Lab 4 Report
Simulation, Synthesis and Physical Design (ECE558)

Luan Vo (30553600), November 21th 2020 (can be submitted by Saturday 11/21 – mentioned in class on 11/19)

***Purpose & Introduction* — This lab focuses on the automated aspect of VLSI, by using tools such as encounter and synthesis.**

I. SYNTHESIS (STEP 3):

A. Changes to script (TCL files)

Below figures show us the summary of our designs to show that they are working properly under the new setup:

```
****************************************
check_design summary:
Version:     E-2010.12-SP5-2
Date:        Wed Nov 18 17:50:24 2020
****************************************

                    Name                                                  Total
--------------------------------------------------------------------------------
Inputs/Outputs                                                           124
    Unconnected ports (LINT-28)                                            8
    Feedthrough (LINT-29)                                                 74
    Shorted outputs (LINT-31)                                             42

Cells                                                                     27
    Cells do not drive (LINT-1)                                            5
    Connected to power or ground (LINT-32)                                15
    Net is fed into multiple inputs (LINT-33)                              7

Nets                                                                      81
    Unloaded nets (LINT-2)                                                81
--------------------------------------------------------------------------------
```

Figure 1. *check_design summary output*

```
Presto compilation completed successfully.
Current design is now '/home/ece558_2020/luanvo/lab4/noc_files/syn_fifo.db:syn_fifo'
Loaded 1 design.
Current design is 'syn_fifo'.
syn_fifo
source Constraints.tcl
Current design is 'Tile'.
Loading db file '/home/ece558_2020/luanvo/lab4/noc_files/NangateOpenCellLibrary_slow_ccs.db'
Loading db file '/usr/synopsys/E-2010.12-SP5-2/libraries/syn/dw_foundation.sldb'
Information: Checking out the license 'DesignWare'. (SEC-104)
Information: Evaluating DesignWare library utilization. (UISN-27)
```

Figure 2. *Presto successful compilation*

```
 Beginning Delay Optimization Phase
 ----------------------------------

  ELAPSED            WORST NEG TOTAL NEG  DESIGN                              MIN DELAY
    TIME      AREA      SLACK     SLACK   RULE COST         ENDPOINT             COST
 --------- --------- --------- --------- --------- ------------------------- ---------
   0:00:06   59459.0      0.00       0.0    1253.1                               0.00


 Beginning Design Rule Fixing  (max_capacitance)
 ----------------------------

  ELAPSED            WORST NEG TOTAL NEG  DESIGN                              MIN DELAY
    TIME      AREA      SLACK     SLACK   RULE COST         ENDPOINT             COST
 --------- --------- --------- --------- --------- ------------------------- ---------
   0:00:10   59459.0      0.00       0.0    1253.1                               0.00
   0:00:11   59465.4      0.00       0.0    1135.1 CPU/RegFile/n4021             0.00
   0:00:11   59471.5      0.00       0.0    1015.3 InstructionMemory/n9254       0.00
   0:00:11   59478.4      0.00       0.0     926.3 InstructionMemory/n9296       0.00
   0:00:12   59483.7      0.00       0.0     796.3 CPU/RegFile/n1577             0.00
   0:00:13   59487.4      0.00       0.0     717.6 DataMemory/n2500              0.00
   0:00:13   59490.9      0.00       0.0     638.6 CPU/n582                      0.00
   0:00:14   59497.8      0.00       0.0     541.3 CPU/RegFile/n4006             0.00
   0:00:15   59509.3      0.00       0.0     448.0 DataMemory/n10096             0.00
   0:00:16   59513.2      0.00       0.0     391.0 DataMemory/n10697             0.00
   0:00:16   59519.4      0.00       0.0     335.3 DataMemory/n10039             0.00
   0:00:17   59528.4      0.00       0.0     218.8 InstructionMemory/n9202       0.00
   0:00:18   59530.0      0.00       0.0     176.6 CPU/RegFile/n4013             0.00
   0:00:18   59531.1      0.00       0.0     129.1 DataMemory/n9982              0.00
   0:00:18   59538.0      0.00       0.0      89.0 router_inst/east_buff/rd_pointer[0]     0.00
   0:00:20   59553.1      0.00       0.0      38.7 CPU/RegFile/n1635             0.00
   0:00:20   59568.3      0.00       0.0       8.0 DataMemory/n7453              0.00
   0:00:22   59571.5      0.00       0.0       0.0                               0.00
```

Figure 3. *Synthesis' design overview*

From Fig. 1, 2 and 3, we can see that the set up is running correctly and the modifications made towards TCL files were correct. The above figures are achieved after these lines are added to "read.tcl" file:

```
#
# Revision History
#   1/15/03  : Author Shane T. Gehring - from class example
#   2/09/07  : Author Zhengtao Yu      - from class example
#   12/14/07 : Author Ravi Jenkal      - updated to 180 nm & tcl
#
#-------------------------------------------------------------

#-------------------------------------------------------------
# Read in Verilog file and map (synthesize) onto a generic
# library.
# MAKE SURE THAT YOU CORRECT ALL WARNINGS THAT APPEAR
# during the execution of the read command are fixed
# or understood to have no impact.
# ALSO CHECK your latch/flip-flop list for unintended
# latches
#-------------------------------------------------------------

read_verilog $RTL_DIR/Tile.v
read_verilog $RTL_DIR/Processor.v
read_verilog $RTL_DIR/router.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/ram_dp_ar_aw.v
read_verilog $RTL_DIR/RegisterFile.v
read_verilog $RTL_DIR/route_table.v
read_verilog $RTL_DIR/Testbench.v
read_verilog $RTL_DIR/system.v
read_verilog $RTL_DIR/syn_fifo.v
```

Figure 4. *read.tcl file modification (RTL)*

The reason behind these lines is because that these are all the Verilog files included within lab 4 folder (i.e: .v prefix files). While not every Verilog files are needed, it does not pose a potential problem when including more than needed. Among these lines, the one that holds our interest is the Tile.v file, as this will be taken as the module name – as per mentioned during the lab procedure in step 3. ***This means that the following also need to be changed within the "setup.tcl" file:***

Set modname Tile | set clkname clk

This is because the name of our synthesis module is Tile (again, as per mentioned in the lab), and within the Tile.v file, its clock is defined to be clk. ***If an incorrect clock's name is set, synthesis will return an error since clk will then have no value causing the design to fail***.

