RTL TO GDSII FLOW OF RADIX-4 BOOTH MULTIPLIER

CADENCE

By:

Umesh Kanna K B

College Engineering Guindy, Anna University

Chennai-25.

As I already exposed the Design concept of Radix-4 Booth multiplier and its algorithm , Let’s straightly go into the part of complete RTL to GDSII flow using Cadence.

INTRODUCTION

The RTL to GDSII flow is the process of transforming a high-level description of a digital design into a physical layout that can be sent to a semiconductor foundry for fabrication. It begins with the RTL design, where the functionality of the chip is described using hardware description languages like Verilog or VHDL. This stage focuses on defining the logic behavior of the circuit, such as data processing, arithmetic, and control operations, without considering physical constraints.

Once the RTL code is written, it undergoes simulation and verification to ensure it works as intended. This involves creating testbenches to simulate the design's behavior and check for logical correctness. After functional verification, the design moves to synthesis, where the RTL code is converted into a gate-level netlist. This netlist describes the design using standard logic gates and flip-flops, taking into account timing, area, and power constraints to ensure the design can be physically realized.

Next comes floorplanning, where the physical layout of the chip is planned. This includes defining regions for various blocks, such as the core and input/output pads, and specifying the chip’s boundaries. After floorplanning, the design enters the placement stage, where the standard cells are physically arranged on the chip. The objective is to minimize wire lengths and optimize performance while respecting area constraints.

Clock Tree Synthesis (CTS) follows, which ensures that the clock signal is distributed evenly across the chip to minimize clock skew. This is critical for ensuring synchronization between different parts of the chip. Then comes the routing stage, where the placed cells are interconnected using metal layers. The goal here is to ensure that all signals are properly routed while minimizing delays and adhering to design rules.

Parasitic extraction is the next step, where the resistive and capacitive effects of the metal wires are calculated. These parasitics affect signal delay and must be taken into account for accurate timing analysis. Static Timing Analysis (STA) follows, which checks whether the timing constraints, such as setup and hold times, are met throughout the design. If violations are found, adjustments may be made to the placement or routing.

Power analysis is then performed to estimate the total power consumption of the design, including both dynamic and static power. The goal is to identify areas where power consumption can be optimized. Once the power analysis is complete, physical verification takes place. This involves two checks: Design Rule Check (DRC), which ensures the layout adheres to the foundry’s design rules, and Layout vs. Schematic (LVS), which verifies that the physical layout matches the original schematic design.

Finally, the design is exported to the GDSII format, which is the standard file format used for manufacturing the chip. The GDSII file contains all the details of the chip’s physical layout and is sent to the foundry for the fabrication process. This completes the RTL to GDSII flow, transforming a high-level design into a physical chip ready for manufacturing.

PROCEDURE:

Create a new folder in desktop → right click → open terminal. In the terminal enter the following commands: i) $ csh ii) $ source /home/install/cshrc iii) $ gedit

