Cole Maxwell (25017014): ECE 658

Lab 4: Network on Chip

November 2020

Part B: Synthesis

**Step 3: Synthesize Design**

Changes to TCL files:

In *setup.tcl*, I changed the *clkname* variable to “clk” and modname variable to “Tile” to match the names in *Tile.v*.

In *read.tcl*, I added the following files to be read as design input in dc shell:

read\_verilog $RTL\_DIR/Tile.v

read\_verilog $RTL\_DIR/ALU32.v

read\_verilog $RTL\_DIR/Controller.v

read\_verilog $RTL\_DIR/crossbar.v

read\_verilog $RTL\_DIR/InstructionDecoder.v

read\_verilog $RTL\_DIR/matrix5arb.v

read\_verilog $RTL\_DIR/Memory.v

read\_verilog $RTL\_DIR/NangateOpenCellLibrary.v

read\_verilog $RTL\_DIR/Processor.v

read\_verilog $RTL\_DIR/ram\_dp\_ar\_aw.v

read\_verilog $RTL\_DIR/RegisterFile.v

read\_verilog $RTL\_DIR/route\_table.v

read\_verilog $RTL\_DIR/router.v

read\_verilog $RTL\_DIR/syn\_fifo.v

In *Constraints.tcl*, I loosened the clock period to make for easier synthesis by setting the *CLK\_PER* variable to “20”ps.

Cell Instances:

There are **121** cell instances in the design consisting of:

* 113 logic gates
* 2 flip-flops
* 1 CPU
* 1 CPU2Router
* 1 DataMemory
* 1 InstructionMemory
* 1 Router2CPU
* 1 router\_inst.

Total Area:

The total cell area of the tile is **59571.4974 square microns**, found in *cell\_report\_final.rpt.*

Maximum Clock Frequency: 0.**15ns or 6.67 GHz.**

I used Design Compiler to run synthesis in batch mode with *run\_synth.tcl*. I modified the *CLK\_PER* value for each batch and looked in *timing\_max\_slow\_holdfixed\_tut1.rpt* to see if there was a timing violation or not.

Critical Path:

There are 99 logic gates in the critical path. In *timing\_max\_slow\_holdfixed\_tut1.rpt* (which is the critical path report for the slow corner), the path is described, where each line is a step in the path. Thus, there are 99 gates listed in the critical path.

Part C: Place and Route

**Step 5: Placement**

Images:

Floorplan view:

A picture containing screenshot

Description automatically generated

Amoeba view:

A screenshot of a computer

Description automatically generated

Physical view:

A picture containing clock

Description automatically generated

Dimensions:

Dimensions are roughly 258 \* 258 = **66,497.8752** square microns.

Consistency:

Compared to the area report of roughly 244 \* 244 = **59,571.4974** square microns, the results are consistent overall, but the tile appears to be a bit larger in the physical view. Since the initial place and route is a trial route to estimate the required metal layers, it may contain extra wires adding to the area and accounting for this difference.

**Step 6: Detailed Routing**

Metal Layers:

According to *encounter.log*, there are 10 metal layers used in detailed routing, described toward the end of the file in the Complete Detail Routing section. Metal layers 2, 3, 4 and have the most routing out of the 10, with 3 having the most overall. This is reasonable, because in the highest layers, wires are fewer in number, and in the lowest layers, wires are very short. It seems reasonable that metal layers 2, 3, and 4 have the most traffic and would have a high percentage of routing – connecting the many gates to each other in small to medium connection lengths.

Image of routing with uppermost metal wiring only:

A picture containing circuit, computer

Description automatically generated

Width of metals:

|  |  |
| --- | --- |
| **Metal Layer** | **Width (microns)** |
| M2 | 0.070 |
| M4 | 0.140 |
| M6 | 0.140 |
| M10 | 0.800 |

The width of metal layers increases in higher layers. This is expected since wires in higher layers typically drive over longer distances, so they must be wider to reduce resistance.

Estimated the size of a mesh network:

For a chip that is 1mm by 1mm, or 1000um by 1000um, I could almost fit 4 tiles at 258um by 258um, but it would be slightly oversized, so I could really only fit 2.