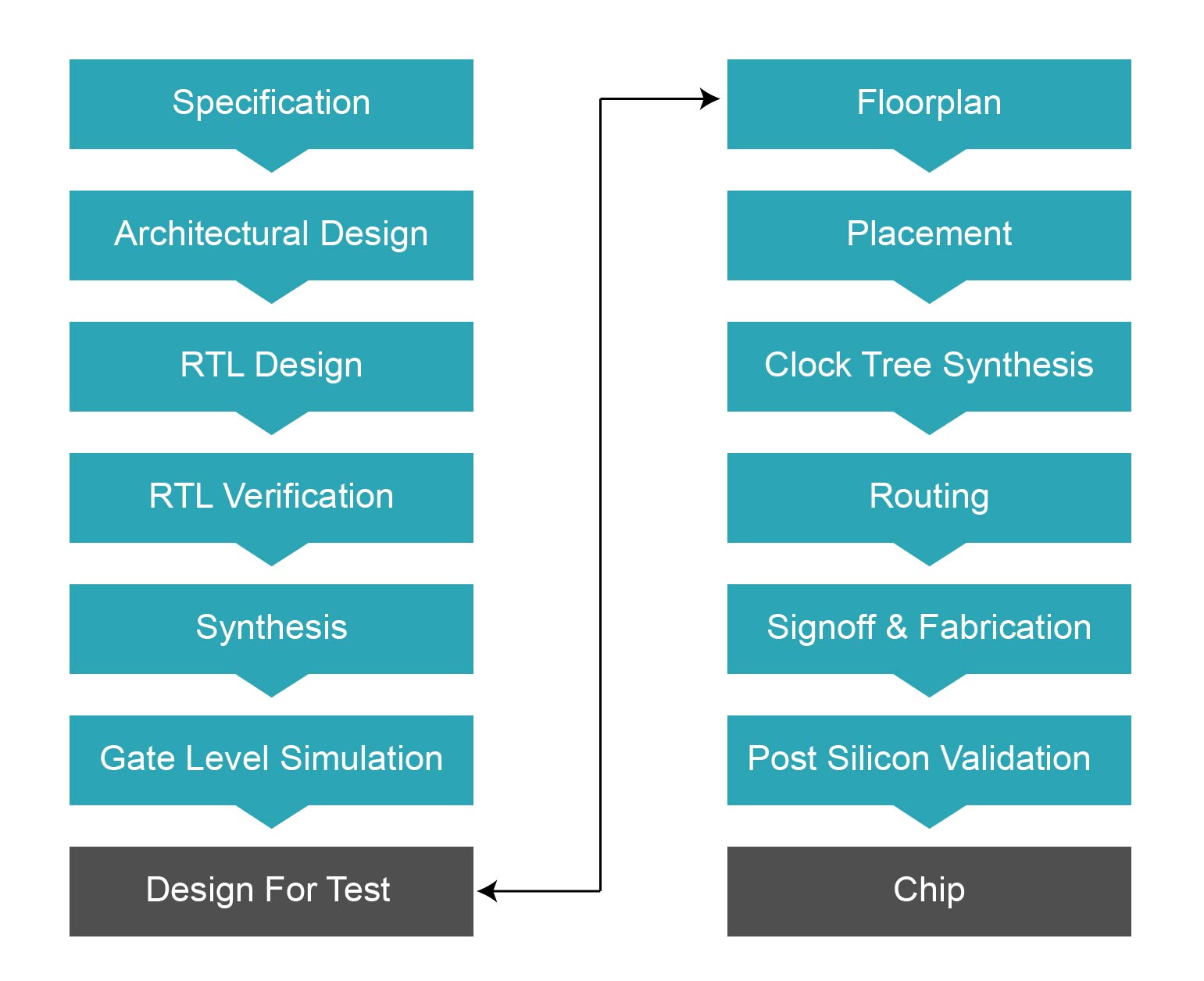
**report**

**Maven SILIcon (vlsi)**

**Physical design**

What is physical design?

* Physical design is a process of converting logical design(cells) to physical cells (placement of cells) on chip.
* Generally VLSI industry divided into two streams front end and backend.
* Where frontend is related to logic design of code using Xilinx and other software, which is called as DESIGN VERIFICATION.
* Frontend process involves different steps.
* Coming to backend here logic cells are converted to physical cells.
* **Design flow of {VLSI}**



Physical design starts from(BACKEND)

Gate level simulation to chip.