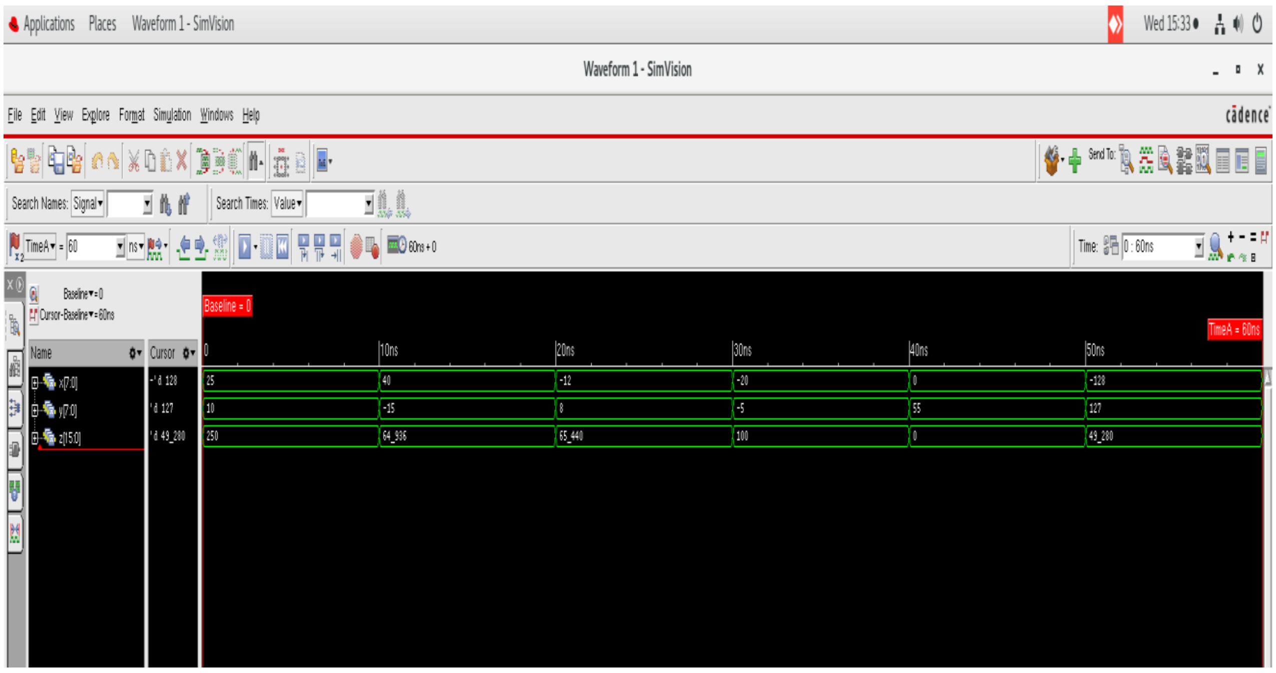
The gedit window opens. Write the Verilog code and save it as .v. Click new and write the code for testbench file and save it as tb.v. Now, in the folder where the codes are present, right click → open terminal and write the following commands. i) $ csh ii) $ source /home/install/cshrc iii) nclaunch , Nclaunch window opens.

Select multiple step option. Go to file → set design directory → select create cds.lib file→ click save. Tick the box don't include any libraries and click ok. Now select both the files and select launch Verilog compiler with current selection. Proceed if no errors found. Select the tb file under worklib and click launch elaborator with current selection.

Proceed to the next step if there are no errors. Now select the module file under snapshots and click launch simulator with current selection. The simvision window opens. In this window click the tb file, variables used in that module file will be listed.

 Select all the variables and click send to target waveform. Waveform window opens. In the waveform window, give run button. The ouput waveform appears.Verify it.

SIMULATION WAVEFORM:

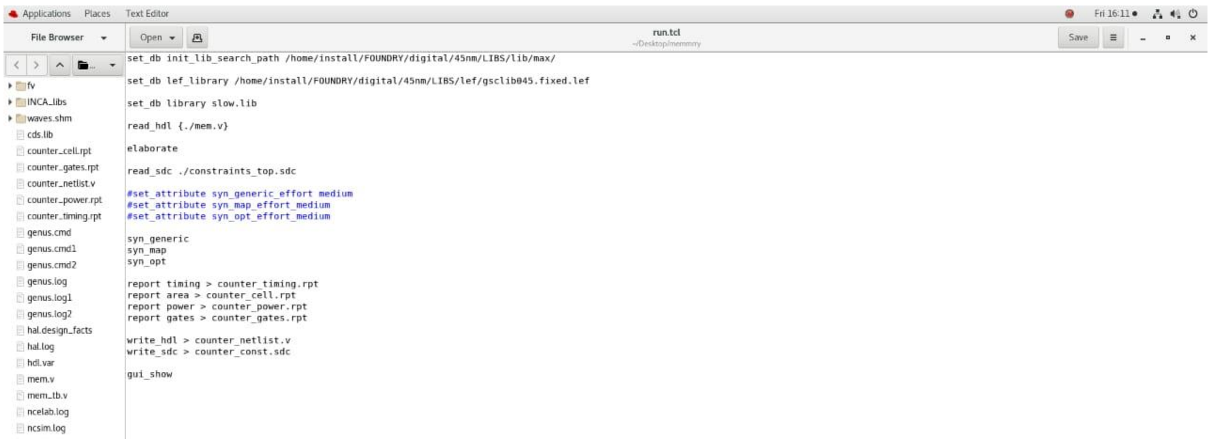


```
x[7:0]   -'d 128   25    40      -12     -20   0    -128
y[7:0]   'd 127    10    -15     8       -5    55   127
z[15:0]  'd 49_280 250   64_936  65_440  100   0    49_280
```

3

Include run.tcl and constraints.sdc file to the folder.