B. Design validation: Max frequency and critical path

The maximum frequency will be detected at the frequency just before timing violation occurs. This means that if we can find the frequency that gives the first timing violation, we have reached the frequency threshold. ***By having the CLK_PER variable to be within the precision of 10s. This is to say that the precision will be the magnitude of 10s – this adheres to the standard set forth during lab 3***. Also, it is difficult to guess the exact time, so a rough estimate of CLK_PER in the 10s precision will work out just fine.

The answer for both critical path and maximum frequency is given in "timing_max_slow_holdfixed_tut1.rpt", because this file contains critical path report for slow corner. By observing the report taken from, we can make the following conclusions:

```
clock clk (rise edge)                                  10.0000    10.0000
clock network delay (ideal)                             0.0000    10.0000
clock uncertainty                                      -0.0500     9.9500
DataMemory/Memory_reg[168][5]/CK (DFF_X1)               0.0000     9.9500 r
library setup time                                     -0.1779     9.7721
data required time                                                 9.7721
--------------------------------------------------------------------------
data required time                                                 9.7721
data arrival time                                                -10.0720
--------------------------------------------------------------------------
slack (VIOLATED)                                                  -0.2999
```

Figure 5. *Timing Violation at CLK_PER = 10ns*

Fig. 5 indicates that (with a precision of 10s) the system failed at 10ns, which means that the maximum operational frequency is at 20ns, this is because 10ns is the first time that synthesis reports a timing violation. To prove this, observe the figure below at 20ns:

```
clock clk (rise edge)                                  20.0000    20.0000
clock network delay (ideal)                             0.0000    20.0000
clock uncertainty                                      -0.0500    19.9500
CPU/RegFile/regFile_reg[11][0]/CK (DFF_X1)              0.0000    19.9500 r
library setup time                                     -0.1853    19.7647
data required time                                                19.7647
--------------------------------------------------------------------------
data required time                                                19.7647
data arrival time                                                -15.6826
--------------------------------------------------------------------------
slack (MET)                                                        4.0821
```

Figure 6. *Timing MET at CLK_PER = 20ns*

As shown in Fig. 6, the timing constraints is met at 20ns, it means that the maximum operational frequency is 50MHz. From the same file, we can determine the critical path by looking at the following:

```
 # A fanout number of 1000 was used for high fanout net computations.

Operating Conditions: slow   Library: NangateOpenCellLibrary
Wire Load Model Mode: top

  Startpoint: InstructionMemory/out_Data_reg[29]
              (rising edge-triggered flip-flop clocked by clk)
  Endpoint: CPU/RegFile/regFile_reg[11][0]
            (rising edge-triggered flip-flop clocked by clk)
  Path Group: clk
  Path Type: max
```

Figure 7. *Tile's critical path at CLK_PER = 20ns*

From Fig. 7, it is evident that the starting point is from IMEM/Data_Register[29] to CPU/RegFile_reff[11]. This means that the path above takes the longest time out of the entire network for the data to fully flow from the start to endpoint. To observe the critical delay time (in terms of logic gates, we look at the data arrival time parameter at the end of Fig. 7's script. ***Fig. 6 shows that the critical delay time is around 15.68 logic gates,*** whereas the required time indicates the logic gate threshold needed.

C. Cell instances and area

The information regarding number of cells and total area of synthesis design is contained within "cell_report_final.rpt". By observing said file, we have the following table:

Table 1. *Design cells and area statistics*

Time (ns)	Cells	Area (um^2)
20	30577	58951.45

To show this, observe the figure below for cell report at 20ns:

```
router_inst/west_buff/wr_pointer_reg[1]
                          DFF_X1          NangateOpenCellLibrary
                                                          4.5220  n
router_inst/west_buff/wr_pointer_reg[2]
                          DFF_X1          NangateOpenCellLibrary
                                                          4.5220  n
valid_data_reg            DFF_X1          NangateOpenCellLibrary
                                                          4.5220  n
--------------------------------------------------------------------------------
Total 30577 cells                                      58951.4514
1
```

Figure 8. *Timing Violation at CLK_PER = 10ns*

This means that at 20ns, synthesis produces a design that consists of 30577 cells, with an area of 58951.45 um². It is worth noting that the 30577 cells represent a sum of all gates and flops within the design.

II. PLACEMENT (STEP 5)

A. Layout design at density = 0.9 (official measurements)

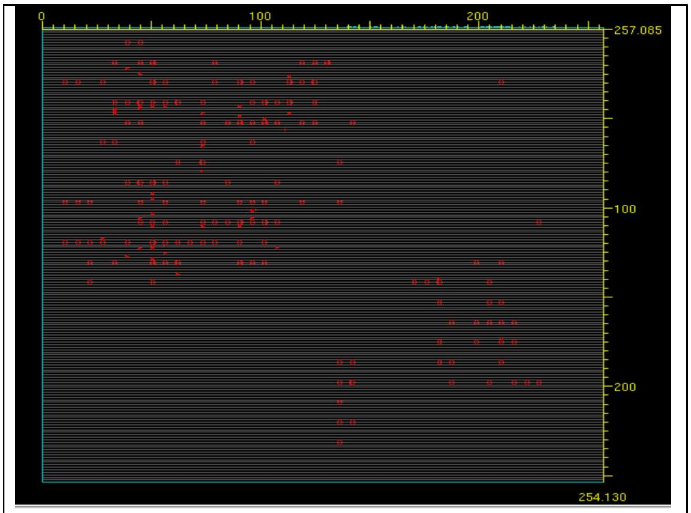

Figure 9. *Encounter simulation of Tile's floorplan, d = 0.9*

Fig. 9 shows that the resulting area comes out to be around:

Dimension / Area = 257 um x 254 um = 65278 um²

Below are the design floorplans with and without wiring/vias:

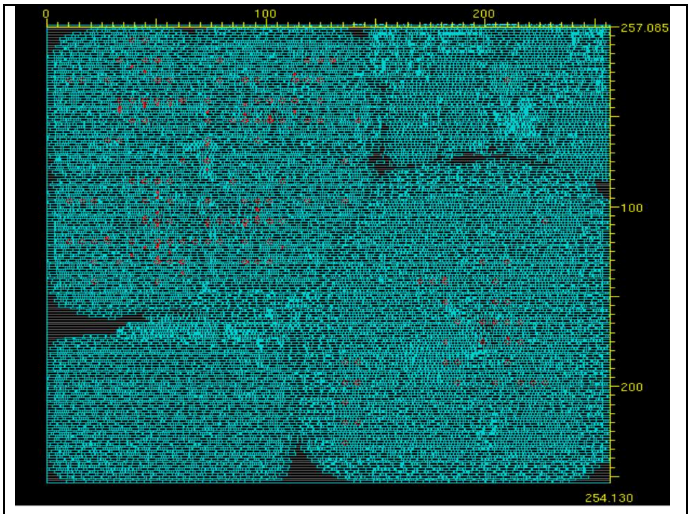

Figure 10. *Encounter – Tile's physical view no wires and vias*

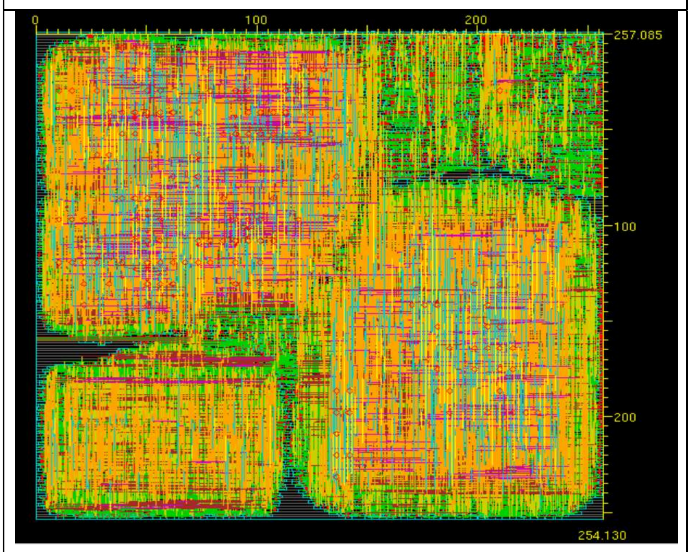

Figure 11. *Encounter – Tile's physical view wires and vias*

Observe the figures above (specifically 10 and 11), where 10 depicts the physical view of tile's design with no wires and vias. Whereas Fig. 11 shows the entire tile's design with both wires and vias. Similar to lab 3 premises, tools are not perfect, and we can see some gaps in between the design.

B. Dimensions and Area of design

Below are the cells and area report from encounter's netlist:

```
#std cell=30577 #block=0 (0 floating + 0 preplaced) #ioInst=0 #net=33642 #
#floatPin=161
stdCell: 30577 single + 0 double + 0 multi
Total standard cell length = 42.1082 (mm), area = 0.0590 (mm^2)
Average module density = 0.900.
Density for the design = 0.900.
       = stdcell_area 221622 (58951 um^2) / alloc_area 246246 (65501 um^2)
Pin Density = 0.553.
            = total # of pins 122483 / total Instance area 221622.
Checking spec file integrity...
```

Figure 12. *Encounter's netlist – at density = 0.9*

With this, we can conclude that encounter produced the following design parameter:

Table 2. *Encounter's cells and area statistics*

Density	Cells	Area (um^2)
0.9	30577	65501

Compared the results from table 1 and 2, the only big difference comes from the total area used for the design. There are three major reason for this difference:

1. Density definition in config file, this affects how condensed the resulting simulation is going to be, this is also the reason why we see some unused areas.
2. The definition of our design's aspect ratio is currently a perfect square. While it won't cause a major impact, however, in some cases, it does make a difference.
3. Tools/automated processes are not perfect.

Therefore, the dimension is (257um x 254um) – according to the ruler. In addition, we can tell that even though the area isn't perfectly in sync with what we had in table 1, the stdcell_area in Fig. 12 is exactly the same with design compiler's output.

C. Layout design at density = 1.0 (extra measurements)

To prove the concept above, if the density is equal to 1, the resulting area for this design would be the same with what we had in table 1. This proof is shown in the figure below:

```
stdCell: 30577 single + 0 double + 0 multi
Total standard cell length = 42.1082 (mm), area = 0.0590 (mm^2)
Average module density = 1.000.
Density for the design = 1.000.
       = stdcell_area 221622 (58951 um^2) / alloc_area 221613 (58949 um^2).
Pin Density = 0.553.
            = total # of pins 122483 / total Instance area 221622.
```

Figure 13. *Encounter's netlist – at density = 1.0*

We see from Fig. 13 that the resulting area turns out to be 58949um². In addition, the number of cells remain the same (when compared to when density = 0.9, and in table 1). ***With density = 1, we can see that the resulting outcome is much closer to what we had in table 1.*** Observe the figures below for design layout at density = 1:

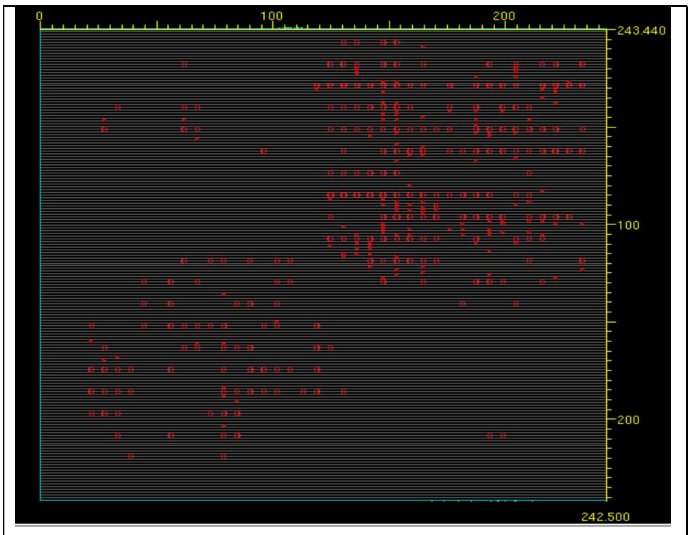

Figure 14. *Encounter simulation of Tile's floorplan, d = 1.0*

Below are the design floorplans with and without wiring/vias:

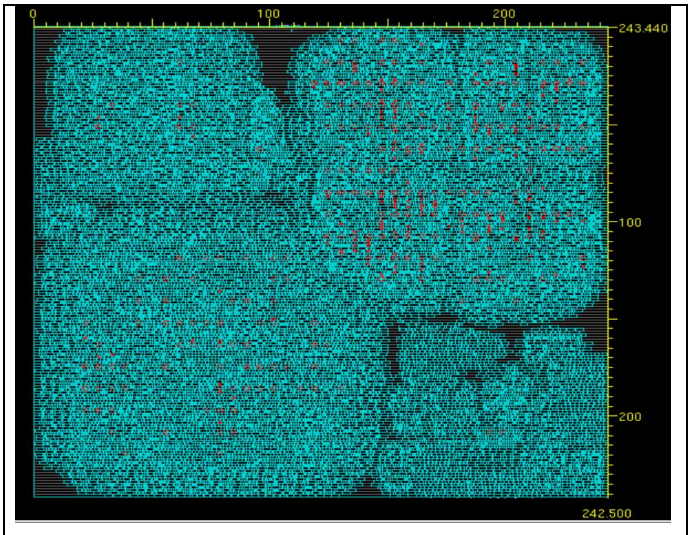

Figure 15. *Encounter – Tile's physical view no wires and vias*

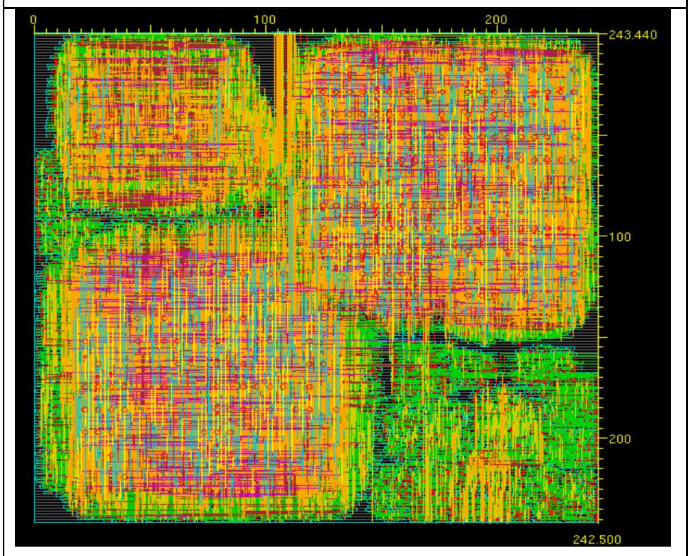

Figure 16. *Encounter – Tile's physical view wires and vias*

Ultimately, density = 1 provides a better result, and thus, we have the following parameters:

Table 3. *Encounter's cells and area statistics*

Density	Cells	Area (um^2)
1	30577	58949

While density = 1 gives us better design area, we would not want to use a density of 1 for an automated process. This is because when d = 1 for a large network like this, there would be many violations (refer to section 4 for nano-route of d = 1)

III. DETAILED ROUTING (STEP 6 – AT DENSITY = 0.9):

A. Nano-routing output:

Figure 17. *Encounter – Tile's physical view with nano routing*

In this design, we have the following information about metal layers and routing within the log file generated by nano routing step (design has no DRC violation):

```
#  Resource Analysis:
#
#                Routing  #Avail      #Track     #Total     %Gcell
#  Layer       Direction   Track      Blocked      Gcell    Blocked
#  ---------------------------------------------------------------
#  Metal 1         H         1820          0         8010    96.22%
#  Metal 2         V         1353          0         8010     0.00%
#  Metal 3         H         1789          0         8010     0.00%
#  Metal 4         V          901          0         8010     0.00%
#  Metal 5         H          893          0         8010     0.00%
#  Metal 6         V          901          0         8010     0.00%
#  Metal 7         H          303          0         8010     0.00%
#  Metal 8         V          305          0         8010     0.00%
#  Metal 9         H          159          0         8010     0.00%
#  Metal 10        V          152          0         8010     0.00%
#  ---------------------------------------------------------------
#  Total                     8576       0.00%  80100      9.62%
```