* **SPECIFICATION:**

It consist of hardware and software.

Hardware specifications are:

ASIC, FPGA, PLD.

Software specifications are:

Software codes.

* **ARCHITECTURE:**

It is generally a block diagram or an overview of design.

* **BEHAVIOURAL MODEL:**

To verify the functionality of a design we are using two design modules

**TLM – Transaction Level model**

**RTL- DUT (Design Under test)**

* **RTL CODING:**

Here data transferred from register to register.

* **VERIFICATION:**

What ever the code we written it needs to be check functionality by test bench code.

Here verification done in three levels:

**WHITE BOX VERIFICATION**

**SUBSYSTEM LEVEL**

**CHIP LEVEL** **(BLACK BOX)**

* **SYNTHESIS:**

Rtl code converted to gate level(net list).

To verify the functionality of netlist we use gate level simulation.

* **GATE LEVEL SIMULATION:**

Verification done in two ways:

* **Sequential equivalence checking**
* **Verifying functionality**
* **DFT INSERTION:**

Design for test here, we try to replace existing flipflops with scan flipflops.

Every scan flipflop consist mux has input.

Here we can operate chip in two modes:

Scan mode

Normal mode.

Here, we can identify errors also by using

**stuck at 0**

**stuck at 1 (faults).**

* **Floor planning:**

Here we need to think about that how efficiently we can use.

At the time of floor planning

We decide:

**What is the area reserved for IO pads?**

**Where to put each and every block?**

**Size of every block, IO pins connection.**

* **PLACEMENT:**

Determine the exact shape and position

(location) on the layout.

**TYPES:**

**Linear placement**

**2D placement.**

Main objective of placement is to reduce the interconnections and avoid congestions.

* **CLOCK TREE SYNTHESIS:**

The main aim of the process is to clock needs to be reach all the flipflops at same time without any delay.

* **ROUTING:**

Making the connections

Here we take care about no short circuit in the circuits and congestions.

* **STATIC TIMING ANALYSIS:**

It is one of the most critical process because timing is important among three design constraints.

Here it consist of setup time and hold time inorder to register the data into the flipflop.

It also consit of setup time and hold time

Violations.

**SETUP TIME VOILATION:**

If the data is changed during the setuptime then it is called setup time violation.

**HOLD TIME VOILATION:**

If the data is changed during holdtime then it is said to be hold time violation.

If the timing was not **ok** then we go back to the placement process.

* **PHYSICAL LAYOUT:**

It is simply stick diagrams.

* **DESIGN RULE CHECK:**

It tells about:

Space between metal layers.

Space between Diffusion layers.

Space between Polysilicon layers.

**LAYOUT VS SCHEMATIC:**

This is the last process of design flow where final routed netlist verify schematically and once again we perform equivalence checking.

**FINALLY WE RELEASED TO GDS-II FOUNDRY..**

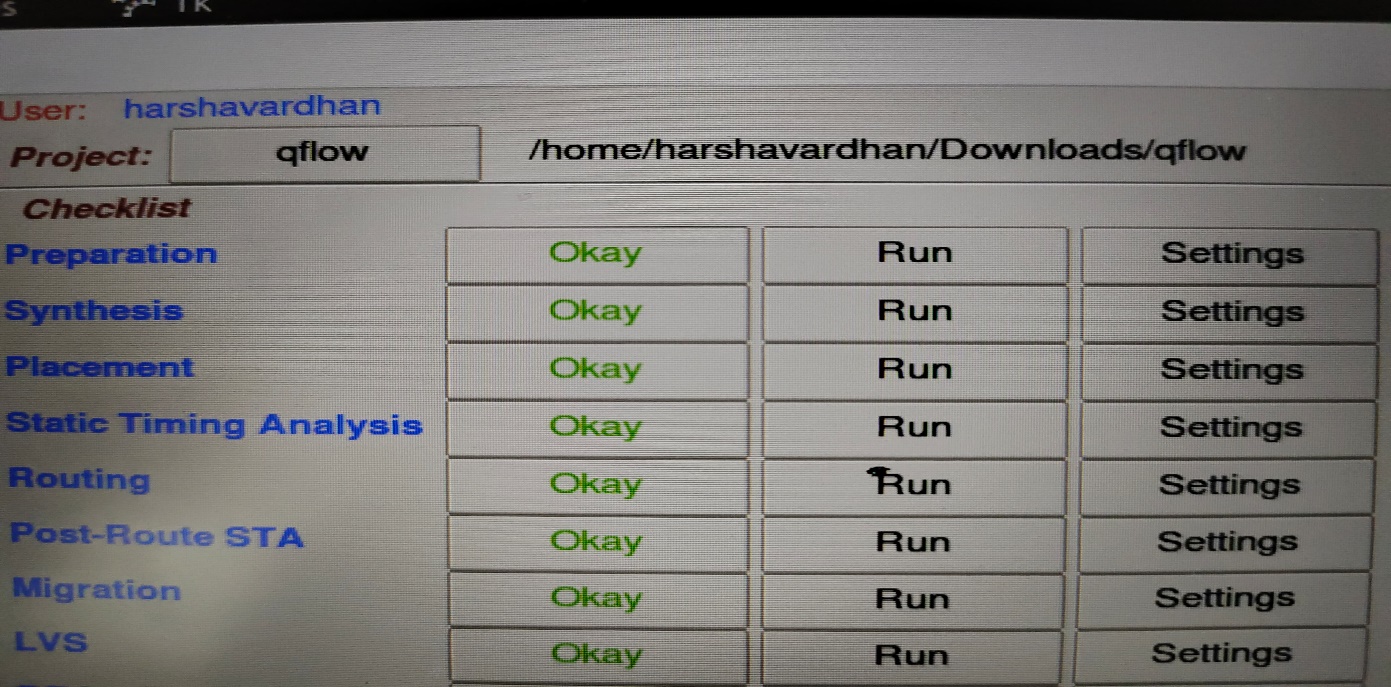
**[GDS-II] – Graphic Design System Information Interchange.**

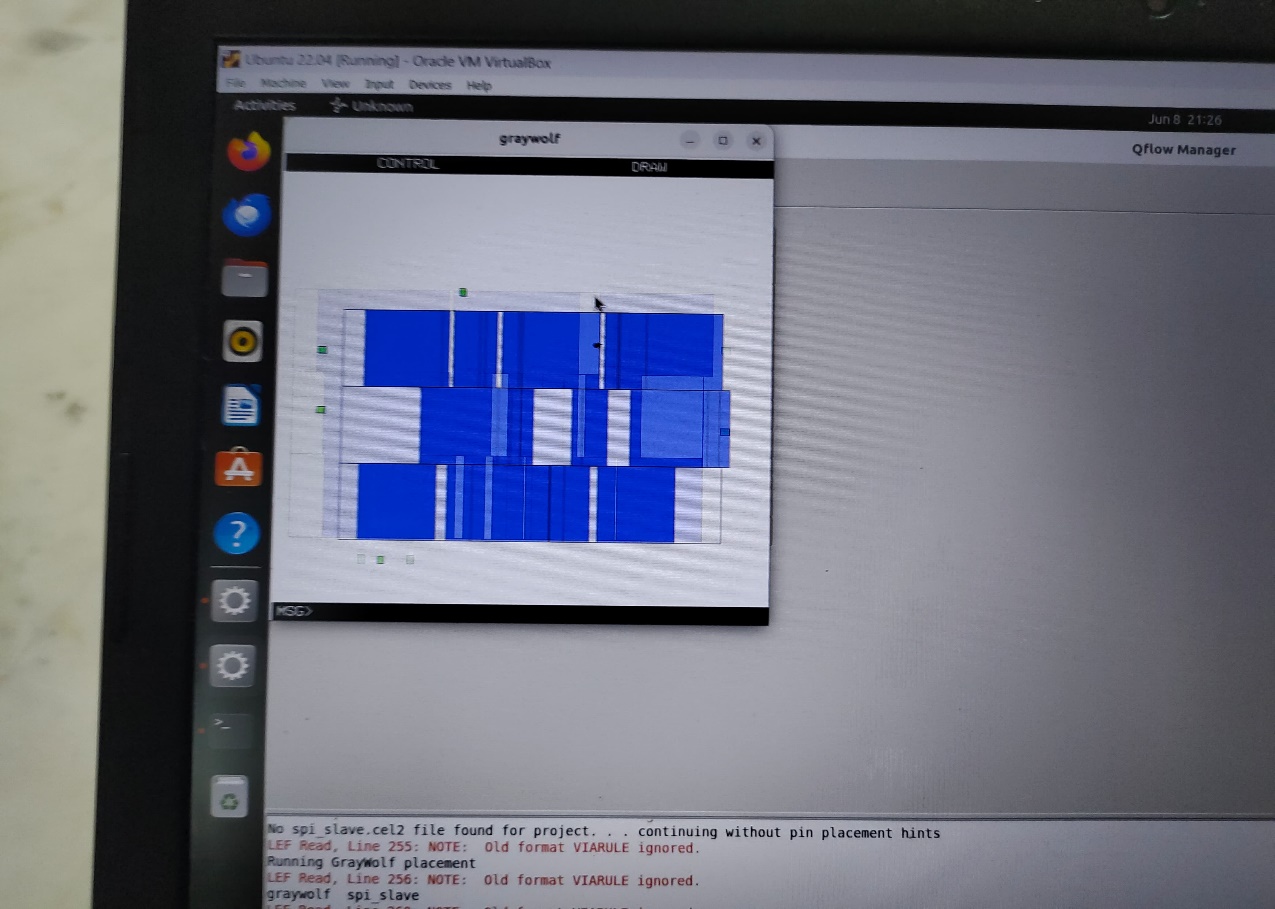
**PROJECT**

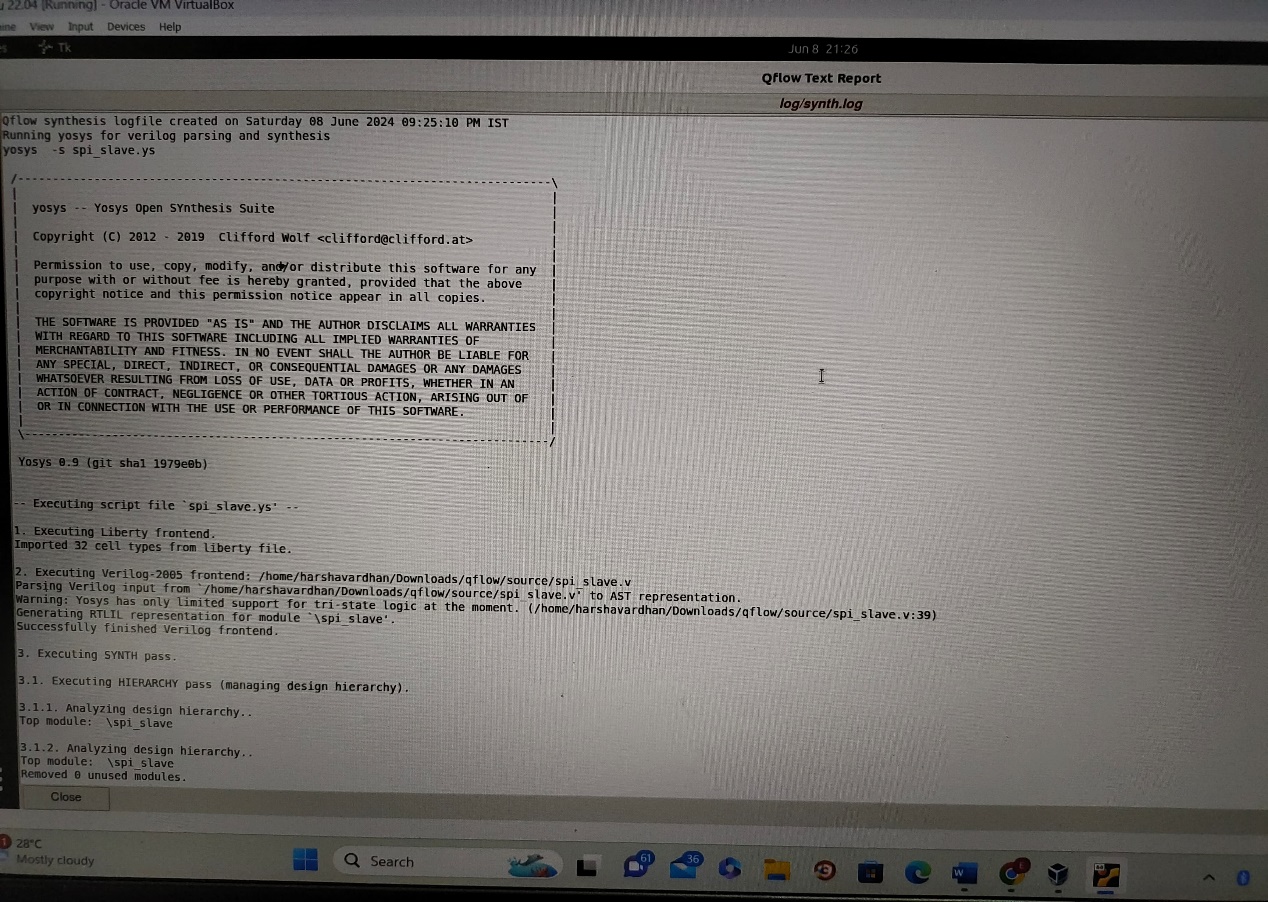
**DESIGN CODE:**

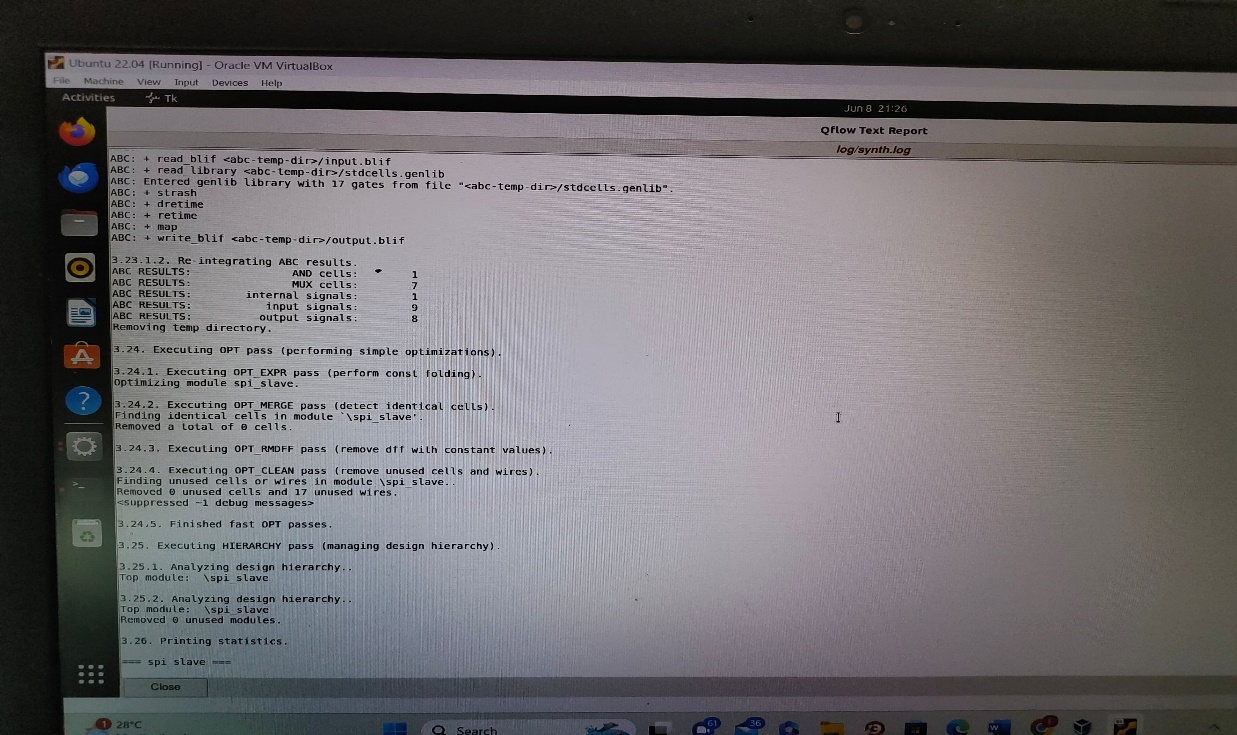
  -----------------------------------------------------------------------  
  Project Name : SPI Controller  
  Module Name  : spi\_slave  
  Type         : SystemVerilog Module  
  Description  : SPI Slave  
  Author       : Sushil Kumar  
  Company      : Maven Silicon Softech Pvt. Ltd.  
  Date         : Feb 4, 2009  
  Version      : 1.0  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
`include "timescale.v"  
module spi\_slave  (sck,  
                   cs,  
                   mosi,  
                   miso  
                  );  
  
 input sck;    
 input cs;    
 input mosi;    
                   
 output miso;  
   