TCL file:



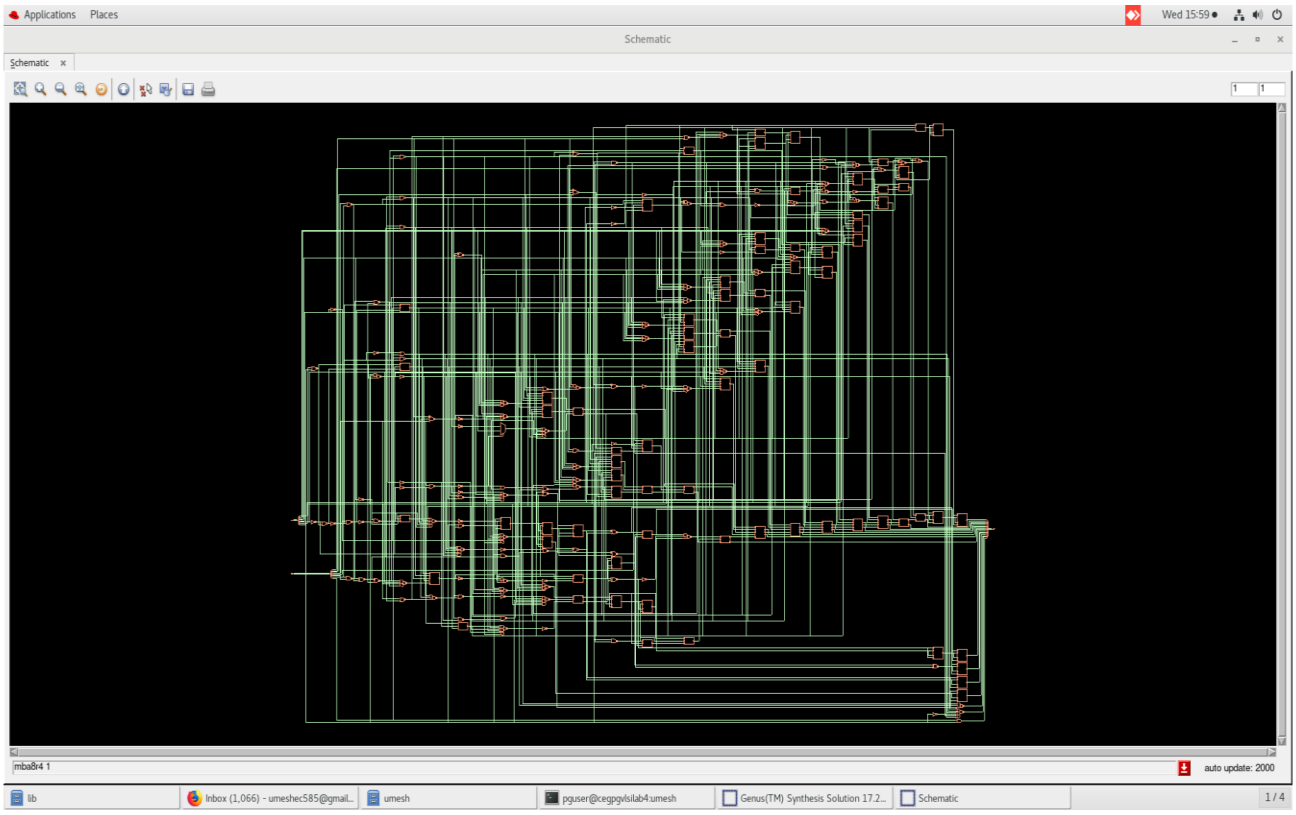
```
set_db init_lib_search_path /home/install/FOUNDRY/digital/45nm/LIBS/lib/max/

set_db lef_library /home/install/FOUNDRY/digital/45nm/LIBS/lef/gsclib045.fixed.lef

set_db library slow.lib

read_hdl {./mem.v}

elaborate

read_sdc ./constraints_top.sdc

#set_attribute syn_generic_effort medium
#set_attribute syn_map_effort_medium
#set_attribute syn_opt_effort_medium

syn_generic
syn_map
syn_opt

report timing > counter_timing.rpt
report area > counter_cell.rpt
report power > counter_power.rpt
report gates > counter_gates.rpt

write_hdl > counter_netlist.v
write_sdc > counter_const.sdc

gui_show
```

SDC file:

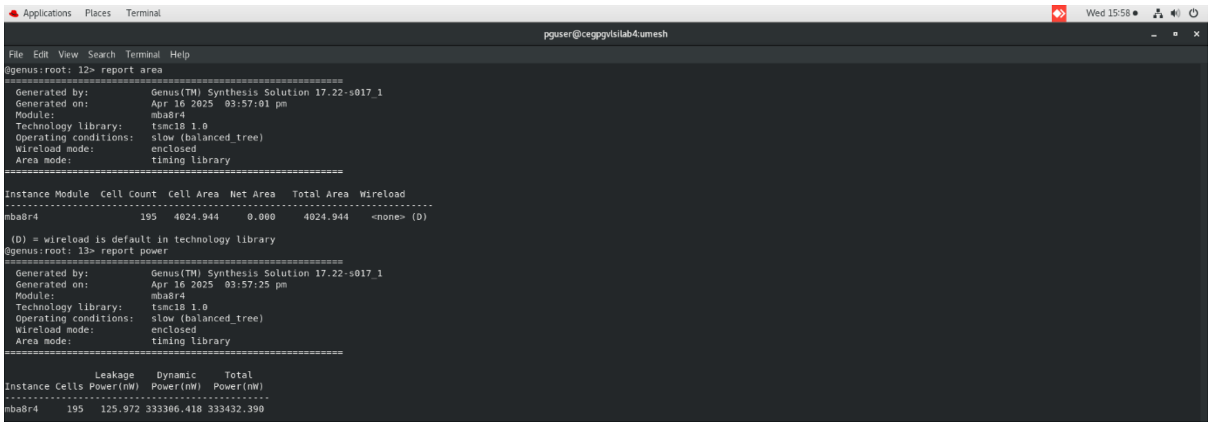
```
create_clock -name clk -period 10 -waveform {0 5} [get_ports "clk"]
set_clock_transition -rise 0.1 [get_clocks "clk"]
set_clock_transition -fall 0.1 [get_clocks "clk"]
set_clock_uncertainty 0.01 [get_ports "clk"]
set_input_delay -max 1.0 [get_ports "rst"] -clock [get_clocks "clk"]
set_output_delay -max 1.0 [get_ports "count"] -clock [get_clocks "clk"]
```

Open Terminal . Genus -> source run.tcl. Schematic, Area, Power, Timing Reports will be generated . RTL PART Completed.

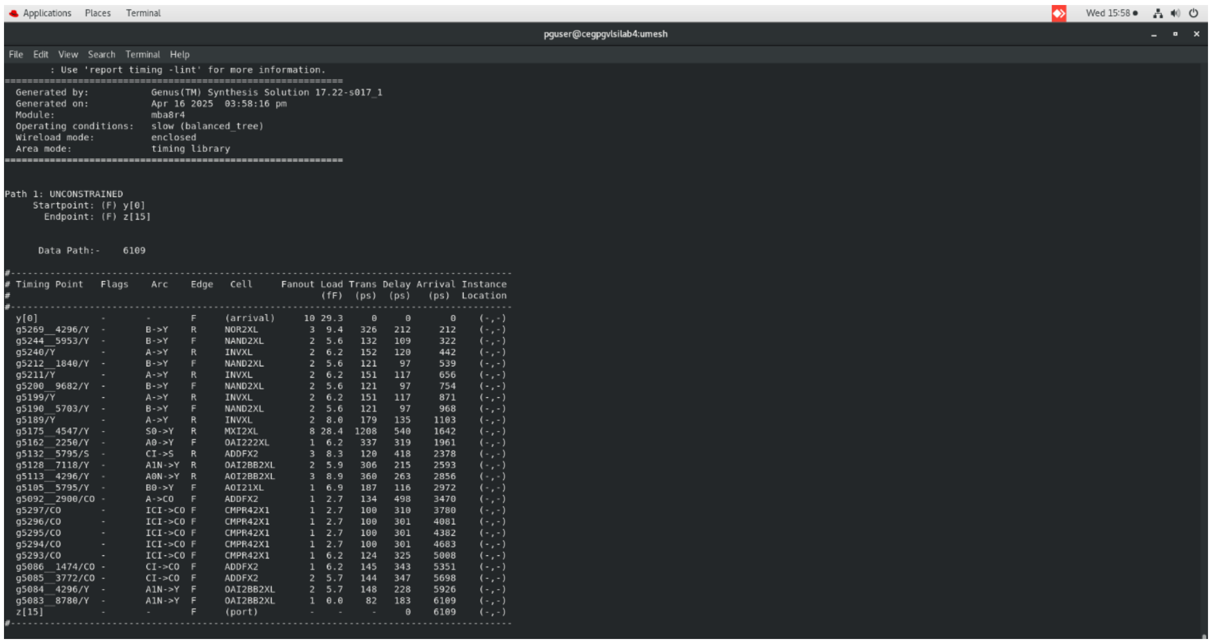
SCHEMATIC:


Applications
Places
Wed 15:59
Schematic
mba8r4 1
auto update: 2000
lib
Inbox (1,066) - umeshec585@gmail...
umesh
pguser@cegpgvlsilab4:umesh
Genus(TM) Synthesis Solution 17.2...
Schematic
1 / 4

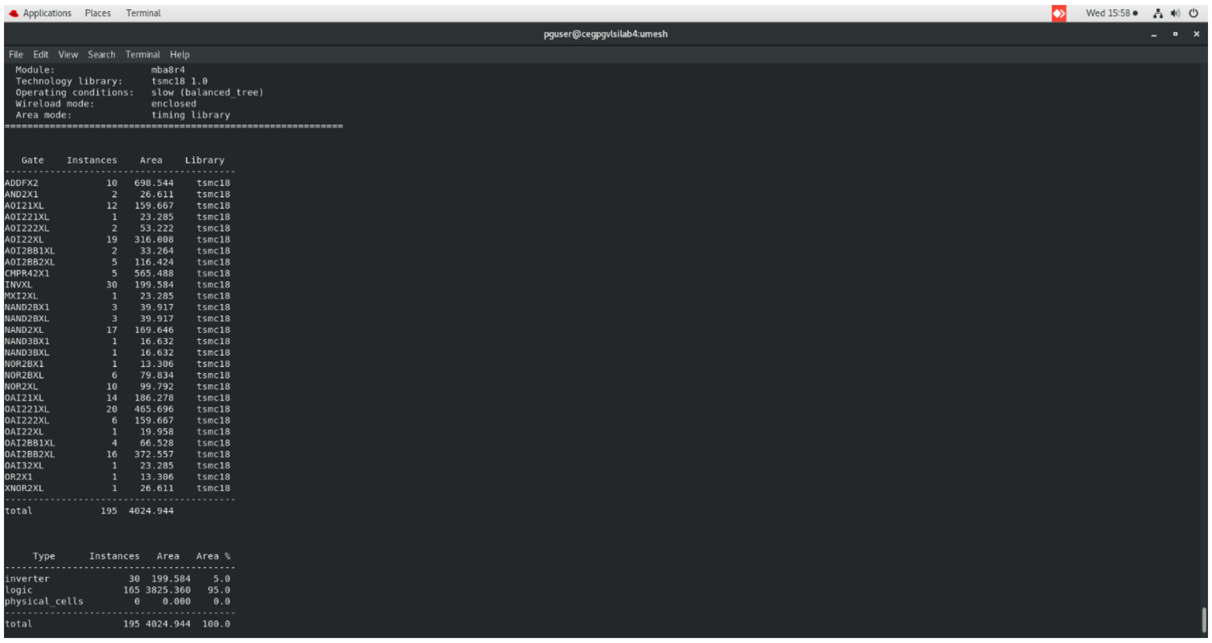
AREA, POWER, TIMING UNCONSTRAINED REPORTS:



```
@genus:root: 12> report area
============================================================
  Generated by:           Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:           Apr 16 2025  03:57:01 pm
  Module:                 mba8r4
  Technology library:     tsmc18 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

Instance Module  Cell Count  Cell Area  Net Area   Total Area  Wireload
--------------------------------------------------------------------------
mba8r4                  195   4024.944     0.000     4024.944    <none> (D)

 (D) = wireload is default in technology library
@genus:root: 13> report power
============================================================
  Generated by:           Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:           Apr 16 2025  03:57:25 pm
  Module:                 mba8r4
  Technology library:     tsmc18 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

                Leakage    Dynamic     Total
Instance Cells Power(nW)  Power(nW)  Power(nW)
-----------------------------------------------
mba8r4     195   125.972 333306.418 333432.390
```



```
        : Use 'report timing -lint' for more information.
============================================================
  Generated by:           Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:           Apr 16 2025  03:58:16 pm
  Module:                 mba8r4
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


Path 1: UNCONSTRAINED
     Startpoint: (F) y[0]
       Endpoint: (F) z[15]


      Data Path:-    6109

#------------------------------------------------------------------------------------------
# Timing Point   Flags    Arc    Edge   Cell       Fanout Load Trans Delay Arrival Instance
#                                                         (fF)  (ps)  (ps)    (ps) Location
#------------------------------------------------------------------------------------------
  y[0]           -        -      F      (arrival)      10 29.3     0     0       0    (-,-)
  g5269__4296/Y  -        B->Y   R      NOR2XL          3  9.4   326   212     212    (-,-)
  g5244__5953/Y  -        B->Y   F      NAND2XL         2  5.6   132   109     322    (-,-)
  g5240/Y        -        A->Y   R      INVXL           2  6.2   152   120     442    (-,-)
  g5212__1840/Y  -        B->Y   F      NAND2XL         2  5.6   121    97     539    (-,-)
  g5211/Y        -        A->Y   R      INVXL           2  6.2   151   117     656    (-,-)
  g5200__9682/Y  -        B->Y   F      NAND2XL         2  5.6   121    97     754    (-,-)
  g5199/Y        -        A->Y   R      INVXL           2  6.2   151   117     871    (-,-)
  g5190__5703/Y  -        B->Y   F      NAND2XL         2  5.6   121    97     968    (-,-)
  g5189/Y        -        A->Y   R      INVXL           2  8.0   179   135    1103    (-,-)
  g5175__4547/Y  -        S0->Y  R      MXI2XL          8 28.4  1208   540    1642    (-,-)
  g5162__2250/Y  -        A0->Y  F      OAI222XL        1  6.2   337   319    1961    (-,-)