Figure 18. *Tile's nano routing resource analysis*

```
#  Congestion Analysis: (blocked Gcells are excluded)
#
#                OverCon       OverCon       OverCon       OverCon
#                 #Gcell        #Gcell        #Gcell        #Gcell    %Gcell
#    Layer         (1-3)         (4-7)         (8-11)       (12-15)   OverCon
#  --------------------------------------------------------------------------
#   Metal 1      0(0.00%)      0(0.00%)      0(0.00%)      0(0.00%)   (0.00%)
#   Metal 2   3035(37.9%)   1108(13.8%)    306(3.82%)    35(0.44%)   (56.0%)
#   Metal 3   1348(16.8%)     82(1.02%)      0(0.00%)      0(0.00%)   (17.9%)
#   Metal 4   1128(14.1%)      0(0.00%)      0(0.00%)      0(0.00%)   (14.1%)
#   Metal 5    297(3.71%)      0(0.00%)      0(0.00%)      0(0.00%)   (3.71%)
#   Metal 6    250(3.12%)      0(0.00%)      0(0.00%)      0(0.00%)   (3.12%)
#   Metal 7     16(0.20%)      0(0.00%)      0(0.00%)      0(0.00%)   (0.20%)
#   Metal 8     10(0.12%)      0(0.00%)      0(0.00%)      0(0.00%)   (0.12%)
#   Metal 9      0(0.00%)      0(0.00%)      0(0.00%)      0(0.00%)   (0.00%)
#  Metal 10      0(0.00%)      0(0.00%)      0(0.00%)      0(0.00%)   (0.00%)
#  --------------------------------------------------------------------------
#     Total   6084(8.34%)   1190(1.63%)    306(0.42%)    35(0.05%)   (10.4%)
#
#  The worst congested Gcell overcon (routing demand over resource in number of tracks) = 15
#
#Complete Global Routing.
#Total wire length = 678797 um.
#Total half perimeter of net bounding box = 453524 um.
#Total wire length on LAYER metal1 = 145 um.
#Total wire length on LAYER metal2 = 97203 um.
#Total wire length on LAYER metal3 = 200748 um.
#Total wire length on LAYER metal4 = 114630 um.
#Total wire length on LAYER metal5 = 99113 um.
#Total wire length on LAYER metal6 = 101389 um.
#Total wire length on LAYER metal7 = 25483 um.
#Total wire length on LAYER metal8 = 22871 um.
#Total wire length on LAYER metal9 = 9793 um.
#Total wire length on LAYER metal10 = 7421 um.
#Total number of vias = 218309
```

Figure 19. *Tile's nano routing congestion analysis*

From Fig. 18 and 19, we can see that there are 10 layers being used in total. However, while layer 1 has the widest track (largest available track number in Fig. 18), it does not have the highest congestion (i.e: routing). Below is the image of the topmost layer and the bottom most layer (i.e: metal 1 and metal 10 respectively):

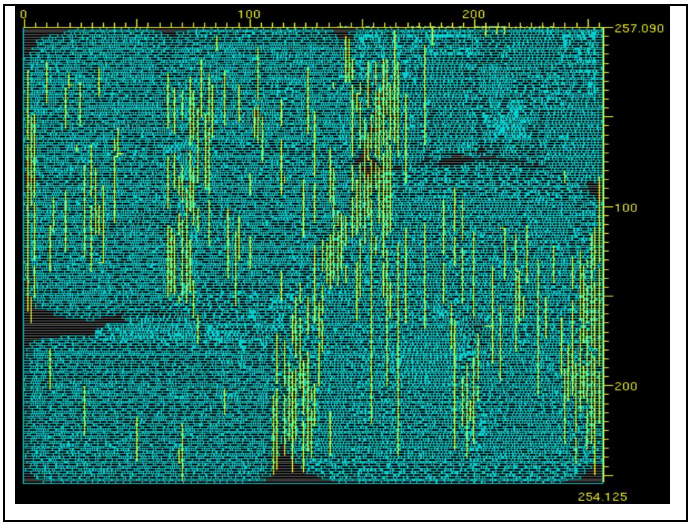

Figure 20. *Tile's bottommost metal layer (metal 10)*

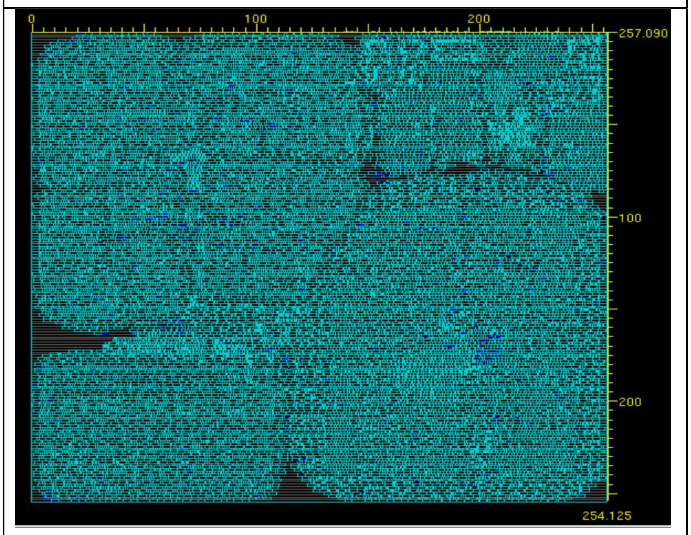

Figure 21. *Tile's topmost layer (layer 1)*

From the above information, we can conclude the followings:

Table 4. *Encounter's nano routing analysis*

Density	Metal layers	Most routing (highest congestion)
0.9	10	Layer 2 (metal 2)

In my opinion, it doesn't make sense when layer 2 has more routing than that of layer 1, and in this case – a lot more than layer 1 (Refer to Fig. 19's complete global routing analysis). ***Since I thought that layer 1 seems to have the highest tracks usage within the entire network.*** So, it is a little weird that layer 2 would have the highest congestion instead of 1. ***Ultimately, layer 2 has the highest congestion, therefore, highest routing.***