 reg [7:0]interna\_reg;  
 //reg [7:0] slv\_reg=8'hFF;  
 reg [7:0] slv\_reg=8'b10100010;  
 reg [7:0] in\_reg=0;  
 reg [7:0] rec\_reg=0;  
 reg [7:0] count=0;  
  
 assign miso = (cs==1'b0) ? slv\_reg[7] : 1'bz;  
 always@(negedge sck)  
 begin  
    if (!cs)  
        slv\_reg<=slv\_reg<<1'b1;  
 end  
   
 always@(posedge sck)  
 begin  
    if (!cs)  
        rec\_reg<={rec\_reg[6:0],mosi};  
 end  
  
 always@(posedge sck)  
 begin  
    if (!cs)  
       count<=count+1;  
    else  
       count=0;  
 end  
   
 always@(posedge sck)  
 if (count==7)  
     begin  
        in\_reg<={rec\_reg[6:0],mosi};  
     end  
endmodule.

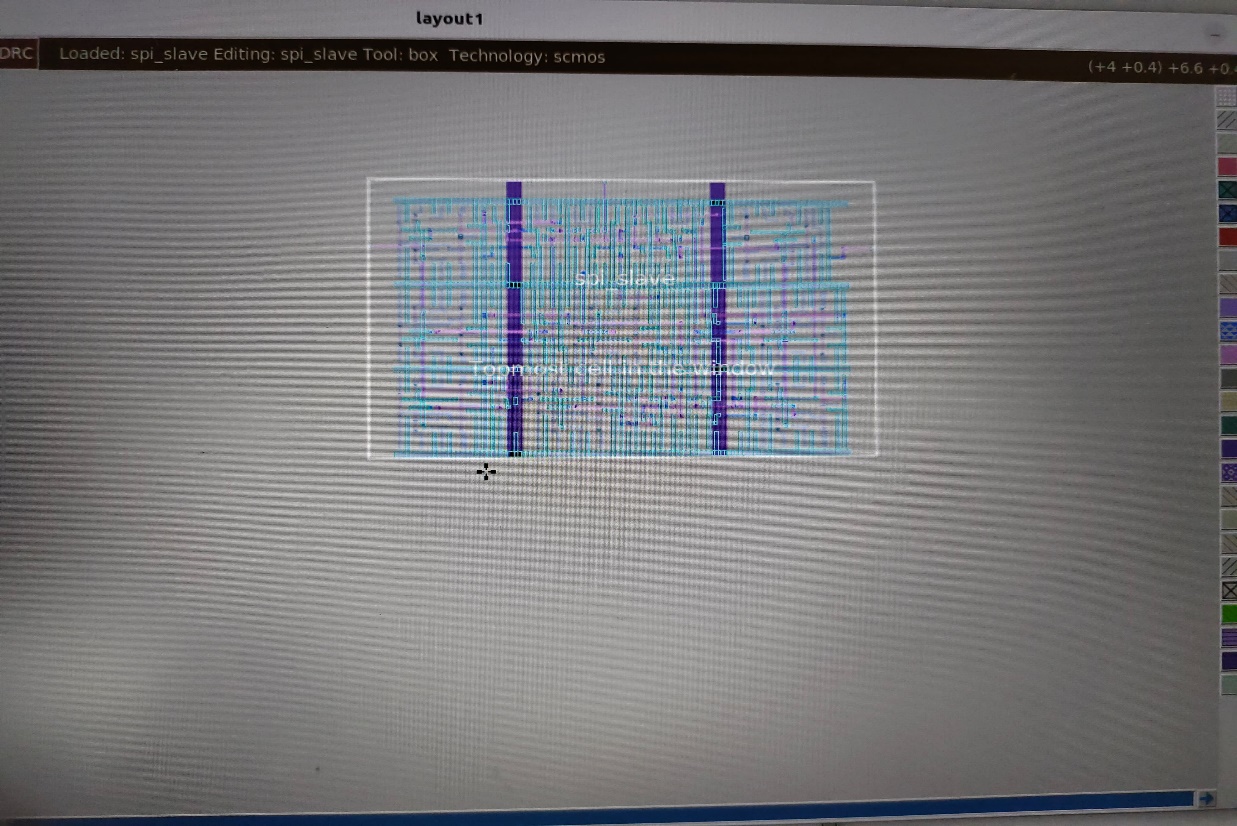
**OUTPUT:**

****

****

****



****

**BY**

**ERUKULLA HARSHVARDAHAN.**