  g5132__5795/S  -        CI->S  R      ADDFX2          3  8.3   120   418    2378    (-,-)
  g5128__7118/Y  -        A1N->Y R      OAI2BB2XL       2  5.9   306   215    2593    (-,-)
  g5113__4296/Y  -        A0N->Y R      AOI2BB2XL       3  8.9   360   263    2856    (-,-)
  g5105__5795/Y  -        B0->Y  F      AOI21XL         1  6.9   187   116    2972    (-,-)
  g5092__2900/CO -        A->CO  F      ADDFX2          1  2.7   134   498    3470    (-,-)
  g5297/CO       -        ICI->CO F     CMPR42X1        1  2.7   100   310    3780    (-,-)
  g5296/CO       -        ICI->CO F     CMPR42X1        1  2.7   100   301    4081    (-,-)
  g5295/CO       -        ICI->CO F     CMPR42X1        1  2.7   100   301    4382    (-,-)
  g5294/CO       -        ICI->CO F     CMPR42X1        1  2.7   100   301    4683    (-,-)
  g5293/CO       -        ICI->CO F     CMPR42X1        1  6.2   124   325    5008    (-,-)
  g5086__1474/CO -        CI->CO F      ADDFX2          1  6.2   145   343    5351    (-,-)
  g5085__3772/CO -        CI->CO F      ADDFX2          2  5.7   144   347    5698    (-,-)
  g5084__4296/Y  -        A1N->Y F      OAI2BB2XL       2  5.7   148   228    5926    (-,-)
  g5083__8780/Y  -        A1N->Y F      OAI2BB2XL       1  0.0    82   183    6109    (-,-)
  z[15]          -        -      F      (port)          -    -     -     0    6109    (-,-)
#------------------------------------------------------------------------------------------
```



```
  Module:                 mba8r4
  Technology library:     tsmc18 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================


   Gate     Instances    Area     Library
-----------------------------------------
ADDFX2             10   698.544    tsmc18
AND2X1              2    26.611    tsmc18
AOI21XL            12   159.667    tsmc18
AOI221XL            1    23.285    tsmc18
AOI222XL            2    53.222    tsmc18
AOI22XL            19   316.008    tsmc18
AOI2BB1XL           2    33.264    tsmc18
AOI2BB2XL           5   116.424    tsmc18
CMPR42X1            5   565.488    tsmc18
INVXL              30   199.584    tsmc18
MXI2XL              1    23.285    tsmc18
NAND2BX1            3    39.917    tsmc18
NAND2BXL            3    39.917    tsmc18
NAND2XL            17   169.646    tsmc18
NAND3BX1            1    16.632    tsmc18
NAND3BXL            1    16.632    tsmc18
NOR2BX1             1    13.306    tsmc18
NOR2BXL             6    79.834    tsmc18
NOR2XL             10    99.792    tsmc18
OAI21XL            14   186.278    tsmc18
OAI221XL           20   465.696    tsmc18
OAI222XL            6   159.667    tsmc18
OAI22XL             1    19.958    tsmc18
OAI2BB1XL           4    66.528    tsmc18
OAI2BB2XL          16   372.557    tsmc18
OAI32XL             1    23.285    tsmc18
OR2X1               1    13.306    tsmc18
XNOR2XL             1    26.611    tsmc18
-----------------------------------------
total             195  4024.944


     Type      Instances   Area   Area %
-----------------------------------------
inverter              30  199.584    5.0
logic                165 3825.360   95.0
physical_cells         0    0.000    0.0
-----------------------------------------
total                195 4024.944  100.0
```