B. Metal Width measurements

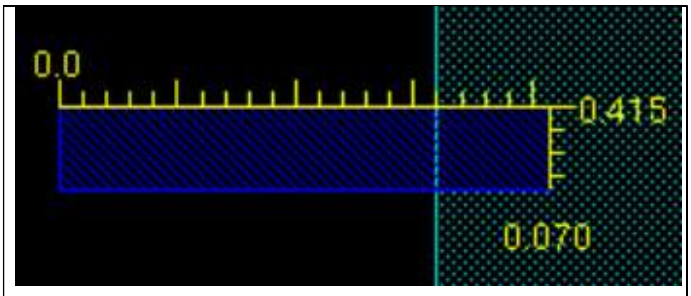

Figure 22. *Metal Layer 1 measurements*

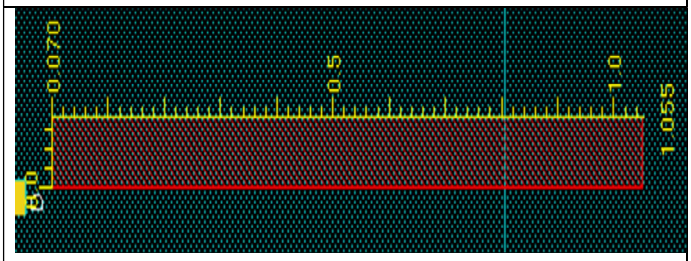

Figure 23. *Metal Layer 2 measurements*

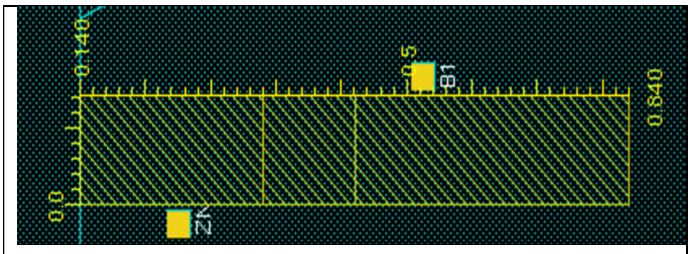

Figure 24. *Layer 4 measurements*

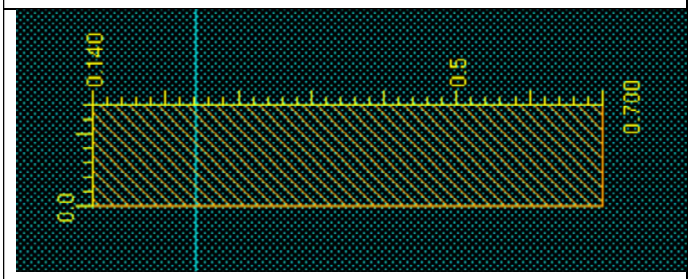

Figure 25. *Layer 6 measurements*

From the above figures, we have the following measurements:

Table 5. *Encounter's width measurements*

Density	Metal layer position	Width
0.9	1	0.07
0.9	2	0.07
0.9	4	0.14
0.9	6	0.14

By only viewing each metal individually, I was able to measure the metal's width by zooming into the layout and use ruler to estimate the metal's width. Since the length is dependent on how far apart some cells are, they could be varied, which is why we only want to measure the width. ***In addition, we can see that metal 1 and 2 has identical width, similarly metal 4 and 6 has identical width***. From lecture, we have:

Layer	t (nm)	w (nm)	s (nm)	pitch (nm)
M9	7 μm	17.5 μm	13 μm	30.5 μm
M8	720	400	410	810
M7	504	280	280	560
M6	324	180	180	360
M5	252	140	140	280
M4	216	120	120	240
M3	144	80	80	160
M2	144	80	80	160
M1	144	80	80	160

Figure 26. *Metal's width lecture 7 slide 3*

Compare Fig. 26 to table 5, we can see that M1's width and M2's width should be equal to each other, whereas M4 and M6 should not. The inconsistency could be due to the automated process not a perfect tool, and this is a large network which could also impact the program's quality.

C. If 1mm x 1mm chip:

First, let's assume that our tile is (257 um x 254um) – where density = 0.9 (as per measured by ruler in section 2 part a, b). Convert this to mm, we have tile's area of 0.257mm x 0.254mm. Therefore, we have the following calculations:

Table 6. *How many tiles on can be fitted on chip?*

Tile's dimension (mm x mm)	0.243 x 0.242	
Chip's dimension (mm x mm)	1 x 1	
Tile's area (mm^2)	Chip's area (mm^2)	How many can be fitted on chip?
0.0652	1	15.3

Table 6 shows that the number of tiles that can be fitted on the given 1mm x 1mm chip is around 15 tiles. The formula used for this was: # tiles = chip's area / tile's area

