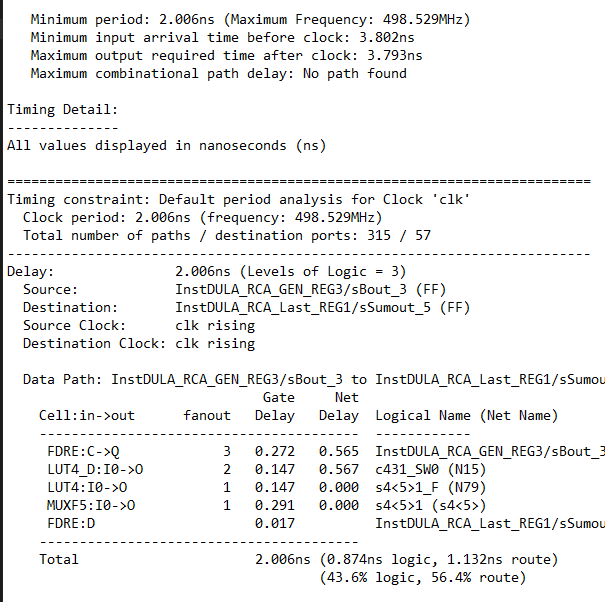
The Combinational design longest delay shown below is from the input pad to output pad. The path is made up of the input buffer IBUF, one LUT2, 22 MUXCYs, the output buffer OBUF and takes 6.157 ns.

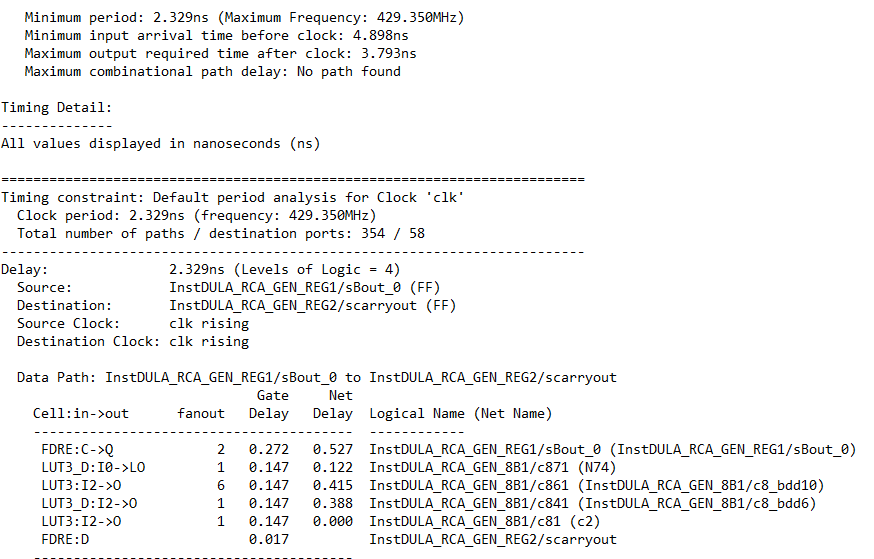


The RCA \_2B design critical path shown below is from the output of general Register 11 to the FDRE. The path is made up of a SRL16E, an FDRE flip-flop, took 1.819ns.



The RCA \_4B design critical path shown below is from the output of Register 5 to the outputs of the last Register. The path is made up of a FDRE flip-flop, three LUT3s, another FDRE flip-flop and takes 1.767ns. 

The RCA \_6B design critical path shown below is from the output of Register 3 to the outputs of the last Register. The path is made up of a FDRE flip-flop, two LUT4s, a MUXF5, another FDRE flip-flop and takes 2.006ns.

The RCA\_8B design critical path shown below is from the output of Register 1 to the outputs of Register 2. The path is made up of a FDRE flip-flop, 4 LUT3s, another FDRE flip-flop and takes 2.329ns.

**Comparison Of Five Designs:**

In this section the five RCA’s are compared in terms of usage, critical paths and Frequency. The differences in logic and routing usage is also analyzed.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Slices | Slice Flip Flops | LUT\_4s | Max Freq | Crit path route% | Crit path logic% |
| COMB | 12/49152 | N/A | 24/10944 | 162.4MHz | 88.6.0% | 11.4% |
| RCA\_2B | 107/49152 | 178/98304 | 143/98304 | 549.753MHz | 100% | 0% |
| RCA\_4B | 100/49152 | 154/98304 | 120/98304 | 549.753MHz | 100% | 0% |
| RCA\_6B | 87/49152 | 136/98304 | 74/98304 | 498.529MHz | 43.6% | 56.4% |
| RCA\_8B | 67/49152 | 99/98304 | 73/9804 | 429.35MHz | 37.7% | 62.3% |

All four clock designs use less than 1% of the available Slices, Slice Flip Flops and LUTs so in order to effectively compare the designs they are compared in terms of Max Frequency, route% and logic%. Out of all the clocked designs both RCA\_2B and RCA\_4B have the highest frequency of 549.753MHz but have a route% of 100% due to the path consisting only of a Shift Register and FDRE requiring little logic. However because devices with 100% routing are unlikely to pass static timing analysis these two designs are not the best. RCA\_6B has the next highest frequency of 498.529MHz with a route% of 43.6 and 57%. RCA\_8B has the lowest route% and highest logic% of 37.7% and 62.3% but also the lowest Max Frequency at 428.350 MHz. Thus, since all the designs are similar in device utilization RCA\_6B is the best design due to its high Max Frequency and viable route% and logic%.

**Troubleshooting:**

While designing the RCA’s the bug that gave our team the most trouble was getting the correct sum value at the correct clock cycle. Originally each block in the design passed its sum to the next register in the ripple, that register would then pass its part of the final sum to the final sum signal. The problem this caused is that because of the ripple design of the RCA the registers would get their sums at different times meaning the final sum wouldn't be ready on the same clock cycle. This issue was fixed by having every sum created by each block passed to the next register and concatenated together and having the final sum come from the last register. Instead of each “block” of the final sum being passed at different times the entire sum would be ready n clock cycles(where n is the number of registers) after the input.