Create a new folder in the desktop. Then copy and paste the bcd_decimal.v file containing Verilog code and following files. ✓ Netlist file [fv_map.v] ✓ Constraint file[.xdc] ✓ Lef file[.lef] ✓ Technology file[.tf] ✓ Logical libraries[.lib] ✓ Physical libraries[xx nm]

LAUNCHING CADENCE TOOL SUITE: • Right click on the folder → Select open in terminal window. • In the terminal window, enter the following commands to invoke C;

• $ csh • $ source /home/install/cshrc Press Enter key • Type, "INNOVUS" and press Enter to open the innovus window.

• After the INNOVUS tool opens, first step is to import the design as shown • Add gate level netlist [.v] by clicking on the three dots icon under netlist Verilog . • Then select top cells:Auto assign.

• LEF Files – LEF files must be uploaded in a sequence and the sequence is tsl180l4.lef, tsl18fs120_scl.lef and then tsl18cio250_4lm.lef. • Power Net – VDD for Standard Cells and VDDO for IO, Ground Net – VSS for standard cells and VSSO for IO, CPF file is optional and not taken as input here.

IO Assignment file – This file is used for assigning the IO pins in a specific order. If this file is not used then tool will automatically assign the input output ports in convenient order. This file also places the IO pads and Corner cells. • View Definition File – This file is actually called Multi Mode Multi Corner (MMMC) view definition file. This file takes timing library files, Capacitance Tables and SDC files as inputs. Then creates Best and Worst case rc_corners for PVT analysis of the chip.

Also, creates Max and Min libraries for timing and delay. • Right click on library sets →new. • Name :max_timing→ library_file→slow.lib. click ok. repeat the same steps for • Name :min_timing→add fast.lib as library file. Click ok. • Right click on delay corners→new. • Name:min_delay→library sets→min_timing. Click ok. • Name :max_delay→library sets→max_timing . click ok. • Right click analysis view→select worst case→ click ok. • Right click on hold analysis view→select best case→click ok. • Power Constraint File – The Common Power Format (CPF) is a file that contains the information regarding power reduction techniques. For example, if multiple voltages are to be applied to reduce power then this should contain details about the multiple voltages.

• Once all the files given then next step is to Save this file import configuration in that folder as [.global]. This is because in future for successive trails the [.globals] file can be loaded instead of importing all the files again individually.

CREATING FLOORPLAN: • Go to floor plan→specify floorplan→set the value for core to left,right,top and bottom fields→click ok. • Tool automatically gives a floorplan using Core utilization factor. Here, Core to die distance is mentioned but core to IO boundary also can be mentioned here. In this case, IO pad length must be taken into account. Lets say Core to die distance = 10 for core area.