IV. DETAILED ROUTING (STEP EXTRA – AT DENSITY = 1):

A. Nano-routing output:

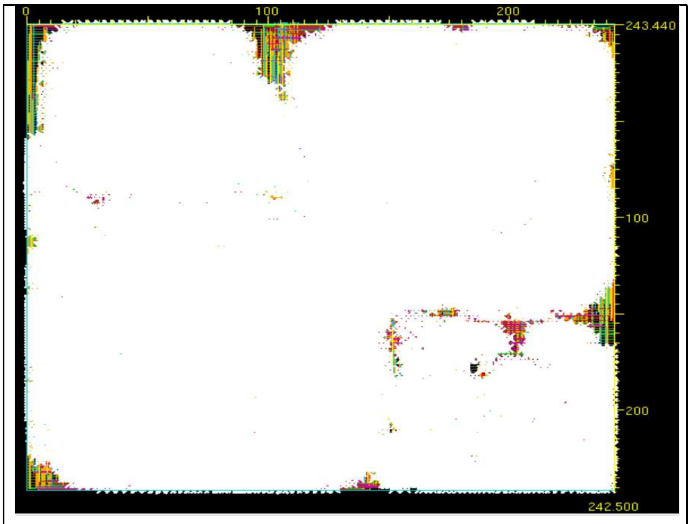

Figure 27. *Encounter – Tile's physical view with nano routing*

We see that when density = 1, our design has a lot of DRC violations (which are represented as the white colored areas):

```
#  Resource Analysis:
#
#                Routing  #Avail      #Track     #Total     %Gcell
#  Layer       Direction   Track      Blocked      Gcell    Blocked
#  ---------------------------------------------------------------
#  Metal 1         H         1730          0         7225    94.13%
#  Metal 2         V         1281          0         7225     0.00%
#  Metal 3         H         1700          0         7225     0.00%
#  Metal 4         V          853          0         7225     0.00%
#  Metal 5         H          849          0         7225     0.00%
#  Metal 6         V          853          0         7225     0.00%
#  Metal 7         H          288          0         7225     0.00%
#  Metal 8         V          289          0         7225     0.00%
#  Metal 9         H          151          0         7225     0.00%
#  Metal 10        V          144          0         7225     0.01%
#  ---------------------------------------------------------------
#  Total                     8138       0.00%  72250      9.41%
#
```

Figure 28. *Tile's nano routing resource analysis*

```
#  Congestion Analysis: (blocked Gcells are excluded)
#
#                 OverCon      OverCon      OverCon      OverCon
#                  #Gcell       #Gcell       #Gcell       #Gcell    %Gcell
#    Layer          (1-4)        (5-9)       (10-14)      (15-19)   OverCon
#  ------------------------------------------------------------------------
#   Metal 1       0(0.00%)     0(0.00%)     0(0.00%)     0(0.00%)   (0.00%)
#   Metal 2    3204(44.3%)  1315(18.2%)   287(3.97%)    36(0.50%)  (67.0%)
#   Metal 3    2579(35.7%)   110(1.52%)     0(0.00%)     0(0.00%)  (37.2%)
#   Metal 4    1880(26.0%)     0(0.00%)     0(0.00%)     0(0.00%)  (26.0%)
#   Metal 5     689(9.54%)     0(0.00%)     0(0.00%)     0(0.00%)  (9.54%)
#   Metal 6     420(5.81%)     0(0.00%)     0(0.00%)     0(0.00%)  (5.81%)
#   Metal 7      42(0.58%)     0(0.00%)     0(0.00%)     0(0.00%)  (0.58%)
#   Metal 8      18(0.25%)     0(0.00%)     0(0.00%)     0(0.00%)  (0.25%)
#   Metal 9       1(0.01%)     0(0.00%)     0(0.00%)     0(0.00%)  (0.01%)
#  Metal 10       6(0.08%)     0(0.00%)     0(0.00%)     0(0.00%)  (0.08%)
#  ------------------------------------------------------------------------
#     Total    8839(13.4%)  1425(2.16%)    287(0.43%)    36(0.05%)  (16.0%)
#
#  The worst congested Gcell overcon (routing demand over resource in number of tracks) = 19
#
#Complete Global Routing.
#Total wire length = 635546 um.
#Total half perimeter of net bounding box = 428317 um.
#Total wire length on LAYER metal1 = 304 um.
#Total wire length on LAYER metal2 = 83617 um.
#Total wire length on LAYER metal3 = 185997 um.
#Total wire length on LAYER metal4 = 103994 um.
#Total wire length on LAYER metal5 = 101501 um.
#Total wire length on LAYER metal6 = 99416 um.
#Total wire length on LAYER metal7 = 25650 um.
#Total wire length on LAYER metal8 = 19411 um.
#Total wire length on LAYER metal9 = 9362 um.
#Total wire length on LAYER metal10 = 6293 um.
#Total number of vias = 217951
```

Figure 29. *Tile's nano routing congestion analysis*

From Fig. 28 and 29, we can see that there are 10 layers being used in total. However, while layer 1 has the widest track (largest available track number in Fig. 29), it does not have the highest congestion (i.e: routing). Below is the image of the topmost layer and the bottom most layer (i.e: metal 1 and metal 10 respectively):

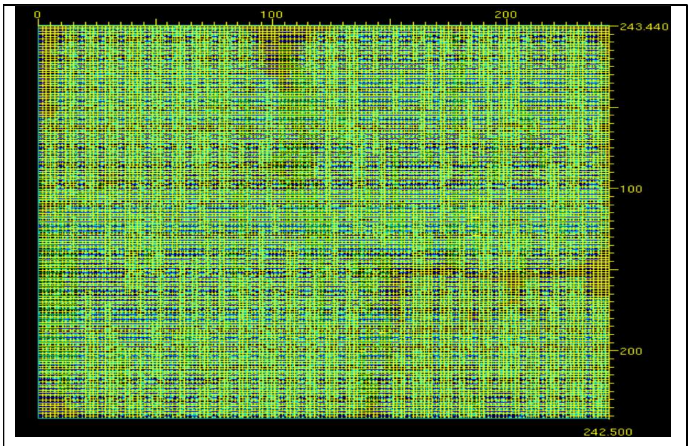

Figure 30. *Tile's bottommost metal layer (metal 10)*

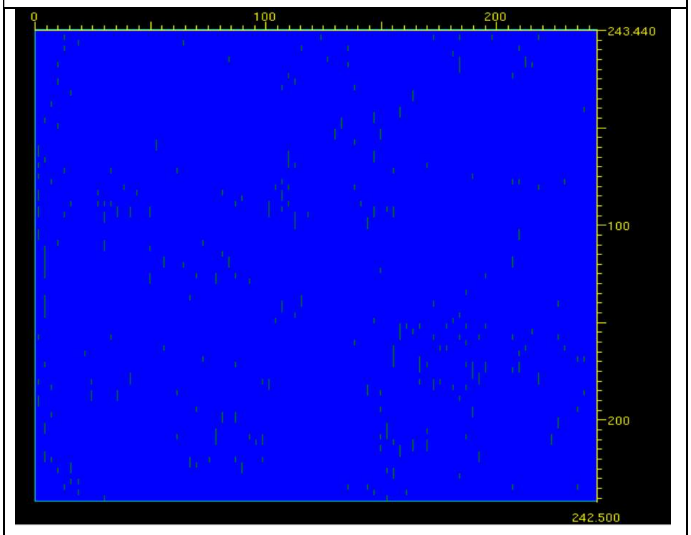

Figure 31. *Tile's topmost layer (layer 1)*

From the above information, we can conclude the followings:

Table 7. *Encounter's nano routing analysis*

Density	Metal layers	Most routing (highest congestion)
1	10	Layer 2 (metal 2)

In my opinion, it doesn't make sense when layer 2 has more routing than that of layer 1, and in this case – a lot more than layer 1 (Refer to Fig. 29's complete global routing analysis). ***Since I thought that layer 1 seems to have the highest tracks usage within the entire network.*** So, it is a little weird that layer 2 would have the highest congestion instead of 1. ***Ultimately, layer 2 has the highest congestion, therefore, highest routing.***

B. Metal Width measurements

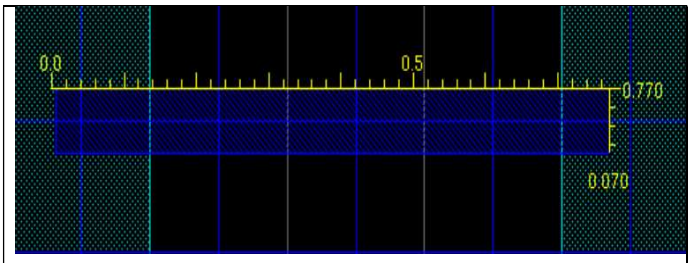

Figure 32. *Metal Layer 1 measurements*

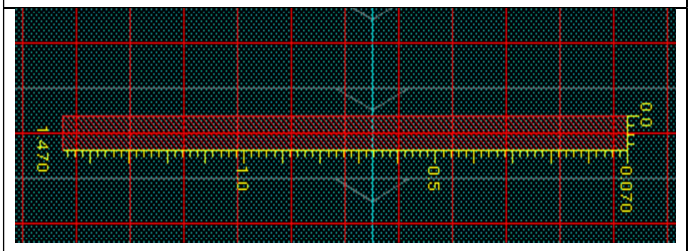

Figure 33. *Metal Layer 2 measurements*

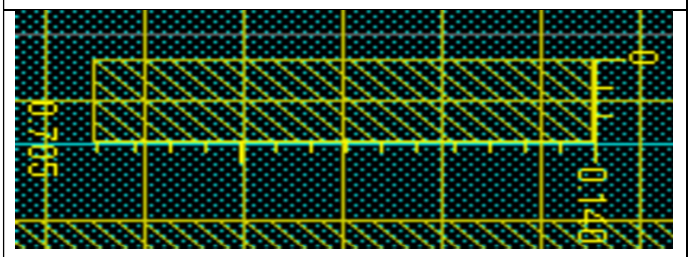

Figure 34. *Layer 4 measurements*

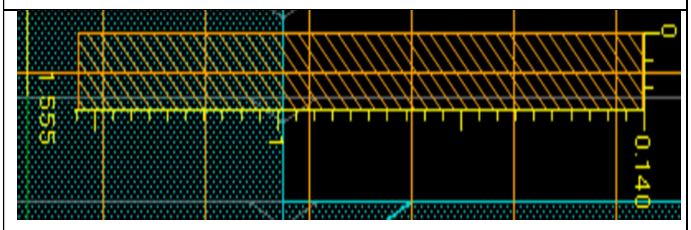

Figure 35. *Layer 6 measurements*

From the above figures, we have the following measurements:

Table 8. *Encounter's width measurements*

Density	Metal layer position	Width
1	1	0.07
1	2	0.07
1	4	0.14
1	6	0.14

By only viewing each metal individually, I was able to measure the metal's width by zooming into the layout and use ruler to estimate the metal's width. Since the length is dependent on how far apart some cells are, they could be varied, which is why we only want to measure the width. ***In addition, we can see that metal 1 and 2 has identical width, similarly metal 4 and 6 has identical width***. From lecture, we have:

Compare Fig. 26 to table 8, we can see that M1's width and M2's width should be equal to each other, whereas M4 and M6 should not. The inconsistency between metal 4 and 6 could be coming from the aspect ratio of 1 caused too much of a strain on the system and thus, making encounter to operate under extreme stress and causing it to provide incorrect results at some places.

C. Why to not use density = 1

As we see within this section, while there isn't much difference in terms of metal widths (for 1, 2, 4 and 6), number of layers and highest congestion layer. However, the design is riddled with DRC violations (as shown in Fig. 27). ***We can also tell that it is wrong by comparing the density of Fig. 20 and 21 vs Fig. 30 and 31, since the density of metal used in Fig. 30 and 31 is too crowded, there are errors bound to occur (being too close, etc) and that (density = 1) caused a total of 25984 DRC violations. Whereas density = 0.9 yields 0 DRC violation.***

V. SUMMARY

In conclusion, if we want to pick a density for the design, I recommend using d = 0.9. While density = 0.9 have a slightly worse (larger) tile's area than that of density = 1. However, density = 0.9 allows encounter a little more room to breath (i.e: more space, and less strict constraints), which makes it to be a little more accurate. While we want the smallest tile possible, depending on whether or not the design has any DRC violations is also an important factor in determining which specifications to make.