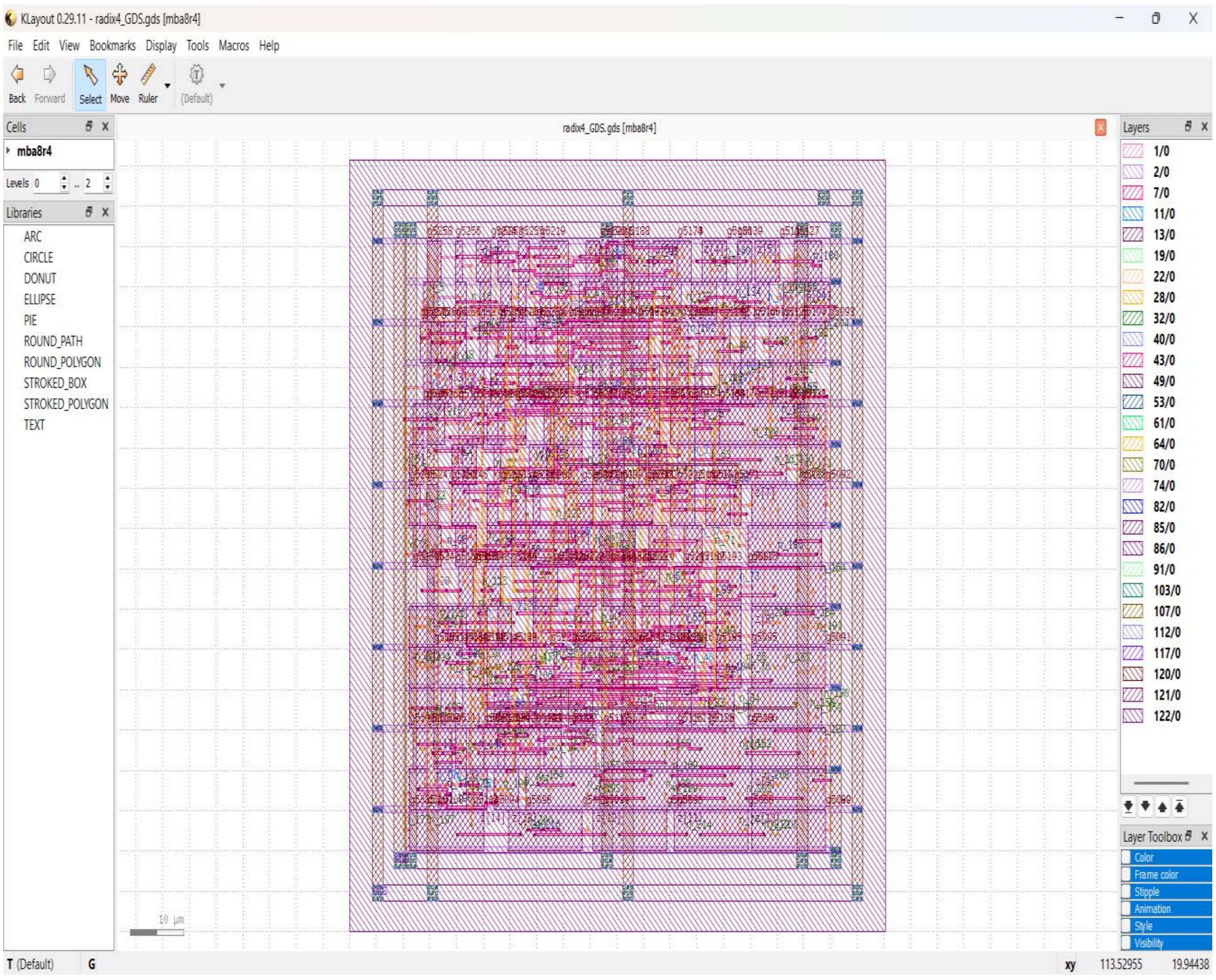
CREATING POWERPLANNING: • Go to power→ power planning → add ring →nets:VDD VSS . • Set width and spacing as 2 for all. → offset : 0.33 • Top & bottom : metal 5H • Right and left : metal 6 H. Click ok. • Power planning → Add stripes. • set nets:VDD VSS. • No. of sets :3 • Layers: Metal 6 • width and spacing:2

ROUTING AND PLACEMENT: • Choose Route – Special Route. • Special Routes connects the Power nets the standard cells. Here, retain all the options as it is in the image. Only select Metal as the bottom layer and TOP Metal as Top layer. • Place →Place standard cells→ Go to mode • Tick box place input and output pins → ok and finally click ok • Place → check placement and placement density can be viewed in terminal window. • Timing → Report timing→ select Pre-CTS→ ok. • Eco →optimize→select pre-CTS→ok.

TIMING ANALYSIS: • Again perform Timing analysis by following above steps. • Route →Nano Route→ Specify attributes →give nets: VDD VSS →ok • Route → Nano route→ route: • Timing driven set effect to 2. Set bottom layer 1: set top layer :9 →ok • Verify →verify geomentry →ok . • check for 0 violations. • Verify →Verify connectivity → ok. • Check for 0 violations.

EXPORTING FILE: • Go to file→GDS/OASIS. • Give name with .gds extension→click ok. • The layout can be viewed using k-layout software.

GDSII Layout in KLayout software:



YOU CAN SEE THE STEP BY STEP BACK-END FLOW WITHIN THIS BROADER VIEWOF .gds FILE: