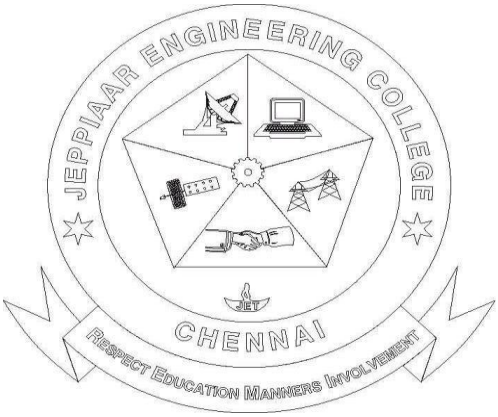
JEPPIAAR ENGINEERING COLLEGE

(A CHRISTIAN MINORITY INSTITUTION)

JEPPIAAR EDUCATIONAL TRUST

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai-600119.



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

EC3561- VLSI DESIGN LABORATORY

III-YEAR V -SEM

LAB MANUAL

NAME :_____

ROLLNO. :_____

REGNO. :_____

YEAR :_____

SECTION :_____

JEPPIAAR ENGINEERING COLLEGE

Vision of the Institute	To build Jeppiaar Engineering College as an institution of academic excellence in technological and management education to become a world class University	
Mission of the Institute	**M1**	To excel in teaching and learning, research and innovation by promoting the principles of scientific analysis and creative thinking
	M2	To participate in the production, development and dissemination of knowledge and interact with national and international communities.
	M3	To equip students with values, ethics and life skills needed to enrich their lives and enable them to meaningfully contribute to the progress of society
	M4	To prepare students for higher studies and lifelong learning, enrich them with the practical and entrepreneurial skills necessary to excel as future professionals and contribute to Nation's economy

DEPARTMENT: ELECTRONICS AND COMMUNICATION ENGINEERING

Vision of the Department	To become a centre of excellence to provide quality education and produce creative engineers in the field of Electronics and Communication Engineering to excel at international level.	
Mission of the Department	**M1**	Inculcate creative thinking and zeal for research to excel in teaching-learning process
	M2	Create and disseminate technical knowledge in collaboration with industries
	M3	Provide ethical and value based education by promoting activities for the betterment of the society
	M4	Encourage higher studies, employability skills, entrepreneurship and research to produce efficient professionals thereby adding value to the nation's economy
PROGRAM OUTCOMES (PO)	**PO 1**	**Engineering knowledge**: **Apply** the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
	PO 2	**Problem analysis**: **Identify, formulate, review** research literature, and **analyze** complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
	PO 3	**Design/development of solutions**: **Design solutions** for complex engineering problems and **design** system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
	PO 4	**Conduct investigations of complex problems**: **Use** research-

		based knowledge and research methods including **design** of experiments, **analysis** and **interpretation** of data, and **synthesis** of the information to provide valid conclusions.
	PO 5	**Modern tool usage**: **Create**, **select**, and **apply** appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
	PO 6	**The engineer and society**: **Apply** reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
	PO 7	**Environment and sustainability**: **Understand** the impact of the professional engineering solutions in societal and environmental contexts, and **demonstrate** the knowledge of, and need for sustainable development.
	PO 8	**Ethics**: **Apply** ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
	PO 9	**Individual and team work**: **Function effectively** as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
	PO 10	**Communication**: **Communicate effectively** on complex engineering activities with the engineering community and with society at large, such as, being able **to comprehend** and **write** effective reports and **design** documentation, **make** effective presentations, and **give** and **receive** clear instructions.
	PO 11	**Project management and finance**: **Demonstrate** knowledge and understanding of the engineering and management principles and **apply** these to one's own work, as a member and leader in a team, to **manage** projects and in multidisciplinary environments.
	PO 12	**Life-long learning**: **Recognize** the need for, and have the preparation and ability to **engage** in independent and life-long learning in the broadest context of technological change.
PROGRAM EDUCATIONAL OBJECTIVES (PEOS)	**PEO I**	Produce technically competent graduates with a solid foundation in the field of Electronics and Communication Engineering with the ability to analyze, design, develop, and implement electronic systems.
	PEO II	Motivate the students for choosing the successful career choices in both public and private sectors by imparting professional development activities.
	PEO III	Inculcate the ethical values, effective communication skills and develop the ability to integrate engineering skills to broader social needs to the students.
	PEO IV	Impart professional competence, desire for lifelong learning and leadership skills in the field of Electronics and Communication Engineering.
PROGRAM SPECIFIC OUTCOMES (PSOs)	**PSO 1**	Design, develop and analyze electronic systems through application of relevant electronics, mathematics and engineering principles.
	PSO 2	Design, develop and analyze communication systems through application of fundamentals from communication principles, signal

		processing, and RF System Design & Electromagnetics.
	PSO 3	Adapt to emerging electronics and communication technologies and develop innovative solutions for existing and newer problems.

EC3561 VLSI LABORATORY L T P C 0 0 4 2

COURSE OBJECTIVES:

- To learn Hardware Descriptive Language (Verilog/VHDL).
- To learn the fundamental principles of Digital System Desing using HDL and FPGA.
- To learn the fundamental principles of VLSI circuit design in digital domain
- To learn the fundamental principles of VLSI circuit design in analog domain
- To provide hands on design experience with EDA platforms.

LIST OF EXPERIMENTS:

1. Design of basic combinational and sequential (Flip-flops) circuits using HDL. Simulate it using Xilinx/Altera Software and implement by Xilinx/Altera FPGA

2. Design an Adder ; Multiplier (Min 8 Bit) using HDL. Simulate it using Xilinx/Altera Software and implement by Xilinx/Altera FPGA

3. Design and implement Universal Shift Register using HDL. Simulate it using Xilinx/Altera Software

4. Design Memories using HDL. Simulate it using Xilinx/Altera Software and implement by Xilinx/Altera FPGA

5. Design Finite State Machine (Moore/Mealy) using HDL. Simulate it using Xilinx/Altera Software and implement by Xilinx/Altera FPGA

6. Design 3-bit synchronous up/down counter using HDL. Simulate it using Xilinx/Altera Software and implement by Xilinx/Altera FPGA

7. Design 4-bit Asynchronous up/down counter using HDL. Simulate it using Xilinx/Altera Software and implement by Xilinx/Altera FPGA

8. Design and simulate a CMOS Basic Gates & Flip-Flops. Generate Manual/Automatic Layout .

9. Design and simulate a 4-bit synchronous counter using a Flip-Flops. Generate Manual/Automatic Layout

10. Design and Simulate a CMOS Inverting Amplifier.

11. Design and Simulate basic Common Source, Common Gate and Common Drain Amplifiers.

12. Design and simulate simple 5 transistor differential amplifier.

COURSE OUTCOMES:

On completion of the course, students will be able to:

CO1: Write HDL code for basic as well as advanced digital integrated circuit

CO2: Import the logic modules into FPGA Boards

CO3: Synthesize Place and Route the digital Ips

CO4: Design, Simulate and Extract the layouts of Digital & Analog IC Blocks using EDA tools

CO5: Test and Verification of IC design TOTAL: 60 PERIOD

INDEX

S.No	Date	Name of the Experiment	Page No	Date of submission	Mark	Sign

S.No	Date	Name of the Experiment	Page No	Date of submission	Mark	Sign

Exp. No. : Date:	DESIGN OF BASIC COMBINATIONAL AND SEQUENTIAL (FLIP-FLOPS) CIRCUITS USING HDL. SIMULATE IT USING XILINX/ALTERA SOFTWARE AND IMPLEMENT BY XILINX/ALTERA FPGA

AIM:

To develop the source code for flip flops by using VERILOG and Obtained the simulation, synthesis, place and route and implement into FPGA.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

PROCEDURE: (Design Entry and Simulation)

1. Start the Xilinx ISE by using Start Program files Xilinx ISE project navigator
2. Click File New Project
3. Enter the Project Name and select the location then click next
4. Select the Device and other category and click next twice and finish.
5. Click on the symbol of FPGA device and then right click click on new source.
6. Select the Verilog Module and give the file name click next and define ports click next and finish.
7. Writing the Verilog Code in Verilog Editor.
8. Run the Check syntax Process window synthesize double click check syntax. If any errors found then remove the errors with proper syntax & coding.
9. Click on the symbol of FPGA device and then right click click on new source.
10. Select the Test Bench Waveform and give the file name select entity click next and finish.
11. Select the desired parameters for simulating your design. In this case combinational circuit and simulation time click finish.
12. Assign all input signal using just click on graph and save file.
13. From the source process window. Click Behavioral simulation from drop-down menu
14. Select the test bench file (.tbw) and click process button double click the Simulation Behavioral Model.
15. Verify your design in wave window by seeing behavior of output signal with respect to input signal.

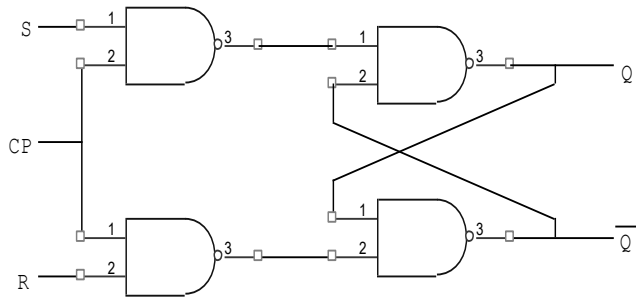
ALGORITHM:

Step1: Define the specifications and initialize the design.
Step2: Declare the name of the module by using VERILOG source code.
Step3: Write the source code in VERILOG.
Step4: Check the syntax and debug the errors if found, obtain the synthesis report.
Step5: Verify the output by simulating the source code.
Step6: Write all possible combinations of input using the test bench.
Step7: Obtain the place and route report.

LOGIC DIAGRAM:

SR FLIPFLOP:

LOGIC DIAGRAM:



TRUTH TABLE:

Q(t)	S	R	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	X
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X

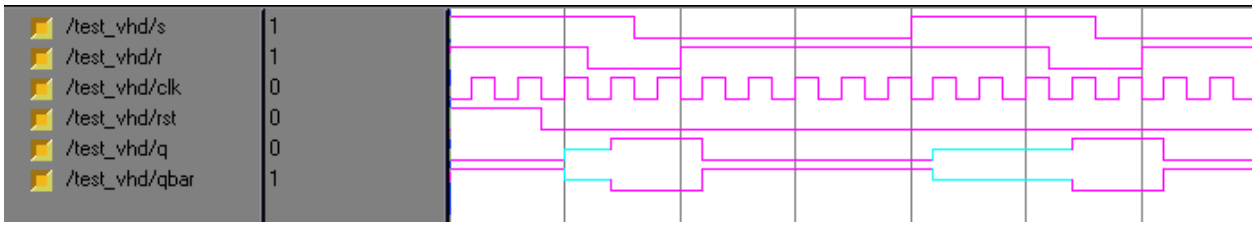
VERILOG SOURCE CODE:

Behavioral Modeling:

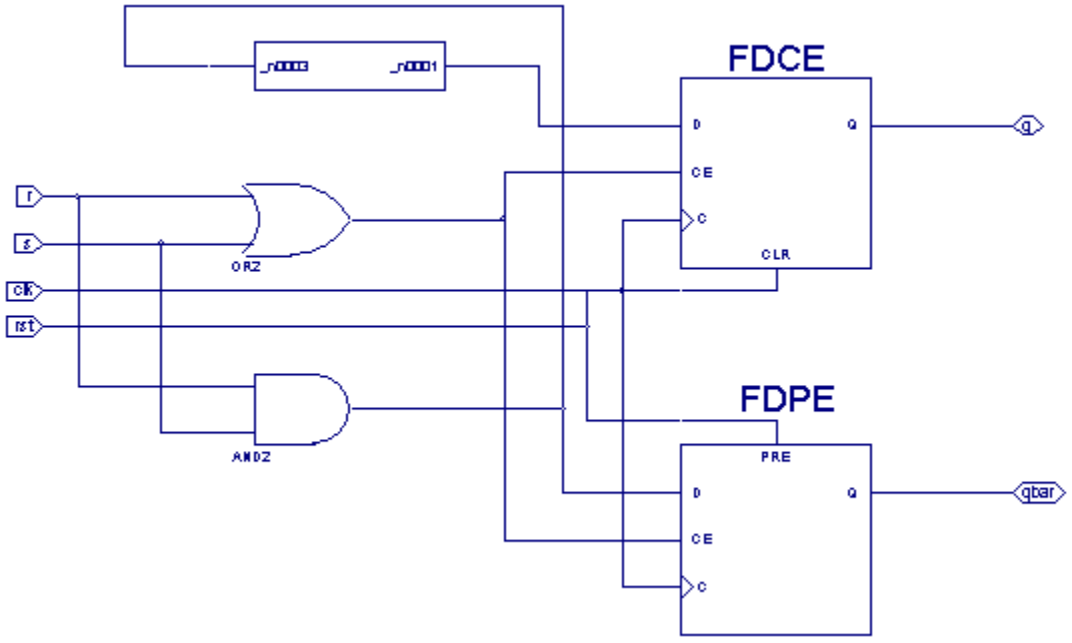
```
module srflipflop(s, r, clk, rst, q, qbar);
  input s;
  input r;
  input clk;
  input rst;
  output q;
  output qbar;
        reg q,qbar;
        always @ (posedge(clk) or posedge(rst)) begin
        if(rst==1'b1) begin
        q= 1'b0;qbar= 1'b1;
        end
        else if(s==1'b0 &&          r==1'b0)
         begin
        q=q; qbar=qbar;
        end
                else if(s==1'b0 &&          r==1'b1)
                 begin
        q= 1'b0; qbar= 1'b1;
```

```
        end
                else if(s==1'b1 &&          r==1'b0)
                 begin
        q= 1'b1; qbar= 1'b0;
        end
        else
        begin
        q=1'bx;qbar=1'bx;
        end
        end
endmodule
```

Simulation output:



Synthesis RTL Schematic:



Synthesis report:

```
=====================================================================
*                  Final Report                     *
=====================================================================
Final Results
RTL Top Level Output File Name     : srff.ngr
Top Level Output File Name         : srff
Output Format                      : NGC
Optimization Goal                  : Speed
```

Keep Hierarchy : NO

Design Statistics
IOs : 6

Macro Statistics :
Registers : 2
1-bit register : 2

Cell Usage :
BELS : 3
LUT2 : 3
FlipFlops/Latches : 2
FDCE : 1
FDPE : 1
Clock Buffers : 1
BUFGP : 1
IO Buffers : 5
IBUF : 3
OBUF : 2
===

Device utilization summary:

Selected Device : 3s400tq144-5

Number of Slices: 2 out of 3584 0%
Number of Slice Flip Flops: 2 out of 7168 0%
Number of 4 input LUTs: 3 out of 7168 0%
Number of bonded IOBs: 6 out of 97 6%
Number of GCLKs: 1 out of 8 12%

===
TIMING REPORT

NOTE:

THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

CLOCK INFORMATION:

Clock Signal | Clock buffer(FF name) | Load |
---
clk | BUFGP | 2 |

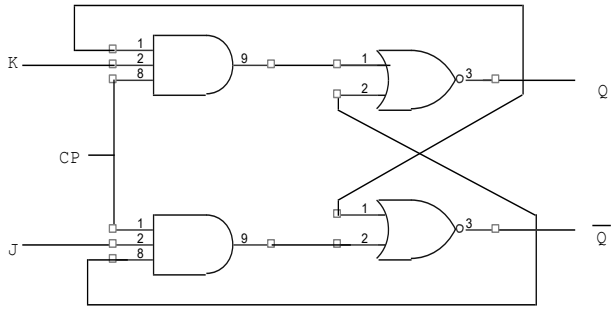
Timing Summary:

Speed Grade: -5

Minimum period: No path found
Minimum input arrival time before clock: 3.529ns
Maximum output required time after clock: 6.216ns
Maximum combinational path delay: No path found

JK FLIPFLOP:

LOGIC DIAGRAM:



TRUTH TABLE:

Q(t)	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

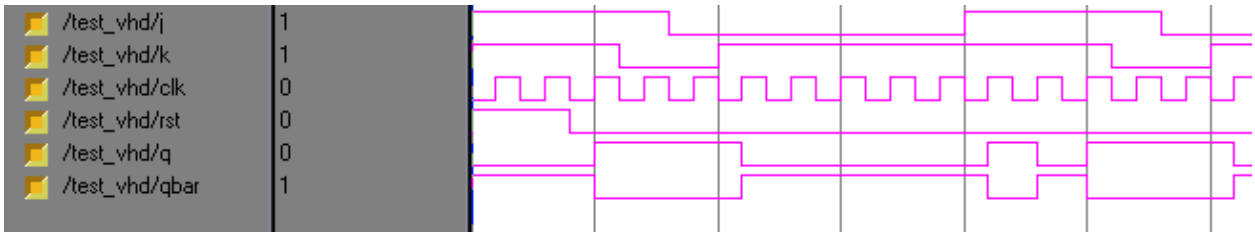
VERILOG SOURCE CODE:

BEHAVIORAL MODELING:

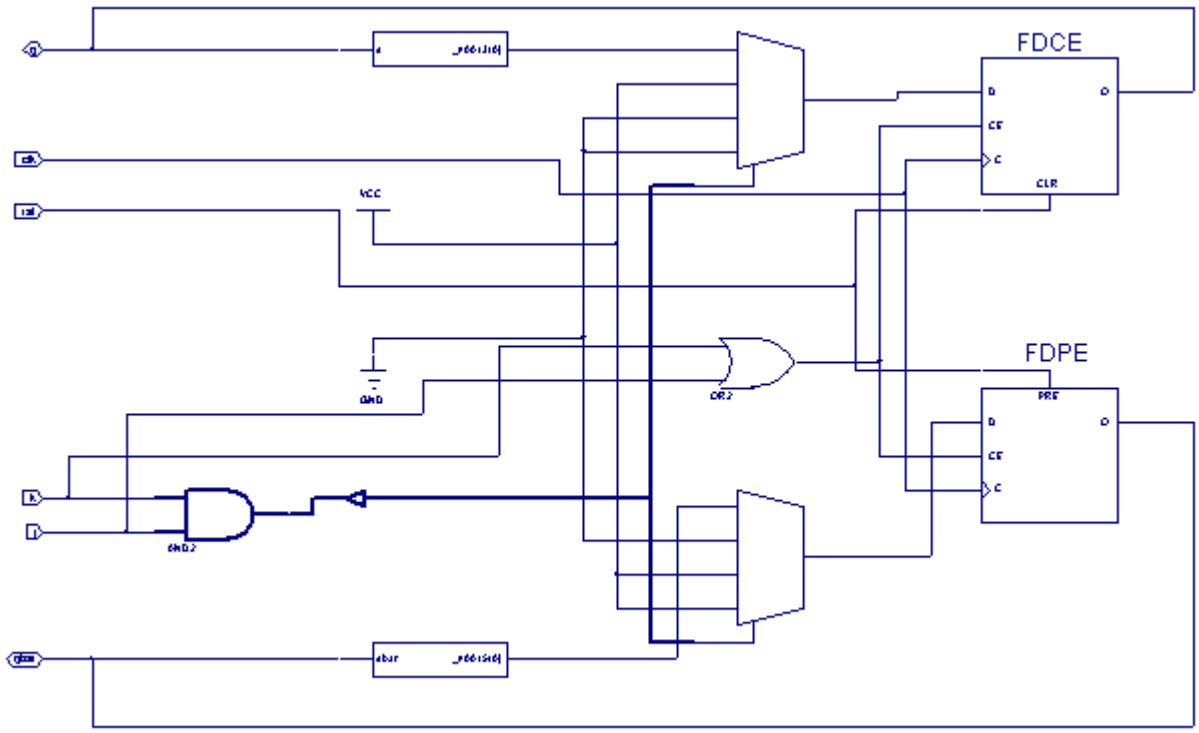
```
module jkff(j, k, clk, rst, q, qbar);
   input j;
   input k;
   input clk;
   input rst;
   output q;
   output qbar;
            reg q;
            reg qbar;
            always @ (posedge(clk) or posedge(rst))  begin
            if (rst==1'b1)
            begin
            q=1'b0;
            qbar=1'b1;
            end
            else if (j==1'b0 && k==1'b0)
            begin
            q=q;
            qbar=qbar;
            end
            else if (j==1'b0 && k==1'b1)
            begin
            q=1'b0;
            qbar=1'b1;
            end
            else if (j==1'b1 && k==1'b0)
```

```
begin
q=1'b1;
qbar=1'b0;
end
else
begin
q=~q;
qbar=~qbar;
end
end
endmodule
```

SIMULATION OUTPUT:



Synthesis RTL Schematic:



SYNTHESIS REPORT:

```
=====================================================================
*                  Final Report                     *
=====================================================================
Device utilization summary:
---------------------------

Selected Device : 3s400tq144-5

 Number of Slices:                  2  out of   3584     0%
 Number of Slice Flip Flops:        2  out of   7168     0%
 Number of 4 input LUTs:            3  out of   7168     0%
 Number of bonded IOBs:             6  out of     97     6%
 Number of GCLKs:                   1  out of      8    12%


=====================================================================
TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.
```

CLOCK INFORMATION:

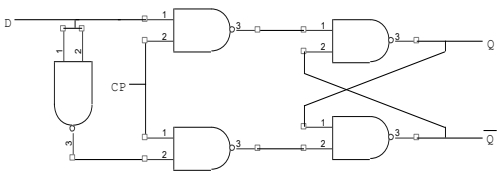
```
------------------
-----------------------------------+------------------------+-------+
Clock Signal                       | Clock buffer(FF name)  | Load  |
-----------------------------------+------------------------+-------+
clk                                | BUFGP                  | 2     |
-----------------------------------+------------------------+-------+
```

TIMING SUMMARY:

```
---------------
Speed Grade: -5

  Minimum period: 2.085ns (Maximum Frequency: 479.513MHz)
  Minimum input arrival time before clock: 3.529ns
  Maximum output required time after clock: 6.280ns
  Maximum combinational path delay: No path found
```

D FLIPFLOP:

LOGIC DIAGRAM:



TRUTH TABLE:

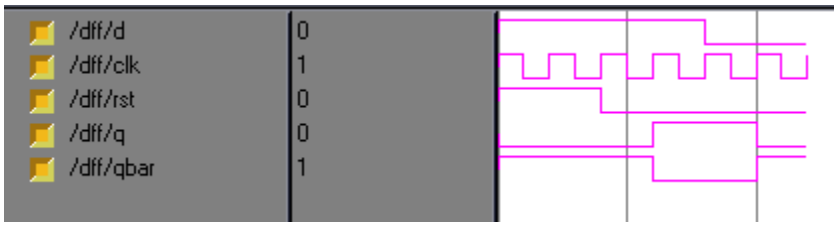
Q(t)	D	Q(t+1)
0	0	0
0	1	1
1	0	0
1	1	1

VERILOG SOURCE CODE:

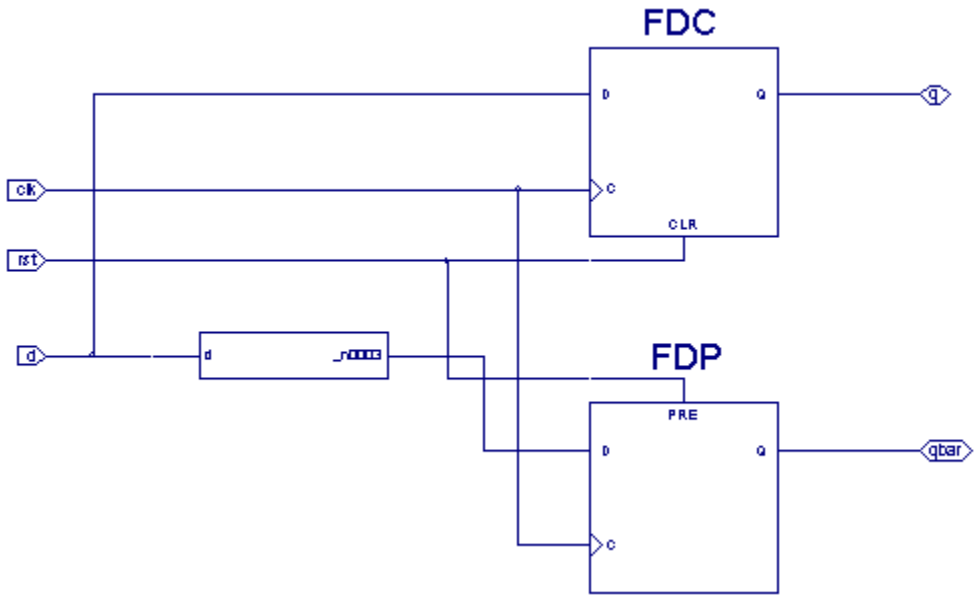
BEHAVIORAL MODELING:

```
module dff(d, clk, rst, q, qbar);
   input d;
   input clk;
   input rst;
   output q;
   output qbar;
            reg q;
            reg qbar;
            always @ (posedge(clk) or posedge(rst)) begin
            if (rst==1'b1)
            begin
            q=1'b0;
            qbar=1'b1;
            end
            else if (d==1'b0)
            begin
            q=1'b0;
            qbar=1'b1;
            end
            else
            begin
            q=1'b1;
            qbar=1'b0;
            end
           end
           endmodule
```

SIMULATION OUTPUT:



SYNTHESIS RTL SCHEMATIC:



SYNTHESIS REPORT:

==========

* Final Report *

==========

DEVICE UTILIZATION SUMMARY:

Selected Device : 3s400tq144-5

Number of Slices:	1	out of 3584	0%
Number of Slice Flip Flops:	2	out of 7168	0%
Number of bonded IOBs:	5	out of 97	5%
Number of GCLKs:	1	out of 8	12%

==========

TIMING REPORT

NOTE:

THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

CLOCK INFORMATION:

Clock Signal	Clock buffer(FF name)	Load
clk	BUFGP	2

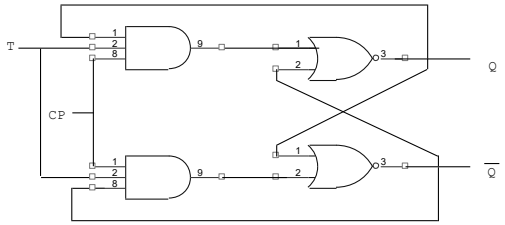
TIMING SUMMARY:

SPEED GRADE: -5

Minimum period: No path found
Minimum input arrival time before clock: 2.796ns
Maximum output required time after clock: 6.216ns
Maximum combinational path delay: No path found

T FLIPFLOP:

LOGIC DIAGRAM:



TRUTH TABLE:

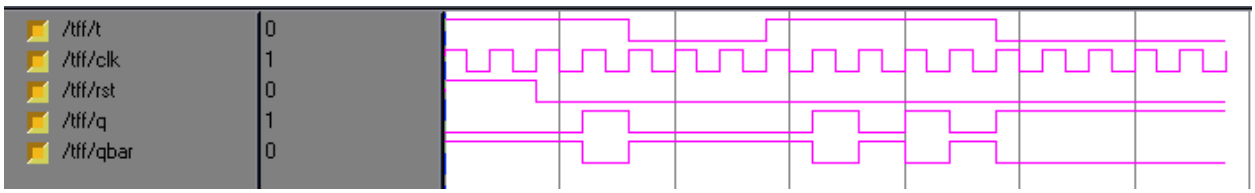
Q(t)	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

VERILOG SOURCE CODE:

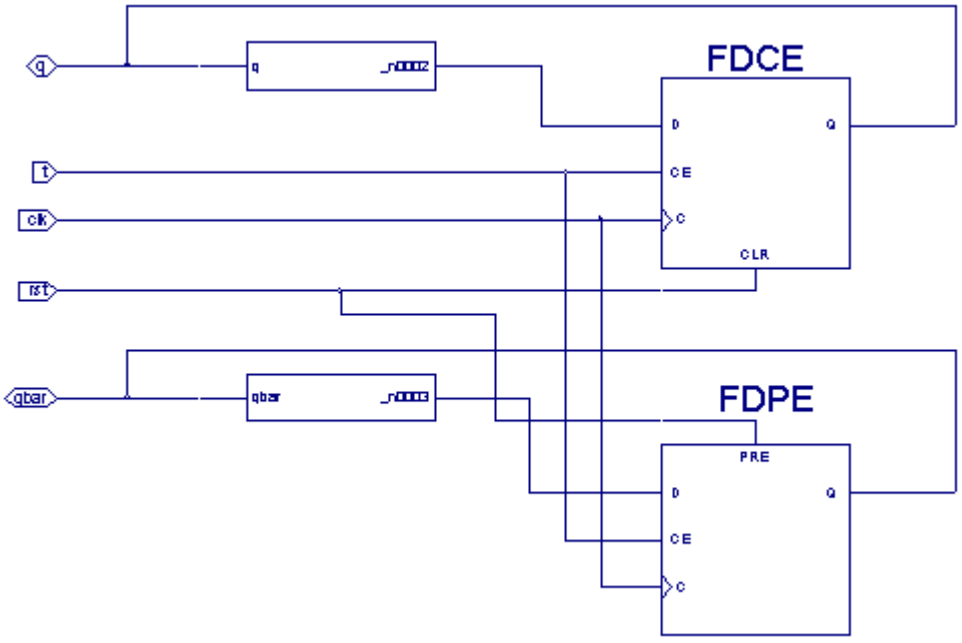
BEHAVIORAL MODELING:

```
module tff(t, clk, rst, q, qbar);
   input t;
   input clk;
   input rst;
   output q;
   output qbar;
            reg q,qbar;
            always @ (posedge(clk) or posedge(rst)) begin
            if(rst==1'b1) begin
            q= 1'b0;qbar= 1'b1;
            end
            else if (t==1'b0)
            begin
            q=q; qbar=qbar;

          end
            else
            begin
            q=~q; qbar=~qbar;
            end
            end
            endmodule
```

SIMULATION OUTPUT:



SYNTHESIS RTL SCHEMATIC:



SYNTHESIS REPORT:

==========

* Final Report *

==========

DEVICE UTILIZATION SUMMARY:

Selected Device : 3s400tq144-5

Number of Slices:	1	out of 3584	0%
Number of Slice Flip Flops:	2	out of 7168	0%
Number of bonded IOBs:	5	out of 97	5%
Number of GCLKs:	1	out of 8	12%

==========

TIMING REPORT

NOTE:

THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

CLOCK INFORMATION:

Clock Signal	Clock buffer(FF name)	Load
clk	BUFGP	2

TIMING SUMMARY:

SPEED GRADE: -5

Minimum period: 2.707ns (Maximum Frequency: 369.372MHz)
Minimum input arrival time before clock: 1.984ns
Maximum output required time after clock: 6.280ns
Maximum combinational path delay: No path found

RESULT:

Exp. No. : Date:	DESIGN ENTRY, SIMULATION, SYNTHESIS, P&R, POST P&R SIMULATION AND HARDWARE FUSING OF FULL ADDER & FULL SUBTRACTOR USING XILINX

AIM:

To write a Verilog code for the full adder and full subtractor and to simulate, synthesis, P&R, post P&R simulation and hardware fusing using Xilinx project navigator.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

PROCEDURE: (Design Entry and Simulation)

1. Start the Xilinx ISE by using Start Program files Xilinx ISE project navigator
2. Click File New Project
3. Enter the Project Name and select the location then click next
4. Select the Device and other category and click next twice and finish.
5. Click on the symbol of FPGA device and then right click click on new source.
6. Select the Verilog Module and give the file name click next and define ports click next and finish.
7. Writing the Verilog Code in Verilog Editor.
8. Run the Check syntax Process window synthesize double click check syntax. If any errors found then remove the errors with proper syntax & coding.
9. Click on the symbol of FPGA device and then right click click on new source.
10. Select the Test Bench Waveform and give the file name select entity click next and finish.
11. Select the desired parameters for simulating your design. In this case combinational circuit and simulation time click finish.
12. Assign all input signal using just click on graph and save file.
13. From the source process window. Click Behavioral simulation from drop-down menu
14. Select the test bench file (.tbw) and click process button double click the Simulation Behavioral Model.
15. Verify your design in wave window by seeing behavior of output signal with respect to input signal.

PROCEDURE: (Synthesis)

1. Start the Xilinx ISE by using Start Program files Xilinx ISE project navigator
2. Click File New Project
3. Enter the Project Name and select the location then click next
4. Select the Device and other category and click next twice and finish.
5. Click on the symbol of FPGA device and then right click click on new source.
6. Select the Verilog Module and give the file name click next and define ports click next and finish.

7. Writing the Verilog Code in Verilog Editor.
8. Run the Check syntax Process window synthesize double click check syntax. If any errors found then remove the errors with proper syntax & coding.
9. Synthesis your design, from the source window select, Synthesis from the window Now double click the
10. After the HDL synthesis phase of the synthesis process, you can display a schematic representation of your synthesized source file. This schematic shows a representation of the pre-optimized design in terms of generic symbols, such as adders, multipliers, counters, AND gates, and OR gates double click View RTL Schematic
11. Double click the schematic to internal view
12. Double click outside the schematic to move one-level back
13. This schematic shows a representation of the design in terms of logic elements optimized to the target device. For example, in terms of LUTs(Look Up Table), carry logic, I/O buffers, and other technology-specific components Double click View Technology Schematic
14. Double click the schematic to inner view
15. Double click the LUT to inner view. This is Gate Level view of LUT, if you want see Truth Table and K-Map for your design just click the respective tabs.
16. Click Generate post synthesis simulation model and view its report.

PROCEDURE: (Place and Route)

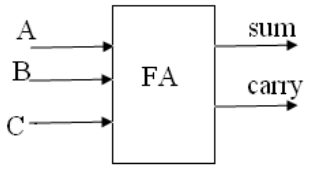
1. Start the Xilinx ISE by using Start Program files Xilinx ISE project navigator
2. Click File New Project
3. Enter the Project Name and select the location then click next
4. Select the Device and other category and click next twice and finish.
5. Click on the symbol of FPGA device and then right click click on new source.
6. Select the Verilog Module and give the file name click next and define ports click next and finish.
7. Writing the Verilog Code in Verilog Editor.
8. Run the Check syntax Process window synthesize double click check syntax. If any errors found then remove the errors with proper syntax & coding.
9. Synthesis your design, from the source window select, synthesis/implementation from the window Now double click the Synthesis -XST
10. After Synthesis you assign the Pin Value for your design so, double click the Assign Package Pins
11. Enter the Pin value for your input and output signals. if you want see your Pin assignment in FPGA zoom in Architecture View or Package View
12. You see the Pins in FPGA. Save file as XST Default click ok and close the window
13. Design Implementation begins with the mapping or fitting of a logical design file to a specific device and is complete when the physical design is successfully routed and a bit stream is generated. Double Click Implementation Design.
14. After finishing the Implementation, you can view the Implementation report.
15. After implementation you see Design Summary, you get the all details about your design. If you want edit the place and route double click View/Edit placed design
16. You see where your IOs are placed in FPGA. And zoom to view how Pins are placed in FPGA. You can see where your pins are placed
17. Just double click View/Edit Routed Design to view interconnection wires and blocks
18. Click the pin to see where its placed in FPGA. And Zoom particular areato see Place and Routing.
19. If you want to change the place of the design, click and trace to another slice. See, You changed place and route of the design

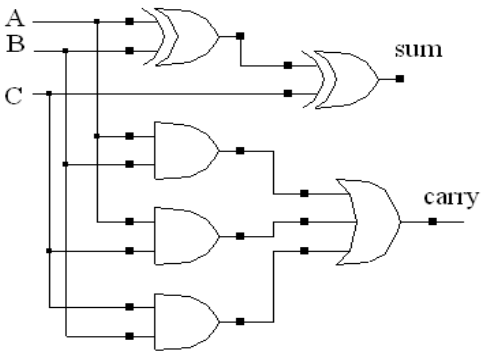
20. Double click Back annotated Pin Location. Once back annotation is completed, constraint file is generated.

PROCEDURE: (HARDWARE FUSING AND TESTING)

1. Start the Xilinx ISE by using Start Program files Xilinx ISE project navigator
2. Click File New Project
3. Enter the Project Name and select the location then click next
4. Select the Device and other category and click next twice and finish.
5. Click on the symbol of FPGA device and then right click click on new source.
6. Select the Verilog Module and give the file name click next and define ports click next and finish.
7. Writing the Verilog Code in Verilog Editor.
8. Run the Check syntax Process window Synthesize double click check syntax. If any errors found then remove the errors with proper syntax & coding.
9. Synthesis your design, from the source window select, synthesis/implementation from the window Now double click the Synthesis -XST.
10. After Synthesis, Click on the symbol of FPGA device and Right click and select New Source, Select Implementation Constraints File and type file name and click next.
11. Type the Net list and click save.
12. Implement the design by double clicking Implement design in the process window.
13. Then double click Generate Programming File, Double click Configure Target Device and click OK.
14. Double click Create PROM File in the ISE iMPACT window, Select Storage Target Device as Xilinx Flash PROM and click forward.
15. Add storage Device as xcf01s [2 M] and click forward, Type Output File Name and Location and click OK.
16. Select the corresponding .bit file and click Open, Click No to Add Another Device and Click OK.
17. Double click Generate File.
18. Double click Boundary Scan and Right click on the window and select Initialize Chain, Now Select the corresponding .mcs file and click open.
19. Click OK in the Device Programming Properties window, Download the Program on to the kit by Right clicking on the device icon and select program.
20. Verify the output in the target device.

FULL ADDER CIRCUIT
TOP BLOCK





TRUTH TABLE:

A	B	C	SUM	CARRY
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

CODING

```
module fa_dataflow(
   output sum,
   output carry,
   input a,
   input b,
   input c
   );
assign sum = a^b^c;
assign carry =(a&b)|(a&c)|(b&c);
endmodule
```

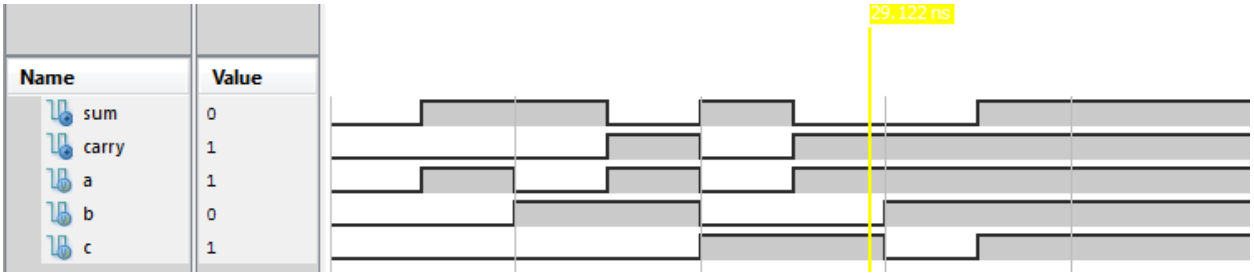
```
module fa_struct(
   output sum,
   output carry,
   input a,
   input b,
   input c
   );
wire w1,w2,w3;
xor x1 (sum,a,b,c);
and x2 (w1,a,b);
and x3 (w2,a,c);
and x4 (w3,b,c);
or x5 (carry,w1,w2,w3);
endmodule
```

```
module fa_beh(
   output sum,
   output carry,
   input a,
   input b,
   input c
   );
reg sum,carry;
always @(a,b,c)
begin
 sum = a^b^c;
 carry =(a&b)|(a&c)|(b&c);
end
endmodule
```

TEST BENCH:

```
module fa_data_test;
        reg a,b,c;
        wire sum,carry;
        fa_data uut (sum, carry,a,b,c);
        initial
        begin
                a = 0; b = 0;c = 0;
        # 5     a = 1; b = 0;c = 0;
        # 5     a = 0; b = 1;c = 0;
        # 5     a = 1; b = 1;c = 0;
        # 5     a = 0; b = 0;c = 1;
        # 5     a = 1; b = 0;c = 1;
        # 5     a = 1; b = 1;c = 0;
        # 5     a = 1; b = 1;c = 1;
end
 initial
$monitor ($time, "a = %d; b = %d;c = %d;sum=%d;carry=%d\n",a,b,c,sum,carry);
initial #50 $stop;
endmodule
```

SIMULATED WAVEFORM:

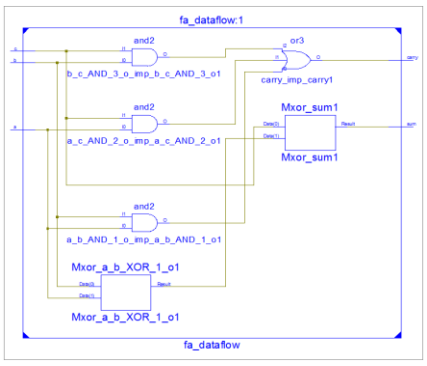


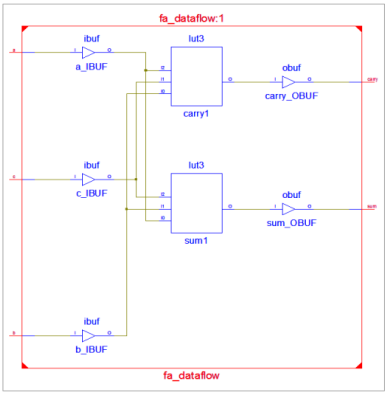
0a = 0; b = 0;c = 0;sum=0;carry=0
5a = 1; b = 0;c = 0;sum=1;carry=0
10a = 0; b = 1;c = 0;sum=1;carry=0
15a = 1; b = 1;c = 0;sum=0;carry=1
20a = 0; b = 0;c = 1;sum=1;carry=0
25a = 1; b = 0;c = 1;sum=0;carry=1
30a = 1; b = 1;c = 0;sum=0;carry=1
35a = 1; b = 1;c = 1;sum=1;carry=1

SIMULATED RESULT:

```
Timing Summary:
---------------
Speed Grade: -3
   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 5.602ns
Timing Details:
---------------
All values displayed in nanoseconds (ns)
=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 6 / 2
-------------------------------------------------------------------------
Delay:               5.602ns (Levels of Logic = 3)
  Source:            a (PAD)
  Destination:       sum (PAD)
  Data Path: a to sum
                                   Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             2   1.228   0.845  a_IBUF (a_IBUF)
     LUT3:I0->O            1   0.235   0.579  sum1 (sum_OBUF)
     OBUF:I->O                 2.715          sum_OBUF (sum)
    ----------------------------------------
    Total                      5.602ns (4.178ns logic, 1.424ns route)
                                       (74.6% logic, 25.4% route)
```

RTL VIEW& TECHNOLOGY VIEW:





POST PAR STATIC TIMING

```
Data Sheet report:
-----------------
All values displayed in nanoseconds (ns)
Pad to Pad
---------------+---------------+---------+
Source Pad     |Destination Pad|  Delay  |
---------------+---------------+---------+
a              |carry          |    7.833|
a              |sum            |    7.755|
b              |carry          |    8.048|
b              |sum            |    7.970|
c              |carry          |    7.735|
c              |sum            |    7.657|
---------------+---------------+---------+
```

POWER ANALYSIS CODE

```
module fa_power( output sum, output carry, output resetout, input a, input b,  input c, input clk, input
reset );
reg sum, carry;
wire dcm_clk;
dcm_clk x1    ( .CLK_IN1 clk), .CLK_OUT1 (dcm_clk),    .RESET (reset),   .LOCKED          (resetout));
always @ (posedge dcm_clk or posedge reset)
begin
if (reset)
begin
sum =1'b0; carry =1'b0;
end
else
begin
 sum = a^b^c;  carry =(a&b)|(a&c)|(b&c);
 end
 end
endmodule
```

POWER REPORT

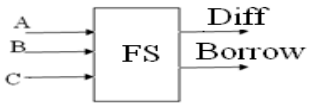
```
------------------------------------------------------------------------------
|                           On-Chip Power Summary                            |
------------------------------------------------------------------------------
|        On-Chip        | Power (mW) |  Used  | Available | Utilization (%) |
------------------------------------------------------------------------------
| Clocks                |       0.00 |      0 |    ---    |       ---       |
| Logic                 |       0.00 |      1 |      2400 |             0.1 |
| Signals               |       0.00 |      5 |    ---    |       ---       |
| IOs                   |       0.00 |      5 |       102 |             4.9 |
| Quiescent             |      13.69 |        |           |                 |
| Total                 |      13.69 |        |           |                 |

------------------------------------------------------------------------------
```

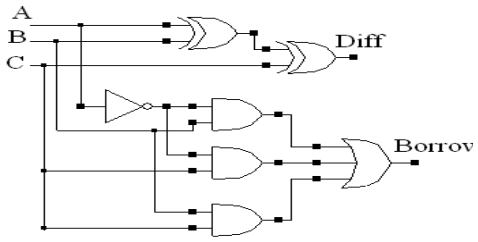
Thermal Summary	
Effective TJA (C/W)	42.4
Max Ambient (C)	84.4
Junction Temp (C)	25.6

Power Supply Summary			
	Total	Dynamic	Quiescent
Supply Power (mW)	13.69	0.00	13.69

RESULT

FULL SUBTRACTOR CIRCUIT
TOP BLOCK





TRUTH TABLE:

A	B	C	DIFFERENCE	BORROW
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

CODING

```
module fullsub_struct(
   output diff,
   output borrow,
   input a,
   input b,
   input c
   );
wire w1,w2,w3;
not x (w,a);
xor x1 (diff,a,b,c);
and x2 (w1,w,b);
and x3 (w2,w,c);
and x4 (w3,b,c);
or x5 (borrow,w1,w2,w3);
endmodule
```

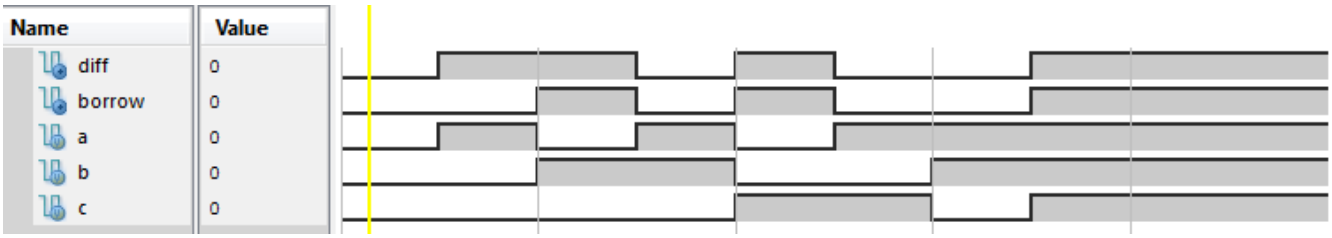
```
module fullsub_dataflow(
   output diff,
   output borrow,
   input a,
   input b,
   input c
   );
assign sum = a^b^c;
assign carry =(~a&b)|(~a&c)|(b&c);
endmodule
```

```
module fullsub_beh(
   output diff,
   output borrow,
   input a,
   input b,
   input c
   );
reg diff,borrow;
always @(a,b,c)
begin
 diff = a^b^c;
 borrow =(~a&b)|(~a&c)|(b&c);
end
endmodule
```

TEST BENCH:

```
module fullsub_test;
        reg a;
        reg b;
        reg c;
        wire diff;
        wire borrow;
        fullsub_beh uut (.diff(diff), .borrow(borrow), .a(a),   .b(b), .c(c));
initial
        begin
a = 0; b = 0;c = 0;
        # 5      a = 1; b = 0;c = 0;
        # 5      a = 0; b = 1;c = 0;
        # 5      a = 1; b = 1;c = 0;
        # 5      a = 0; b = 0;c = 1;
        # 5      a = 1; b = 0;c = 1;
        # 5      a = 1; b = 1;c = 0;
        # 5      a = 1; b = 1;c = 1;
        end
initial
$monitor ($time, "a = %d; b = %d;c = %d;diff=%d;borrow=%d\n",a,b,c,diff,borrow);
initial #50 $stop;
endmodule
```

SIMULATED WAVEFORM:

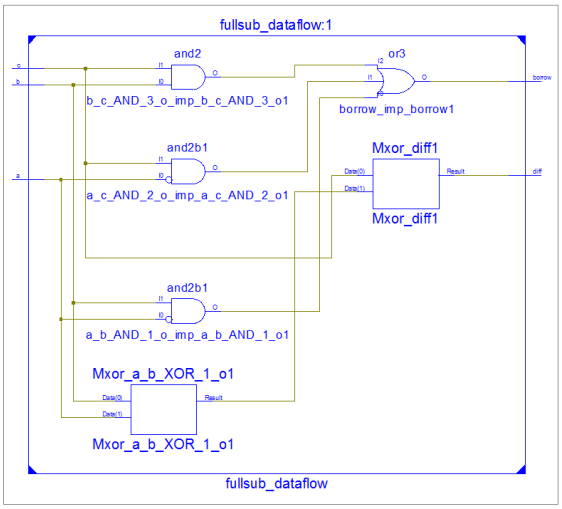


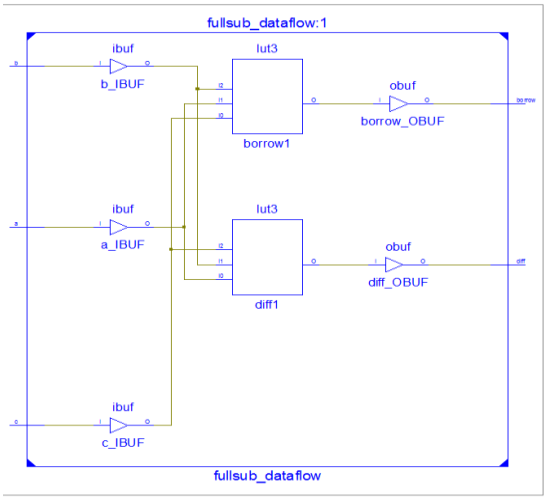
0a = 0; b = 0;c = 0;diff=0;borrow=0
5a = 1; b = 0;c = 0;diff=1;borrow=0
10a = 0; b = 1;c = 0;diff=1;borrow=1
15a = 1; b = 1;c = 0;diff=0;borrow=0
20a = 0; b = 0;c = 1;diff=1;borrow=1
25a = 1; b = 0;c = 1;diff=0;borrow=0
30a = 1; b = 1;c = 0;diff=0;borrow=0
35a = 1; b = 1;c = 1;diff=1;borrow=1

SIMULATED RESULT:

```
Timing Summary:
---------------
Speed Grade: -3
   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 5.602ns
Timing Details:
---------------
All values displayed in nanoseconds (ns)
=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 6 / 2
-------------------------------------------------------------------------
Delay:               5.602ns (Levels of Logic = 3)
  Source:            a (PAD)
  Destination:       diff (PAD)
  Data Path: a to diff
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             2   1.228   0.845  a_IBUF (a_IBUF)
     LUT3:I0->O            1   0.235   0.579  diff1 (diff_OBUF)
     OBUF:I->O                 2.715          diff_OBUF (diff)
    ----------------------------------------
    Total                      5.602ns (4.178ns logic, 1.424ns route)
                                       (74.6% logic, 25.4% route)
```

RTL VIEW:TECHNOLOGY VIEW:





POST PAR STATIC TIMING

Data Sheet report:

All values displayed in nanoseconds (ns)

Pad to Pad

Source Pad	Destination Pad	Delay
a	borrow	7.711
a	diff	7.981
b	borrow	7.210
b	diff	7.480
c	borrow	7.356
c	diff	7.626

POWER ANALYSIS CODE

```
module fs_power( output diff, output borrow, output resetout, input a, input b, input c, input clk, input reset );
reg sum, carry;
wire dcm_clk;
dcm_clk x1    ( .CLK_IN1 clk), .CLK_OUT1 (dcm_clk),    .RESET (reset),   .LOCKED  (resetout));
reg diff,borrow;
always @ (posedge dcm_clk or posedge reset)
begin
if (reset)
begin
diff =1'b0; borrow =1'b0;
end
else
begin
 diff = a^b^c;  borrow =(~a&b)|(~a&c)|(b&c);
 end
 end
endmodule
```

POWER REPORT

On-Chip Power Summary				
On-Chip	Power (mW)	Used	Available	Utilization (%)
Clocks	2.92	1	---	---
Logic	0.01	1	2400	0.1
Signals	0.04	9	---	---
IOs	2.06	8	102	7.8
DCMs	14.11	1	4	25.0
Quiescent	13.89			
Total	33.03			

Thermal Summary	
Effective TJA (C/W)	42.4
Max Ambient (C)	83.6
Junction Temp (C)	26.4

Power Supply Summary			
	Total	Dynamic	Quiescent
Supply Power (mW)	33.03	8.42	24.60

RESULT

Exp. No. :	DESIGN ENTRY, SIMULATION, SYNTHESIS, P&R, POST P&R SIMULATION OF 8-BIT PARALLEL ADDER
Date:	

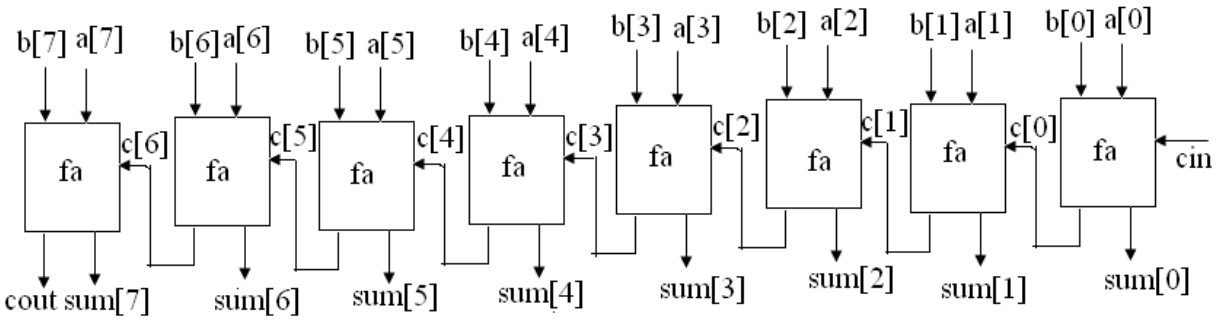
AIM:

To write a Verilog code for the 8-bit parallel adder and to simulate, synthesis, P&R, post P&R simulation and hardware fusingusing Xilinx project navigator.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

8-BIT PARALLEL ADDER (TOP BLOCK)



SOURCE CODE:

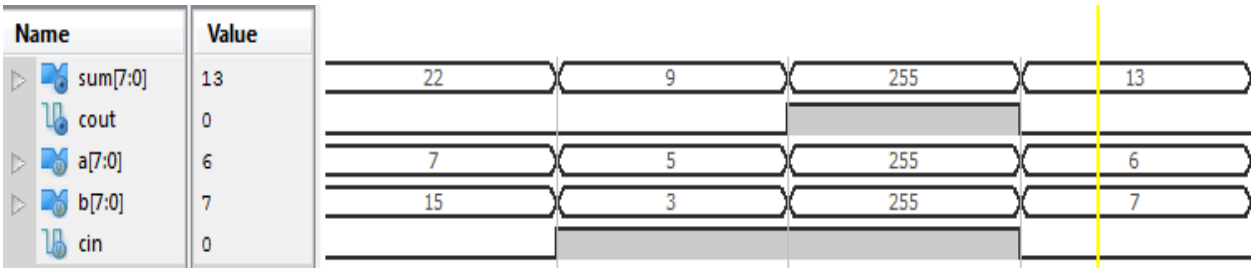
```
module adder8(
   output [7:0] sum,
   output cout,
   input [7:0] a,
   input [7:0] b,
   input cin
   );
wire [6:0] c;
fa_struct x1(sum[0],c[0],a[0],b[0],cin);
fa_struct x2(sum[1],c[1],a[1],b[1],c[0]);
fa_struct x3(sum[2],c[2],a[2],b[2],c[1]);
fa_struct x4(sum[3],c[3],a[3],b[3],c[2]);
fa_struct x5(sum[4],c[4],a[4],b[4],c[3]);
```

```
fa_struct x6(sum[5],c[5],a[5],b[5],c[4]);
fa_struct x7(sum[6],c[6],a[6],b[6],c[5]);
fa_struct x8(sum[7],cout,a[7],b[7],c[6]);
endmodule
```

TEST BENCH:

```
module adder8_test;
        reg [7:0] a,b;
        reg cin;
        wire [7:0] sum;
        wire cout;
        adder8 uut (.sum(sum),.cout(cout),.a(a),.b(b),.cin(cin));
initial
begin
    a=8'b00000111;b=8'b00001111;cin=1'b0;
#5 a=8'b00000101;b=8'b00000011;cin=1'b1;
#5 a=8'b11111111;b=8'b11111111;cin=1'b1;
#5 a=8'b00000110;b=8'b00000111;cin=1'b0;
endinitial $monitor($time,"a=%d,b=%d,cin=%d,sum=%d,cout=%d\n",a,b,cin,sum,cout);
initial #20 $stop;
endmodule
```

SIMULATED WAVEFORM:



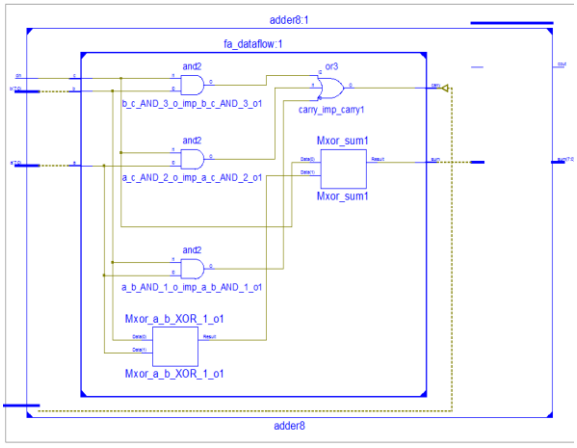
0a=7,b=15,cin=0,sum= 22,cout=0
5a= 5,b= 3,cin=1,sum= 9,cout=0
10a=255,b=255,cin=1,sum=255,cout=1
15a= 6,b= 7,cin=0,sum= 13,cout=0

SIMULATED RESULT:

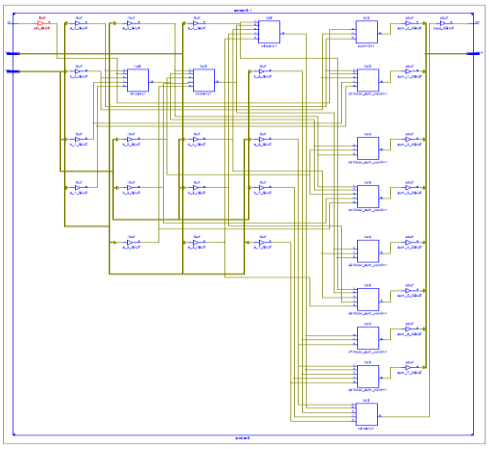
```
Timing Details:
---------------
All values displayed in nanoseconds (ns)
=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 97 / 9
-------------------------------------------------------------------------
Delay:               8.851ns (Levels of Logic = 6)
  Source:            a<1> (PAD)
  Destination:       sum<7> (PAD)
  Data Path: a<1> to sum<7>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
```

```
 IBUF:I->O             2    1.228   1.047  a_1_IBUF (a_1_IBUF)
 LUT5:I0->O            3    0.254   0.759  x2/carry1 (c<1>)
 LUT5:I3->O            3    0.250   0.759  x4/carry1 (c<3>)
 LUT5:I3->O            3    0.250   0.759  x6/carry1 (c<5>)
 LUT5:I3->O            1    0.250   0.579  x8/carry1 (cout_OBUF)
 OBUF:I->O                  2.715          cout_OBUF (cout)
----------------------------------------
Total                      8.851ns (4.947ns logic, 3.904ns route)
                                   (55.9% logic, 44.1% route)
```

RTL VIEW:



TECHNOLOGY VIEW:



POWER ANALYSIS CODE

```
module adder8_power( output [7:0] sum,output cout, resetout, input [7:0] a, b, input cin, clk, reset );
wire [7:0] sum1; wire [6:0] c;
wire dcm_clk,cout1; reg [7:0] sum;reg cout;
fa_struct x1(sum1[0],c[0],a[0],b[0],cin); fa_struct x2(sum1[1],c[1],a[1],b[1],c[0]);
fa_struct x3(sum1[2],c[2],a[2],b[2],c[1]); fa_struct x4(sum1[3],c[3],a[3],b[3],c[2]);
fa_struct x5(sum1[4],c[4],a[4],b[4],c[3]); fa_struct x6(sum1[5],c[5],a[5],b[5],c[4]);
fa_struct x7(sum1[6],c[6],a[6],b[6],c[5]); fa_struct x8(sum1[7],cout1,a[7],b[7],c[6]);
dcm_clk uuu(clk,dcm_clk,reset,resetout);
always @ (posedge dcm_clk or posedge reset)
begin
if (reset)
begin
sum<=8'b00000000; cout<=8'b00000000;
end
else
begin
sum<=sum1; cout<=cout1;
end
end
endmodule
```

POWER REPORT

On-Chip Power Summary				
On-Chip	Power (mW)	Used	Available	Utilization (%)
Clocks	3.01	1	---	---
Logic	0.09	8	2400	0.3
Signals	0.77	40	---	---
IOs	11.75	29	102	28.4
DCMs	14.11	1	4	25.0
Quiescent	14.00			
Total	43.73			

Thermal Summary	
Effective TJA (C/W)	42.4
Max Ambient (C)	83.1
Junction Temp (C)	26.9

Power Supply Summary			
	Total	Dynamic	Quiescent
Supply Power (mW)	43.73	19.01	24.71

RESULT

Exp. No. : Date:	DESIGN ENTRY, SIMULATION, SYNTHESIS, P&R, POST P&R SIMULATION OF 4-BIT ARRAY MULTIPLIER

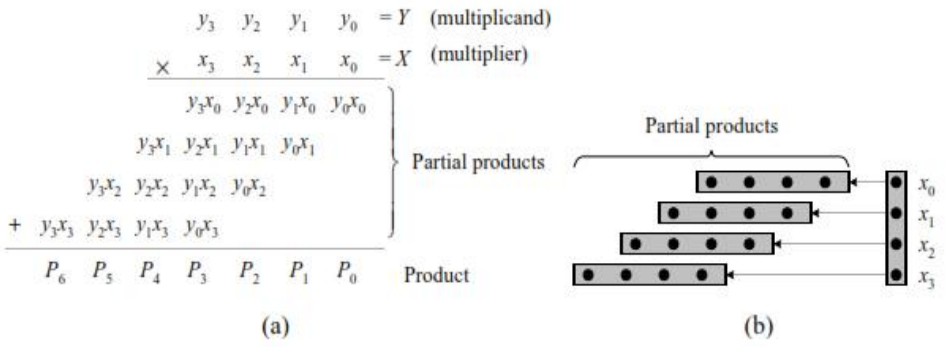
AIM:

To write a Verilog code for the 4-bit array multiplier and to simulate, synthesis, P&R, post P&R simulation and hardware fusingusing Xilinx project navigator.

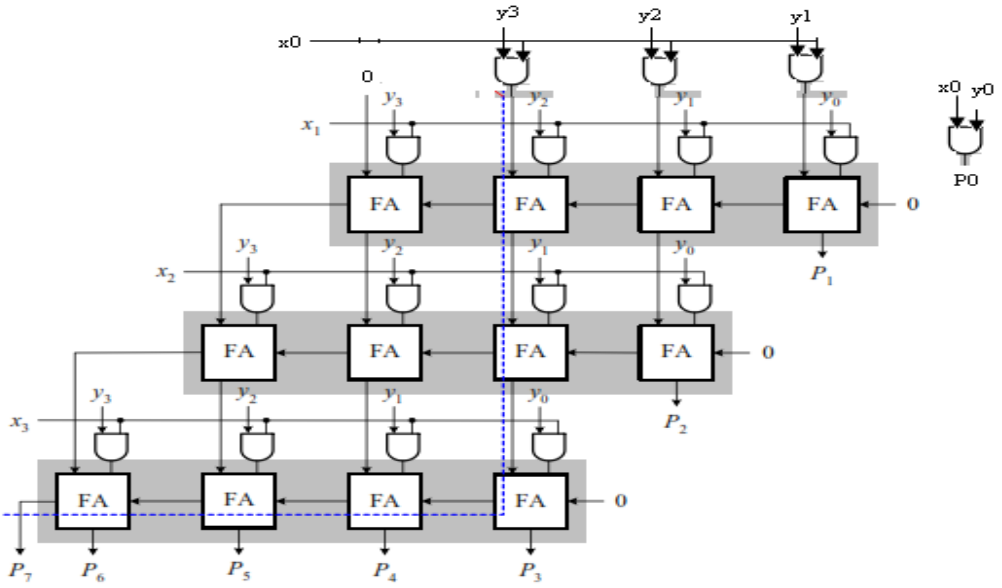
APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

4-BIT ARRAY MULTIPLIER



The operation of multiplication with two 4-bit operands: (a) partial products; (b) dot diagram.



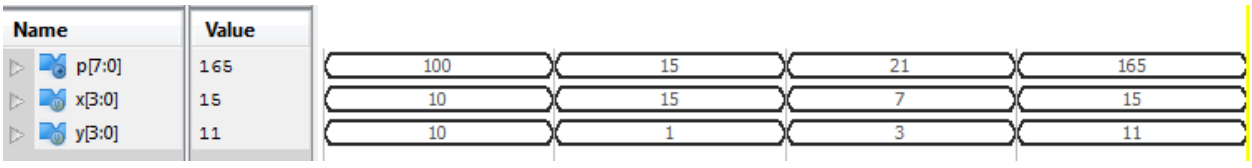
SOURCE CODE:

```
module arraymul( output [7:0] p, input [3:0] x,  input [3:0] y   );
wire [3:1] a; wire [4:1] b;
wire w1,w2,w3,c1,c2,c3,c4;
wire [4:1] e;
wire f1,f2,f3,v1,v2,v3; wire g1,g2,g3;
wire [4:1] d;
assign p[0]=x[0]&y[0];
assign a[1]=x[0]&y[1]; assign a[2]=x[0]&y[2]; assign a[3]=x[0]&y[3]; assign b[1]=x[1]&y[0];
assign b[2]=x[1]&y[1]; assign b[3]=x[1]&y[2]; assign b[4]=x[1]&y[3];
fa_dataflow u1(p[1],c1,a[1],b[1],1'b0);
fa_dataflow u2(w1,c2,a[2],b[2],c1);
fa_dataflow u3(w2,c3,a[3],b[3],c2);
fa_dataflow u4(w3,c4,1'b0,b[4],c3);
assign d[1]=x[2]&y[0]; assign d[2]=x[2]&y[1]; assign d[3]=x[2]&y[2]; assign d[4]=x[2]&y[3];
fa_dataflow u5(p[2],g1,d[1],w1,1'b0);
fa_dataflow u6(v1,g2,d[2],w2,g1);
fa_dataflow u7(v2,g3,d[3],w3,g2);
fa_dataflow u8(v3,g4,c4,d[4],g3);
assign e[1]=x[3]&y[0];assign e[2]=x[3]&y[1];assign e[3]=x[3]&y[2];assign e[4]=x[3]&y[3];
fa_dataflow u9(p[3],f1,e[1],v1,1'b0);
fa_dataflow u10(p[4],f2,e[2],v2,f1);
fa_dataflow u11(p[5],f3,e[3],v3,f2);
fa_dataflow u12(p[6],p[7],e[4],g4,f3);
endmodule
```

TEST BENCH:

```
module arraymul_test;
reg [3:0] x;reg [3:0] y;
wire [7:0] p;
arraymul uut (.p(p),.x(x),.y(y));
initial
begin
x=4'b1010;y=4'b1010;#5 x=4'b1111;y=4'b0001;
#5 x=4'b0111;y=4'b0011;#5 x=4'b1111;y=4'b1011;
end
initial $monitor($time,"x=%d,y=%d,p=%d\n",x,y,p);
initial #20 $stop;
endmodule
```

SIMULATED WAVEFORM:

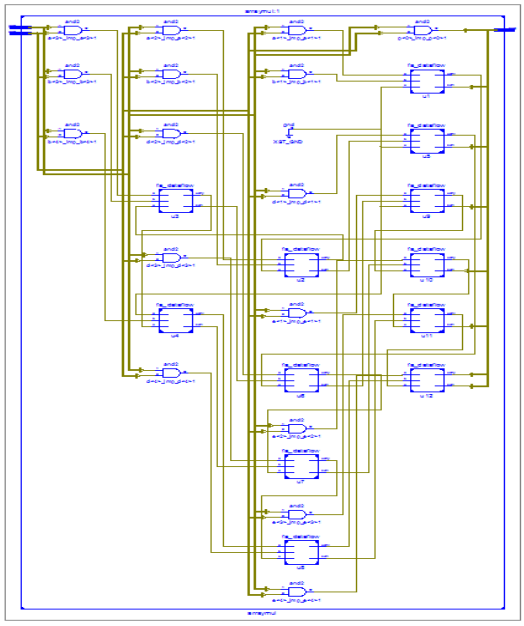


0x=10,y=10,p=100
5x=15,y= 1,p= 15
10x= 7,y= 3,p= 21
15x=15,y=11,p=165

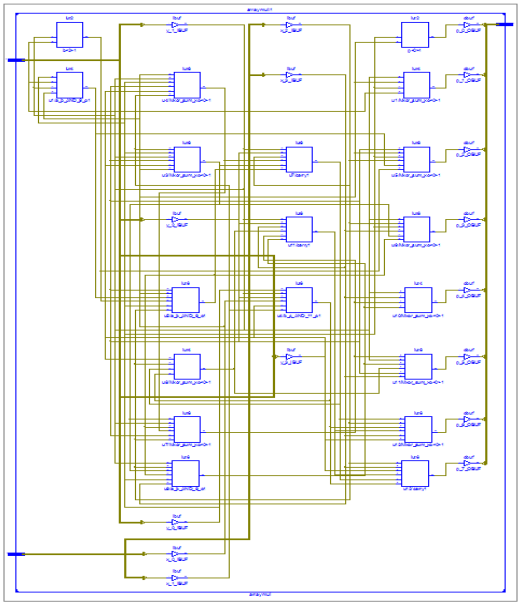
SIMULATED RESULT:

```
Timing Summary:
---------------
Speed Grade: -3
   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 11.800ns
All values displayed in nanoseconds (ns)
=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 394 / 8
-------------------------------------------------------------------------
Delay:               11.800ns (Levels of Logic = 8)
  Source:            y<1> (PAD)
  Destination:       p<7> (PAD)
  Data Path: y<1> to p<7>
                                Gate     Net
    Cell:in->out      fanout   Delay    Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             13   1.228    1.161  y_1_IBUF (y_1_IBUF)
     LUT4:I1->O             2   0.235    0.845  u1/a_b_AND_9_o1 (c1)
     LUT6:I3->O             4   0.235    1.139  u5/a_b_AND_9_o1 (g1)
     LUT6:I0->O             3   0.254    0.879  u7/carry1 (g3)
     LUT4:I1->O             2   0.235    0.893  u8/Mxor_sum_xo<0>1 (v3)
     LUT6:I2->O             2   0.254    0.893  u11/carry1 (f3)
     LUT6:I2->O             1   0.254    0.579  u12/carry1 (p_7_OBUF)
     OBUF:I->O                  2.715           p_7_OBUF (p<7>)
    ----------------------------------------
    Total                      11.800ns (5.410ns logic, 6.390ns route)
                                        (45.8% logic, 54.2% route)
```

RTL VIEW:



TECHNOLOGY VIEW:



POWER ANALYSIS CODE

```
module arraymul_power( output [7:0] p, output resetout, input [3:0] x, input [3:0] y, input clk, input reset);
wire dcm_clk;
wire [3:1] a;wire [4:1] b;
wire w1,w2,w3,c1,c2,c3,c4; wire [4:1] e;
wire f1,f2,f3,v1,v2,v3; wire [4:1] d;
wire g1,g2,g3; reg [7:0] p;
wire [7:0] p1;
assign p1[0]=x[0]&y[0]; assign a[1]=x[0]&y[1]; assign a[2]=x[0]&y[2]; assign a[3]=x[0]&y[3]; assign b[1]=x[1]&y[0];
assign b[2]=x[1]&y[1]; assign b[3]=x[1]&y[2]; assign b[4]=x[1]&y[3]; fa_dataflow u1(p1[1],c1,a[1],b[1],1'b0);
fa_dataflow u2(w1,c2,a[2],b[2],c1);fa_dataflow u3(w2,c3,a[3],b[3],c2);
fa_dataflow u4(w3,c4,1'b0,b[4],c3);
assign d[1]=x[2]&y[0]; assign d[2]=x[2]&y[1];assign d[3]=x[2]&y[2];assign d[4]=x[2]&y[3];
fa_dataflow u5(p1[2],g1,d[1],w1,1'b0); fa_dataflow u6(v1,g2,d[2],w2,g1);
fa_dataflow u7(v2,g3,d[3],w3,g2); fa_dataflow u8(v3,g4,c4,d[4],g3);
assign e[1]=x[3]&y[0]; assign e[2]=x[3]&y[1]; assign e[3]=x[3]&y[2]; assign e[4]=x[3]&y[3];
fa_dataflow u9(p1[3],f1,e[1],v1,1'b0); fa_dataflow u10(p1[4],f2,e[2],v2,f1);
fa_dataflow u11(p1[5],f3,e[3],v3,f2); fa_dataflow u12(p1[6],p1[7],e[4],g4,f3);
dcm_clk uuu(clk,dcm_clk,reset,resetout);
always @ (posedge dcm_clk or posedge reset)
begin
if (reset)
p<=8'b00000000;
else
begin
p<=p1;
end
end
endmodule
```

POWER REPORT

On-Chip Power Summary				
On-Chip	Power (mW)	Used	Available	Utilization (%)
Clocks	3.00	1	---	---
Logic	0.10	17	2400	0.7
Signals	0.24	37	---	---
IOs	8.67	19	102	18.6
DCMs	14.11	1	4	25.0
Quiescent	13.96			
Total	40.08			

Thermal Summary	
Effective TJA (C/W)	42.4
Max Ambient (C)	83.3
Junction Temp (C)	26.7

Power Supply Summary			
	Total	Dynamic	Quiescent
Supply Power (mW)	40.08	15.40	24.68

RESULT

Exp. No. : 3 Date:	DESIGN ENTRY, SIMULATION, SYNTHESIS, P&R, POST P&R SIMULATION OF ALU

AIM:

To write a Verilog code for the 4-bit arithmetic and logic unit and to simulate, synthesis, P&R, post P&R simulation and hardware fusingusing Xilinx project navigator.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

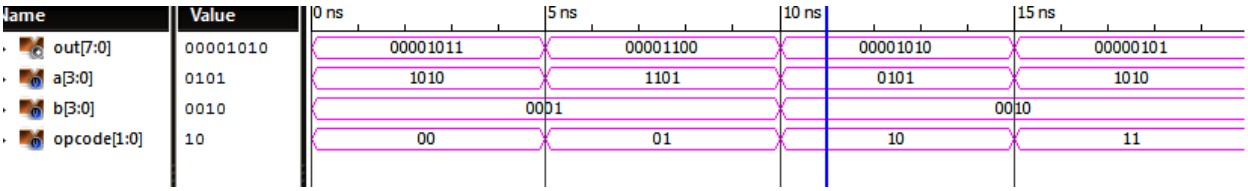
SOURCE CODE:

```
module beh_alu(out,a,b,opcode);
output[7:0] out;
input [3:0] a,b;
input [1:0] opcode;
reg [7:0] out;
parameter
ADD = 2'b00,
SUB = 2'b01,
MUL = 2'b10,
DIV = 2'b11;
always @(a,b,opcode)
case(opcode)
ADD: out=a+b;
SUB: out=a-b;
MUL: out=a*b;
DIV: out=a/b;
endcase
endmodule
```

TEST BENCH:

```
module stimu_beh_alu;
wire [7:0] out;
reg [3:0] a,b;
reg [1:0] opcode;
beh_alu u1(out,a,b,opcode);
initial
begin
```

```
opcode = 2'b00;a=4'b1010;b=4'b0001;
#5 opcode = 2'b01;a=4'b1101;b=4'b0001;
#5 opcode = 2'b10;a=4'b0101;b=4'b0010;
#5 opcode = 2'b11;a=4'b1010;b=4'b0010;
end
initial
$monitor($time,"opcode=%d,a=%d,b=%d,out=%d\n",opcode,a,b,out);
initial #20 $stop;
endmodule
```



SIMULATION OUTPUT

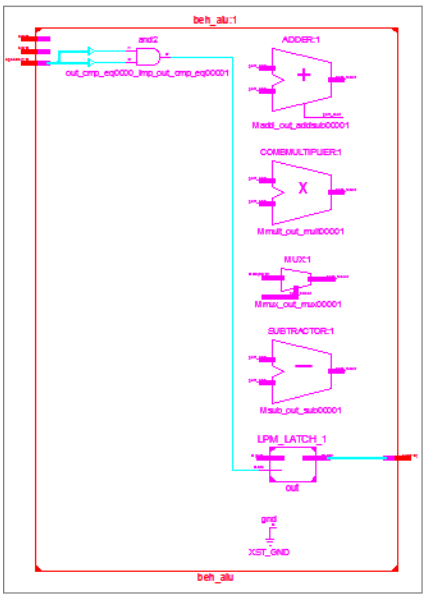
```
# 0opcode=0,a=10,b= 1,out= 11
# 5opcode=1,a=13,b= 1,out= 12
# 10opcode=2,a= 5,b= 2,out= 10
# 15opcode=3,a=10,b= 2,out= 5
```

TIMING SUMMARY:

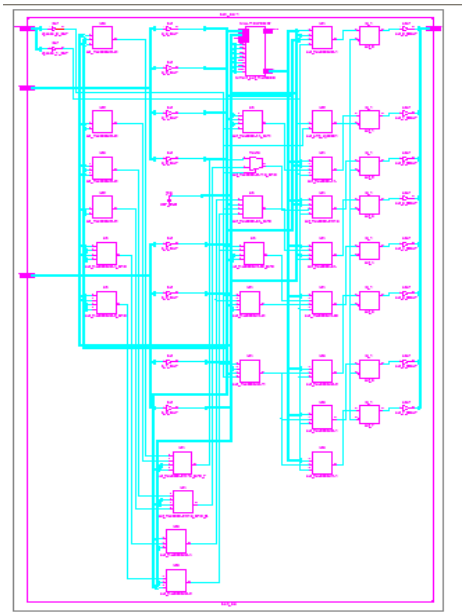
```
---------------
Speed Grade: -4
  Minimum period: No path found
  Minimum input arrival time before clock: 7.786ns
  Maximum output required time after clock: 4.368ns
  Maximum combinational path delay: No path found

Timing Detail:
--------------
All values displayed in nanoseconds (ns)
=========================================================================
Timing constraint: Default OFFSET IN BEFORE for Clock 'out_cmp_eq0000'
 Total number of paths / destination ports: 143 / 8
-------------------------------------------------------------------------
Offset:            7.786ns (Levels of Logic = 3)
 Source:           b<1> (PAD)
 Destination:      out_7 (LATCH)
 Destination Clock: out_cmp_eq0000 rising

 Data Path: b<1> to out_7
                       Gate     Net
  Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
  ----------------------------------------  ------------
   IBUF:I->O             8   1.218   0.757  b_1_IBUF (b_1_IBUF)
   MULT18X18SIO:B1->P7   1   4.344   0.455  Mmult_out_mult0000 (out_mult0000<7>)
   LUT3:I2->O            1   0.704   0.000  out_mux0000<7>1 (out_mux0000<7>)
   LD_1:D                    0.308          out_7
  ----------------------------------------
```

```
    Total                     7.786ns (6.574ns logic, 1.212ns route)
                              (84.4% logic, 15.6% route)

=========================================================================
Timing constraint: Default OFFSET OUT AFTER for Clock 'out_cmp_eq0000'
  Total number of paths / destination ports: 8 / 8
-------------------------------------------------------------------------
Offset:              4.368ns (Levels of Logic = 1)
  Source:            out_7 (LATCH)
  Destination:       out<7> (PAD)
  Source Clock:      out_cmp_eq0000 rising

  Data Path: out_7 to out<7>
                            Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     LD_1:G->Q             1   0.676   0.420  out_7 (out_7)
     OBUF:I->O                 3.272          out_7_OBUF (out<7>)
    ----------------------------------------
    Total                     4.368ns (3.948ns logic, 0.420ns route)
                              (90.4% logic, 9.6% route)
```

RTL VIEW



TECHNOLOGY VIEW



RESULT:

Exp. No. : 4 Date:	DESIGN ENTRY, SIMULATION, SYNTHESIS, P&R, POST P&R SIMULATION OF UNIVERSAL SHIFT REGISTER

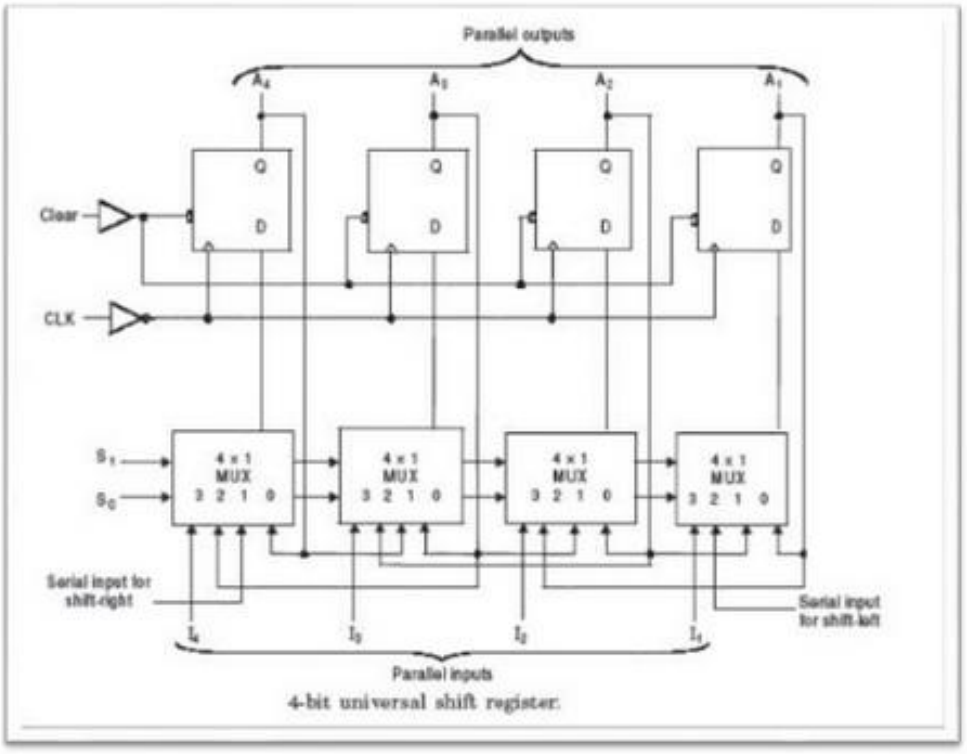
AIM:

To write a Verilog code for 4-bit universal shift register and to simulate, synthesis, P&R, post P&R simulation and hardware fusingusing Xilinx project navigator.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

BLOCK DIAGRAM



4-bit universal shift register.

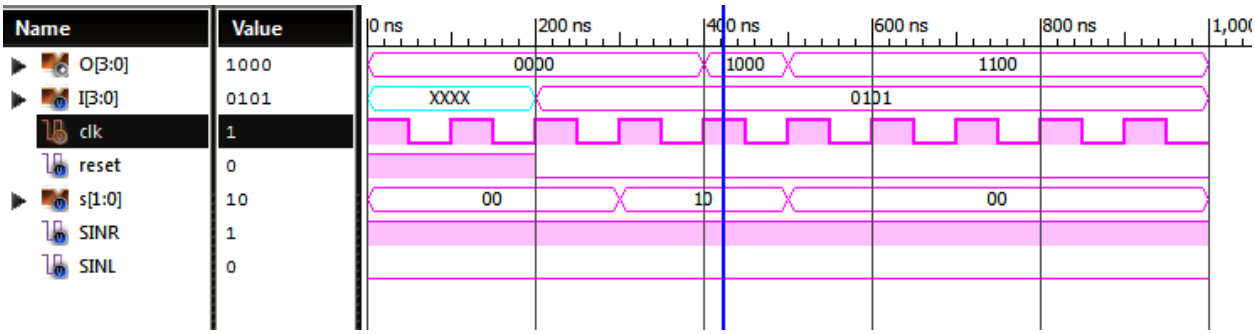
SOURCE CODE

```
module D_FF(q,d,clk,reset);
output reg q;
input d,clk,reset;
always@(posedge clk)
if(reset==1'b1)
q=1'b0;
else
q=d;
endmodule

module mux_4_1(y,s1,s0,i3,i2,i1,i0);
output reg y;
input i3,i2,i1,i0;
input s1,s0;
always@(s1,s0,i3,i2,i1,i0)
begin
if(s0==0&s1==0)
y=i0;
if(s0==0&s1==1)
y=i1;
if(s0==1&s1==0)
y=i2;
if(s0==1&s1==1)
y=i3;
end
endmodule

module USR(O,I,clk,reset,s,SINR,SINL);
wire[3:0]w;
input[3:0]I;
input[1:0]s;
input clk;
input reset,SINR,SINL;
output[3:0]O;
mux_4_1 m1(w[0],s[1],s[0],I[0],SINL,O[1],O[0]);
mux_4_1 m2(w[1],s[1],s[0],I[1],O[0],O[2],O[1]);
mux_4_1 m3(w[2],s[1],s[0],I[2],O[1],O[3],O[2]);
mux_4_1 m4(w[3],s[1],s[0],I[3],O[2],SINR,O[3]);
D_FF d1(O[0],w[0],clk,reset);
D_FF d2(O[1],w[1],clk,reset);
D_FF d3(O[2],w[2],clk,reset);
D_FF d4(O[3],w[3],clk,reset);
endmodule
```

TEST BENCH

```
module uni_testbench;
reg[3:0]I;
reg clk;
reg reset;
reg[1:0]s;
```

```
reg SINR;
reg SINL;
wire[3:0]O;

USR uut( .O(O),.I(I),.clk(clk),.reset(reset),.s(s),.SINR(SINR),.SINL(SINL));
initial begin
//Initialize
//I=4'b0000;
clk=1'b1;
reset=1'b1;
SINR=1'b1;
SINL=1'b0;
s=2'b00;
#100;
reset=1'b1;
#100;
I=4'b0101;
reset=1'b0;
#100;
s=2'b10;
#100;
s=2'b10;
#100;
s=2'b00;
end
always#50 clk=~clk;
endmodule
```



TIMING REPORT:

Minimum period: No path found
Minimum input arrival time before clock: 4.047ns
Maximum output required time after clock: 4.283ns
Maximum combinational path delay: No path found

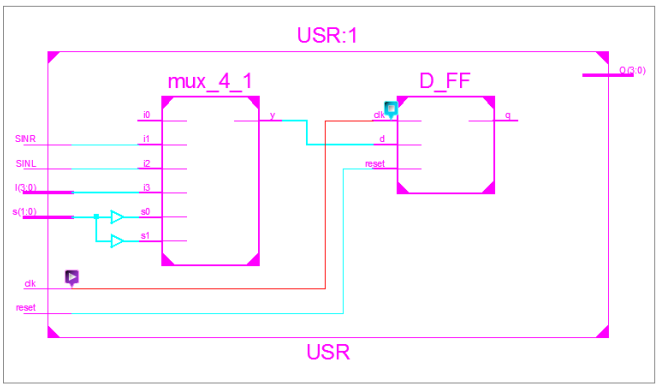
TIMING DETAIL:

All values displayed in nanoseconds (ns)

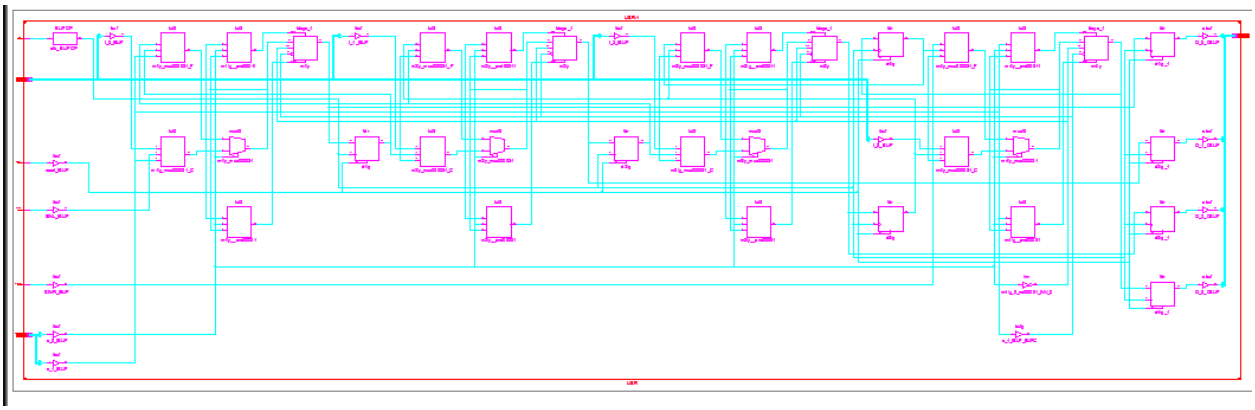
```
================================================================================
Timing constraint: Default OFFSET IN BEFORE for Clock 'clk'
  Total number of paths / destination ports: 8 / 8
--------------------------------------------------------------------------------
Offset:              2.886ns (Levels of Logic = 1)
  Source:            reset (PAD)
```

```
  Destination:       d4/q (FF)
  Destination Clock: clk rising

  Data Path: reset to d4/q
                        Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             8   1.218   0.757  reset_IBUF (reset_IBUF)
     FDR:R                      0.911          d4/q
    ----------------------------------------
    Total                      2.886ns (2.129ns logic, 0.757ns route)
                               (73.8% logic, 26.2% route)
=========================================================================
Timing constraint: Default OFFSET IN BEFORE for Clock 's<1>'
  Total number of paths / destination ports: 22 / 8
-------------------------------------------------------------------------
Offset:              4.047ns (Levels of Logic = 2)
  Source:            s<0> (PAD)
  Destination:       m4/y (LATCH)
  Destination Clock: s<1> rising

  Data Path: s<0> to m4/y
                        Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O            13   1.218   0.983  s_0_IBUF (s_0_IBUF)
     INV:I->O              4   0.704   0.587  m4/y_0_not00001_INV_0 (m1/y_0_not0000)
     LDCPE_1:GE                 0.555          m1/y
    ----------------------------------------
    Total                      4.047ns (2.477ns logic, 1.570ns route)
                               (61.2% logic, 38.8% route)

=========================================================================
Timing constraint: Default OFFSET OUT AFTER for Clock 'clk'
  Total number of paths / destination ports: 4 / 4
-------------------------------------------------------------------------
Offset:              4.283ns (Levels of Logic = 1)
  Source:            d4/q_1 (FF)
  Destination:       O<3> (PAD)
  Source Clock:      clk rising

  Data Path: d4/q_1 to O<3>
                        Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDR:C->Q              1   0.591   0.420  d4/q_1 (d4/q_1)
     OBUF:I->O                     3.272          O_3_OBUF (O<3>)
    ----------------------------------------
    Total                      4.283ns (3.863ns logic, 0.420ns route)
                               (90.2% logic, 9.8% route)
```

RTL VIEW



TECHNOLOGY VIEW



RESULT:

Exp. No. : Date:	**DESIGN ENTRY, SIMULATION, SYNTHESIS, P&R, POST P&R SIMULATION OF FINITE STATE MACHINE**

AIM:

To write a Verilog code for finite state machine and to simulate, synthesis, P&R, post P&R simulation and hardware fusingusing Xilinx project navigator.

APPARATUS REQUIRED:

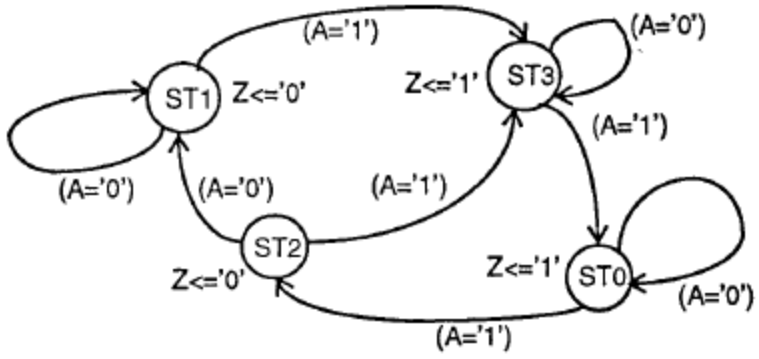
S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

ALGORITHM:

Step1: Define the specifications and initialize the design.
Step2: Declare the name of the entity and architecture by using Verilog source code.
Step3: Write the source code in VERILOG.
Step4: Check the syntax and debug the errors if found, obtain the synthesis report.
Step5: Verify the output by simulating the source code.
Step6: Write all possible combinations of input using the test bench.
Step7: Obtain the place and route report.

LOGIC DIAGRAM:

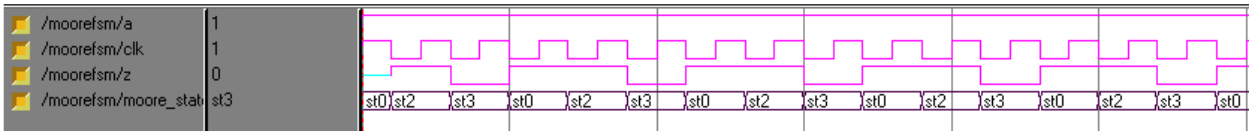
MOORE FSM:



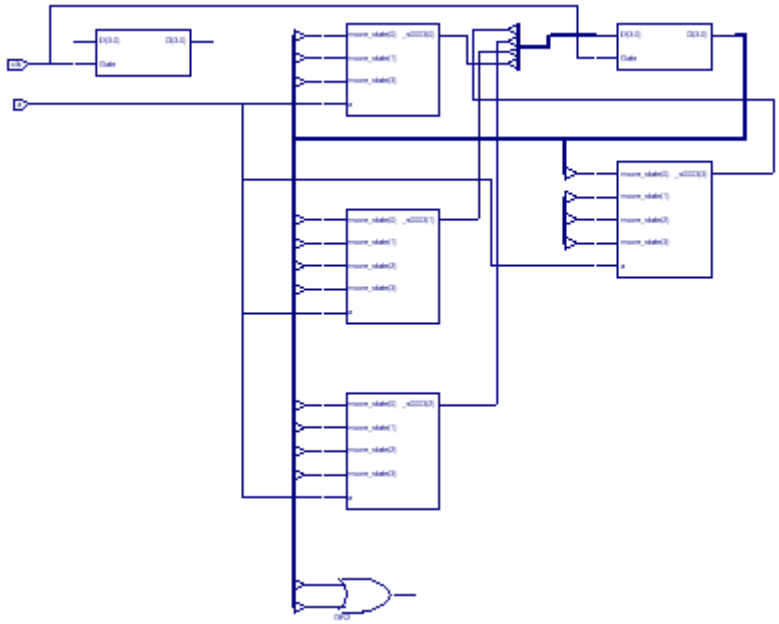
VERILOG SOURCE CODE:
BEHAVIORAL MODELING:

```
module moorefsm(a,clk,z);
input a;
input clk;
output z;
reg z;
parameter st0=0,st1=1,st2=2,st3=3;
reg[0:1]moore_state;
initial
begin
moore_state=st0;
end
always @ (posedge(clk))
case(moore_state)
st0:
begin
z=1;
if(a)
moore_state=st2;
end
st1:
begin
z=0;
if(a)
moore_state=st3;
end
st2:
begin
z=0;
if(~a)
moore_state=st1;
else
moore_state=st3;
end
st3:
begin
z=1;
if(a)
moore_state=st0;
end
endcase
endmodule
```

SIMULATION OUTPUT :



SYNTHESIS RTL SCHEMATIC:



MEALY FSM:
TRUTH TABLE:

Present state	0	1
ST0	ST0 / 0	ST3 / 1
ST1	ST1 / 1	ST0 / 0
ST2	ST2 / 0	ST1 / 1
ST3	ST2 / 0	ST1 / 0

Input A

(Entries in table are next state and output Z)

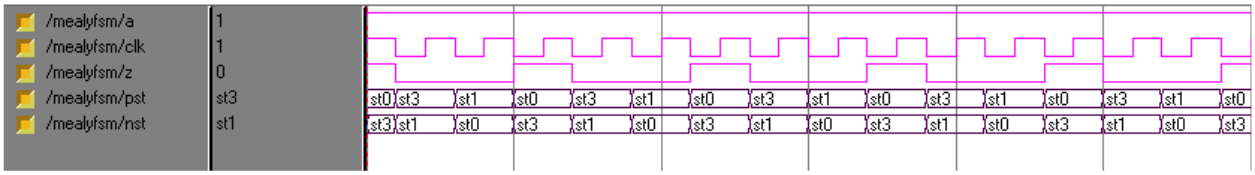
VERILOG SOURCE CODE:

Behavioral Modeling:

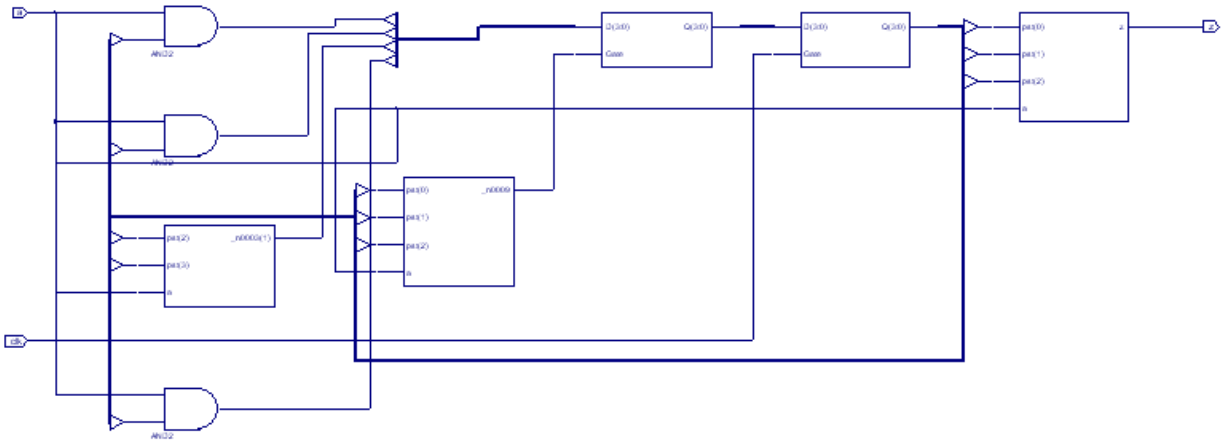
```
module mealayfsm(a, clk, z);
input a;
input clk;
output z;
reg z;
parameter st0=0,st1=1,st2=2,st3=3;
reg[0:1]mealy_state;
initial
```

```
begin
mealy_state=st0;
end
always @ (posedge(clk))
case(mealy_state)
st0:
begin
if(a) begin
z=1;
mealy_state=st3; end
else
z=0;
end
st1:
begin
if(a) begin
z=0;
mealy_state=st0; end
else
z=1;
end
st2:
begin
if(a) begin
z=1;
mealy_state=st1; end
else
z=0;
end
st3:
begin
z=0;
if(a) begin
mealy_state=st1; end
else
mealy_state=st2;
end
endcase
endmodule
```

SIMULATION OUTPUT:



SYNTHESIS RTL SCHEMATIC:



RESULT

Exp. No. : Date:	DESIGN ENTRY, SIMULATION, SYNTHESIS, P&R, POST P&R SIMULATION RAM MEMORY

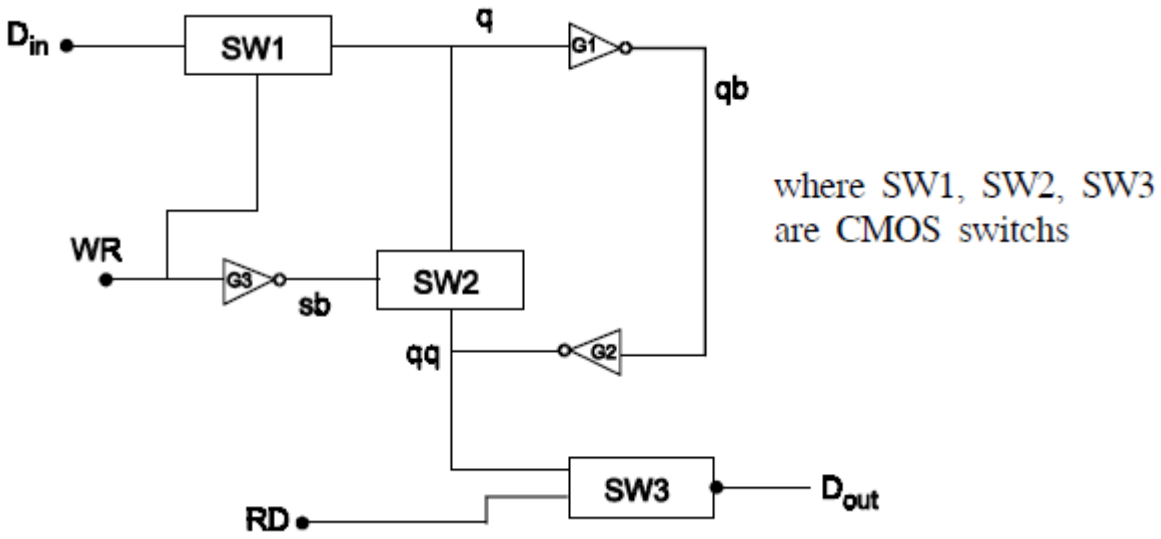
AIM:

To write a Verilog code for RAM and to simulate, synthesis, P&R, post P&R simulation and hardware fusingusing Xilinx project navigator.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

BLOCK DIAGRAM



SOURCE CODE:

```
module ram(dout,din,wr,rd);
// declaration of I/O ports
output dout;
input din,wr,rd;
// declare internal wire
wire q,sb,qq,qb;
// component instantiation
cms sw1(q,din,wr);
```

```
not n1(sb,wr);
cms sw2(q,sb,qq);
not n2(qb,q);
not n3(qq,qb);
cms sw3(dout,qq,rd);
endmodule

module cms(out,in,n_ctr);
// declaration of I/O ports
output out;
input in,n_ctr;
// declare internal wire
wire p_ctr;
assign p_ctr=~n_ctr;
cmos u1(out,in,n_ctr,p_ctr);
endmodule
```

TEST BENCH

```
module stimu_cmos_ram;
// declaration of variables
wire dout;
reg din,wr,rd;
// component instantiation
ram u1(dout,din,wr,rd);
// declaration of inputs
initial
begin
din=1'b0;wr=1'b0;rd=1'b0;
#10 din=1'b0;wr=1'b0;rd=1'b1;
#10 din=1'b0;wr=1'b1;rd=1'b1;
#10 din=1'b0;wr=1'b1;rd=1'b0;
#10 din=1'b0;wr=1'b1;rd=1'b1;
#10 din=1'b1;wr=1'b1;rd=1'b1;
#10 din=1'b1;wr=1'b0;rd=1'b1;
#10 din=1'b1;wr=1'b1;rd=1'b0;
end
initial
monitor($time,"din=%b,wr=%b,rd=%b,dout=%b\n",din,wr,rd,dout);
initial #80 $stop;
endmodule
```

SIMULATED RESULT FOR RAM

0din=0,wr=0,rd=0,dout=z
10din=0,wr=0,rd=1,dout=x
20din=0,wr=1,rd=1,dout=0
30din=0,wr=1,rd=0,dout=z
40din=0,wr=1,rd=1,dout=0
50din=1,wr=1,rd=1,dout=1
60din=1,wr=0,rd=1,dout=1
70din=1,wr=1,rd=0,dout=z

RESULT

Exp. No. : Date:	SYNCHRONOUS AND ASYNCHRONOUS COUNTER

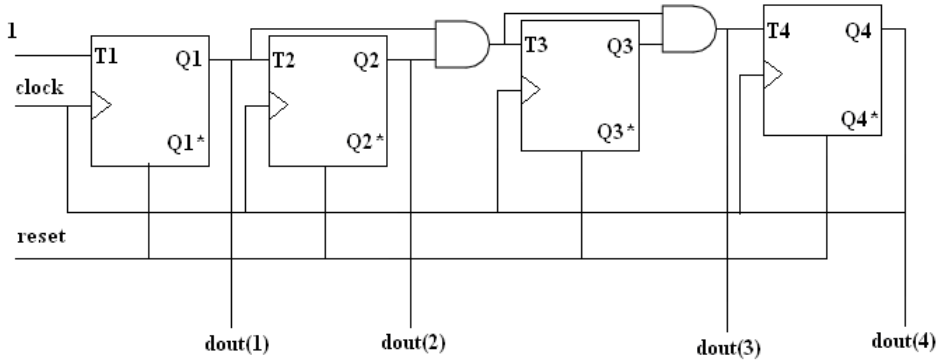
AIM:

To develop the source code for synchronous and asynchronous counter by using VHDL/VERILOG and obtain the simulation, synthesis, place and route and implement into FPGA.

ALGORITM:

Step1: Define the specifications and initialize the design.
Step2: Declare the name of the entity and architecture by using VHDL source code.
Step3: Write the source code in VERILOG.
Step4: Check the syntax and debug the errors if found, obtain the synthesis report.
Step5: Verify the output by simulating the source code.
Step6: Write all possible combinations of input using the test bench.
Step7: Obtain the place and route report.

LOGIC DIAGRAM:



SYNCHRONOUS COUNTER:

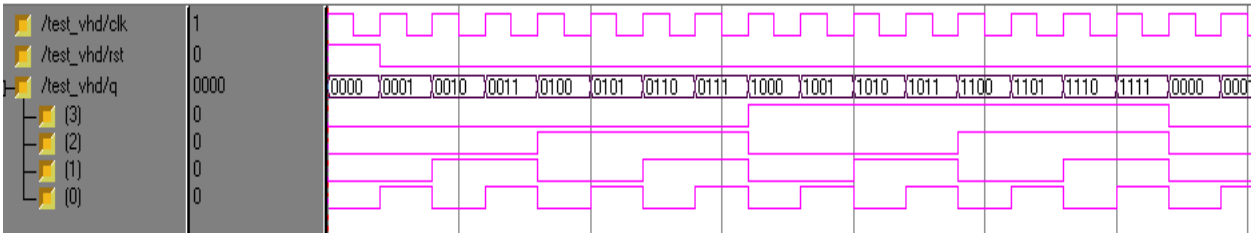
VERILOG SOURCE CODE:

Behavioral Modeling:

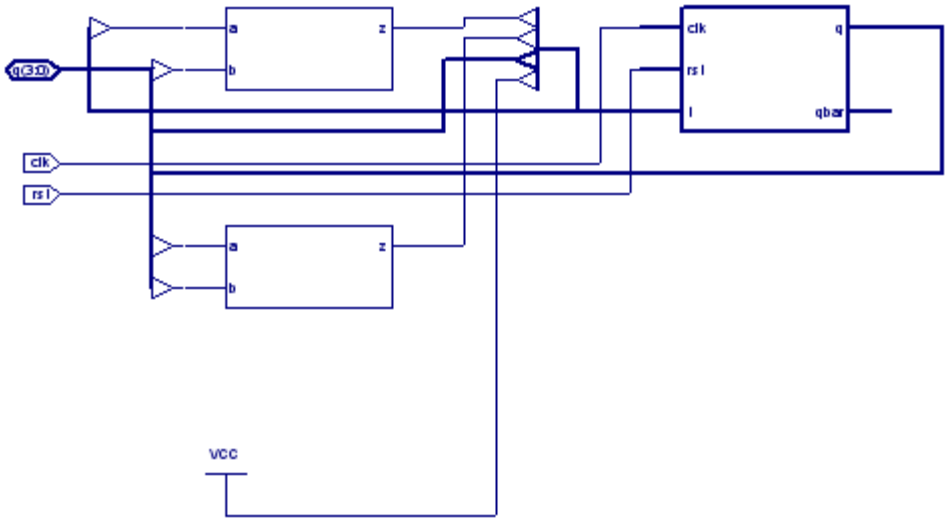
```
module syncntr(clk, rst, q);
  input clk;
  input rst;
  output [3:0]q;
        reg [3:0]q;
        reg [3:0]x=0;
        always @ (posedge(clk) or posedge(rst))
        begin
        if (rst==1'b1)
```

```
begin
q=4'b0;
end
else
begin
x=x+1'b1;
end
q=x;
end
endmodule
```

Simulation output:



Synthesis RTL Schematic:



SYNTHESIS REPORT:

==========

* Final Report *

==========

DEVICE UTILIZATION SUMMARY:

Selected Device : 3s400tq144-5

Number of Slices:	2	out of	3584	0%
Number of Slice Flip Flops:	4	out of	7168	0%
Number of 4 input LUTs:	2	out of	7168	0%

Number of bonded IOBs: 6 out of 97 6%
Number of GCLKs: 1 out of 8 12%

==========

TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

CLOCK INFORMATION:

Clock Signal	Clock buffer(FF name)	Load
clk	BUFGP	4

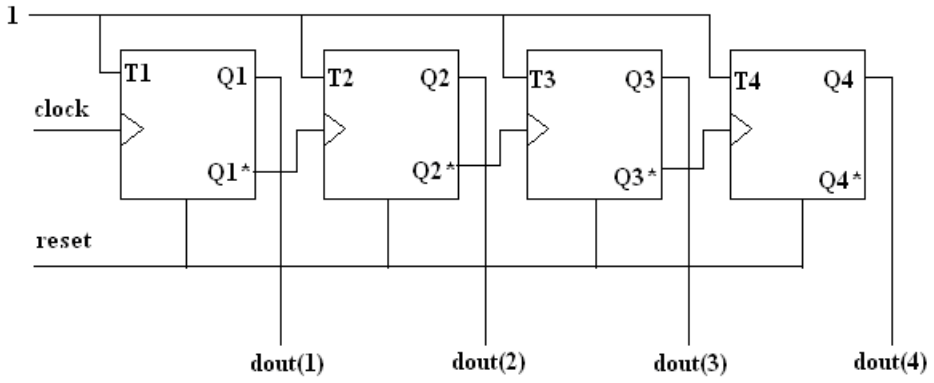
TIMING SUMMARY:

Speed Grade: -5

Minimum period: 3.388ns (Maximum Frequency: 295.186MHz)
Minimum input arrival time before clock: No path found
Maximum output required time after clock: 6.318ns
Maximum combinational path delay: No path found

ASYNCHRONOUS COUNTER:

LOGIC DIAGRAM:



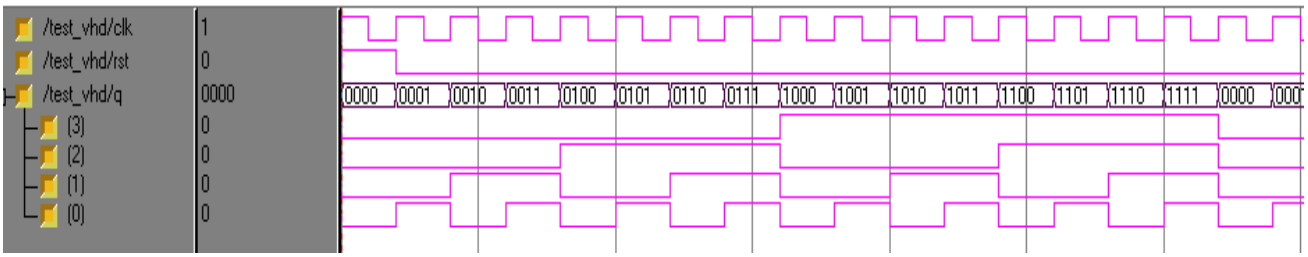
VERILOG SOURCE CODE:

BEHAVIORAL MODELING:

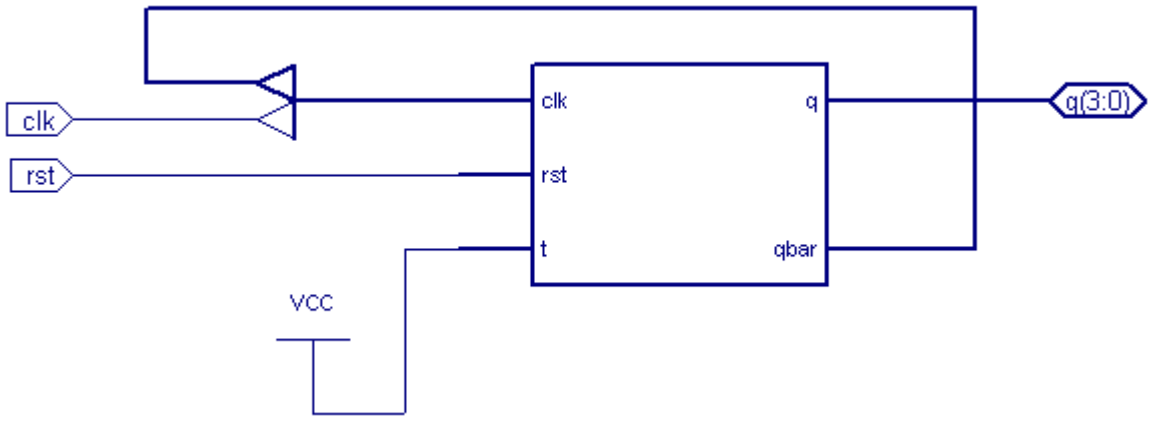
```
module asyncntr(clk, rst, s, q);
   input clk;
   input rst;
   input s;
```

```
output [3:0] q;
        reg [3:0] q;
        reg [3:0] x=0;
        always @ (posedge(clk) or posedge(rst))
        begin
        if (rst==1'b1)
        begin
        q=4'b0;
        end
        else if (s==1'b0)
        begin
        x=x-1'b1;
        end
        else if (s==1'b1)
        begin
        x=x+1'b1;
        end
        q=x;
        end
endmodule
```

Simulation output:



Synthesis RTL Schematic:



Synthesis report:

```
=========================================================================
*                      Final Report                     *
=========================================================================
Device utilization summary:
---------------------------

Selected Device : 3s400tq144-5

 Number of Slices:                  4  out of   3584    0%
 Number of Slice Flip Flops:        7  out of   7168    0%
 Number of bonded IOBs:             6  out of     97    6%
 Number of GCLKs:                   1  out of      8   12%


=========================================================================
```

TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

Clock Information:

```
------------------
-----------------------------------+------------------------+-------+
Clock Signal                       | Clock buffer(FF name)  | Load  |
-----------------------------------+------------------------+-------+
clk                                | BUFGP                  | 2     |
t1/qbar:Q                          | NONE                   | 2     |
t2/qbar:Q                          | NONE                   | 2     |
t3/qbar:Q                          | NONE                   | 1     |
-----------------------------------+------------------------+-------+
```

INFO:Xst:2169 - HDL ADVISOR - Some clock signals were not automatically buffered by XST with BUFG/BUFR resources. Please use the buffer_type constraint in order to insert these buffers to the clock signals to help prevent skew problems.

Timing Summary:

```
---------------
Speed Grade: -5

   Minimum period: 2.733ns (Maximum Frequency: 365.925MHz)
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: 6.280ns
   Maximum combinational path delay: No path found
```

RESULT:

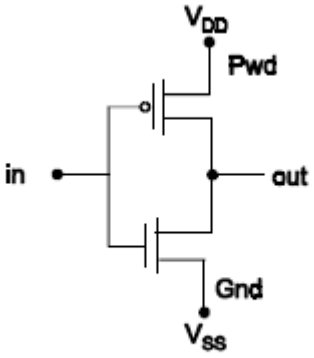
Exp. No. : Date:	**DESIGN ENTRY AND SIMULATION OF CMOS INVERTOR CIRCUIT**

AIM:

To write a Verilog code for **CMOS INVERTOR**simulation and hardware using Xilinx project navigator.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1



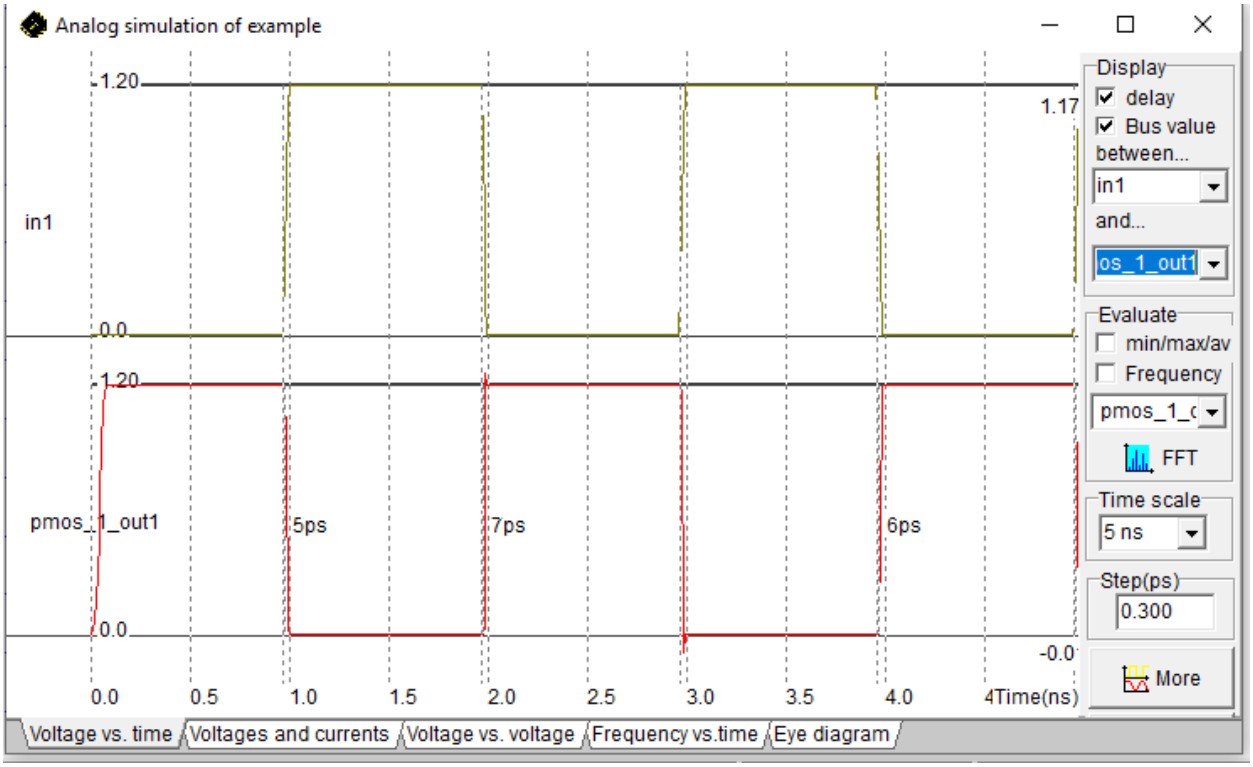
CMOS Inverter

SOURCE CODE

```
module cmos_inverter(out,in);
// declaration of I/O ports
output out;
input in;
// declare power and ground terminals
supply1 pwr;
supply0 gnd;
// component instantiation
pmos u1(out,pwr,in);
nmos u2(out,gnd,in);
endmodule
```

TEST BENCH

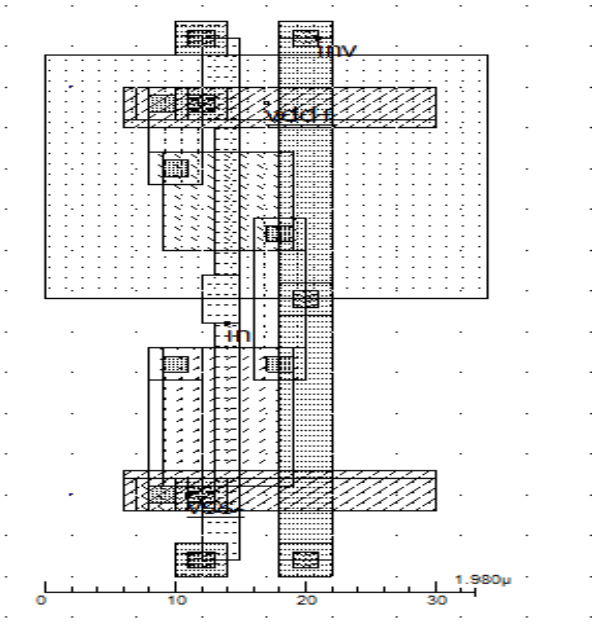
```
module stimu_cmos_invertor;
// declaration of variables
wire out;
reg in;
// component instantiation
cmos_inverter u1(out,in);
// declaration of inputs
initial
begin
in=1'b0;
#10 in=1'b1;
end
initial
$monitor($time,"in=%b,out=%b",in,out);
initial #20 $stop;
endmodule
```



OUTPUT

```
# 0in=0,out=1
# 10in=1,out=0
```

LAYOUT



AREA

35*1.98 = 69.3µ

RESULT

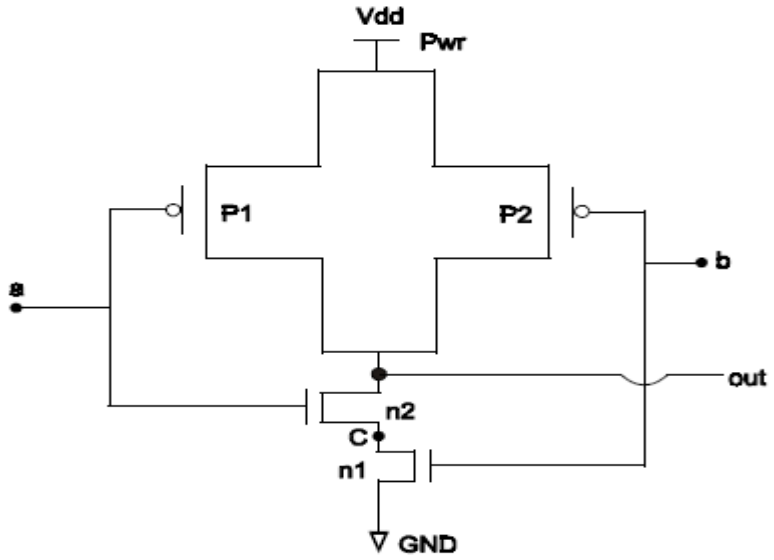
Exp. No. : Date:	**DESIGN ENTRY AND SIMULATION OF CMOS BASIC GATES AND FLIP FLOPS**

AIM:

To write a Verilog code for **CMOS BASIC GATES And FLIP FLOPS** simulation and hardware using Xilinx project navigator.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1



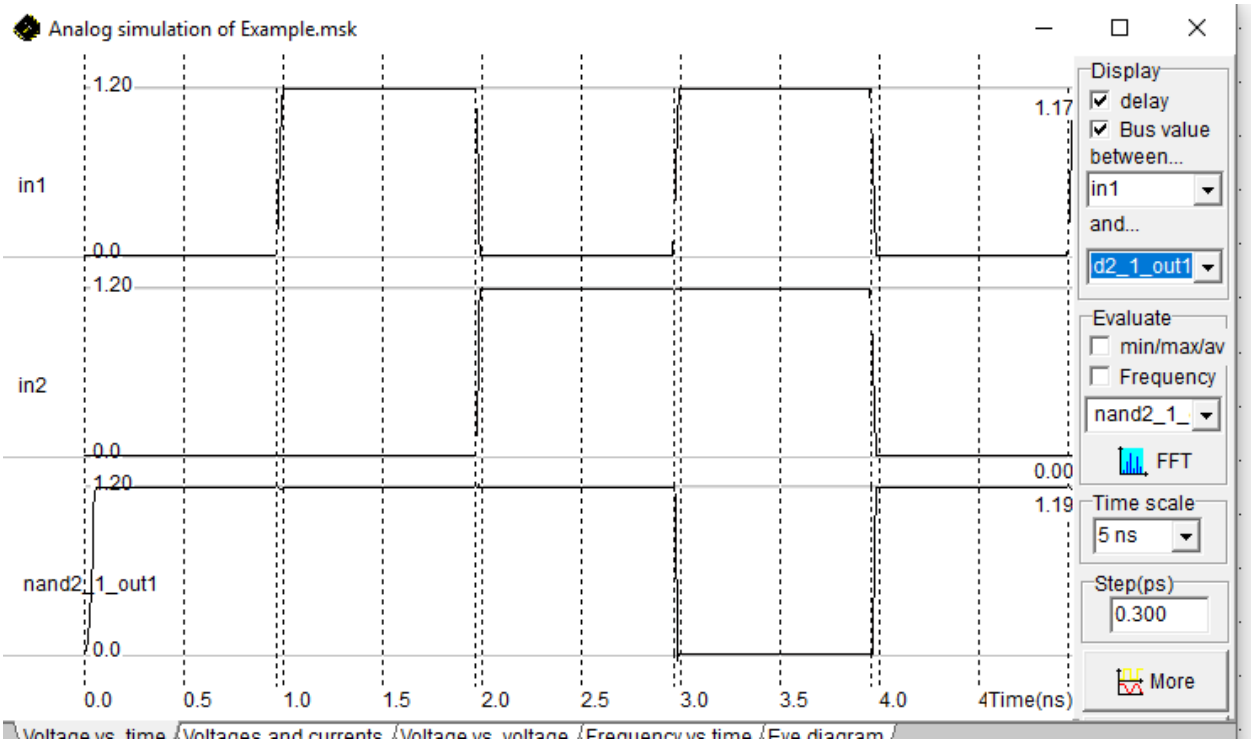

```
module cmos_nand(out,a,b);
output out;
input a,b;
supply1 pwr;
supply0 gnd;
wire c;
nmos u1(out,c,a);
nmos u2(c,gnd,b);
pmos u3(out,pwr,a);
pmos u4(out,pwr,b);
endmodule
```

TEST BENCH

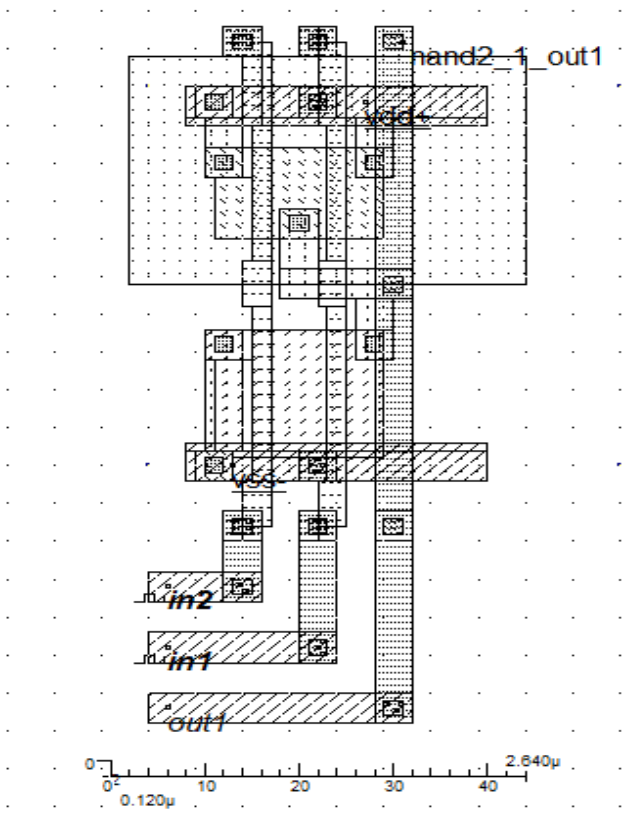
```
module stimu_cmos_nand;
// declaration of variables
wire out;
reg a,b;
// component instantiation
cmos_nand u1(out,a,b);
// declaration of inputs
initial
begin
a=1'b0;b=1'b0;
#10 a=1'b1;b=1'b0;
#10 a=1'b0;b=1'b1;
#10 a=1'b1;b=1'b1;
end
initial
$monitor($time,"a=%b,b=%b,out=%b\n",a,b,out);
initial #40 $stop;
endmodule
```

OUTPUT

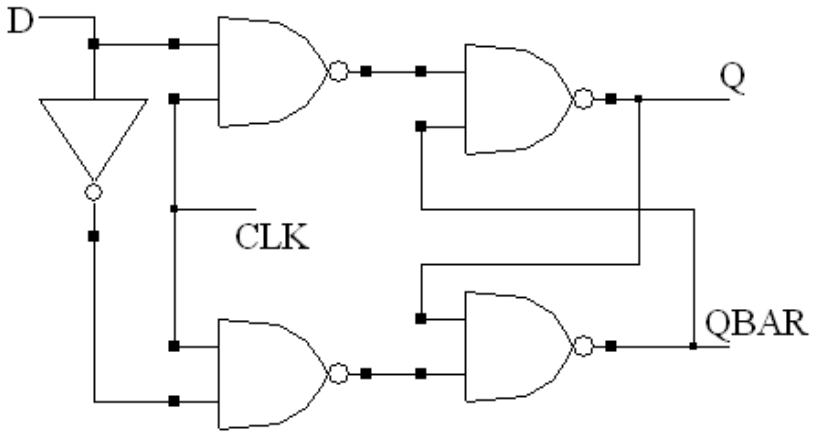
```
# 0a=0,b=0,out=1
# 10a=1,b=0,out=1
# 20a=0,b=1,out=1
# 30a=1,b=1,out=0
```



LAYOUT:



D FLIP FLOP



SOURCE CODE

```
module dff_beh(
    output q,
    output qbar,
    input d,
    input clk
    );
reg q,qbar;
initial
q=1'b0;
always @(negedge clk)
begin
q=d;
qbar=~q;
end
endmodule
```

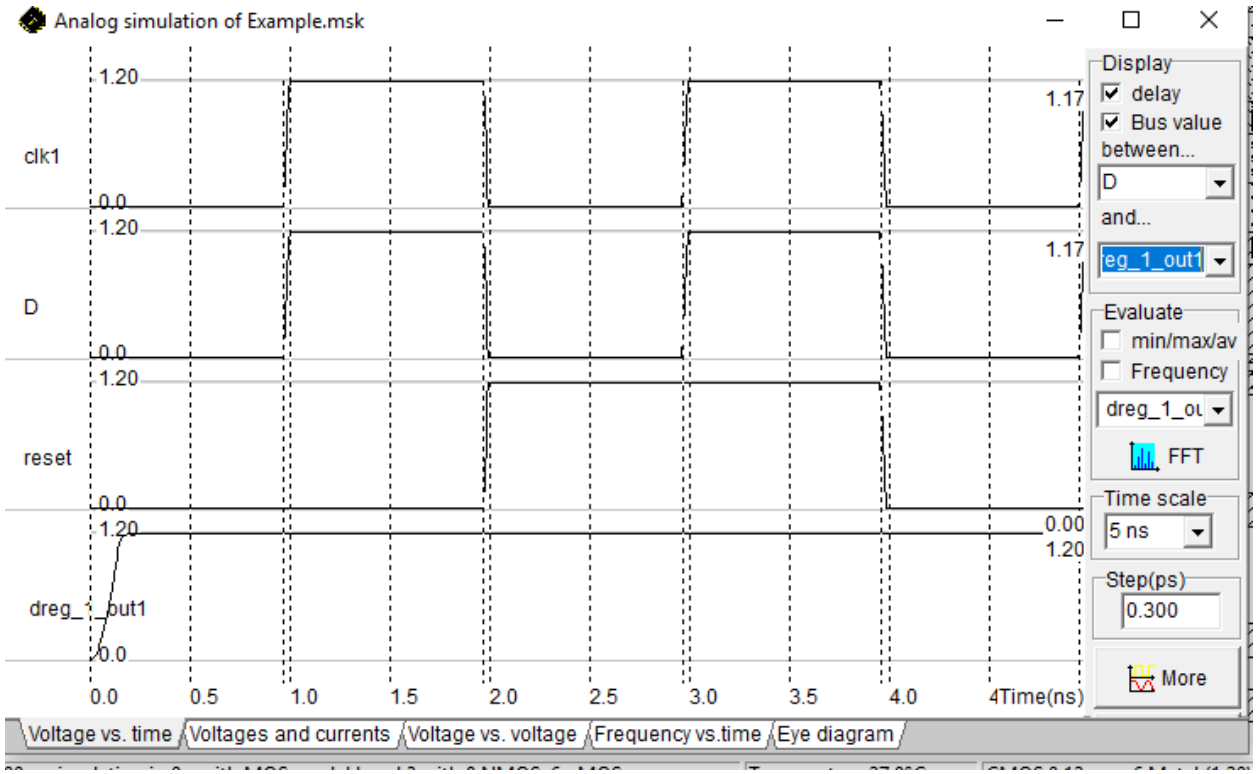
TEST BENCH

```
module dff_test;
reg d;
reg clk;
wire q;
wire qbar;
dff_beh uut ( .q(q), .qbar(qbar), .d(d), .clk(clk));
initial
begin
clk=1'b0;
d=1'b0;
end
always #3 clk=~clk;
always #5 d=~d;
initial $monitor($time,"d=%d,clk=%d,q=%d,qbar=%d\n",d,clk,q,qbar);
initial #30 $stop;
endmodule
```

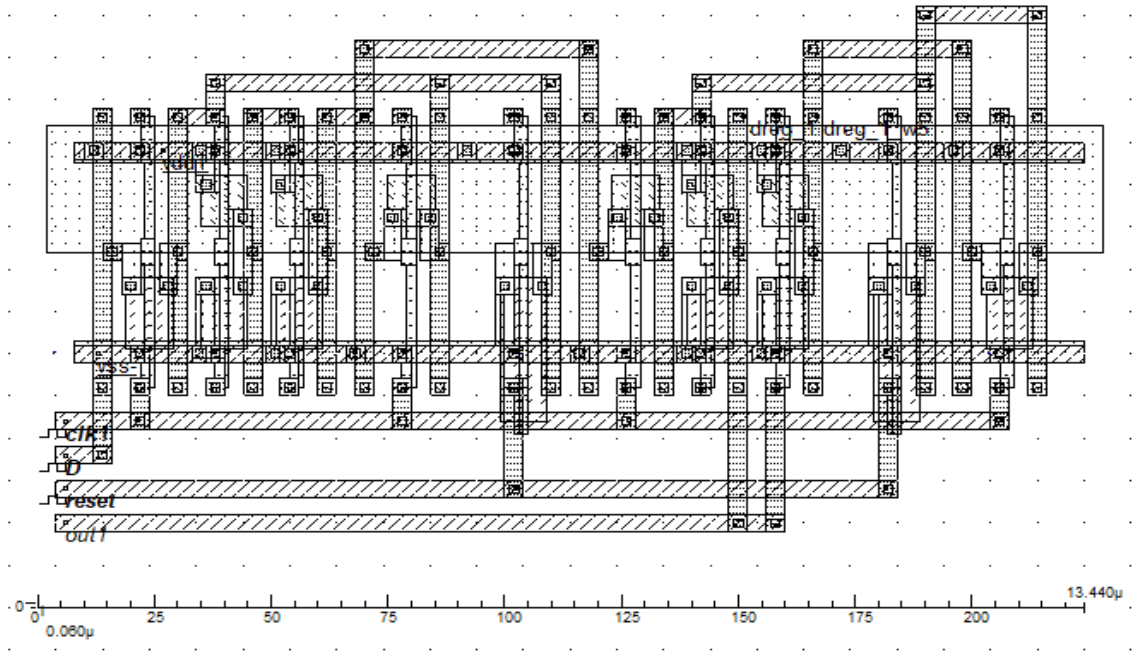
OUTPUT

```
0d=0,clk=0,q=x,qbar=x
3d=0,clk=1,q=0,qbar=1
5d=1,clk=1,q=1,qbar=0
6d=1,clk=0,q=1,qbar=0
9d=1,clk=1,q=1,qbar=0
10d=0,clk=1,q=0,qbar=1
12d=0,clk=0,q=0,qbar=1
15d=1,clk=1,q=1,qbar=0
18d=1,clk=0,q=1,qbar=0
20d=0,clk=0,q=1,qbar=0
21d=0,clk=1,q=0,qbar=1
```

24d=0,clk=0,q=0,qbar=1
25d=1,clk=0,q=0,qbar=1
27d=1,clk=1,q=1,qbar=0



LAYOUT



RESULT

Exp. No. : Date:	DESIGN ENTRY, SIMULATION, SYNTHESIS, P&R, POST P&R SIMULATION UP/DOWN COUNTER

AIM:

To write a Verilog code for up/down counter and to simulate, synthesis, P&R, post P&R simulation and hardware fusingusing Xilinx project navigator.

APPARATUS REQUIRED:

S.No	Name of the equipment/ software	Quantity
1.	PC with Windows	1
2.	Xilinx Project navigator	1

CODING

UPCOUNTER

```
module upcounter(count,clk,n);
output [3:0] count;
input clk;
input [3:0] n;
reg [3:0] count;
initial
count=4'b0000;
always @(negedge clk) count=(count==n)? 4'b0000:count+1'b1;
endmodule
```

DOWNCOUNTER

```
module downcounter(count,clk,n);
output [3:0] count;
input clk;
input [3:0] n;
reg [3:0] count;
initial
count=4'b0000;
always @(negedge clk) count=(count==4'b0000)? n:count-1'b1;
endmodule
```

UPDOWNCOUNTER

```
module updowncounter(count,clk,n,u_d);
output [3:0] count;
input clk,u_d;
input [3:0] n;
reg [3:0] count;
initial
count=4'b0000;
always @(negedge clk) count=(u_d)?((count==n)? 4'b0000: count+ 1'b1)
:(( count==4'b0000)? n:count - 1'b1);
endmodule
```

TEST BENCH:

UPCOUNTER (TESTBENCH)

```
module upcounter_test;
reg clk;
reg [3:0] n;
wire [3:0] count;
upcounter uut (.count(count), .clk(clk),.n(n));
initial
begin
clk=0;
n = 4'b1111;
end
always #2 clk=~clk;
initial
$monitor ($time,"clk=%b;,n=%b,count=%b\n",clk,n,count);
initial # 60 $stop;
endmodule
```

DOWNCOUNTER (TESTBENCH)

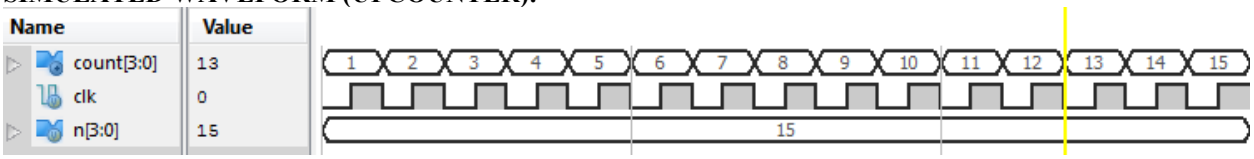
```
module downcounter_test;
wire [3:0] count;
reg clk;
reg [3:0] n;
downcounter u1(count,clk,n);
initial
begin
clk=0;
n = 4'b1111;
end
always #2 clk=~clk;
initial
```

```
$monitor ($time,"clk=%b;,n=%b,count=%b\n",clk,n,count);
initial # 60 $stop;
endmodule
```

UPDOWNCOUNTER (TESTBENCH)

```
module updowncounter_test;
wire [3:0] count;
reg clk,u_d;
reg [3:0] n;
updowncounter u1(count,clk,n,u_d);
initial
begin
clk=0;
n = 4'b0111;
u_d=1'b0;
end
always #2 clk=~clk;
always #30u_d=~u_d;
initial
$monitor ($time,"clk=%b,u_d=%b,n=%b,count=%b\n",clk,u_d,n,count);
initial # 58 $stop;
endmodule
```

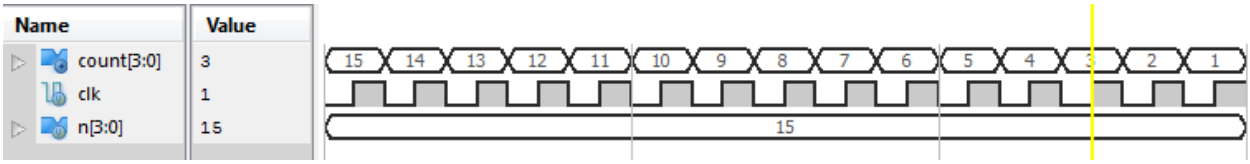
SIMULATED WAVEFORM (UPCOUNTER):




```
0clk=0;,n=1111,count=0001
2clk=1;,n=1111,count=0001
4clk=0;,n=1111,count=0010
6clk=1;,n=1111,count=0010
8clk=0;,n=1111,count=0011
10clk=1;,n=1111,count=0011
12clk=0;,n=1111,count=0100
14clk=1;,n=1111,count=0100
16clk=0;,n=1111,count=0101
18clk=1;,n=1111,count=0101
20clk=0;,n=1111,count=0110
22clk=1;,n=1111,count=0110
24clk=0;,n=1111,count=0111
26clk=1;,n=1111,count=0111
28clk=0;,n=1111,count=1000
30clk=1;,n=1111,count=1000
32clk=0;,n=1111,count=1001
34clk=1;,n=1111,count=1001
36clk=0;,n=1111,count=1010
```

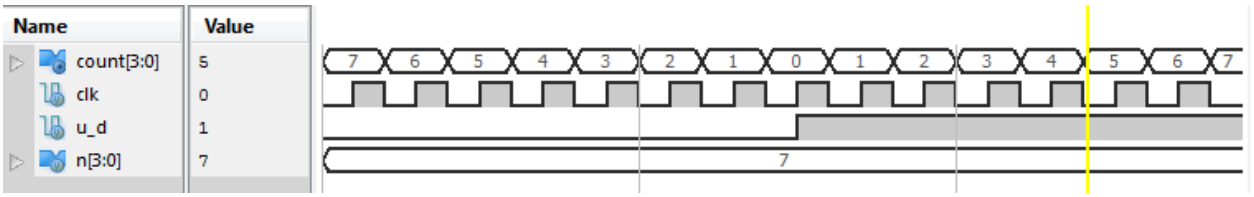
38clk=1;,n=1111,count=1010
40clk=0;,n=1111,count=1011
42clk=1;,n=1111,count=1011
44clk=0;,n=1111,count=1100
46clk=1;,n=1111,count=1100
48clk=0;,n=1111,count=1101
50clk=1;,n=1111,count=1101
52clk=0;,n=1111,count=1110
54clk=1;,n=1111,count=1110
56clk=0;,n=1111,count=1111
58clk=1;,n=1111,count=1111

SIMULATED WAVEFORM (DOWNCOUNTER):



0clk=0;,n=1111,count=1111
2clk=1;,n=1111,count=1111
4clk=0;,n=1111,count=1110
6clk=1;,n=1111,count=1110
8clk=0;,n=1111,count=1101
10clk=1;,n=1111,count=1101
12clk=0;,n=1111,count=1100
14clk=1;,n=1111,count=1100
16clk=0;,n=1111,count=1011
18clk=1;,n=1111,count=1011
20clk=0;,n=1111,count=1010
22clk=1;,n=1111,count=1010
24clk=0;,n=1111,count=1001
26clk=1;,n=1111,count=1001
28clk=0;,n=1111,count=1000
30clk=1;,n=1111,count=1000
32clk=0;,n=1111,count=0111
34clk=1;,n=1111,count=0111
36clk=0;,n=1111,count=0110
38clk=1;,n=1111,count=0110
40clk=0;,n=1111,count=0101
42clk=1;,n=1111,count=0101
44clk=0;,n=1111,count=0100
46clk=1;,n=1111,count=0100
48clk=0;,n=1111,count=0011
50clk=1;,n=1111,count=0011
52clk=0;,n=1111,count=0010
54clk=1;,n=1111,count=0010
56clk=0;,n=1111,count=0001
58clk=1;,n=1111,count=0001

SIMULATED WAVEFORM (UPDOWNCOUNTER):



0clk=0,u_d=0,n=0111,count=0111
2clk=1,u_d=0,n=0111,count=0111
4clk=0,u_d=0,n=0111,count=0110
6clk=1,u_d=0,n=0111,count=0110
8clk=0,u_d=0,n=0111,count=0101
10clk=1,u_d=0,n=0111,count=0101
12clk=0,u_d=0,n=0111,count=0100
14clk=1,u_d=0,n=0111,count=0100
16clk=0,u_d=0,n=0111,count=0011
18clk=1,u_d=0,n=0111,count=0011
20clk=0,u_d=0,n=0111,count=0010
22clk=1,u_d=0,n=0111,count=0010
24clk=0,u_d=0,n=0111,count=0001
26clk=1,u_d=0,n=0111,count=0001
28clk=0,u_d=0,n=0111,count=0000
30clk=1,u_d=1,n=0111,count=0000
32clk=0,u_d=1,n=0111,count=0001
34clk=1,u_d=1,n=0111,count=0001
36clk=0,u_d=1,n=0111,count=0010
38clk=1,u_d=1,n=0111,count=0010
40clk=0,u_d=1,n=0111,count=0011
42clk=1,u_d=1,n=0111,count=0011
44clk=0,u_d=1,n=0111,count=0100
46clk=1,u_d=1,n=0111,count=0100
48clk=0,u_d=1,n=0111,count=0101
50clk=1,u_d=1,n=0111,count=0101
52clk=0,u_d=1,n=0111,count=0110
54clk=1,u_d=1,n=0111,count=0110
56clk=0,u_d=1,n=0111,count=0111

SIMULATED RESULT:
UPCOUNTER:

```
Timing Summary:
---------------
Speed Grade: -3
   Minimum period: 2.610ns (Maximum Frequency: 383.164MHz)
   Minimum input arrival time before clock: 3.349ns
```

```
   Maximum output required time after clock: 3.984ns
   Maximum combinational path delay: No path found
Timing Details:
---------------
All values displayed in nanoseconds (ns)
=========================================================================
Timing constraint: Default period analysis for Clock 'clk'
  Clock period: 2.610ns (frequency: 383.164MHz)
  Total number of paths / destination ports: 23 / 4
-------------------------------------------------------------------------
Delay:               2.610ns (Levels of Logic = 2)
  Source:            count_1 (FF)
  Destination:       count_3 (FF)
  Source Clock:      clk falling
  Destination Clock: clk falling
  Data Path: count_1 to count_3
                              Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FD:C->Q               5   0.525   0.823  count_1 (count_1)
     LUT2:I0->O            1   0.250   0.688  count[3]_n[3]_equal_2_o4_SW2 (N4)
     LUT6:I4->O            1   0.250   0.000  count_3_rstpot (count_3_rstpot)
     FD:D                      0.074          count_3
    ----------------------------------------
    Total                      2.610ns (1.099ns logic, 1.511ns route)
                                       (42.1% logic, 57.9% route)
```

DOWNCOUNTER

```
Timing Summary:
---------------
Speed Grade: -3
   Minimum period: 1.844ns (Maximum Frequency: 542.299MHz)
   Minimum input arrival time before clock: 2.566ns
   Maximum output required time after clock: 3.954ns
   Maximum combinational path delay: No path found
Timing Details:
---------------
All values displayed in nanoseconds (ns)
=========================================================================
Timing constraint: Default period analysis for Clock 'clk'
  Clock period: 1.844ns (frequency: 542.299MHz)
  Total number of paths / destination ports: 16 / 4
-------------------------------------------------------------------------
Delay:               1.844ns (Levels of Logic = 1)
  Source:            count_3 (FF)
  Destination:       count_0 (FF)
  Source Clock:      clk falling
  Destination Clock: clk falling
  Data Path: count_3 to count_0
                              Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FD:C->Q               5   0.525   0.991  count_3 (count_3)
     LUT5:I1->O            1   0.254   0.000  Mcount_count_xor<1>11 (Mcount_count1)
     FD:D                      0.074          count_1
    ----------------------------------------
    Total                      1.844ns (0.853ns logic, 0.991ns route)
```

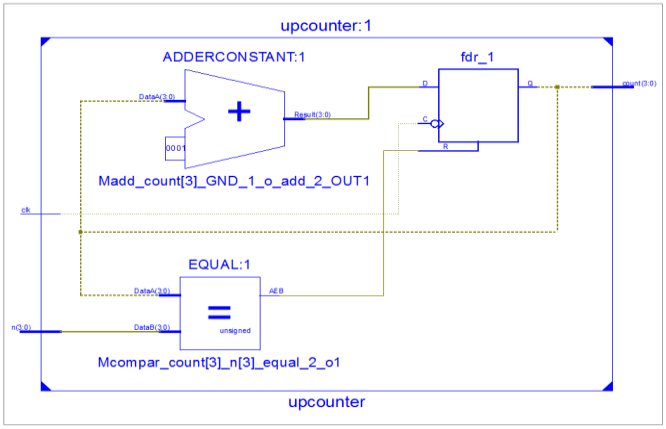
```
                                        (46.3% logic, 53.7% route)
```

UPDOWNCOUNTER

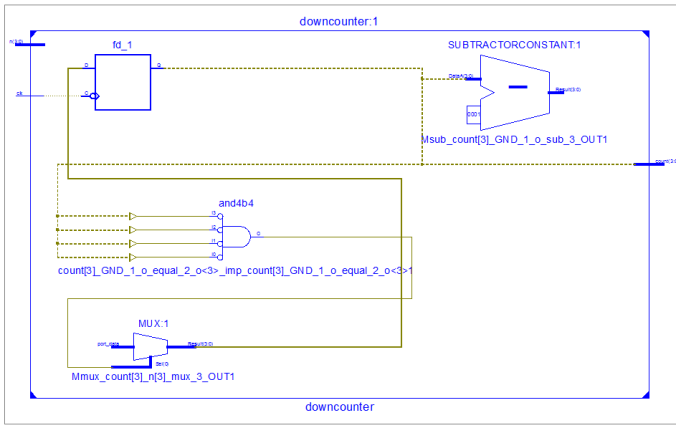
```
Timing Summary:
---------------
Speed Grade: -3
   Minimum period: 2.744ns (Maximum Frequency: 364.392MHz)
   Minimum input arrival time before clock: 3.622ns
   Maximum output required time after clock: 4.069ns
   Maximum combinational path delay: No path found
=========================================================================
Timing constraint: Default period analysis for Clock 'clk'
  Clock period: 2.744ns (frequency: 364.392MHz)
  Total number of paths / destination ports: 32 / 4
-------------------------------------------------------------------------
Delay:               2.744ns (Levels of Logic = 2)
  Source:            count_2 (FF)
  Destination:       count_0 (FF)
  Source Clock:      clk falling
  Destination Clock: clk falling
  Data Path: count_2 to count_0
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FD_1:C->Q             6   0.525   0.745  count_2 (count_2)
     LUT2:I1->O            4   0.254   0.912  _n0019_SW1 (N2)
     LUT6:I3->O            1   0.235   0.000  count_0_rstpot (count_0_rstpot)
     FD_1:D                    0.074          count_0
    ----------------------------------------
    Total                      2.744ns (1.088ns logic, 1.656ns route)
                                       (39.6% logic, 60.4% route)
=========================================================================
Timing constraint: Default OFFSET IN BEFORE for Clock 'clk'
  Total number of paths / destination ports: 28 / 4
-------------------------------------------------------------------------
Offset:              3.622ns (Levels of Logic = 3)
  Source:            u_d (PAD)
  Destination:       count_0 (FF)
  Destination Clock: clk falling
  Data Path: u_d to count_0
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             8   1.228   1.233  u_d_IBUF (u_d_IBUF)
     LUT6:I1->O            1   0.254   0.580  count[3]_count[3]_mux_7_OUT<3>1
(count[3]_count[3]_mux_7_OUT<3>)
     LUT6:I5->O            1   0.254   0.000  count_3_rstpot (count_3_rstpot)
     FD_1:D                    0.074          count_3
    ----------------------------------------
    Total                      3.622ns (1.810ns logic, 1.812ns route)
                                       (50.0% logic, 50.0% route)
=========================================================================
Timing constraint: Default OFFSET OUT AFTER for Clock 'clk'
  Total number of paths / destination ports: 4 / 4
-------------------------------------------------------------------------
Offset:              4.069ns (Levels of Logic = 1)
  Source:            count_3 (FF)
  Destination:       count<3> (PAD)
```

```
Source Clock:      clk falling
Data Path: count_3 to count<3>
                                Gate     Net
  Cell:in->out      fanout    Delay    Delay  Logical Name (Net Name)
  ----------------------------------------  ------------
   FD_1:C->Q             9    0.525    0.829  count_3 (count_3)
   OBUF:I->O                  2.715          count_3_OBUF (count<3>)
  ----------------------------------------
  Total                       4.069ns (3.240ns logic, 0.829ns route)
                                      (79.6% logic, 20.4% route)
```

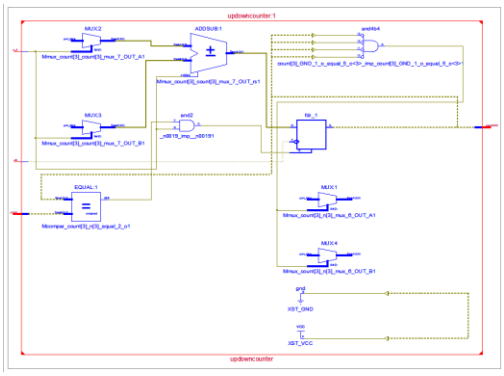
RTL VIEW: (upcounter)



RTL VIEW: (Downcounter)

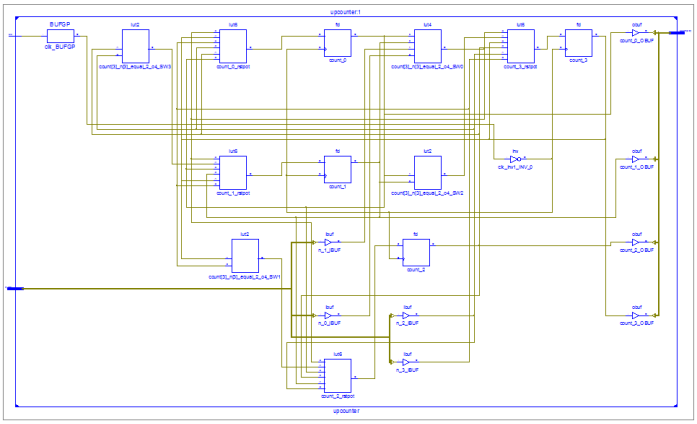


RTL VIEW: (Updowncounter)

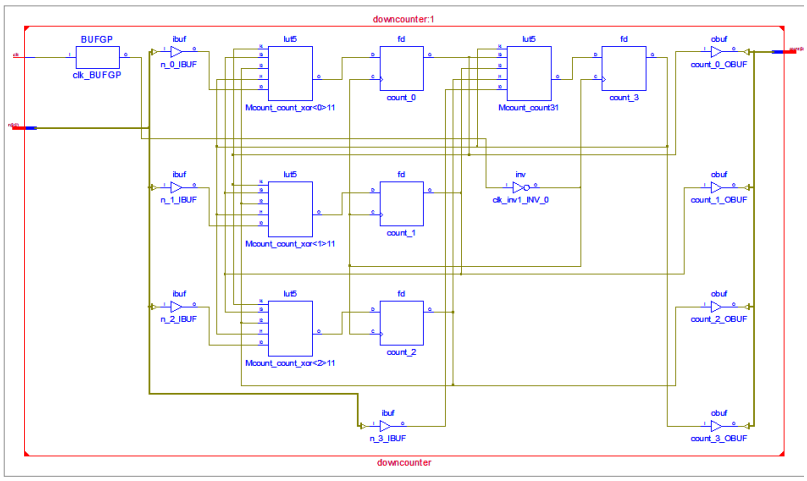


TECHNOLOGY VIEW:

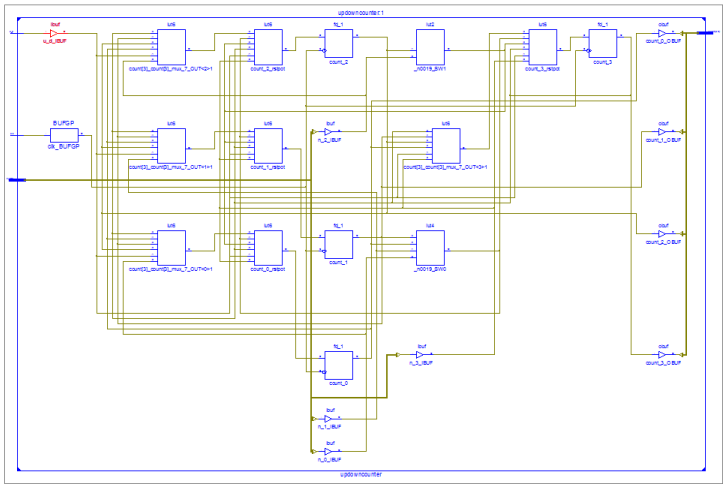
Upcounter:



Downcounter:



Updown counter



POWER ANALYSIS CODE

(upcounter)

```
module upcounter_power(count,clk,n,reset,resetout);
output [3:0] count;
output resetout;
input clk,reset;
input [3:0] n;
reg [3:0] count1;
initial
count1=4'b0000;
dcm_clk uuu (clk,dcm_clk,reset,resetout);
always @(posedge dcm_clk or posedge reset)
begin
if(reset)
 count1=4'b0000;
else
 count1=(count1==n)? 4'b0000:count1+1'b1;
end
assign count=count1;
endmodule
```

DownCounter

```
module downcounter_power(count,clk,n,reset,resetout);
output [3:0] count;
output resetout;
input clk,reset;
input [3:0] n;
reg [3:0] count1;
initial
count1=4'b0000;
dcm_clk uuu (clk,dcm_clk,reset,resetout);
always @(posedge dcm_clk or posedge reset)
begin
if(reset)
 count1=4'b0000;
else
 count1=(count1==4'b0000)?n:count1-1'b1;
end
assign count=count1;
endmodule
```

Updown counter

```
module updowncounter_power(count,clk,u_d,n,reset,resetout);
output [3:0] count;
output resetout;
input clk,reset,u_d;
input [3:0] n;
reg [3:0] count1;
initial
count1=4'b0000;
dcm_clk uuu (clk,dcm_clk,reset,resetout);
always @(posedge dcm_clk or posedge reset)
begin
if(reset)
```

```
 count1=4'b0000;
else
count1=(u_d)?((count1==n)? 4'b0000: count1+ 1'b1)
:(( count1==4'b0000)? n:count1 - 1'b1);
end
assign count=count1;
endmodule
```

POWER REPORT

Upcounter:

On-Chip Power Summary				
On-Chip	Power (mW)	Used	Available	Utilization (%)
Clocks	2.92	1	---	---
Logic	0.03	8	2400	0.3
Signals	0.08	14	---	---
IOs	1.71	11	102	10.8
DCMs	14.11	1	4	25.0
Quiescent	13.88			
Total	32.73			

Effective TJA (C/W)	42.4
Max Ambient (C)	83.6
Junction Temp (C)	26.4

Power Supply Summary			
	Total	Dynamic	Quiescent
Supply Power (mW)	32.73	8.13	24.60

Downcounter

On-Chip Power Summary				
On-Chip	Power (mW)	Used	Available	Utilization (%)
Clocks	2.89	1	---	---
Logic	0.02	4	2400	0.2
Signals	0.06	10	---	---
IOs	2.72	11	102	10.8
DCMs	14.11	1	4	25.0
Quiescent	13.89			
Total	33.69			

Thermal Summary	
Effective TJA (C/W)	42.4
Max Ambient (C)	83.6
Junction Temp (C)	26.4

Power Supply Summary			
	Total	Dynamic	Quiescent
Supply Power (mW)	33.69	9.08	24.61

Updown counter

On-Chip Power Summary				
On-Chip	Power (mW)	Used	Available	Utilization (%)
Clocks	2.95	1	---	---
Logic	0.12	17	2400	0.7
Signals	0.17	23	---	---
IOs	4.02	12	102	11.8
DCMs	14.11	1	4	25.0
Quiescent	13.91			
Total	35.27			

Thermal Summary	
Effective TJA (C/W)	42.4
Max Ambient (C)	83.5
Junction Temp (C)	26.5

Power Supply Summary			
	Total	Dynamic	Quiescent
Supply Power (mW)	35.27	10.64	24.63

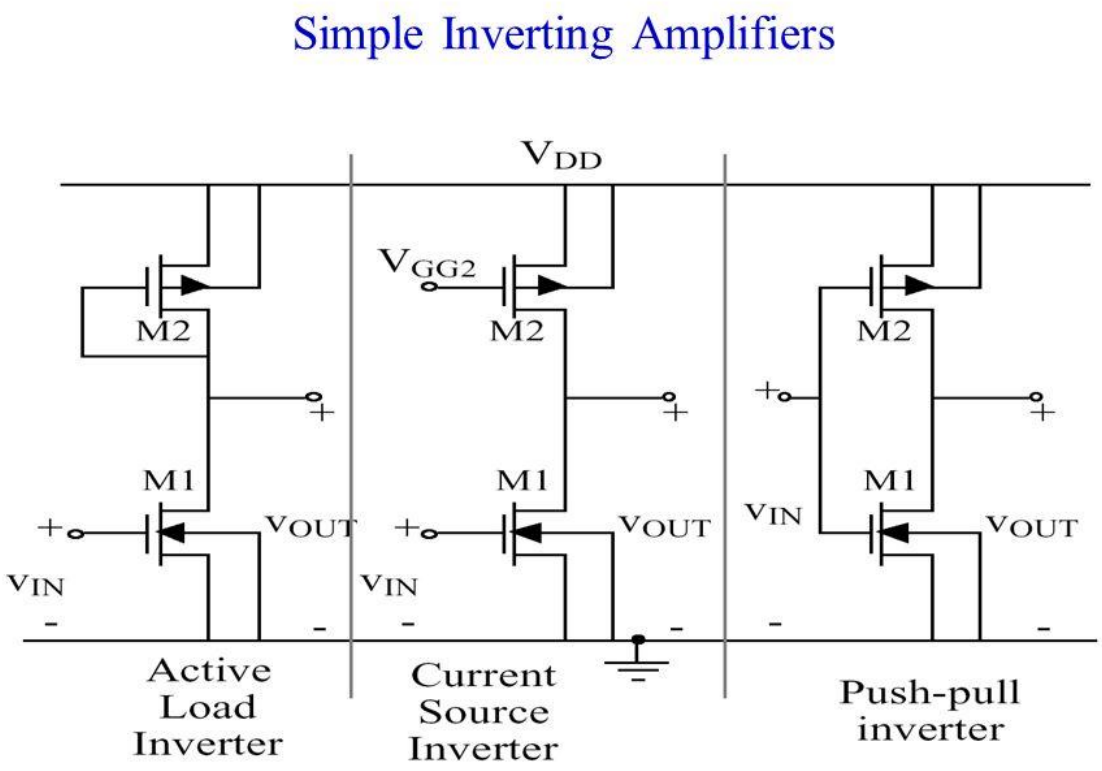
Result

Exp. No. : Date:	DESIGN OF CMOS INVERTING AMPLIFIER

AIM:

To simulate CMOS inverting amplifier using Tanner software

Circuit Diagram



TIMING PARAMETER

VI CHARACTERISTICS

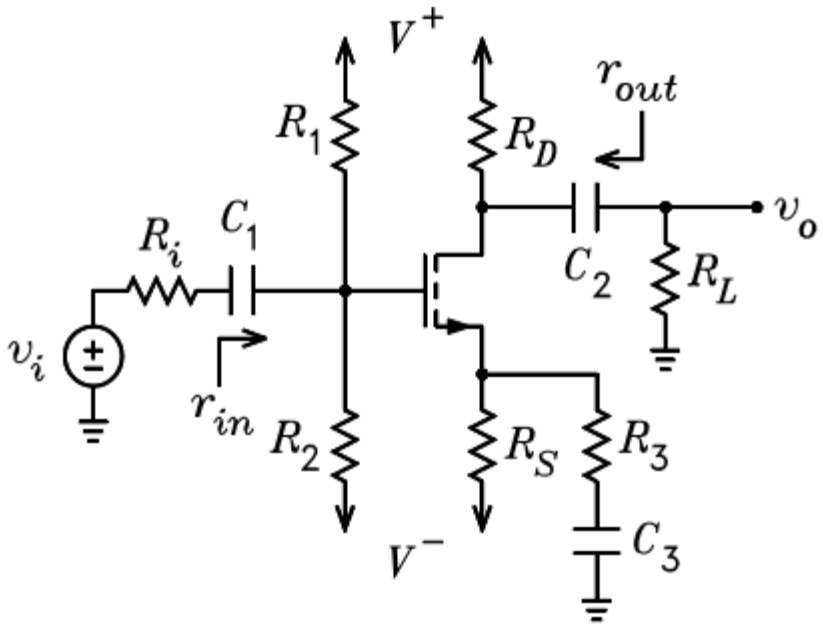
RESULT

Exp. No. :	DESIGN OF BASIC COMMON SOURCE, COMMON GATE AND COMMON DRAIN AMPLIFIERS
Date:	

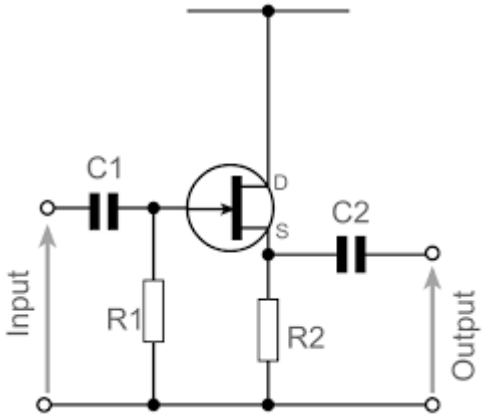
AIM:

To simulatebasic common source, common gate and common drain amplifiersusing Tanner software

Common Source



Common Drain



Timing Parameters

Result

Ex. No:	SCHEMATIC ENTRY AND SPICE SIMULATION CMOS INVERTER
Date:	

AIM

To perform schematic entry of a CMOS inverter and verify its functionality using transient analysis.

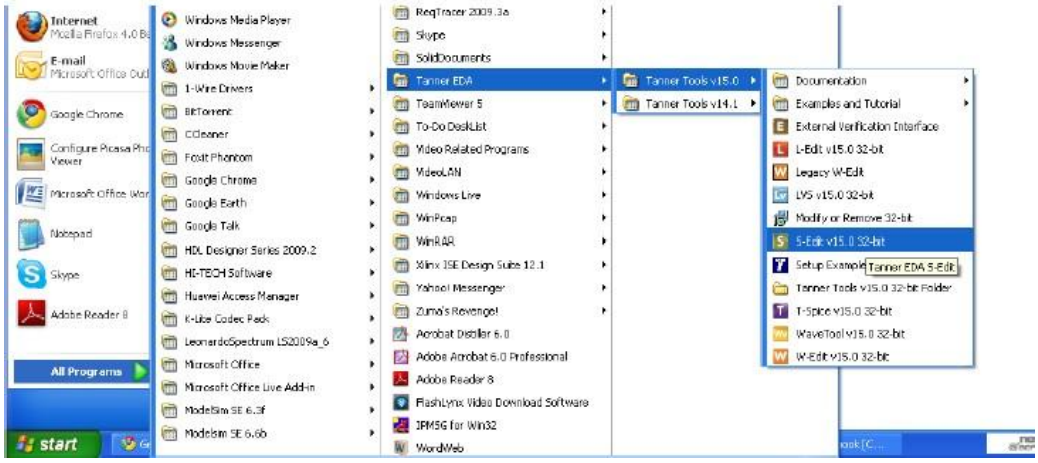
APPARATUS REQUIRED

PC with Tanner EDA tool.

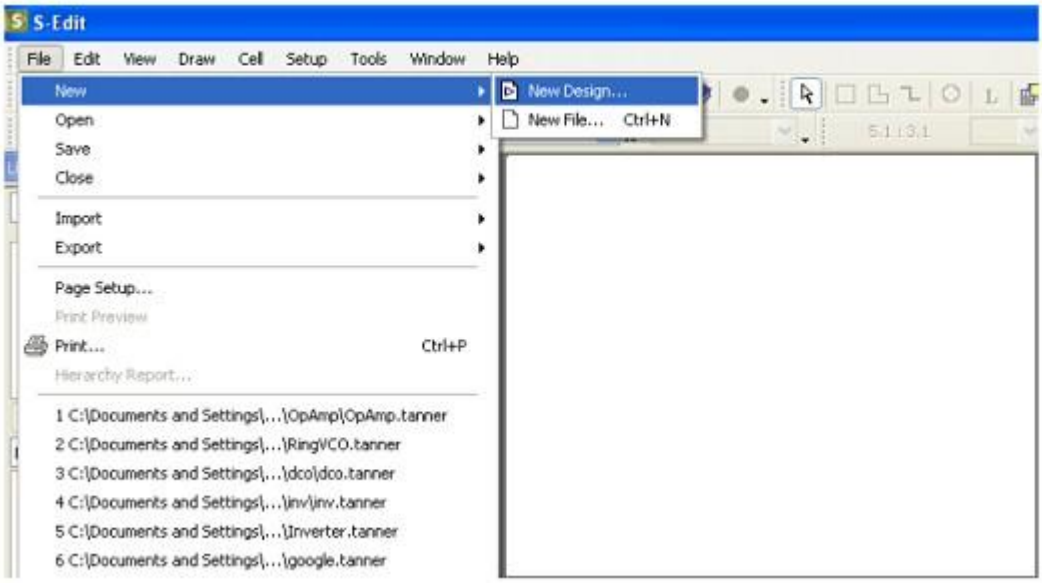
PROCEDURE

- Open S-edit and choose new design.
- Draw the schematic of CMOS Inverter using S-edit.
- Perform Transient Analysis of the CMOS Inverter.
- Go to transient analysis and set the parameters as explained in the instructions
- above. Obtain the output waveform from W-Edit.
- Obtain the spice code using T-Edit.

Start → All Programs → Tanner EDA → Tanner V 15 → S-Edit V15.0 32 Bit



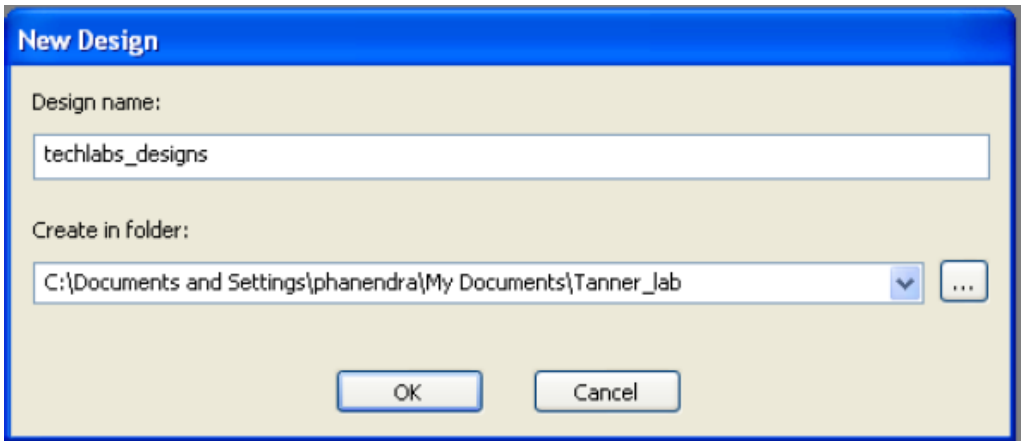
To create a new design : File → New Design



Enter the design name and give the path where it should be saved.

Example : techlabs_designs is the design folder

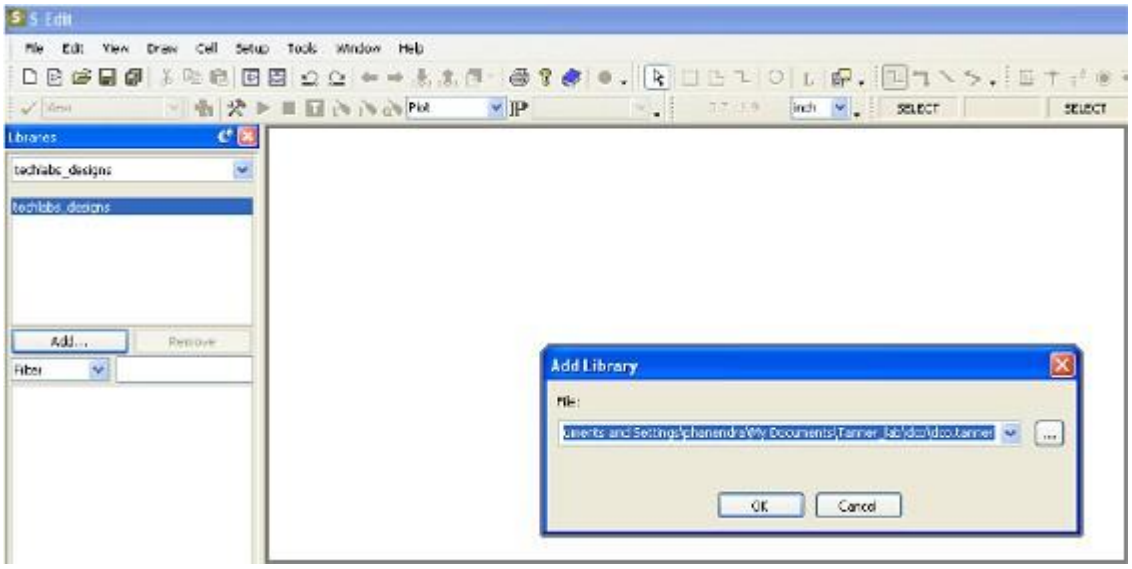
C:\Documents and Settings\phanendra\My Documents\Tanner_lab is the target location of design folder

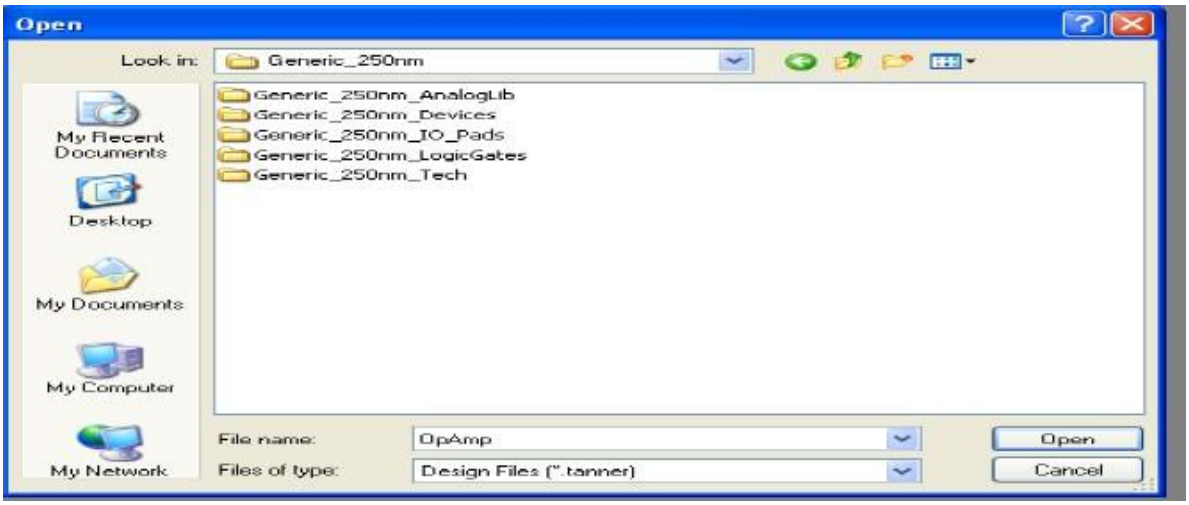


Add component libraries. The path for component libraries is given below.

My Documents\Tanner EDA\Tanner Tools v15.0\Process\Generic_250nm

Click add in S-Edit window for adding the libraries and follow the path of process folder.

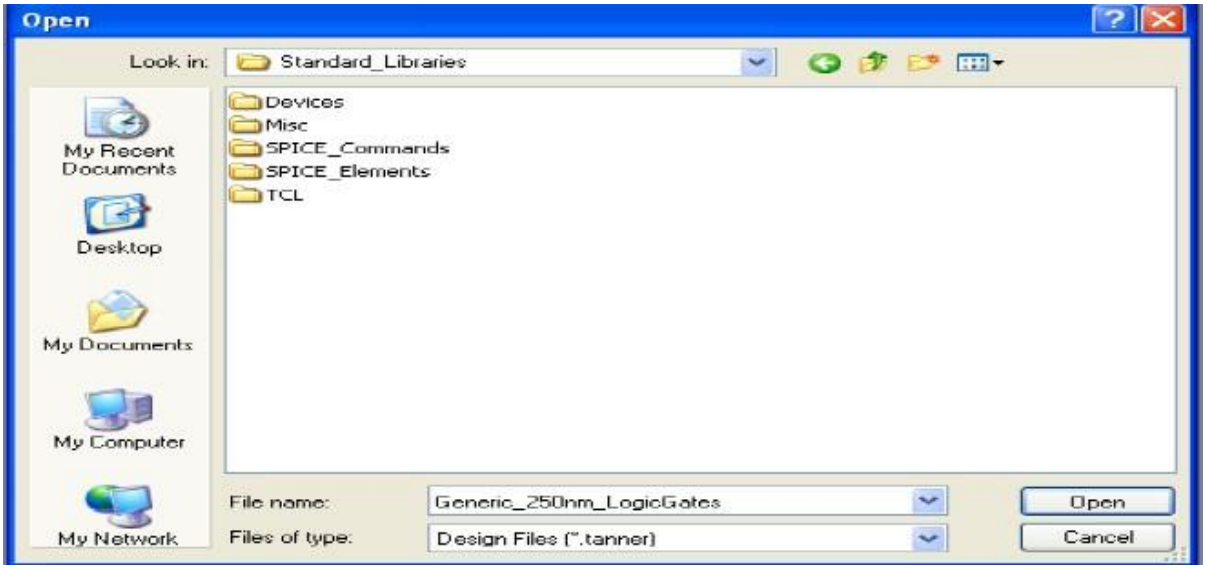


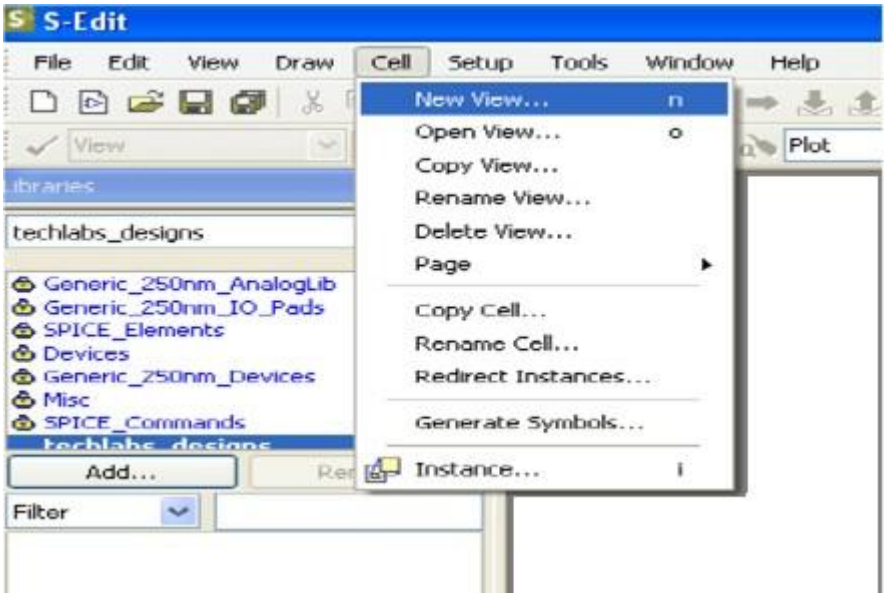


Double click all the component libraries under Generic_250nm and add all tanner database files (.tdb).

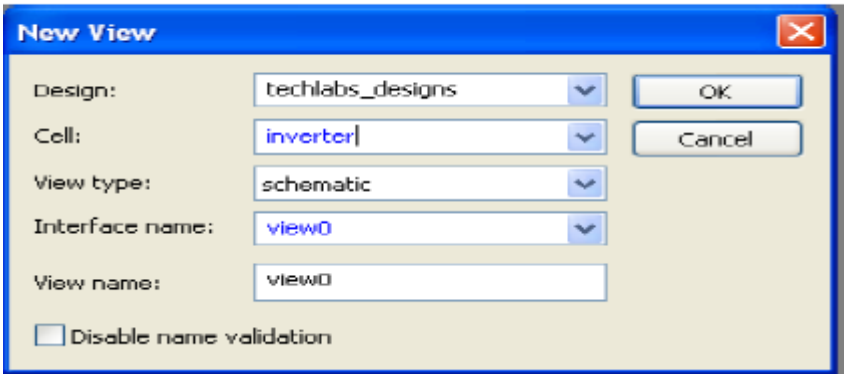
To add Spice commands and Spice Elements for setting spice simulation follow the path.

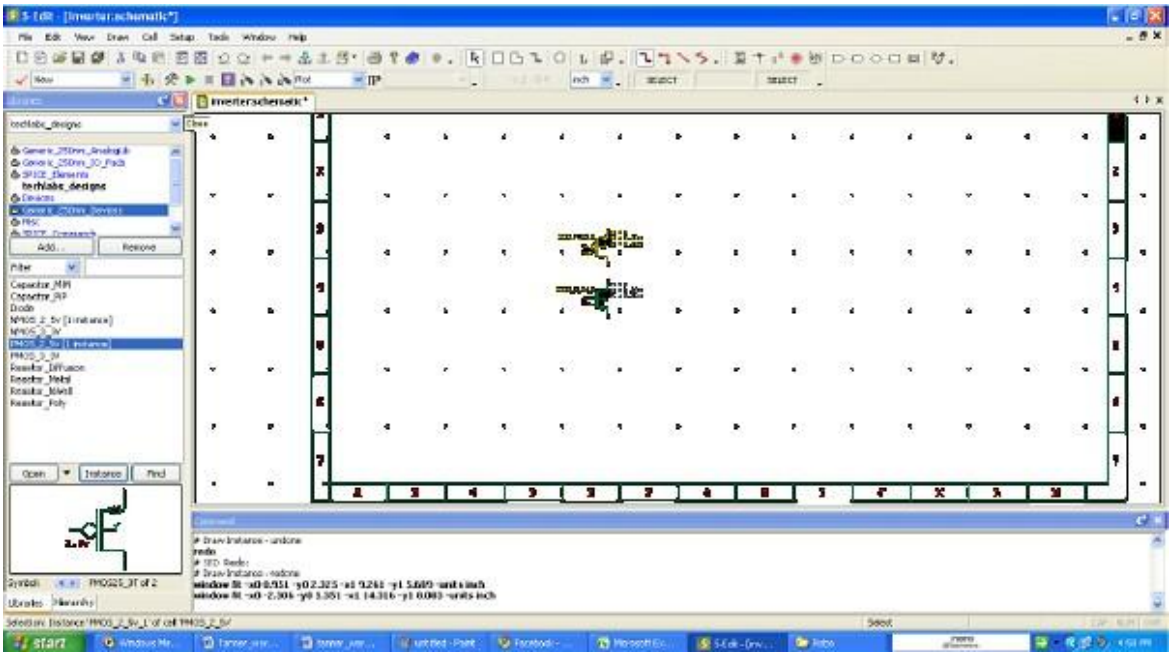
My Documents\Tanner EDA\Tanner Tools v15.0\Process\Standard_Libraries



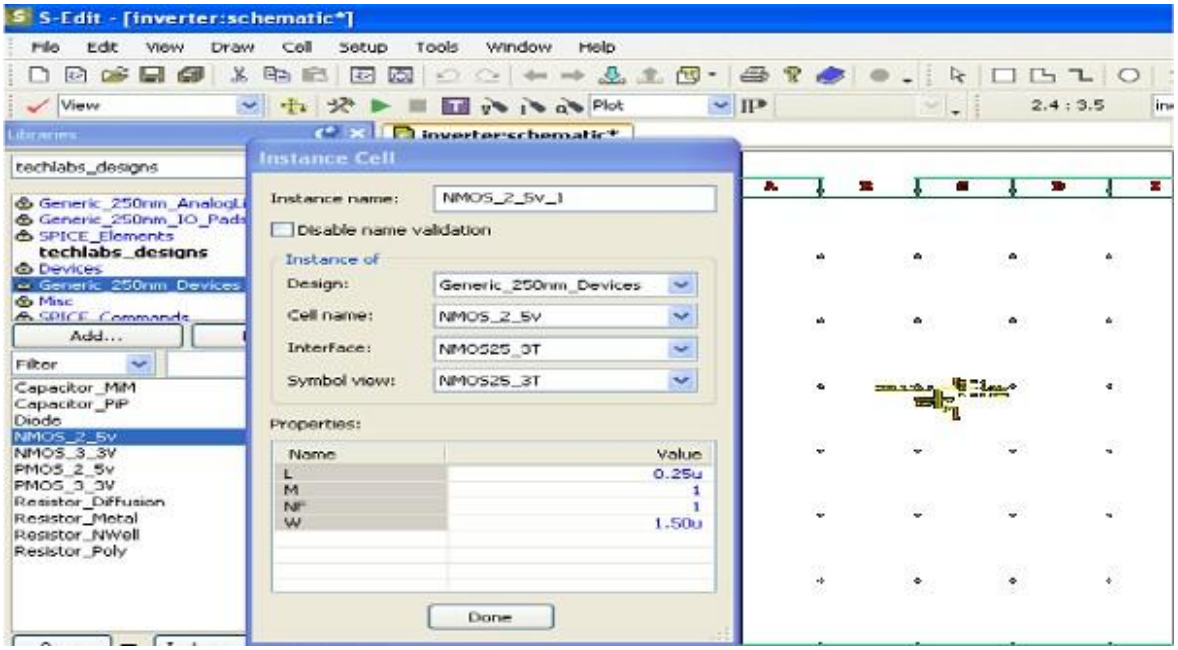


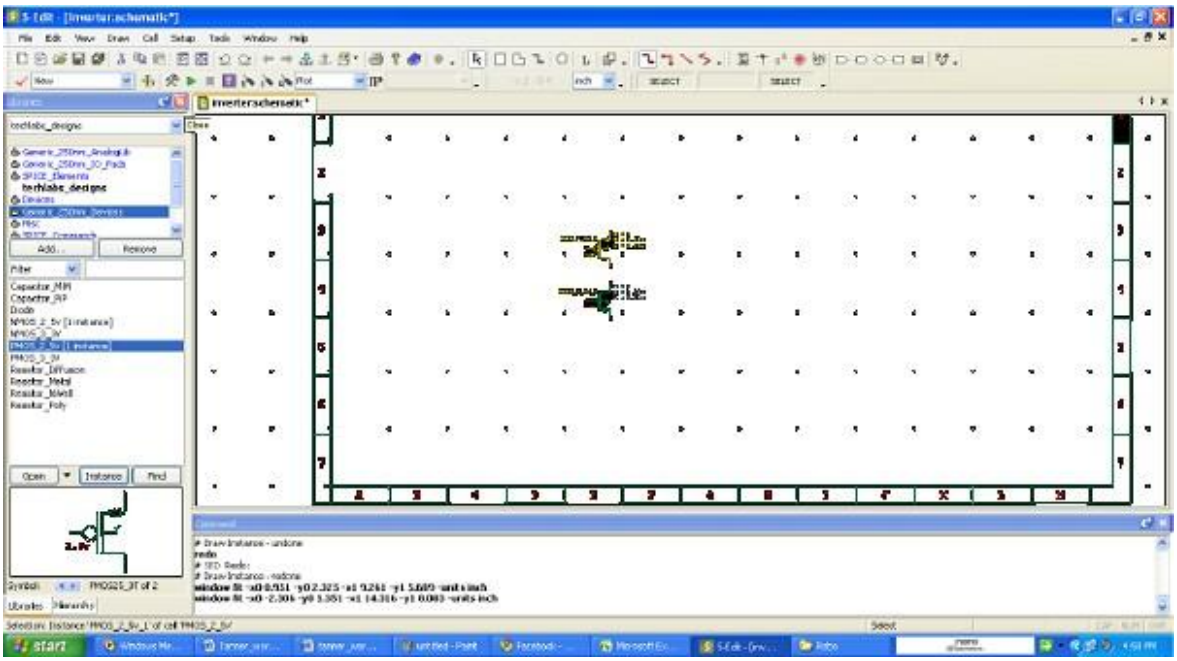
Give a new name for cell



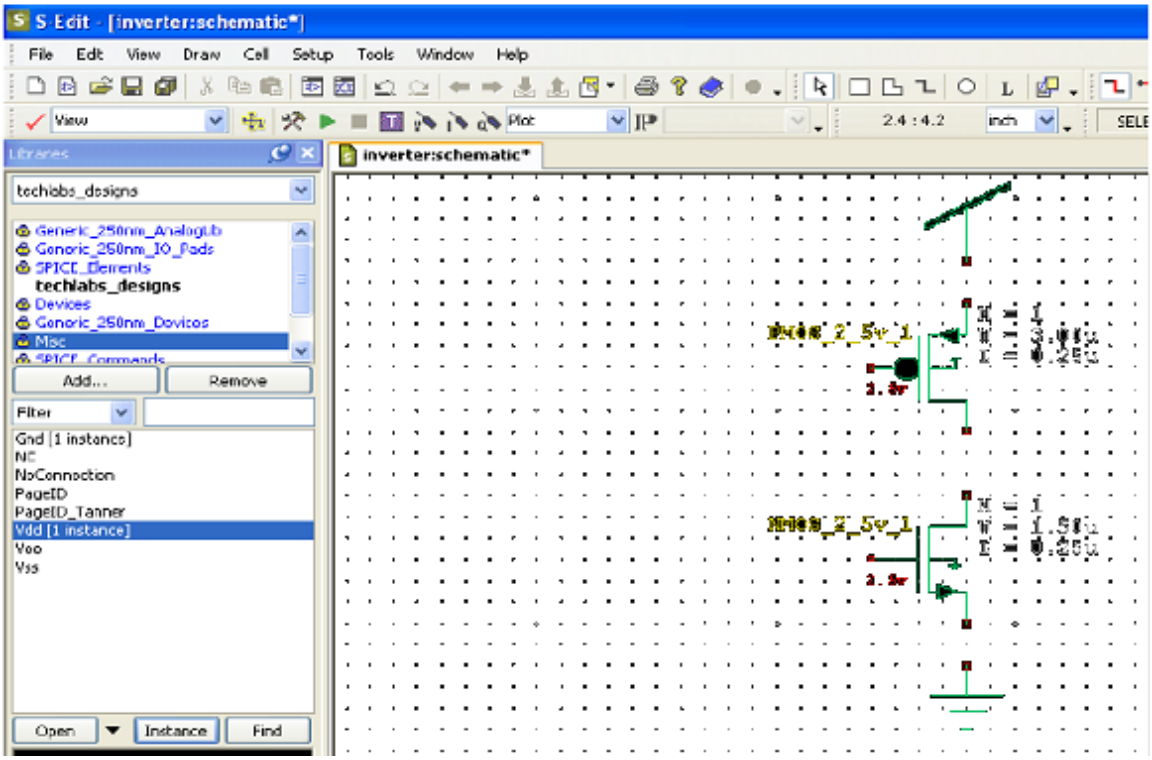


Click on Generic_250nm_Devices folder on libraries

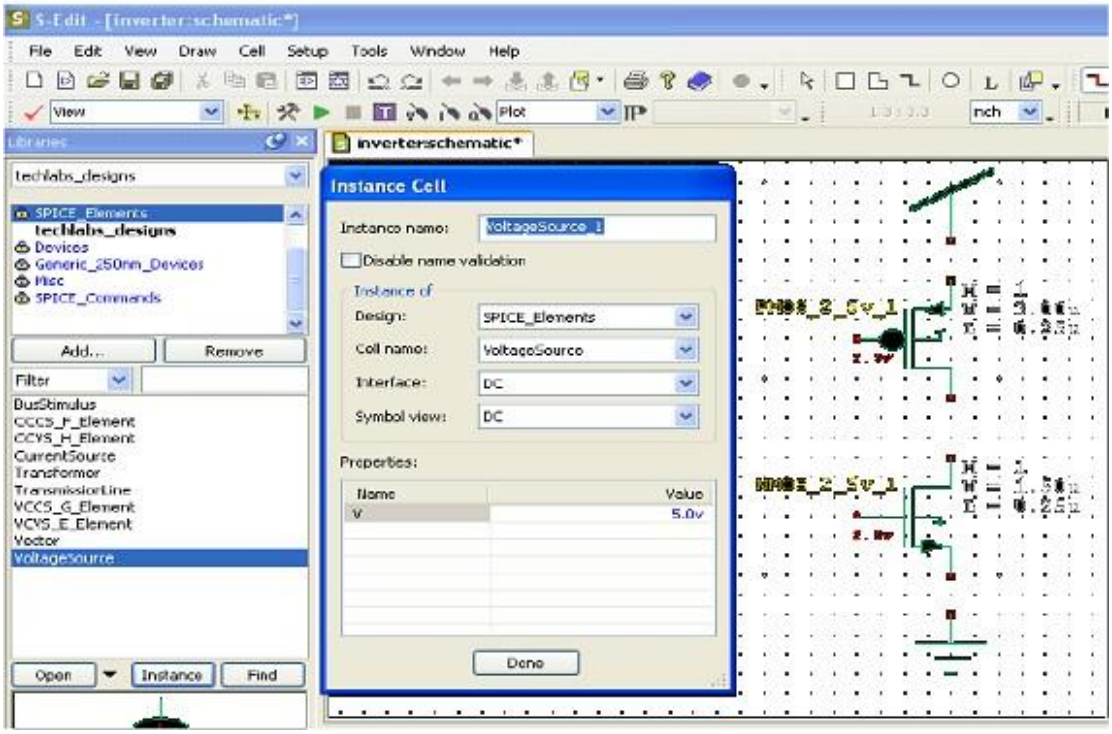

S-Edit - [inverter:schematic*]
File Edit View Draw Cell Setup Tools Window Help
Instance Cell
Instance name:
NMOS_2_5v_1
Disable name validation
Instance of
Design:
Generic_250nm_Devices
Cell name:
NMOS_2_5V
Interface:
NMOS25_3T
Symbol view:
NMOS25_3T
Properties:
Name
Value
L
0.25u
M
1
NF
1
W
1.50u
Done
techlabs_designs
Capacitor_MiM
Capacitor_PiP
Diode
NMOS_2_5V
NMOS_3_3V
PMOS_2_5v
PMOS_3_3V
Resistor_Diffusion
Resistor_Metal
Resistor_NWell
Resistor_Poly



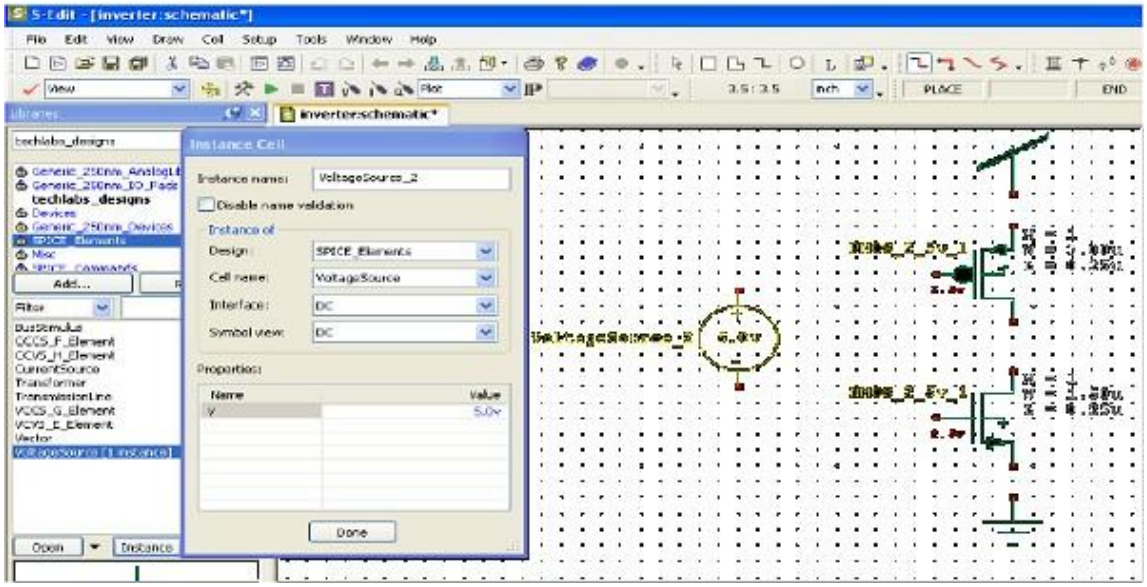
Now place Vdd and Gnd Instances from the Misc folder under Library.



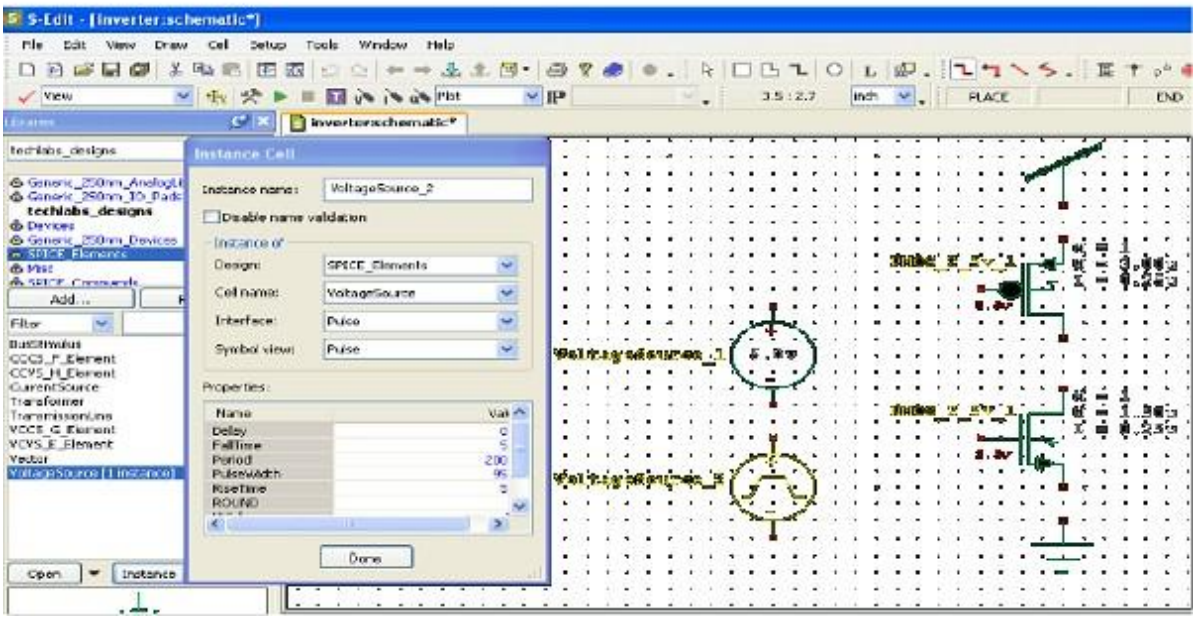
To place a Voltage Source, Click Spice Elements, Under Spice Elements Click Voltage Source and then Instance.



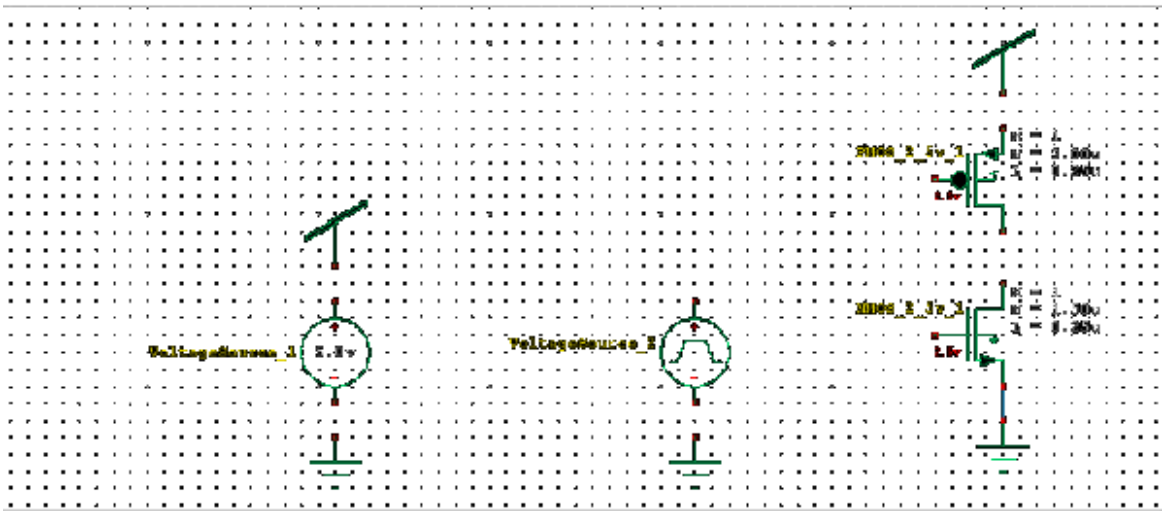
To place a DC voltage source, change the interface to DC and edit the voltage value . Click done only after placing the voltage source on design area.



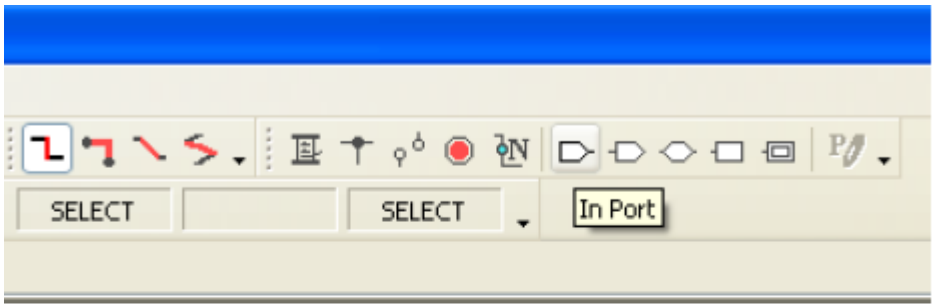
To place a Pulse voltage source, change the interface to DC and edit the voltage value . Click done only after placing the voltage source on design area.

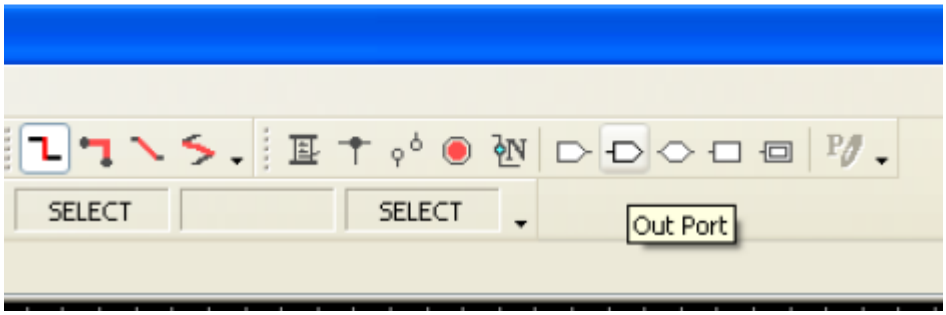


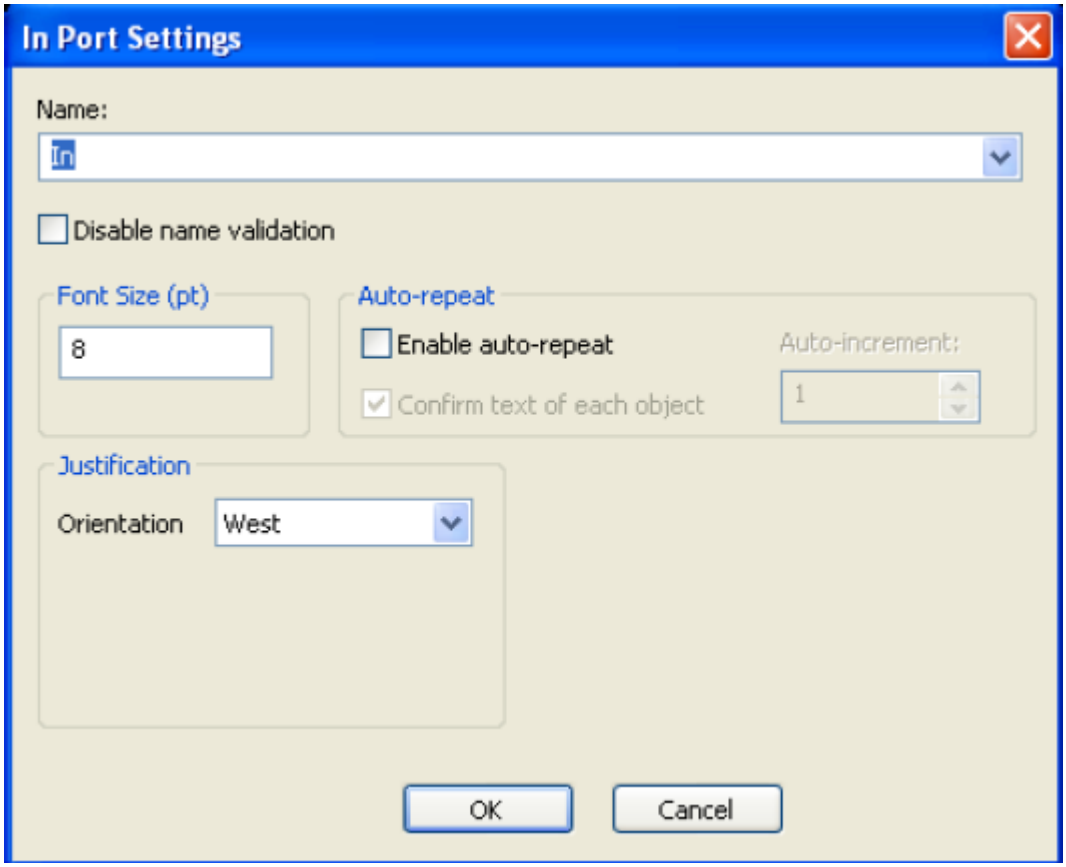
Place Vdd and Gnd even for the voltage sources as shown.

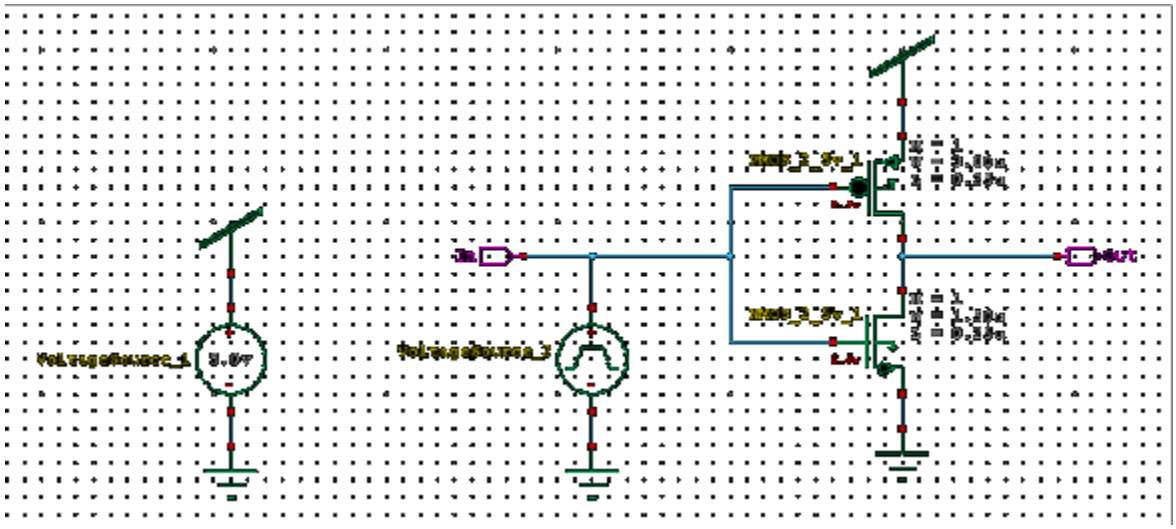


Now place Input and Output ports.

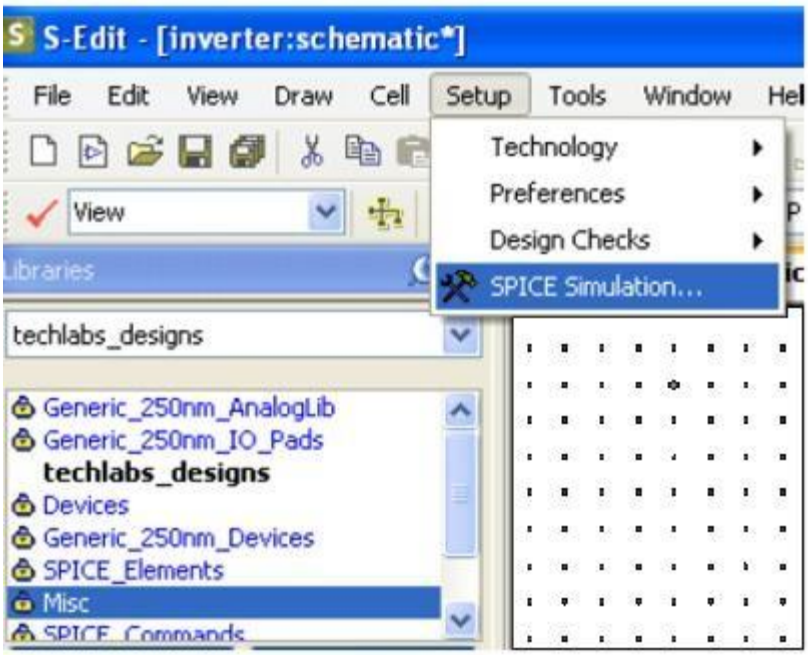


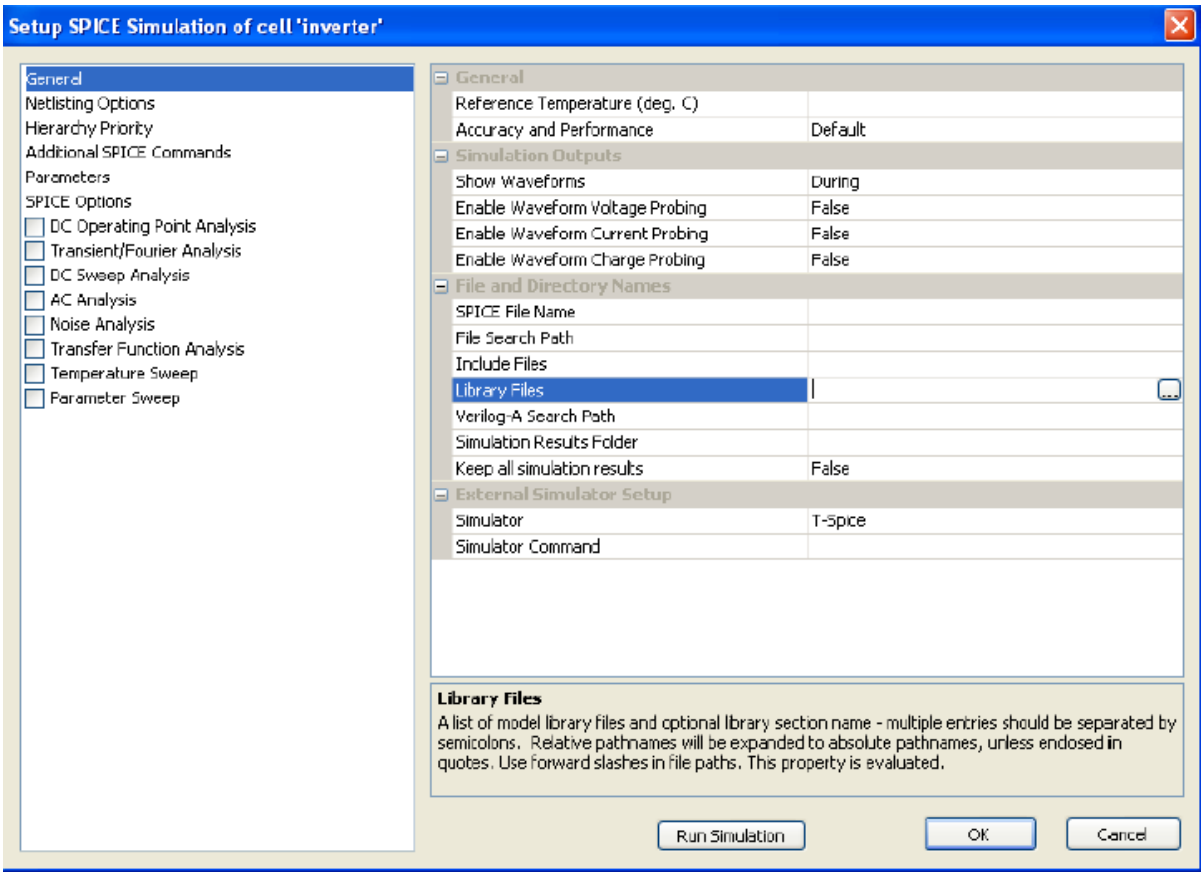



In Port Settings
Name:
In
Disable name validation
Font Size (pt)
8
Auto-repeat
Enable auto-repeat
Auto-increment:
Confirm text of each object
1
Justification
Orientation
West
OK
Cancel



Now we need to set up simulation.





Now , give the path of library file.

My Documents\Tanner EDA\Tanner Tools v15.0\Process\Generic_250nm\ Generic_250nm_Tech \Generic_250nm.lib

After giving the path of Generic_250nm.lib, add TT as shown below.

My Documents\Tanner EDA\Tanner Tools v15.0\Process\Generic_250nm\ Generic_250nm_Tech \Generic_250nm.lib TT

TT is the corner model used . There are different types of Corner models in .lib file

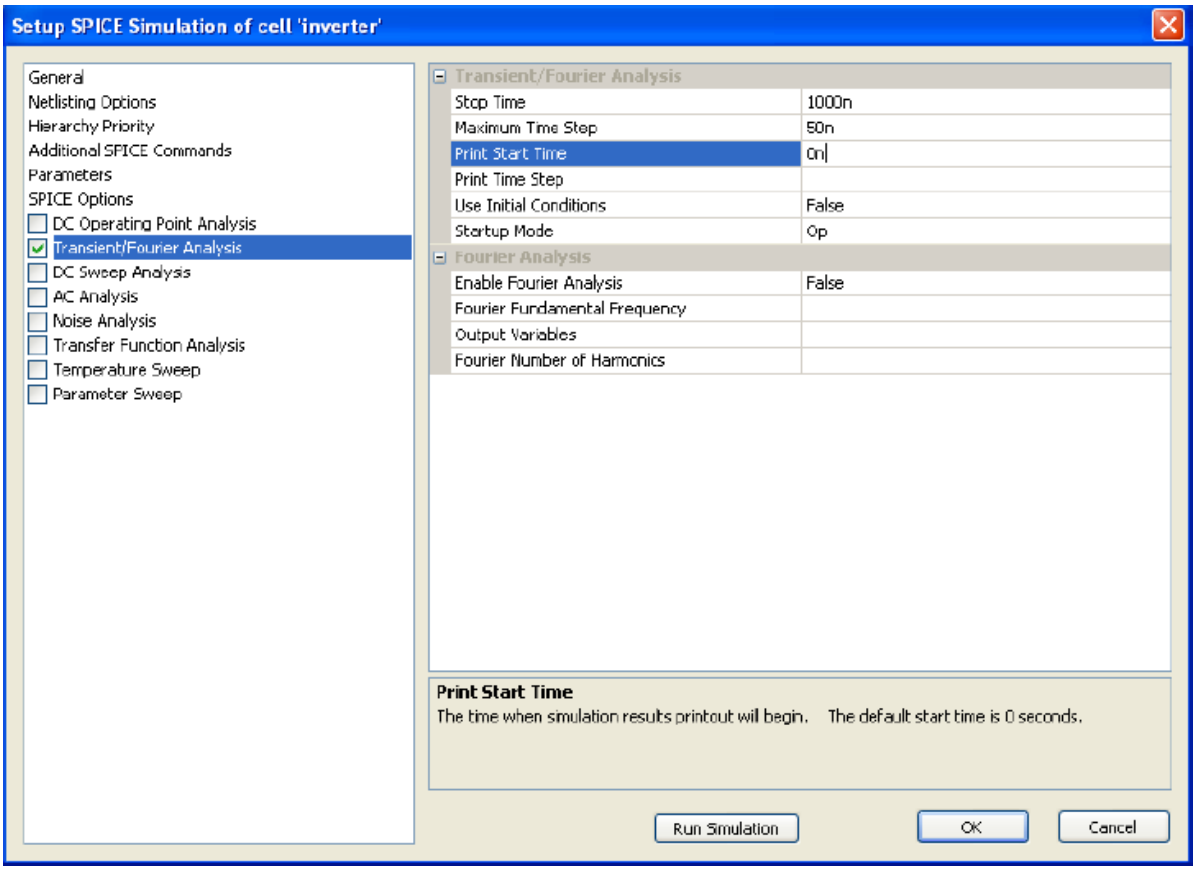
* TT : Typical model for NMOS & PMOS

* SS : Slow NMOS Slow PMOS model

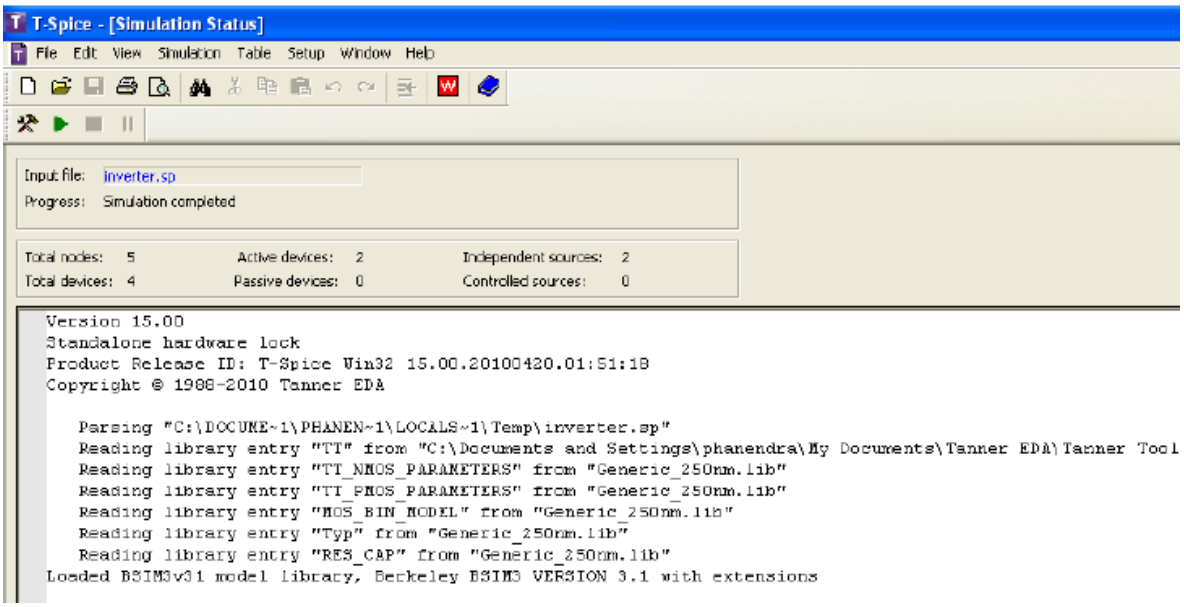
* FF : Fast NMOS Fast PMOS model

* SF : Slow NMOS Fast PMOS model

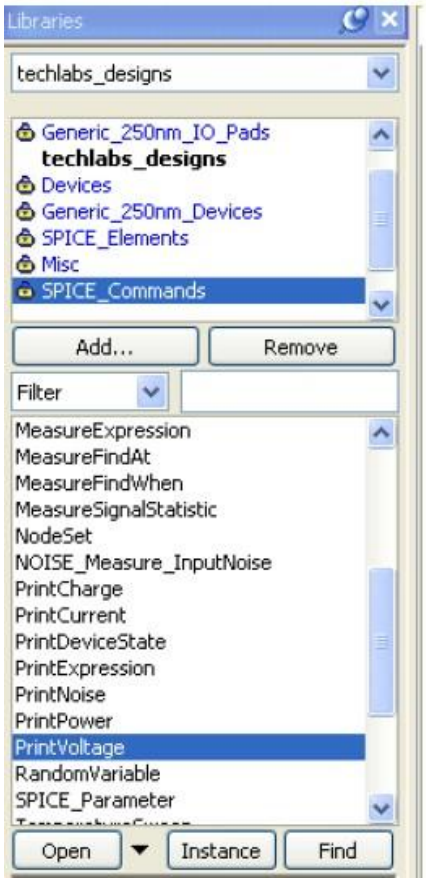
Set Transient Analysis as shown below .

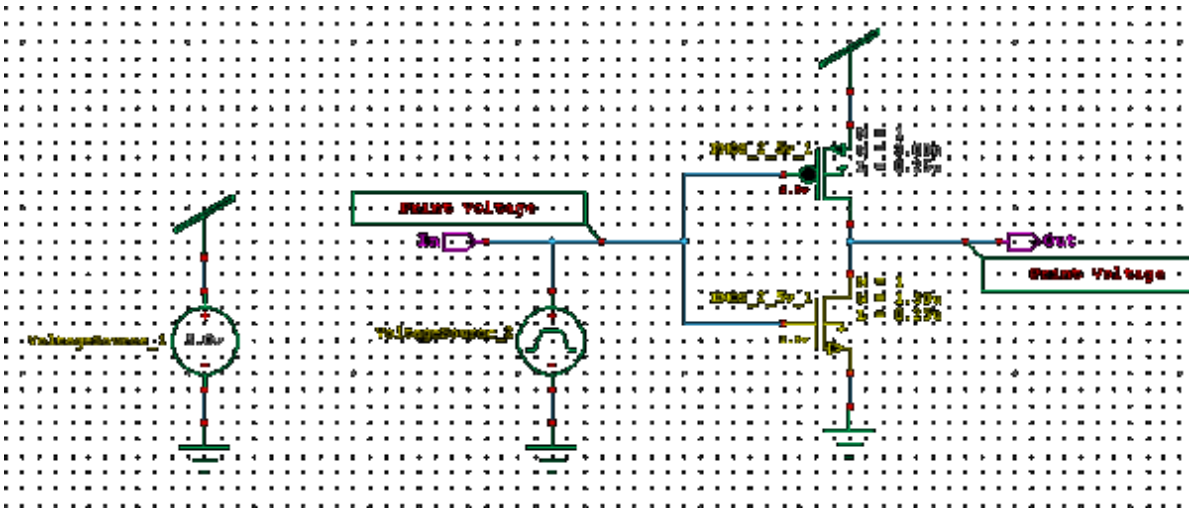


Run Simulation.

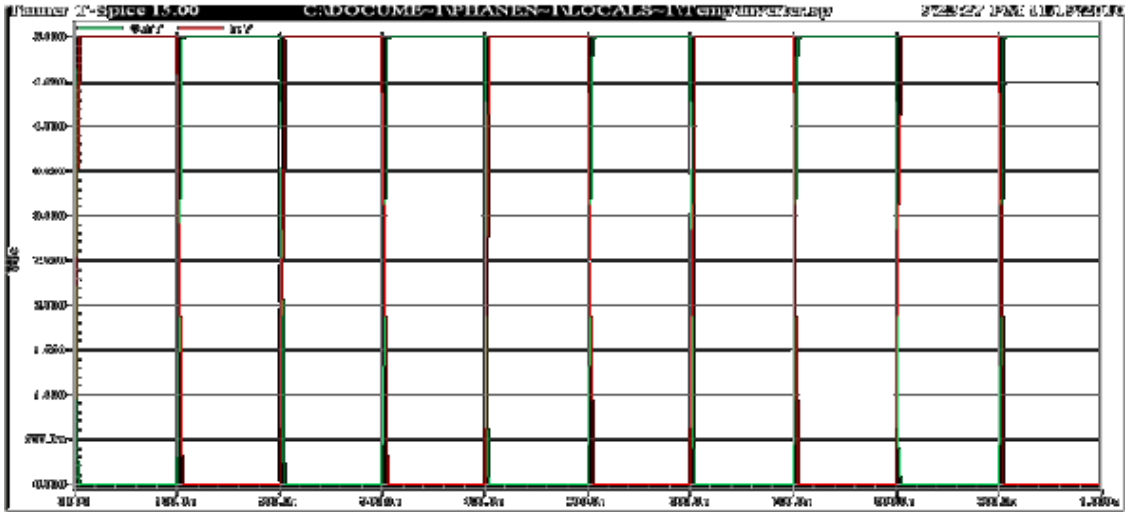


To view waveforms , place PrintVoltage from Spice Commands library.

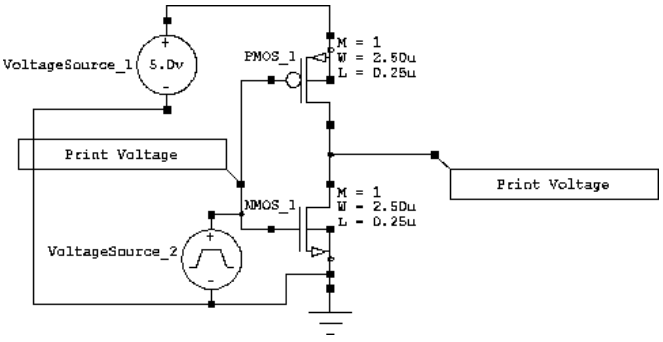




Waveforms can be viewed on W-Edit tool once Simulation is Run again.



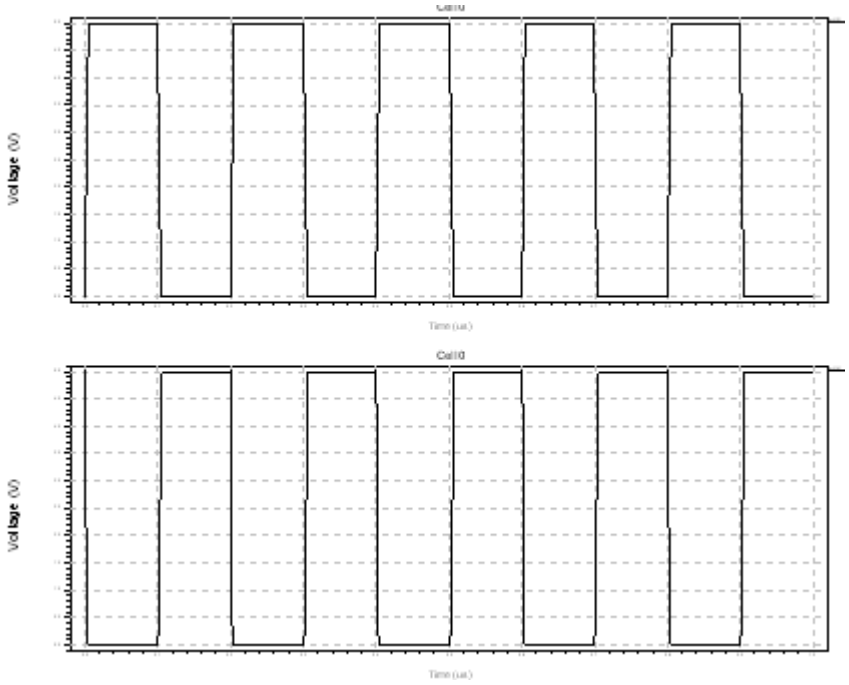
SCHEMATIC:



SPICE CODE

```
********* Simulation Settings - General section *********
.lib "C:\Documents and Settings\user\My Documents\Tanner EDA\Tanner Tools
v14.1\Libraries\Models\Generic_025.lib" tt

********* Simulation Settings - Parameters and SPICE Options *********
*-------- Devices: SPICE.ORDER > 0 --------
MNMOS_1 N_4 N_3 Gnd N_5 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MPMOS_1 N_4 N_3 N_2 N_1 PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
VVoltageSource_1 N_2 Gnd DC 5
VVoltageSource_2 N_3 Gnd PULSE(0 5 0 5n 5n 95n 200n)
.PRINT TRANV(N_4)
.PRINT TRANV(N_3)

********* Simulation Settings - Analysis section *********
.tran 100n 1u

********* Simulation Settings - Additional SPICE commands *********

.end
```

OUTPUT WAVEFORM



RESULT

Ex. No:	Schematic Entry and SPICE simulation of MOS Differential Amplifier
Date:	

AIM

To perform schematic entry of a CMOS inverter and verify its functionality using transient analysis.

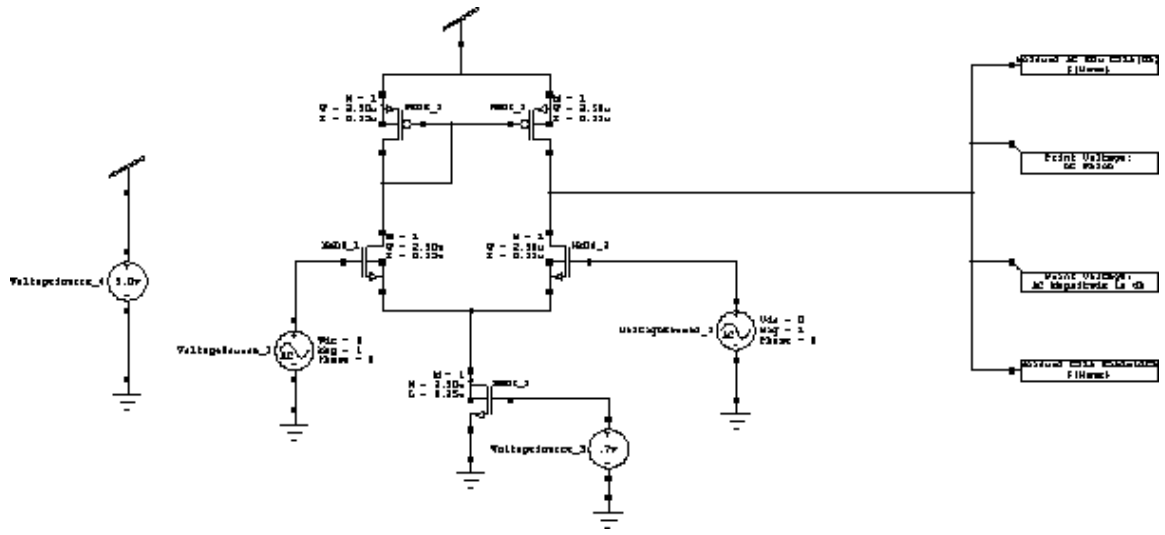
APPARATUS REQUIRED

PC with Tanner EDA tool.

PROCEDURE

- Enter the schematic of differential amplifier using S-Edit.
- Perform AC analysis of the differential amplifier.
- Go to „setup " in that select „spice simulation". Choose „ac analysis"and give the following values.
- Set „start frequency=10",‟stop frequency=10meg". „ no. of frequency=25",‟ sweep type=dec". click on „general" type and give path to Generic _250nm.lib. Then click OK. RUN simulation to get output.
- Obtain the frequency response from W-
- Edit. Obtain the spice code using T-Edit.
-

SCHEMATIC (Common Mode):



The simulation setup is as below

After completing circuit diagram goto setup -> spice simulation.

Choose ac analysis and give the following values . S tart

frequency =10

Stop frequency=10meg

No offrequency=25Sweep Type =dec.

Click on general type and give path to Generic_250nm.lib.

And run simulation to get output. For differential mode choose one of the ac source and change the phase to 180 in the property window.

SPICE CODE

```
********* Simulation Settings - General section *********
.lib "C:\Documents and Settings\user\My Documents\Tanner EDA\Tanner Tools
v14.1\Libraries\Models\Generic_025.lib" ff

********* Simulation Settings - Parameters and SPICE Options *********

*-------- Devices: SPICE.ORDER > 0 --------
MNMOS_1 N_2 N_3 N_4 N_4 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MNMOS_2 N_6 N_7 N_4 N_4 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MNMOS_3 N_4 N_11 Gnd N_5 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MPMOS_1 N_2 N_2 Vdd N_1 PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MPMOS_2 N_6 N_2 Vdd Vdd PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
VVoltageSource_3 N_11 Gnd DC 700m
VVoltageSource_4 Vdd Gnd DC 5
VVoltageSource_1 N_3 Gnd  DC 0 AC 1 0
VVoltageSource_2 N_7 Gnd  DC 0 AC 10
.PRINT AC Vdb(N_6)
.PRINT AC Vp(N_6)
.MEASURE AC AC_Measure_Gain_1 MAX vdb(N_6) ON
.MEASURE AC AC_Measure_GainBandwidthProduct_1_Gain MAX vdb(N_6) OFF
.MEASURE AC AC_Measure_GainBandwidthProduct_1_UGFreq WHEN Vdb(N_6)=0 OFF
.MEASURE AC AC_Measure_GainBandwidthProduct_1
PARAM='AC_Measure_GainBandwidthProduct_1_Gain*AC_Measure_GainBandwidthProduct_1_UGFreq' ON

********* Simulation Settings - Analysis section *********
.ac dec 25 10 10meg

********* Simulation Settings - Additional SPICE commands *********

.end
```

OUTPUT :

```
Generaloptions:
        threads =2
Device and node counts:
          MOSFETs-5              MOSFET geometries - 2
            BJTs-0                    JFETs -0
          MESFETs-0                   Diodes -0
        Capacitors-0                Resistors - 0
         Inductors-0              Mutual inductors -0
  Transmission lines-0     Coupled transmission lines -0
     Voltage sources-4             Current sources -0
            VCVS-0                     VCCS -0
            CCVS-0                     CCCS -0
    V-control switch-0           I-control switch -0
      Macro devices-0           Verilog-A devices - 0
        Subcircuits-0          Subcircuit instances -0
   Model Definitions-2             Computed Models -2
   Independent nodes-5             Boundary nodes - 5
       Total nodes -10

*** 2 WARNING MESSAGES GENERATED DURING SETUP

            Opening output file "C:\DOCUME~1\user\LOCALS~1\Temp\Cell0.out"
```

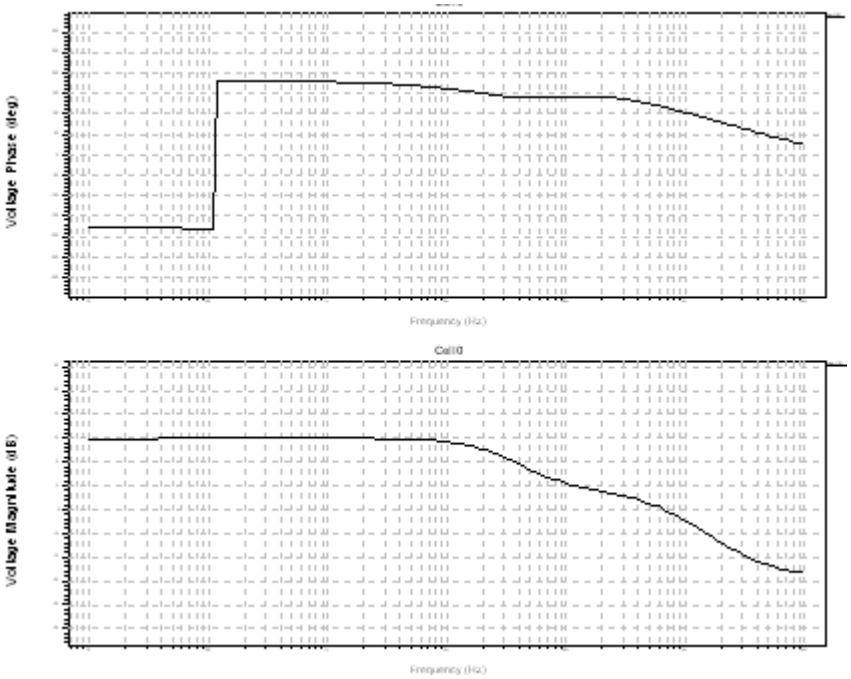
Measurement result summary

AC_Measure_Gain_1 = 9.9534e+000

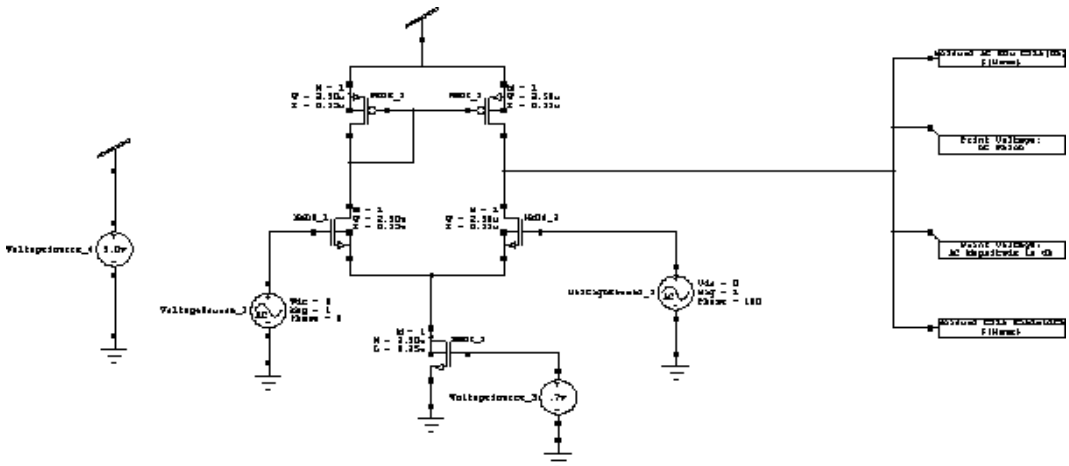
AC_Measure_GainBandwidthProduct_1 = 1.1555e+006

Parsing	0.00 seconds
Setup	0.01 seconds
DCoperatingpoint	0.00 seconds
ACAnalysis	0.00 seconds
Overhead	0.88 seconds
Total	0.89seconds

OUTPUTWAVEFORM:



SCHEMATIC (Differential Mode):



SPICE CODE

```
******** Simulation Settings - General section *********
.lib "C:\Documents and Settings\user\My Documents\Tanner EDA\Tanner Tools
v14.1\Libraries\Models\Generic_025.lib" ff

********* Simulation Settings - Parameters and SPICE Options *********

*-------- Devices: SPICE.ORDER > 0 --------
MNMOS_1 N_2 N_3 N_4 N_4 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MNMOS_2 N_6 N_7 N_4 N_4 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MNMOS_3 N_4 N_11 Gnd N_5 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MPMOS_1 N_2 N_2 Vdd N_1 PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
MPMOS_2 N_6 N_2 Vdd Vdd PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
VVoltageSource_3 N_11 Gnd DC 700m
VVoltageSource_4 Vdd Gnd DC 5
VVoltageSource_1 N_3 Gnd DC 0 AC 1 0
VVoltageSource_2 N_7 Gnd DC 0 AC 1 180
.PRINT AC Vdb(N_6)
.PRINT AC Vp(N_6)
.MEASURE AC AC_Measure_Gain_1 MAX vdb(N_6) ON
.MEASURE AC AC_Measure_GainBandwidthProduct_1_Gain MAX vdb(N_6) OFF
.MEASURE AC AC_Measure_GainBandwidthProduct_1_UGFreq WHEN Vdb(N_6)=0 OFF
.MEASURE AC AC_Measure_GainBandwidthProduct_1
PARAM='AC_Measure_GainBandwidthProduct_1_Gain*AC_Measure_GainBandwidthProduct_1_UGFreq' ON

********* Simulation Settings - Analysis section *********
.ac dec 25 10 10meg

********* Simulation Settings - Additional SPICE commands *********

.end
```

OUTPUT:

```
 Generaloptions:
        threads =2

 Device and node counts:
          MOSFETs-5              MOSFET geometries - 2
            BJTs-0                    JFETs -0
          MESFETs-0                   Diodes -0
        Capacitors-0                Resistors - 0
         Inductors-0              Mutual inductors -0
 Transmission lines-0      Coupled transmission lines -0
    Voltage sources-4            Current sources -0
            VCVS-0                    VCCS -0
            CCVS-0                    CCCS -0
   V-control switch-0           I-control switch -0
     Macro devices-0           Verilog-A devices - 0
       Subcircuits-0          Subcircuit instances -0
  Model Definitions-2              Computed Models -2
 Independent nodes-5               Boundary nodes - 5
      Total nodes -10

*** 2 WARNING MESSAGES GENERATED DURING SETUP

            Opening output file "C:\DOCUME~1\user\LOCALS~1\Temp\Cell0.out"
```

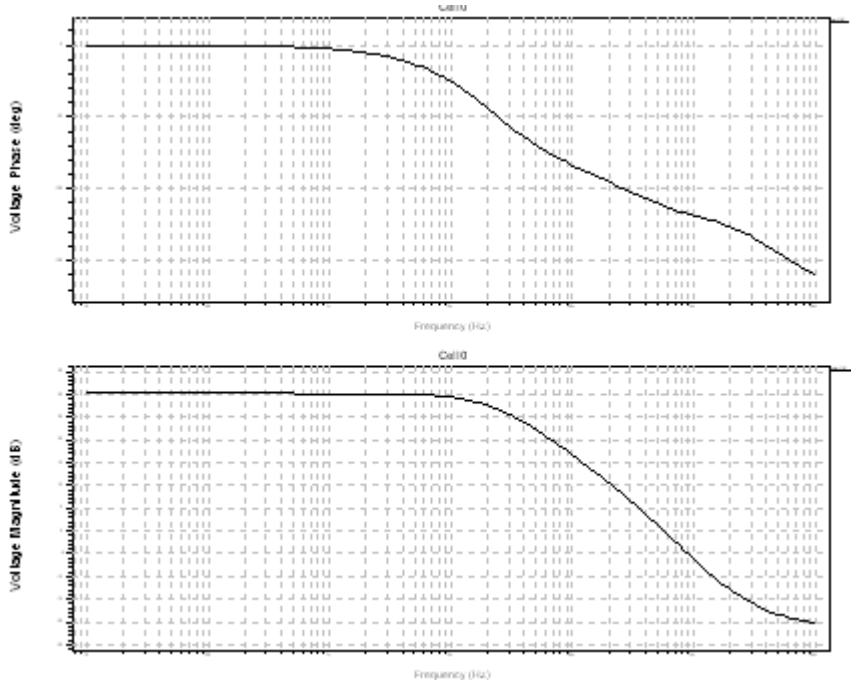
Measurement result summary

AC_Measure_Gain_1 = 3.0225e+001

AC_Measure_GainBandwidthProduct_1 = 1.7161e+007

Parsing	0.02 seconds
Setup	0.01 seconds
DCoperatingpoint	0.00 seconds
ACAnalysis	0.01seconds
Overhead	0.47seconds
Total	0.51 seconds

OUTPUT WAVEFORM



Calculation

CMRR = Diff Mode/Common Mode

= 30.31/9.9

= 3.061

RESULT

Ex. No:	Implementation Layout of A Simple Cmos Inverter
Date:	

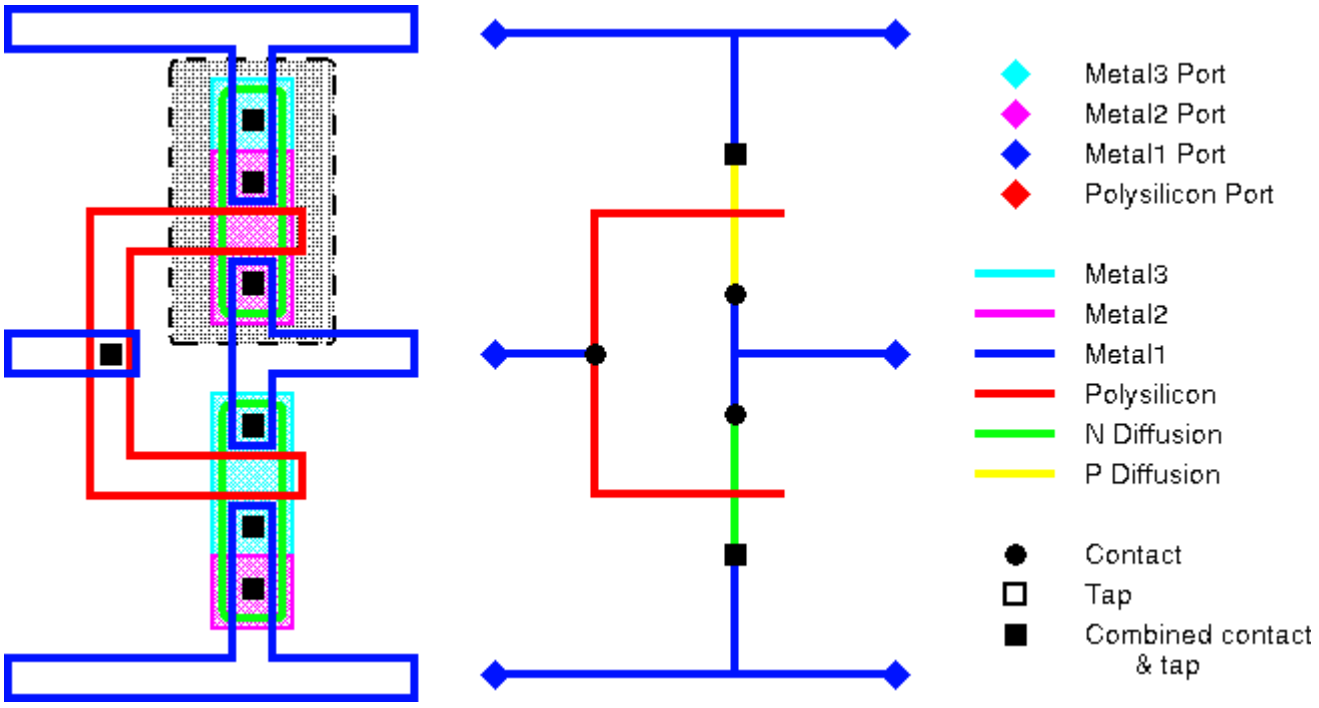
AIM

To implement Layout of a simple CMOS inverter.

SOFTWARE REQUIRED

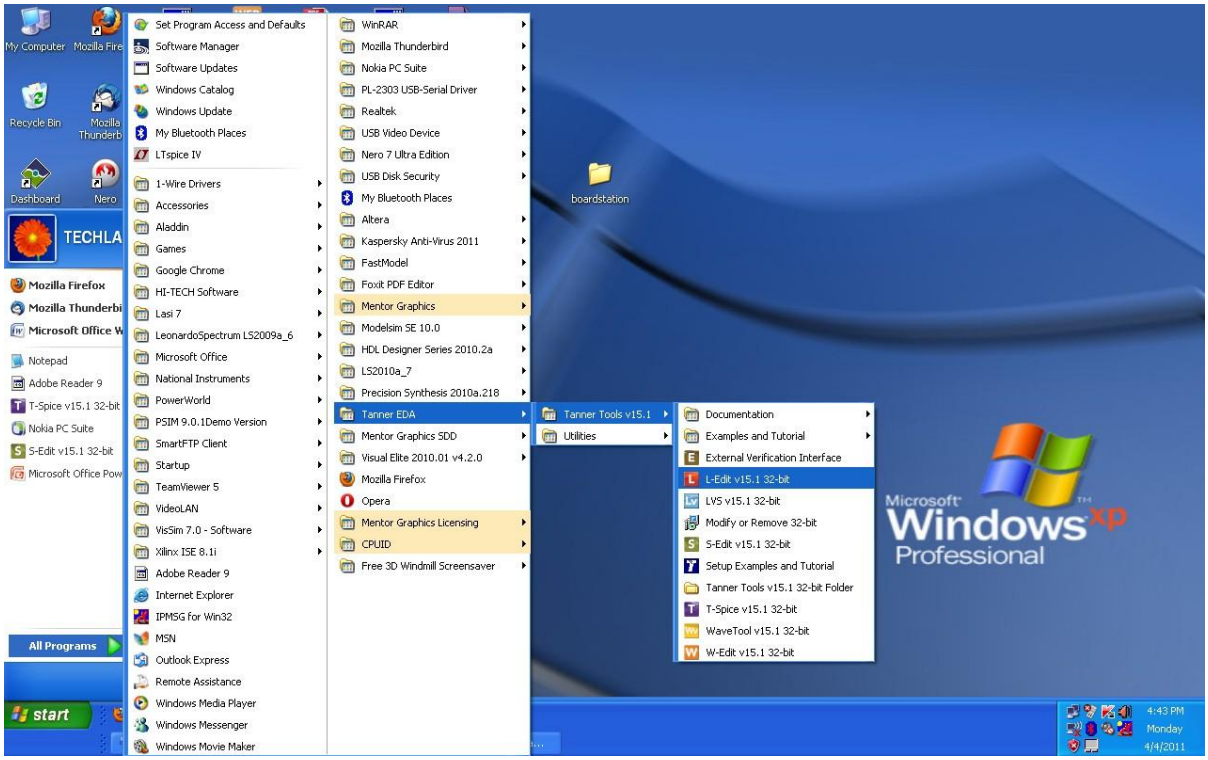
PC with Tanner EDA tool.

DESCRIPTION OF CMOS INVERTER:

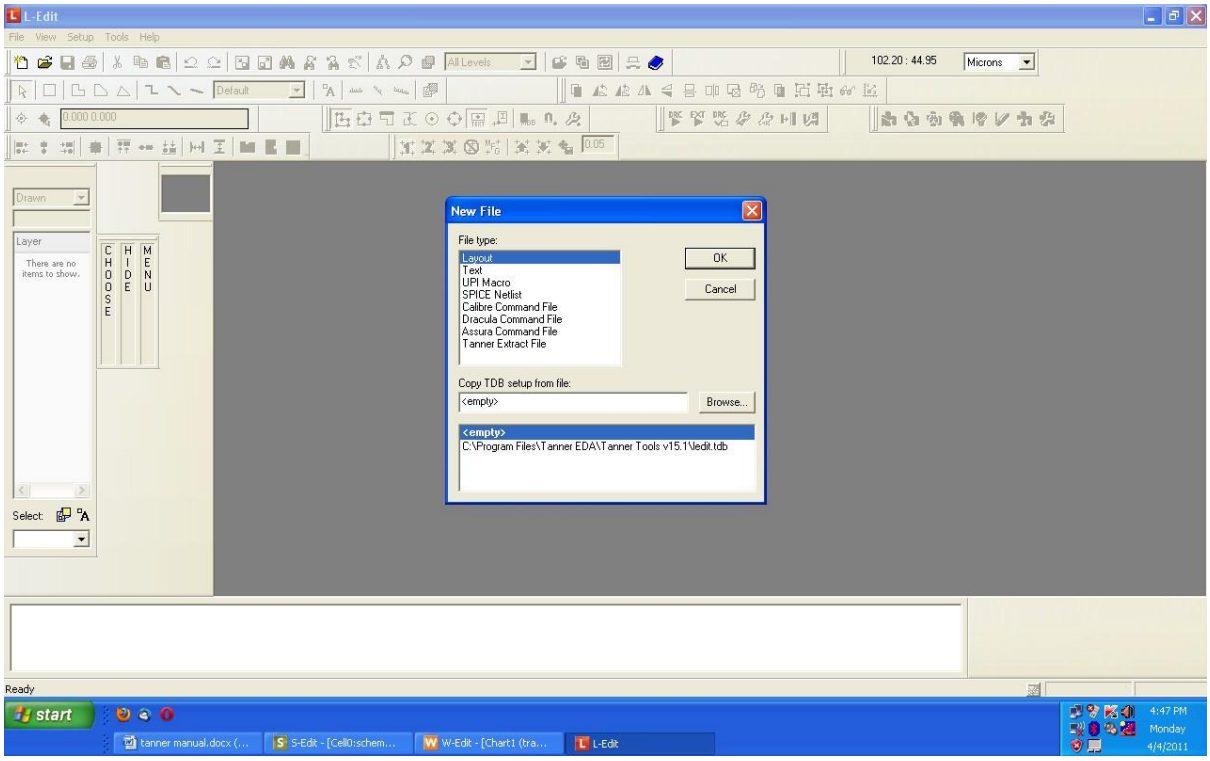


PROCEDURE

- "Create a New File"
- To make the layout of the inverter we need to get the layout of an nmos, a pmos.
- Click on Create and select Rectangle. Draw a rectangle, which covers the gates (red rectangles) of both the nmos and pmos.
- Connect this poly layer with the m1_poly via, by making another rectangle of poly, covering the central square of the via and touching the vertical poly layer connecting the 2gates.
- Now to make power and ground lines select metal1
- on Give terminal names: VDD GND as input output pins.

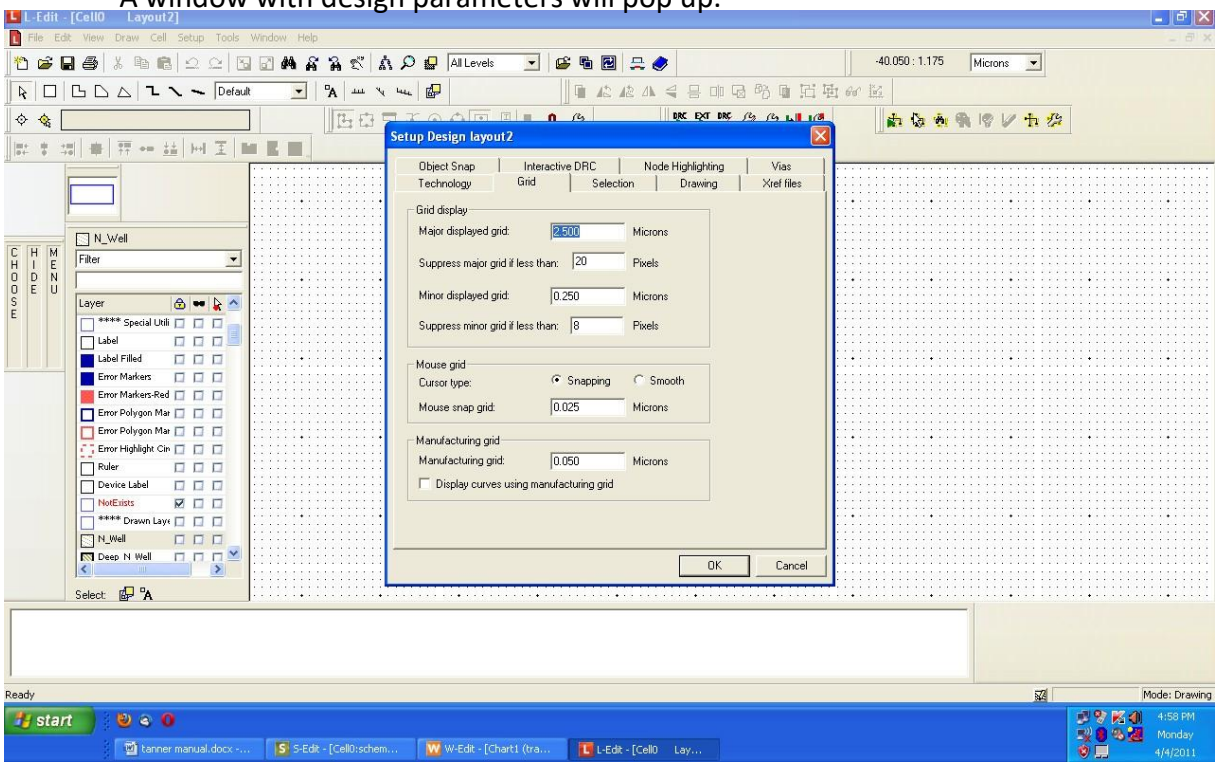


LEdit opens as below. In L-Edit Click on file->new. A window pops up

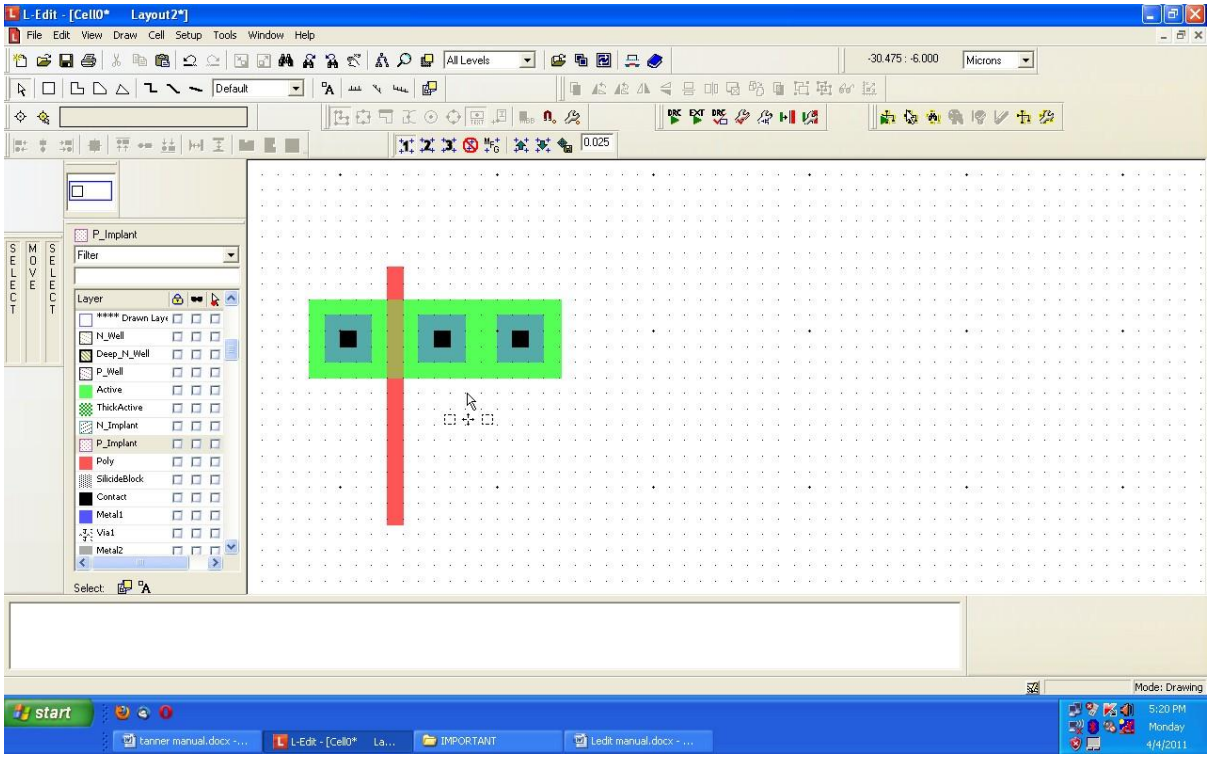


- In this window layout is highlighted and there is a browse button.
- Click on it. And browse to the following path. “My Documents\Tanner EDA\Tanner Tools v15.1\ Process\ Generic_250nm\ Generic_250nm_Tech\ Generic_250nm_TechSetup.tdb” and click ok.

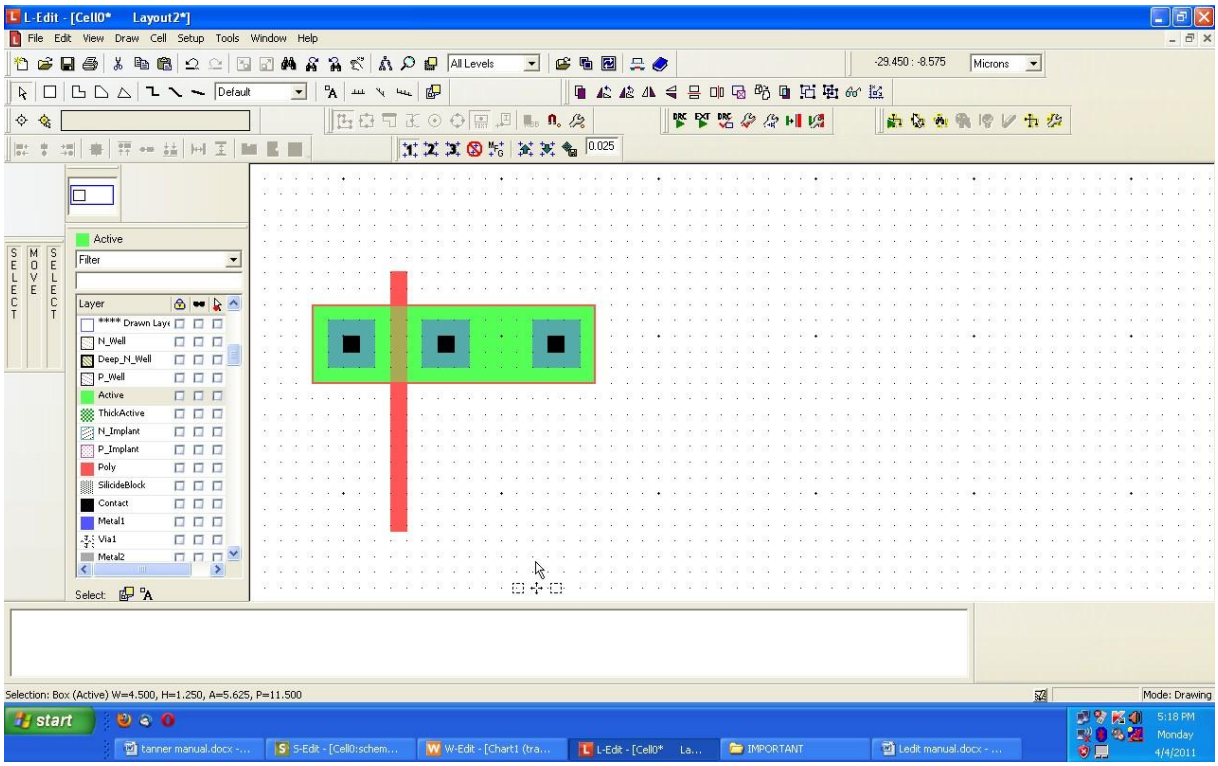
- Use + and – keys in num pad to zoom in and zoom out. Tap twice on + key to have the grids visible. Goto -> view->display->Major grids.
- Now major grids will be visible.
- Goto -> setup->design
- A window with design parameters will pop up.



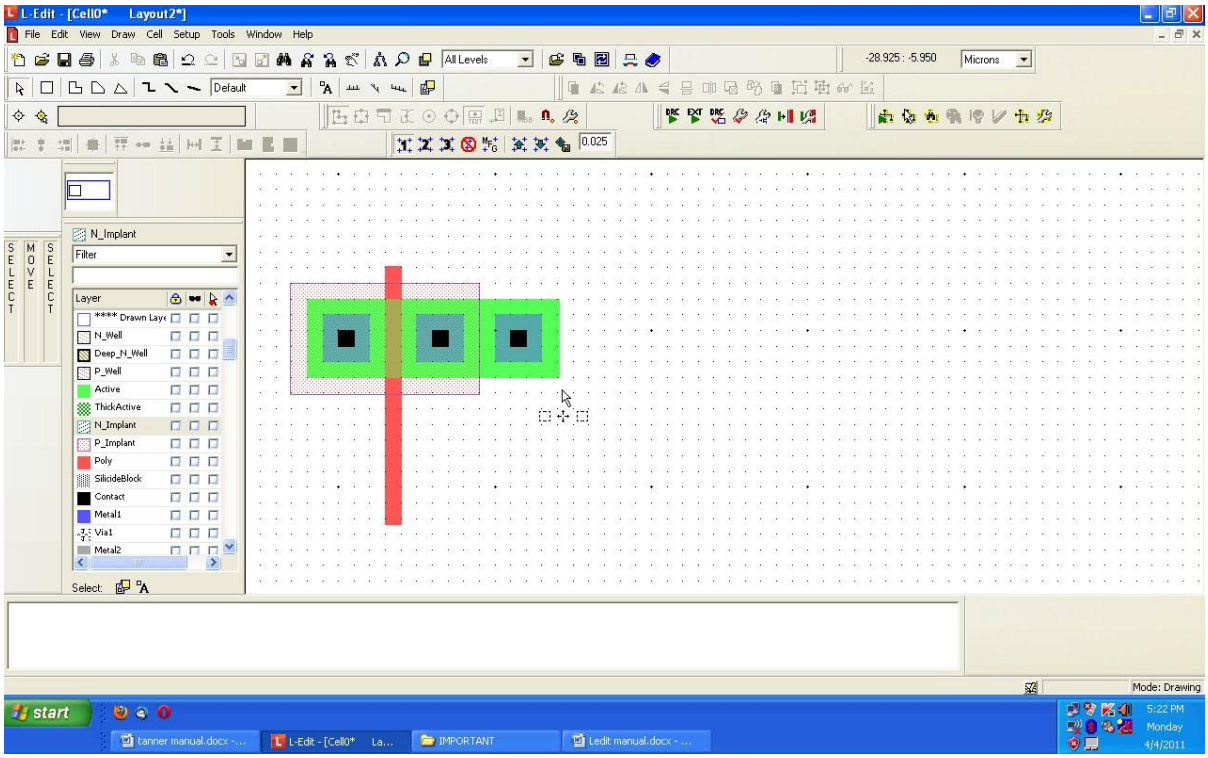
- Click on grid button and we get to see the distance between the 2 grid points as 0.250 Microns.
- In this manual the distance between 2 grid points is called as lambda for our convenience.
- First draw an active contact and click on the square box and draw a square for 1x1 lambda.
- Click on metal and choose square box .
- Metal around any contact is 3x3 lambda. Red Poly 1X X. Spacing between Poly & contact is 1 Lambda.
- Place 2 other metal and contact combination near poly as
- below. Distance between 2 (metal and contact) combination is 3 lambda.



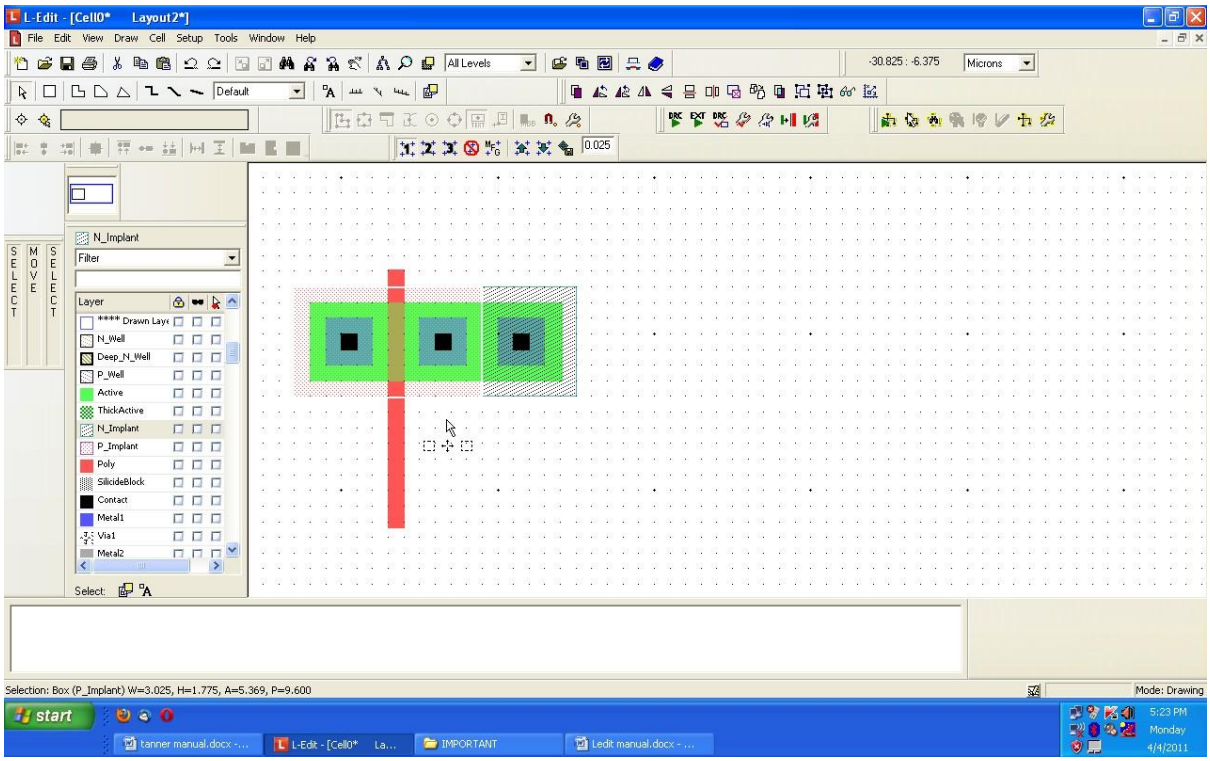
- Active region should cover metal region by 1 Lambda as shownabove



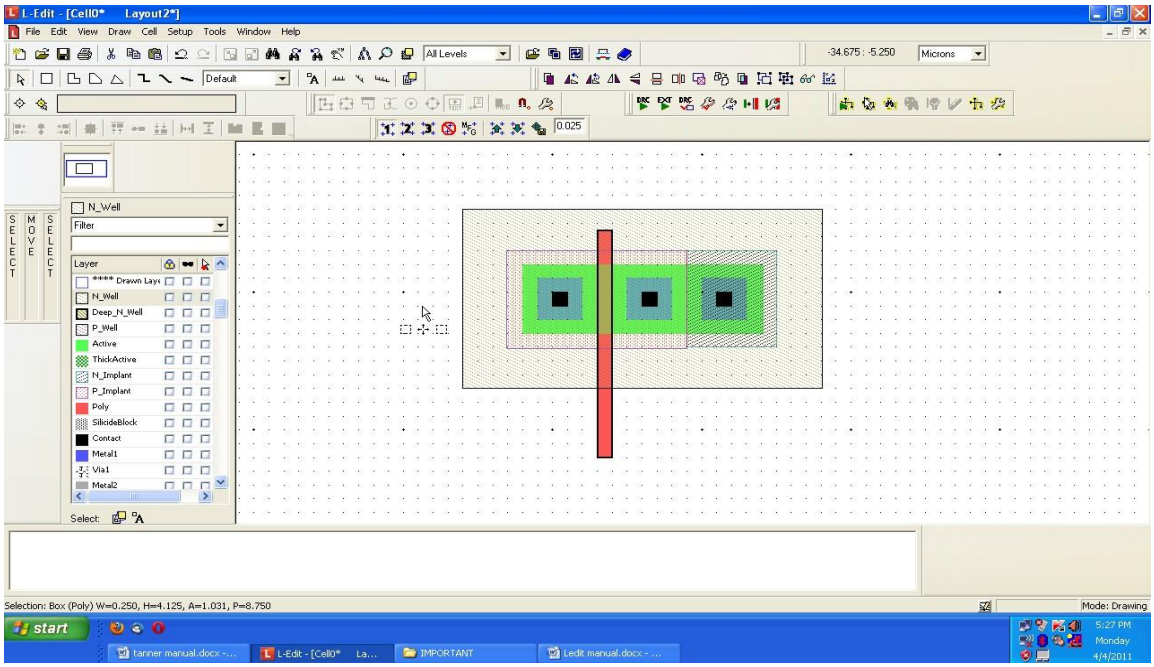
- Pselect should cover metal, poly, metal combination by 1 lambda as below. Pselect defines the transistor as pmos.



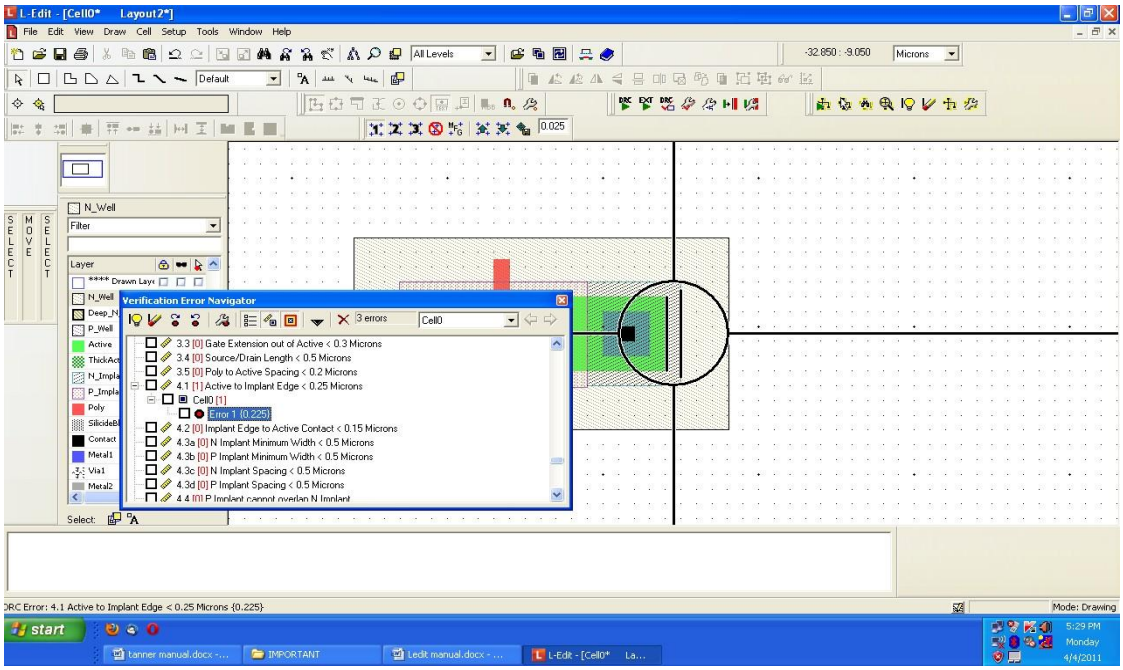
- n-select to coveractive region by 11 lambda as below.



- Pmos is grown in n-well. Distance between (p/nselect or p/n implant) is 3 lambda.



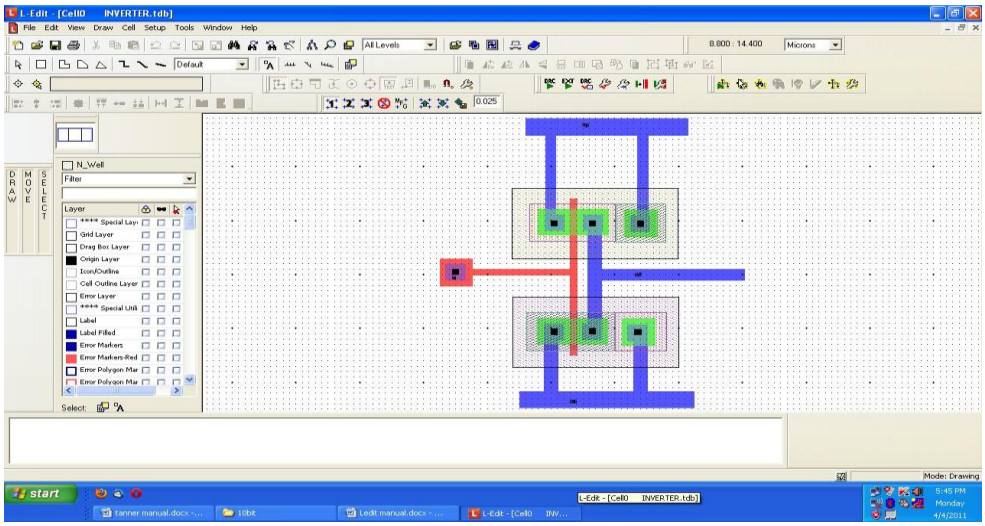
- Pmos is now completed. Run DRC to check for any



- Rectify the errors and clear DRC.
  Errors poly density error and metal density errors can be ignored. For Creating Nmos Copy & paste Pmos
  Change p-implant/select to n-implant/select and vice versa.
  Procedure for changing p-select to n-select is given below.
  Click on the border of Pselect (goto Edit Tab-> Edit objects-> Change to Nselect in the Drop down menu.)

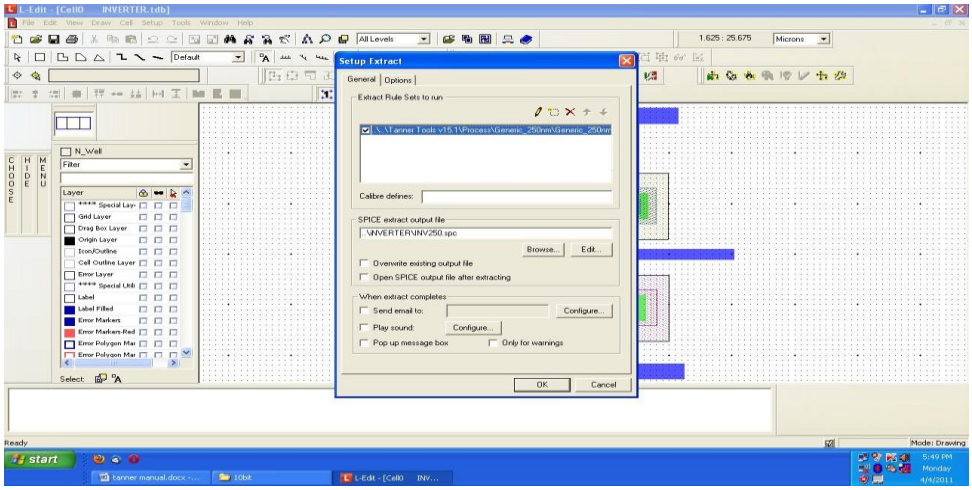
Now We have our Nmos.

Complete the connections and label the ports using Switch to drawing port which looks like A.



Run DRC once again to check for any error.

For extraction goto tools-> extract setup and make a tick on Generic_250.ext. go to options and uncheck anything under hyper verify



Now click on extract button to get the netlist for the layout. The following is the extracted netlist

****

***** * SPICE netlist generated by HiPer Verify's NetList Extractor * *

Extract Date/Time: Mon Feb 04 17:52:49 2013 *

L-Edit Version: L-Edit Win32 15.10.20101227.13:21:06 *

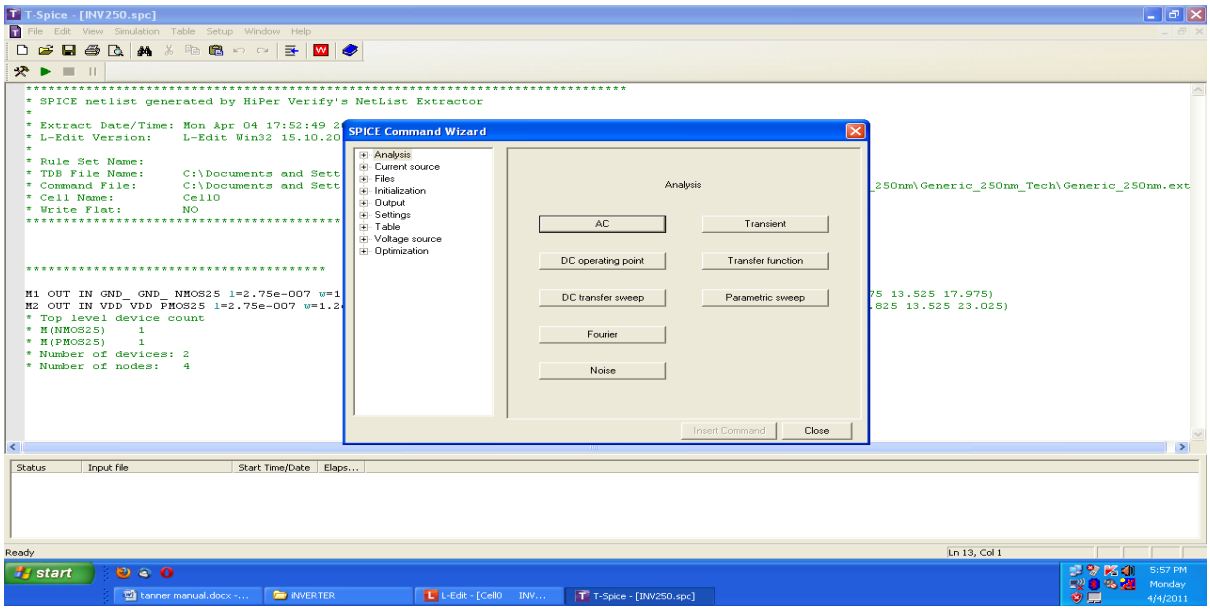
* Rule SetName:

* TDB File Name: C:\Documents and Settings\ramesh\My Documents\Tanner EDA\Work\10bit\INVERTER.tdb

Command File: C:\Documents and Settings\ramesh\My Documents\Tanner EDA\Tanner Tools v15.1\Process\Generic_250nm\Generic_250nm_Tech\Generic_250nm.ext

* Cell Name: Cell0

Write Flat:NO



- Expand files icon and click on library and browse to generic 250.lib the path is as below “My Documents\Tanner EDA\Tanner Tools v15.1\ Process\ Generic_250nm\ Generic_250nm_Tech\Generic_250nm.lib”

- In library section give “tt” and click on insert command to insert the commandin tspice.

- Similarly inser the following command using insert command options or copy them from schematicnetlist.

VVoltageSource_1 Vdd Gnd DC 5 $ $x=4900 $y=2900 $w=400 $h=600

VVoltageSource_2 In Gnd PULSE(0 5 0 5n 5n 95n 200n) $ $x=6600 $y=2200 $w=400 $h=600 .

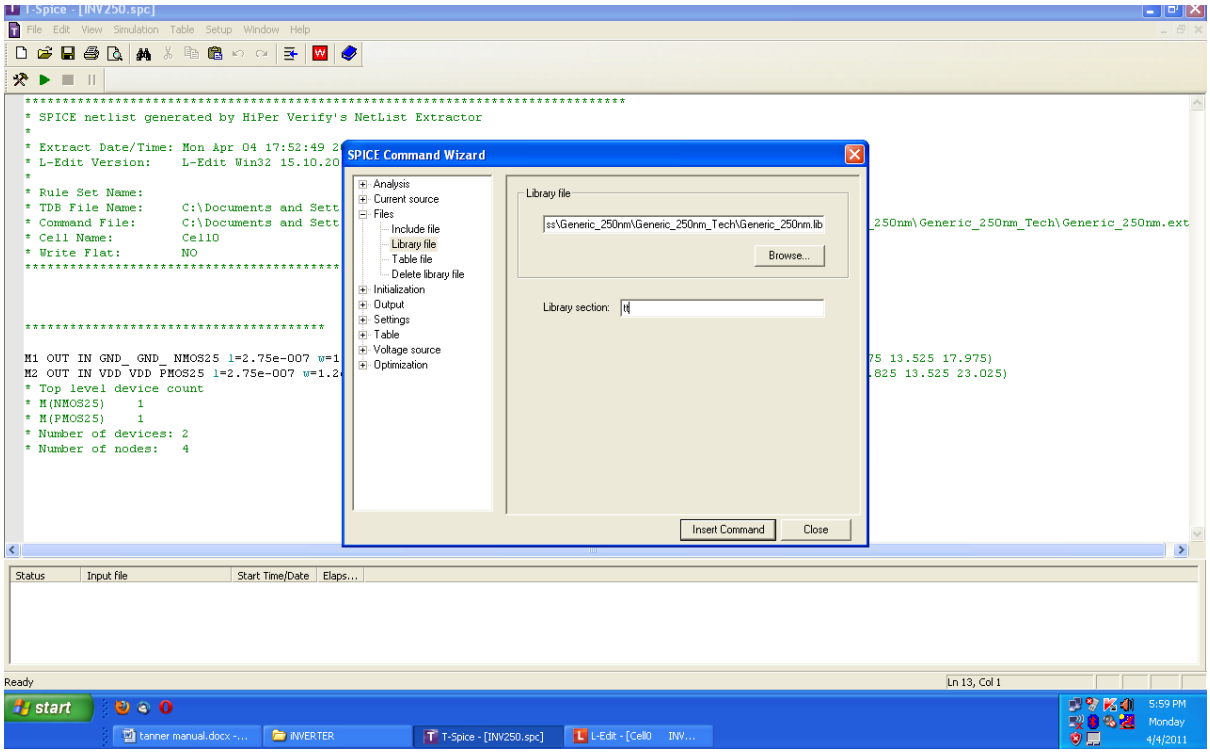
PRINT TRAN V(In) $ $x=5950 $y=2850 $w=1500 $h=300 $r=180 .

PRINT TRAN V(Out) $ $x=8350 $y=2550 $w=1500 $h=300 *******

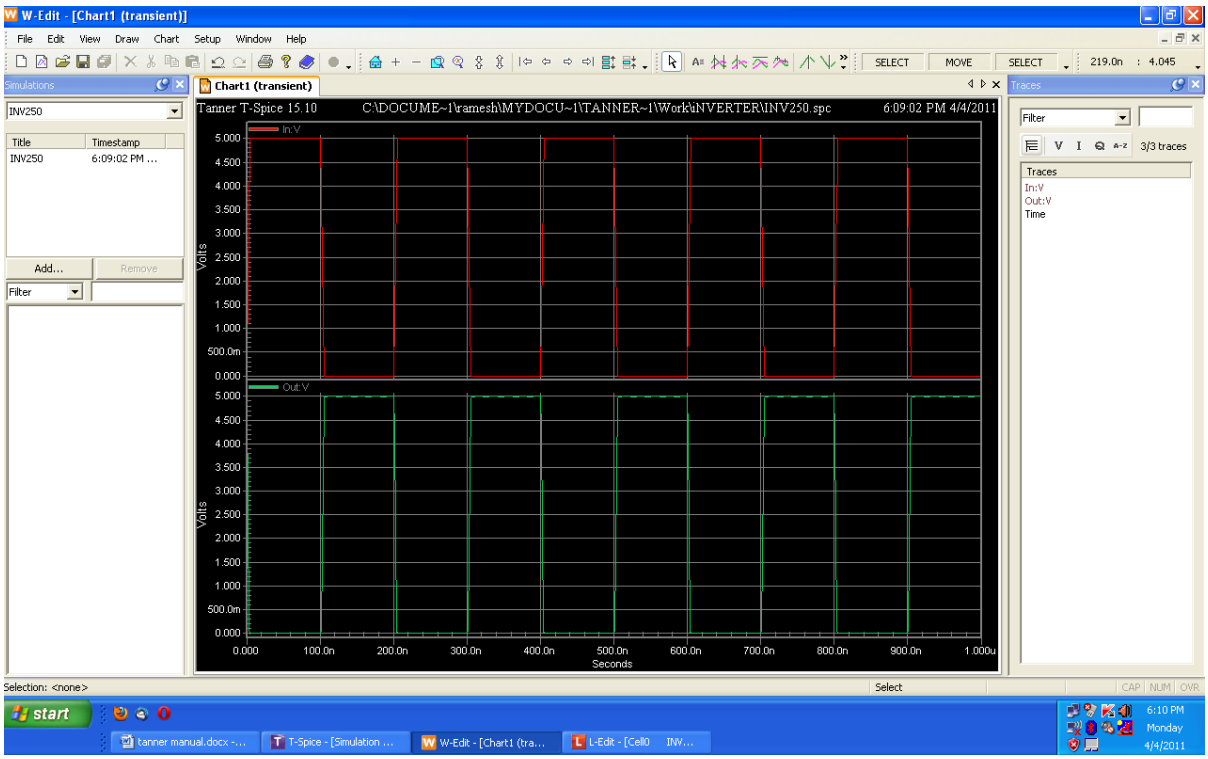
Simulation Settings - Analysis Section *********

.tran 5n 1u start=0**

.end



OUTPUT WAVEFORM



RESULT

Ex. No:	
Date:	**Implementation Layout of A Simple Cmos Inverter By using microwind**

AIM

To develop the layout design for CMOS inverter by using MICROWIND and obtain the simulation results.

MICROWIND

The MICROWIND program allows designing and simulating an integrated circuit at physical description level. The package contains a library of common logic and analog ICs to view and simulate.

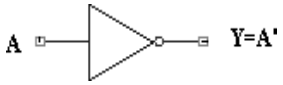
1. MICROWIND LAYOUT DESIGNTOOLS:

	Open a layout file (MSK format)		Extract and simulate the circuit
	Save the layout file in MSK format		Measure the distance in lambda and micron between two points
	Draw a box using the selected layer of the palette		2D vertical aspect of the device
	Delete boxes or text.		Step by step fabrication of the layout in 3D
	Copy boxes or text		Design rule checking of the circuit. Errors are notified in the layout
	Stretch or move elements		Add a text to the layout. The text may include simulation properties.
	Zoom In		Connect the lower to the upper layers at the desired location using appropriate contacts.
	Zoom Out		Static MOS characteristics
	View all the drawing		View the palette
	Extract and view the electrical node pointed by the cursor		Move the layout up, left, right, down

2. PURPOSE OF DESIGN RULES:

1. As all the layouts are prepared with minimum λ design rules .With minimum λ , layout will requireless amount of area on chip, so larger circuits can be implemented with in lessarea.
2. Makes the circuit to consumes less amount power i.e. as area and polysilicon usage can also reduced, resulting less resistance ,which in turn causes to dissipate less amount ofpower.
3. Design rules will give optimizedlayout.

LOGIC DIAGRAM

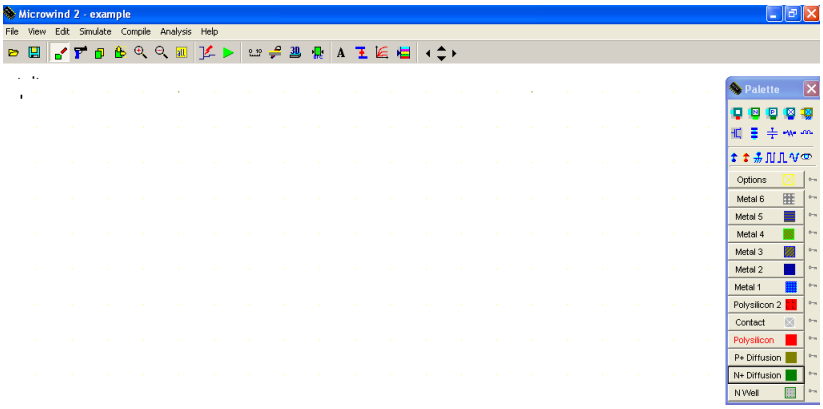


TRUTH TABLE:

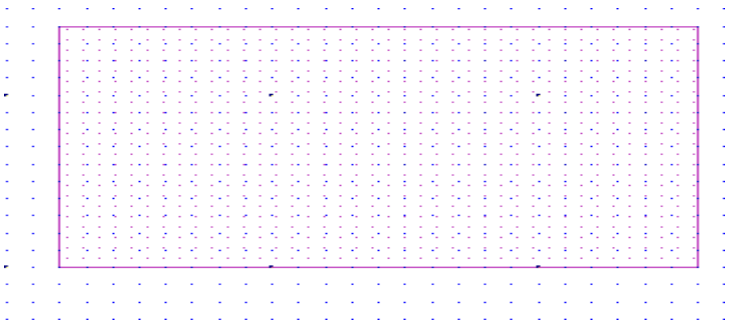
A	Y=A'
0	1
1	0

LAYOUT DESIGN STEPS

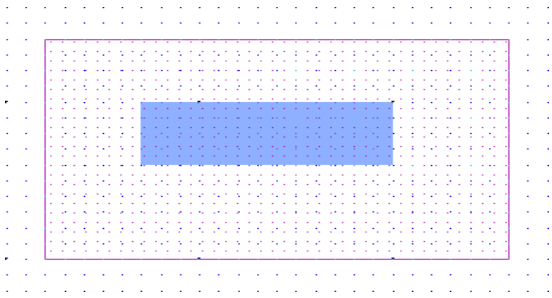
STEP 1: Open MICROWIND window which is considered as PWELL



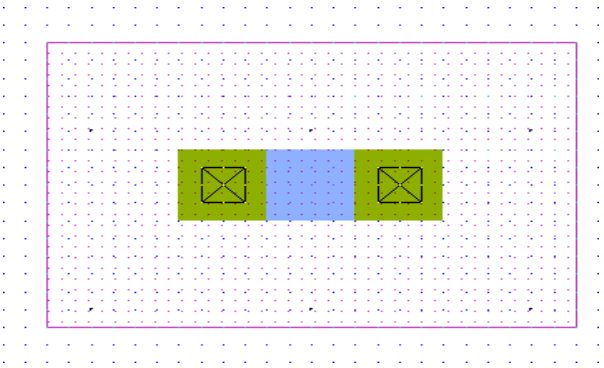
STEP 2: Place n-well of required length



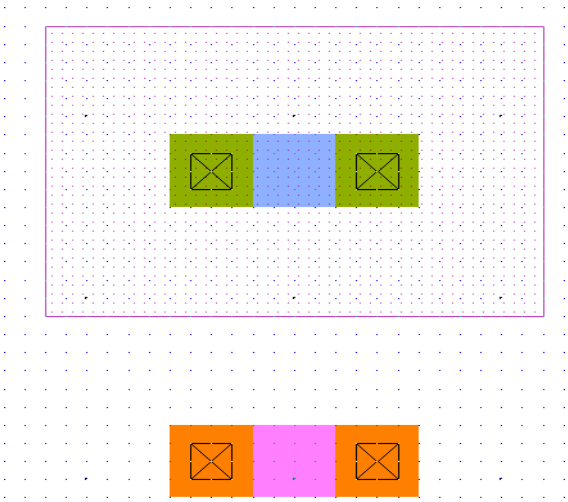
STEP 3: Take P-diffusion of required length and 4λ width



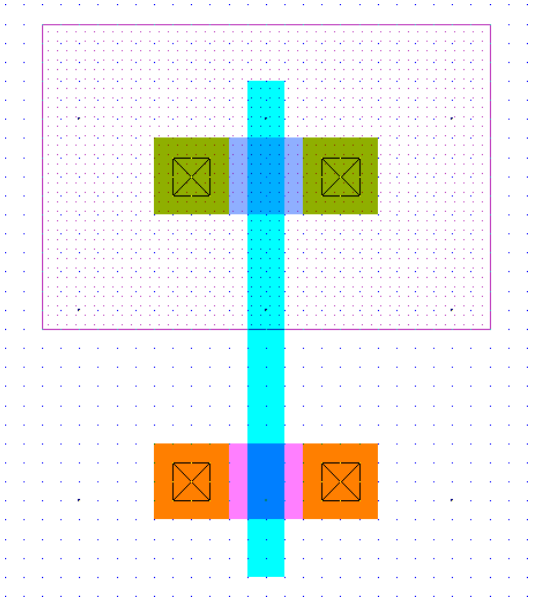
STEP 4: Take NWELL and place it surrounding the P-diffusion, it must have 6λ on all the sides from p-material



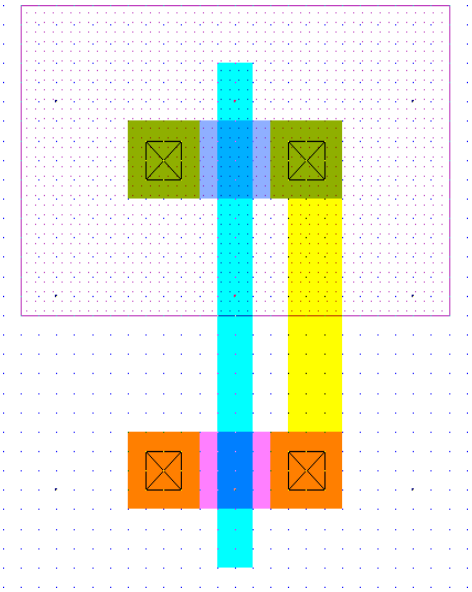
STEP 5: Do same for N-diffusion and maintain a distance of 6λ from NWELL



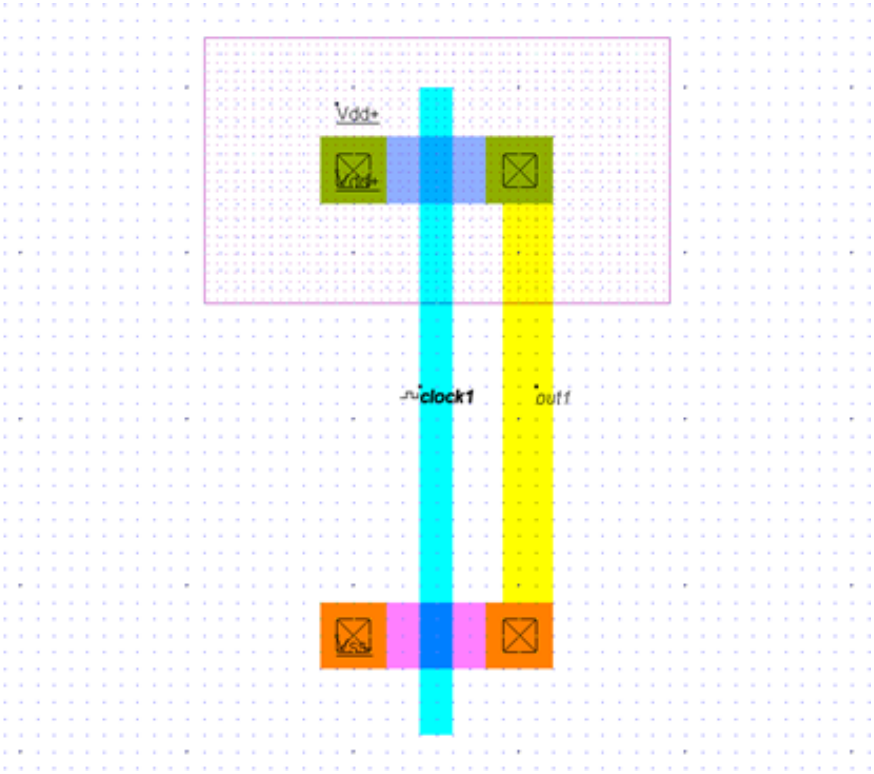
STEP 6: Make gate contact by using poly silicon material of width 2λ and maintain 1λ distance Between n or p contact and extend 3λ above and below the p and n diffusion.



STEP 7: Connect p-contact and n-contact using metal1, which is meant for taking output

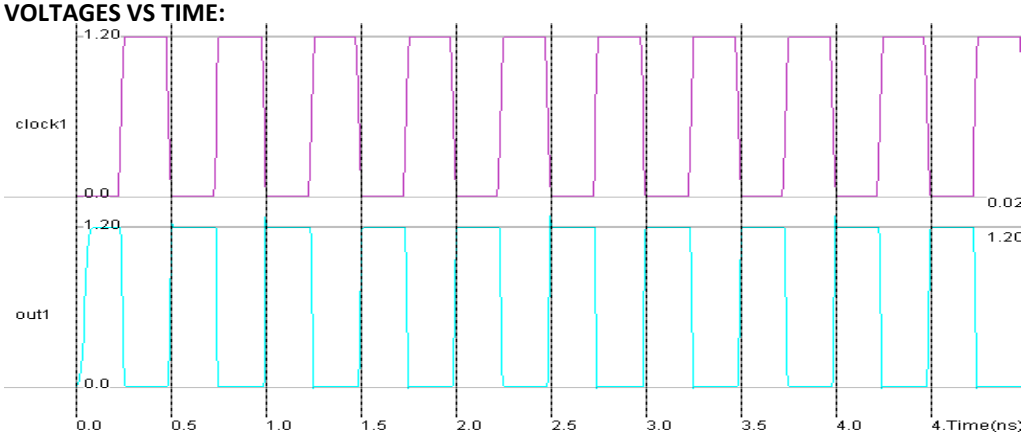


STEP 8: Give supply voltage (Vdd) to p-diffusion and Vss to n-diffusion and finally inverter layout will as follows:

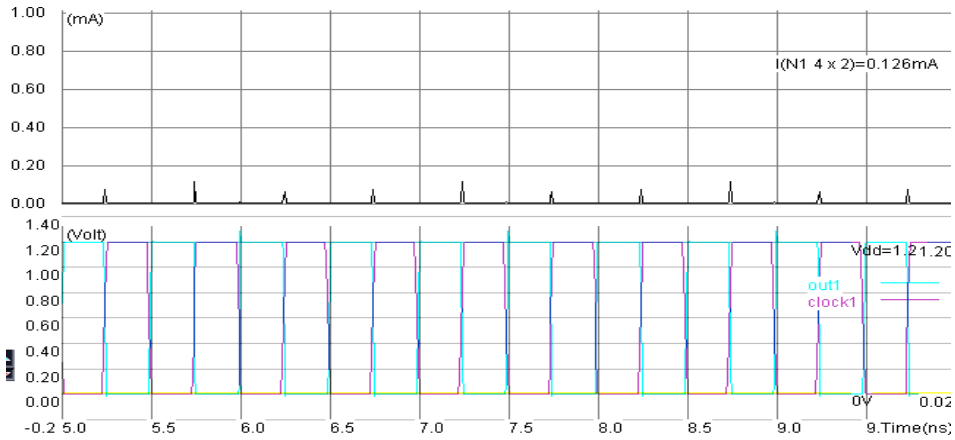


SIMULATION OUTPUT:

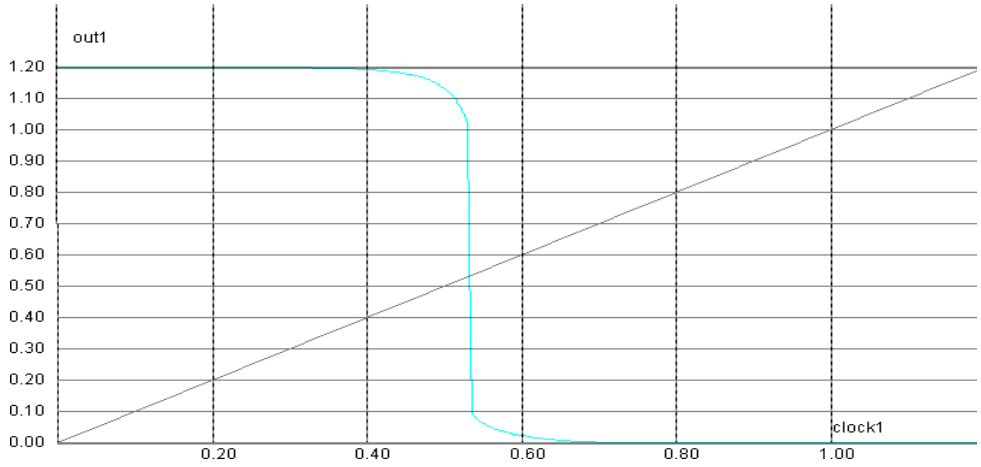
VOLTAGES VS TIME:



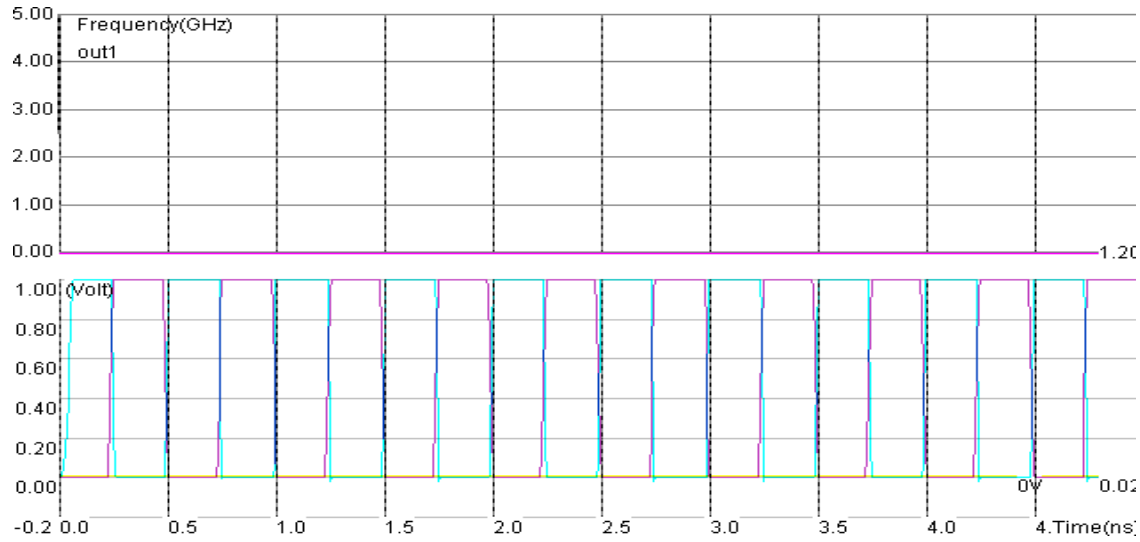
VOLTAGES VS CURRENT:



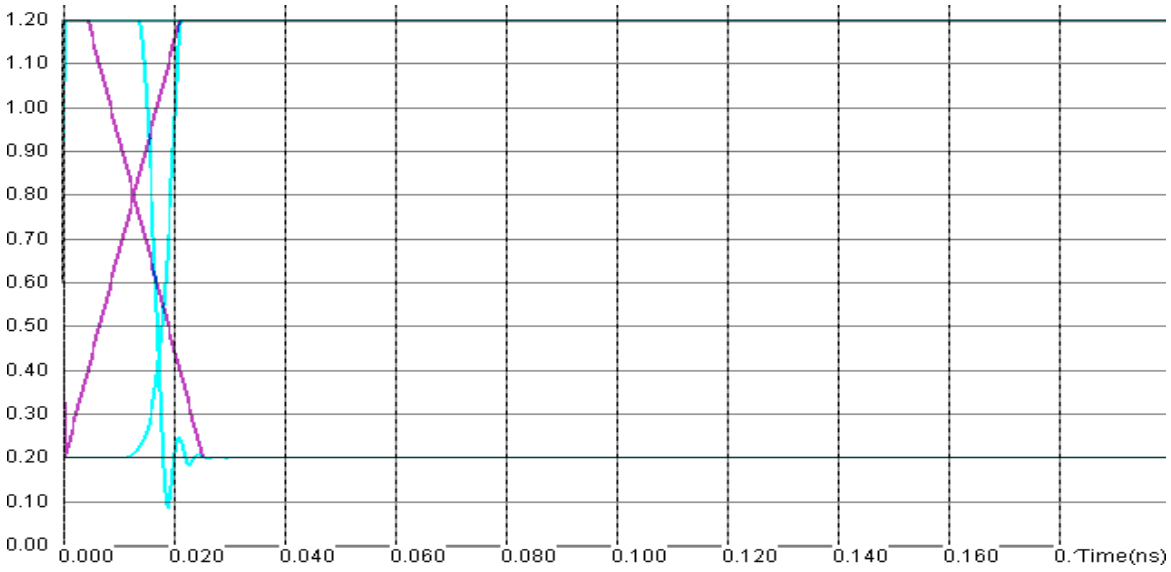
VOLTAGES VS VOLTAGES:



FREQUENCY VS TIME:



EYE DIAGRAM:



OBSERVATION TABLE:

NO. OF TRANSISTORS USED			POWER CONSUMPTION
DEVICE	PMOS TRANSISTOR	NMOS TRANSISTOR	
Inverter	1	1	1.283µw
Total number of transistors	1	1	

PROCEDURE:

PMOS TRANSISTOR

1. The complete window can be considered asP-well.
2. The N-well is placed over the p-well in a rectangularbox.
3. Next we need to place the P-diffusion over theN-well.
4. Width of the P-diffusion should be4λ.
5. The metal contacts can be drawn by selecting a (4λ*4λ)square.

NMOS TRANSISTOR

1. There should be minimum 6λ space between N-well andN-diffusion.
2. The same steps can be followed for Nmos as inPmos.
3. The grid itself is P-well, we can start directly with n-diffusionlayer.
4. The Pmos and Nmos should be connected by using polysilicon layer of width2λ.
5. The polysilicon layer should be extended up to 3λ length above and below P and N diffusionlayers.
6. Leave minimum 1λ space on both sides of polysilicon between thecontacts.
7. The source to drain connection is made using a metal of width3λ.

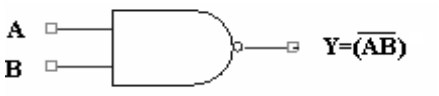
RESULT

Ex. No:	BASIC LOGIC GATES
Date:	

AIM:

To develop the layout design for basic logic gates by using MICROWIND and obtain the simulation results.

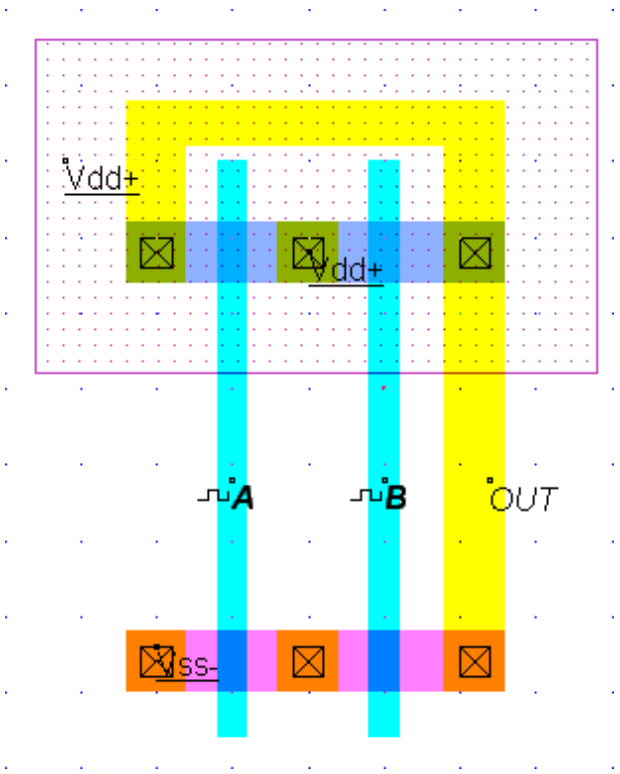
LOGIC DIAGRAM:NANDGATE:



TRUTHTABLE:

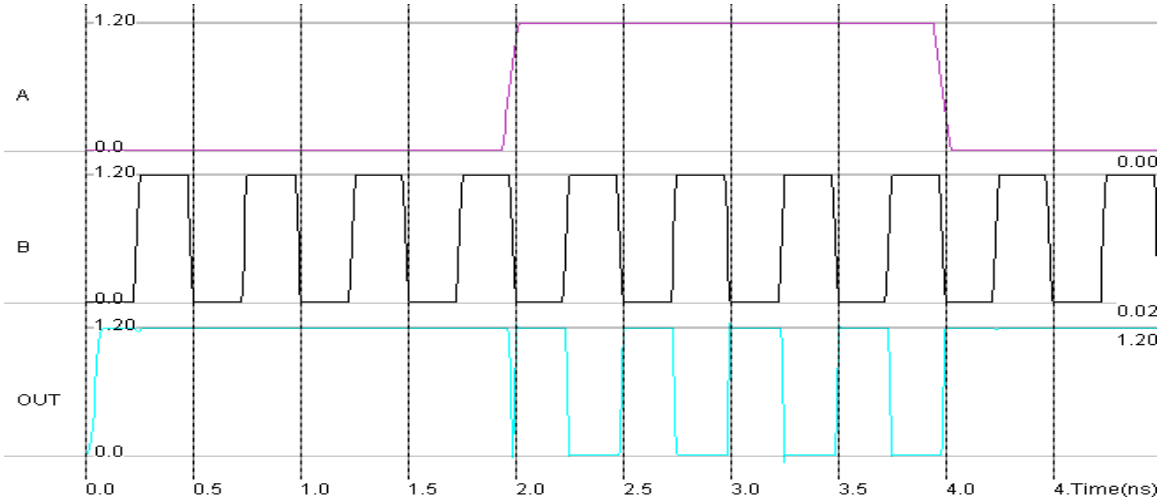
A	B	Y=(AB)'
0	0	1
0	1	1
1	0	1
1	1	0

LAYOUT DESIGN

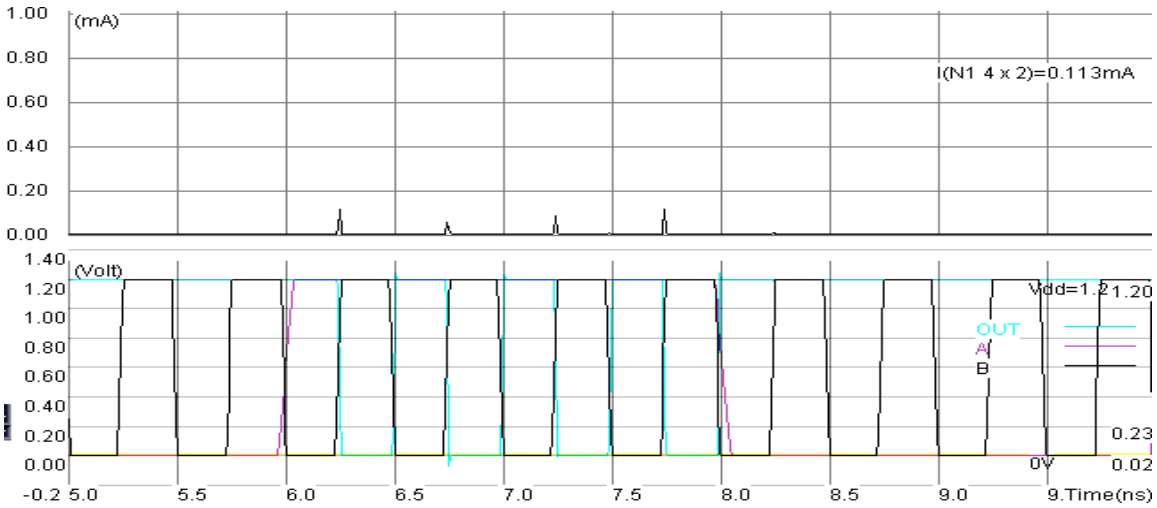


SIMULATION OUTPUT:

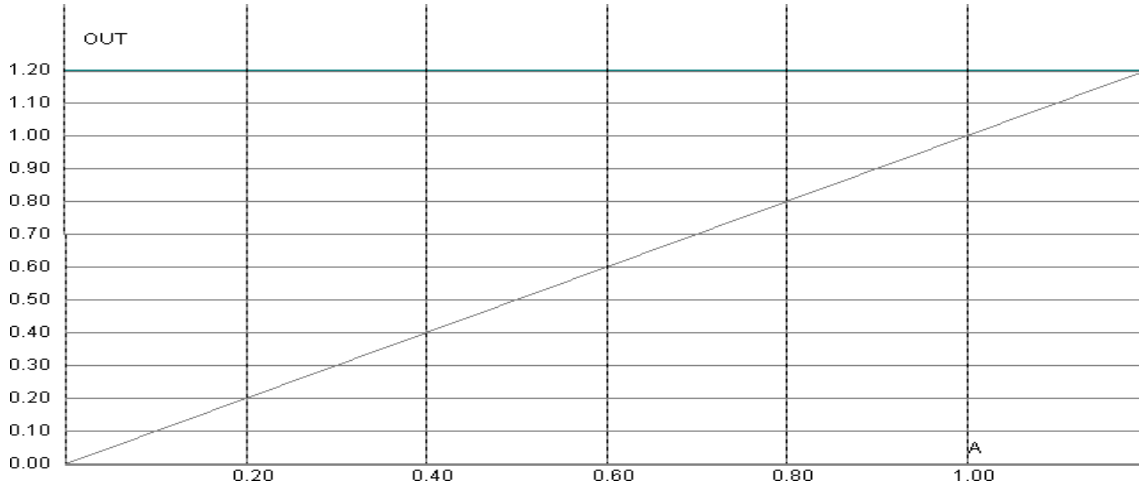
VOLTAGE VS TIME:



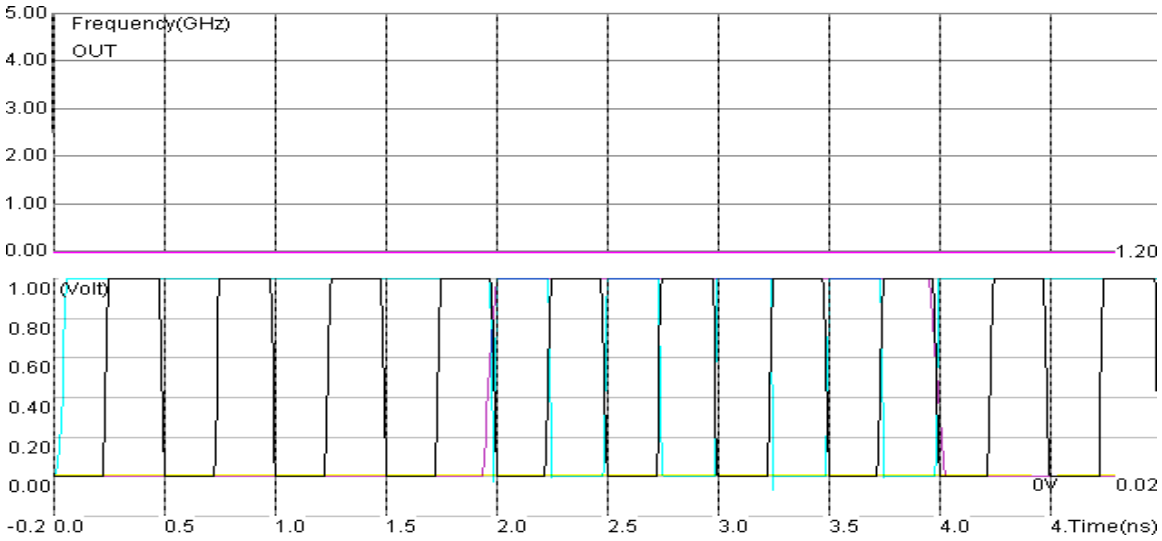
VOLTAGES VSCURRENTS:



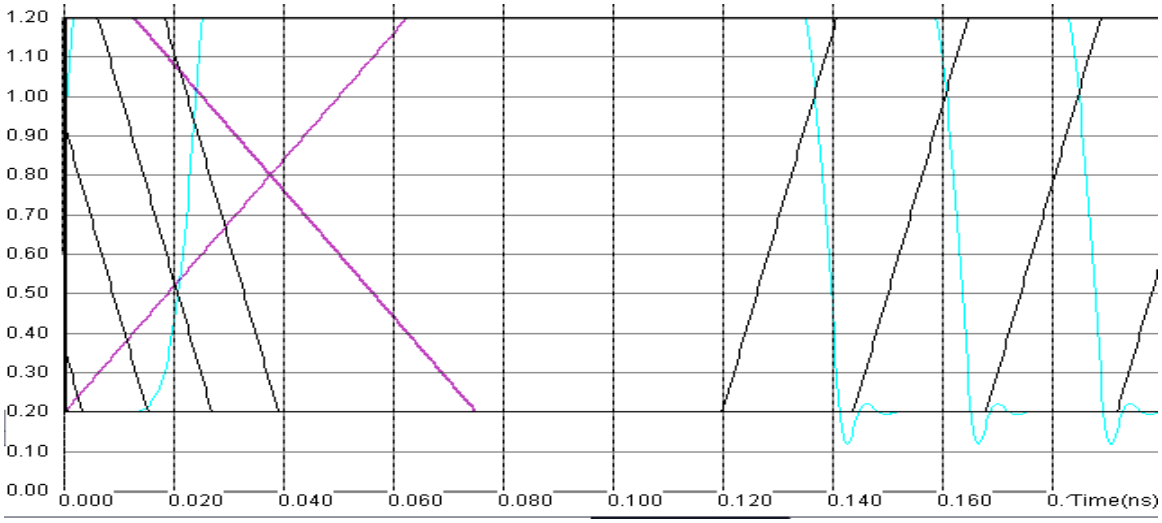
VOLTAGES VSVOLTAGES:



FREQUENCY VS TIME:



EYE DIAGRAM:

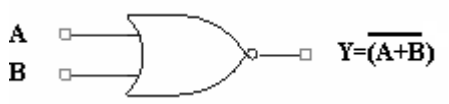


OBSERVATION TABLE

NO. OF TRANSISTORS USED			POWER CONSUMPTION
DEVICE	PMOS TRANSISTOR	NMOS TRANSISTOR	
NAND GATE(2i/p)	2	2	1.306µw
Total numberof transistors	2	2	

NOR GATE:

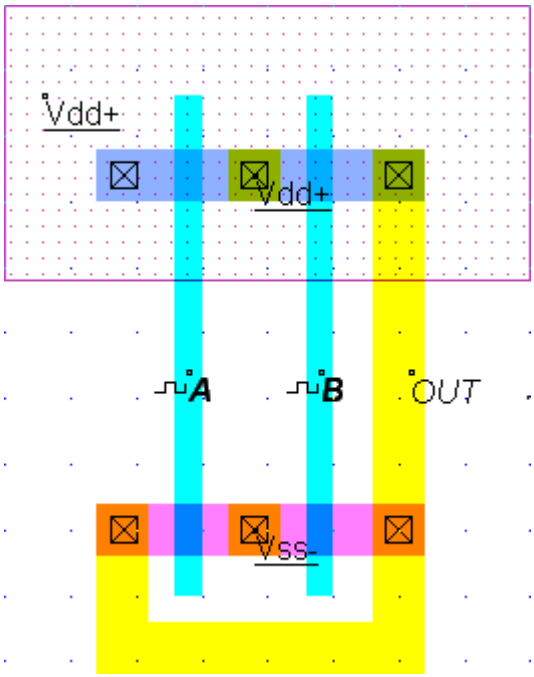
LOGIC DIAGRAM:



TRUTH TABLE:

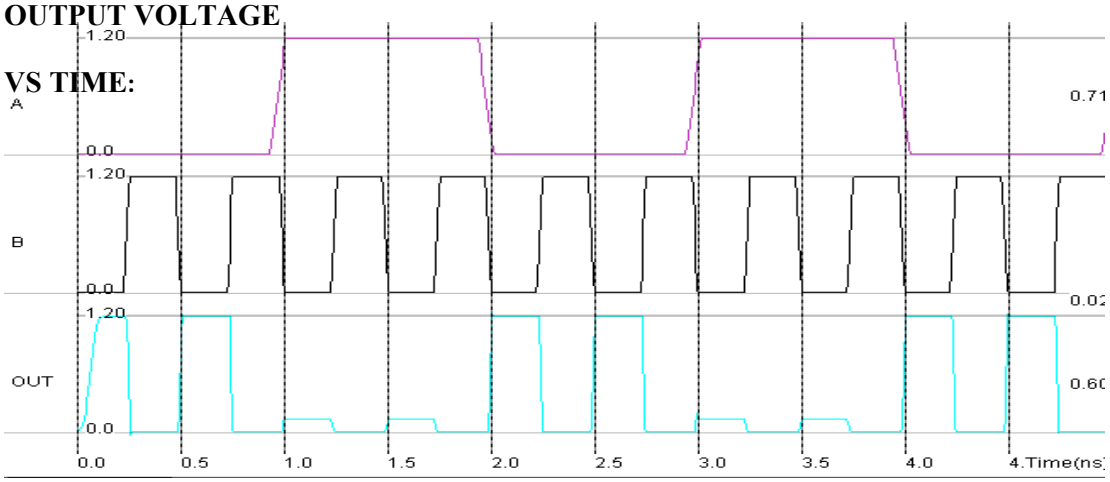
A	B	Y=(A+B)'
0	0	1
0	1	0
1	0	0
1	1	0

LAYOUT DESIGN



SIMULATION

OUTPUT VOLTAGE VS TIME:

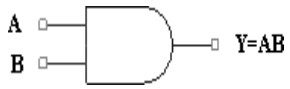


OBSERVATION TABLE:

NO. OF TRANSISTORS USED			POWER CONSUMPTION
DEVICE	PMOS TRANSISTOR	NMOS TRANSISTOR	
NOR GATE(2i/p)	2	2	1.058µw
Total numberof transistors	2	2	

AND GATE:

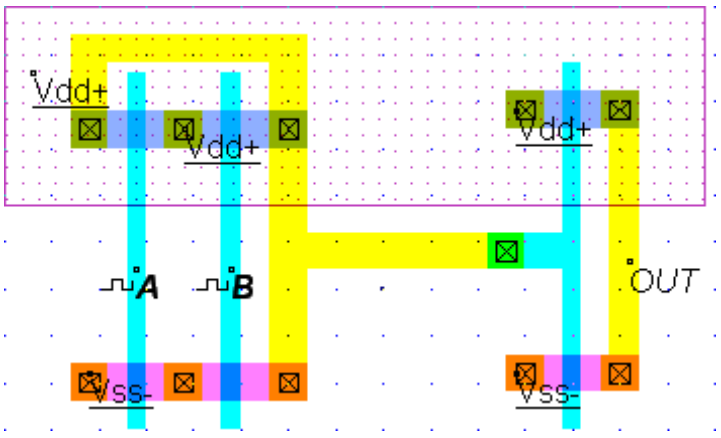
LOGICDIAGRM:



TRUTH TABLE:

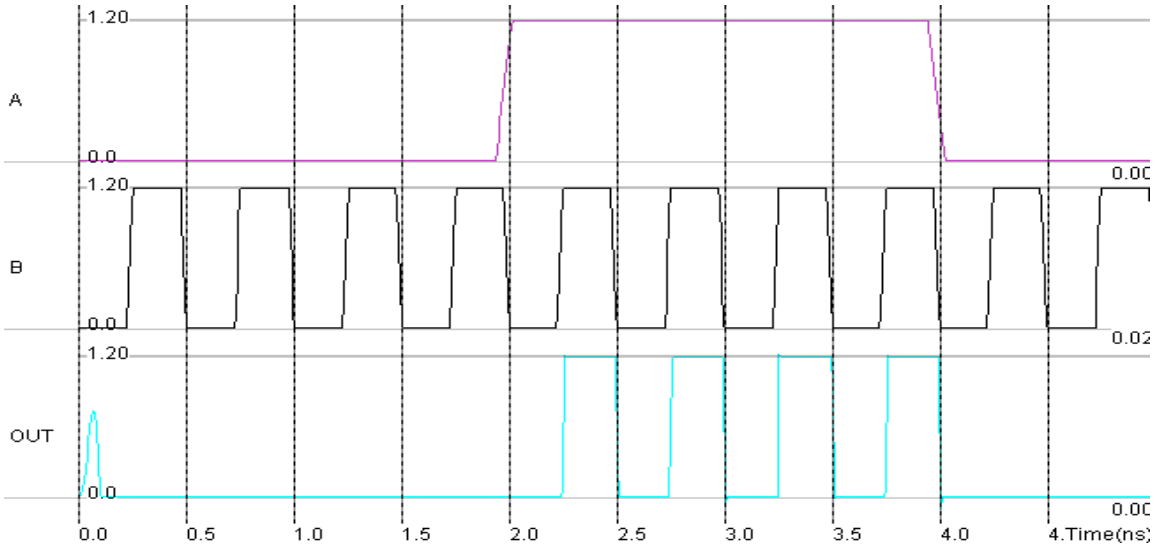
A	B	Y=AB
0	0	0
0	1	0
1	0	0
1	1	1

LAYOUT DESIGN:



SIMULATION OUTPUT:

VOLTAGE VS TIME:

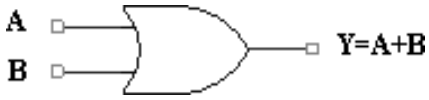


OBSERVATION TABLE:

NO. OF TRANSISTORS USED			POWER CONSUMPTION
DEVICE	PMOS TRANSISTOR	NMOS TRANSISTOR	3.504µw
Inverter	1	1	
Nand gate(2i/p)	2	2	
Total numberof transistors	3	3	

LOGIC DIAGRAM:

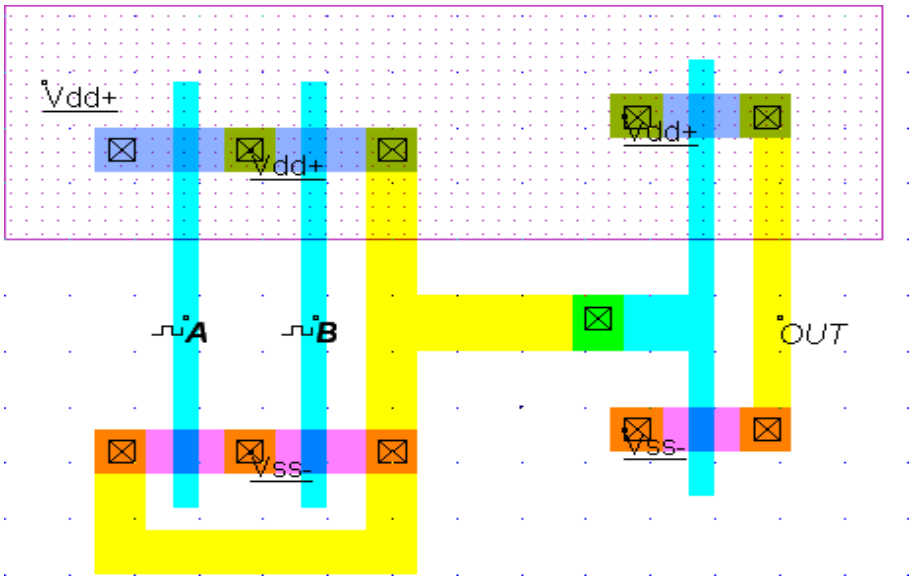
OR GATE:



TRUTH TABLE:

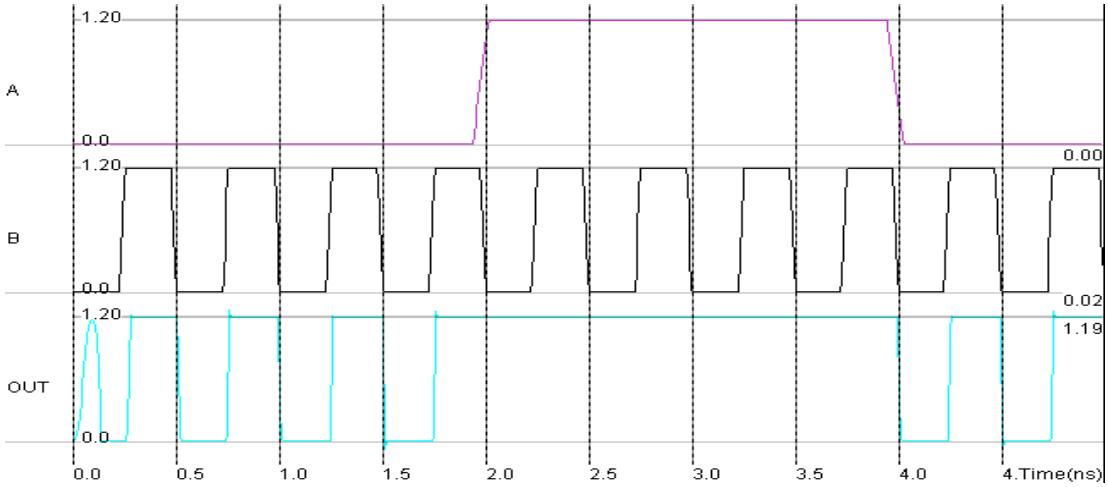
A	B	Y=A+B
0	0	0
0	1	1
1	0	1
1	1	1

LAYOUT DESIGN:



SIMULATION OUTPUT:

VOLTAGE VS TIME:

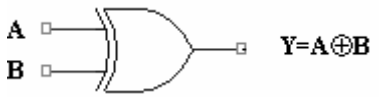


OBSERVATION TABLE:

NO. OF TRANSISTORS USED			POWER CONSUMPTION
DEVICE	PMOS TRANSISTOR	NMOS TRANSISTOR	2.943µw
NOR GATE (2i/p)	2	2	
INVERTER	1	1	

XOR GATE:

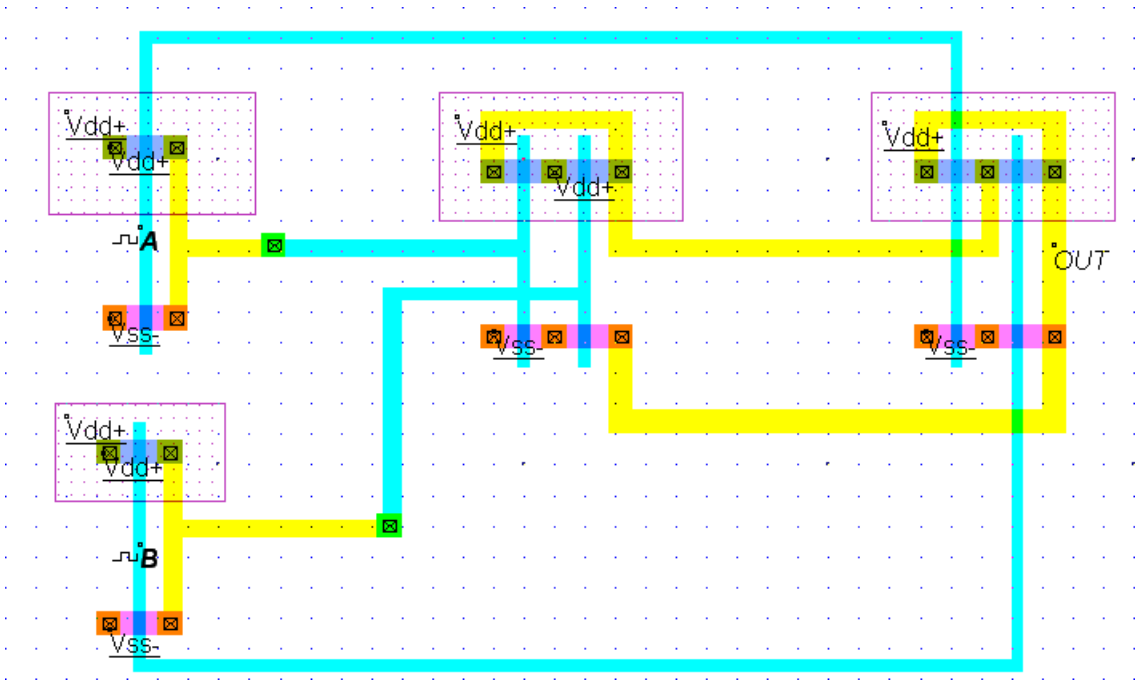
LOGIC DIAGRAM:



TRUTH TABLE:

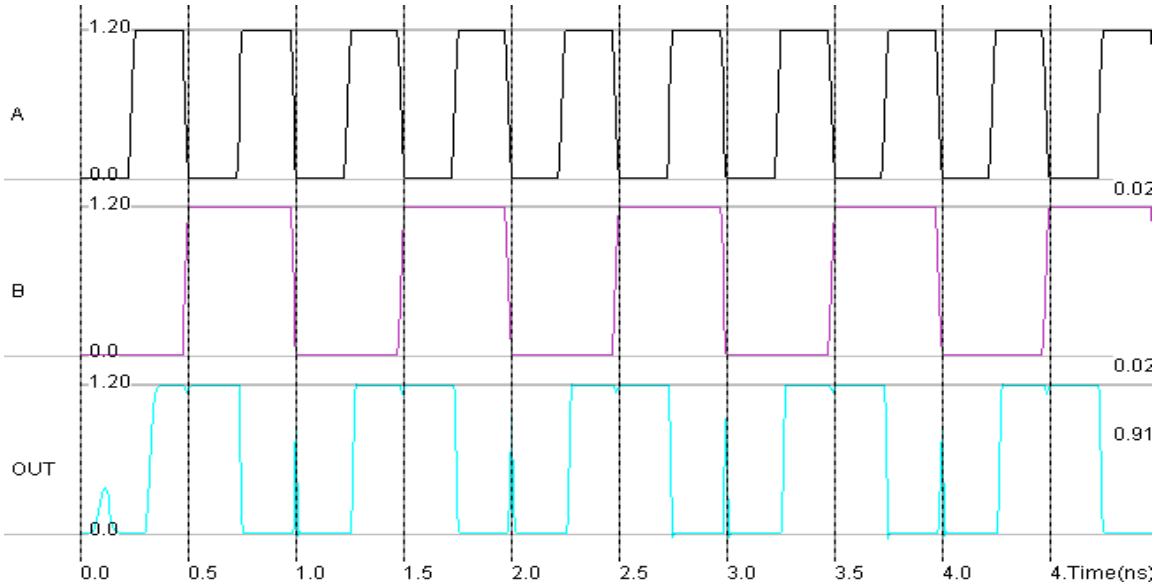
A	B	Y=A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

LAYOUT DESIGN:



SIMULATION OUTPUT:

VOLTAGE VS TIME:

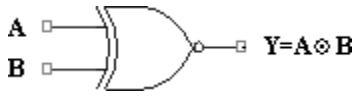


OBSERVATION TABLE:

NO. OF TRANSISTORS USED			POWER CONSUMPTION
DEVICE	PMOS TRANSISTOR	NMOS TRANSISTOR	
XORGATE	6	6	3.434µw
Total number of transistors	6	6	

XNOR GATE:

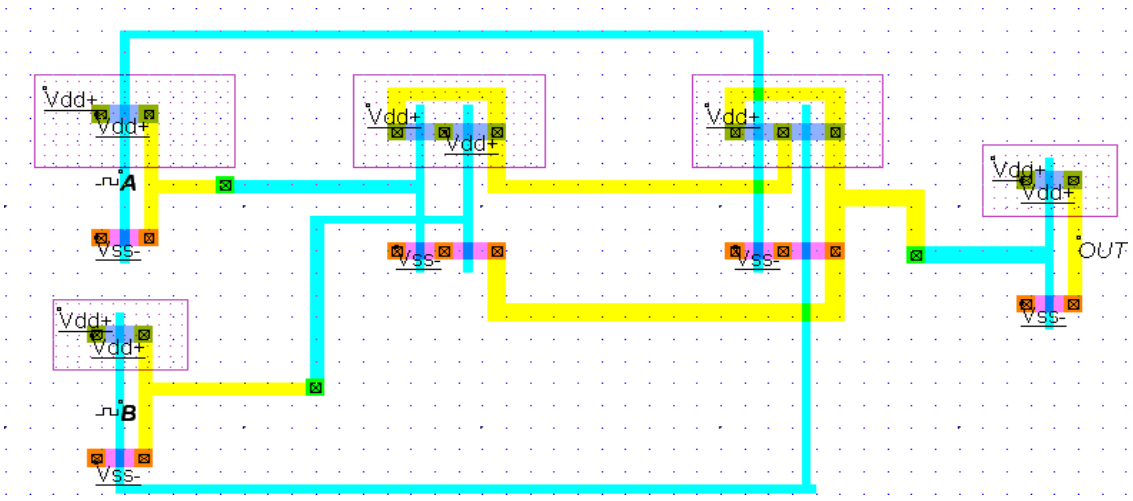
LOGIC DIAGRAM:



TRUTH TABLE:

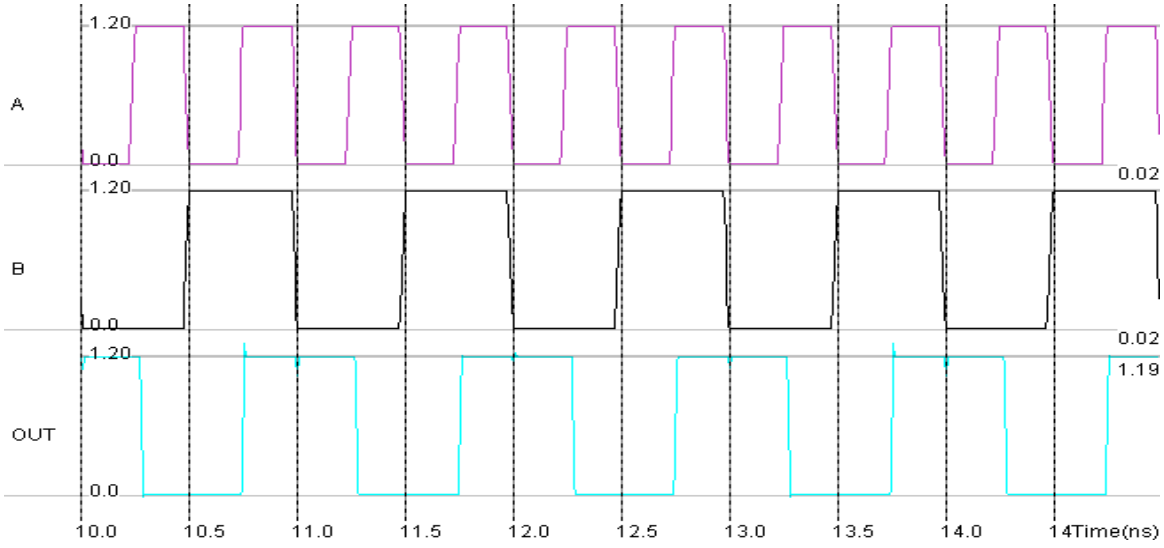
A	B	Y=A⊙B
0	0	1
0	1	0
1	0	0
1	1	1

LAYOUT DESIGN:



SIMULATION OUTPUT:

VOLTAGE VS TIME:



OBSERVATION TABLE:

NO. OF TRANSISTORS USED			POWER CONSUMPTION
DEVICE	PMOS TRANSISTOR	NMOS TRANSISTOR	5.212µw
XNOR GATE	7	7	
Total number of transistors	7	7	

PROCEDURE:

PMOS TRANSISTOR

1. The complete window can be considered asP-well.
2. The N-well is placed over the p-well in a rectangularbox.
3. Next we need to place the P-diffusion over theN-well.
4. Width of the P-diffusion should be4λ.
5. The metal contacts can be drawn by selecting a (4λ*4λ)square.

NMOS TRANSISTOR

1. There should be minimum 6λ space between N-well andN-diffusion.
2. The same steps can be followed for Nmos as inPmos.
3. The grid itself is P-well, we can start directly with n-diffusionlayer.
4. The Pmos and Nmos should be connected by using polysilicon layer of width2λ.
5. The polysilicon layer should be extended up to 3λ length above and below P and N diffusionlayers.
6. Leave minimum 1λ space on both sides of polysilicon between thecontacts.

RESULT

Ex. No:	VERIFICATION OF LOGIC GATES (STUDY EXPERIMENT)
Date:	

AIM:

To develop the source code for logic gates by using VHDL and obtain the simulation, synthesis, place and route and implement into FPGA.

ALGORITM:

Step1: Define the specifications and initialize the design.
Step2: Declare the name of the entity and architecture by using VHDL source code.
Step3: Write the source code in VERILOG.
Step4: Check the syntax and debug the errors if found, obtain the synthesis report.
Step5: Verify the output by simulating the source code.
Step6: Write all possible combinations of input using the test bench.
Step7: Obtain the place and route report.

VHDL SOUCE CODE

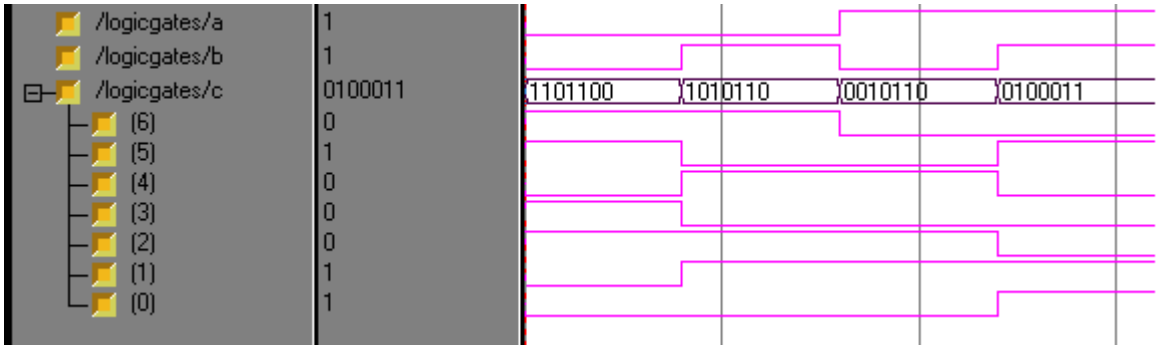
```
        --Design       : VERIFICATION OF LOGIC GATES
        --Description  : To implement LOGIC GATES
        --Author       :
        --Reg no       :
        --Version      :

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity logicgates is
Port ( a : in std_logic;
       b : in std_logic;
       c : out std_logic_vector(6 downto 0));
end logicgates;

architecture dataflow of logicgates is

begin
        c(0)<= a and b;
        c(1)<= a or b;
        c(2)<= a nand b;
        c(3)<= a nor b;
        c(4)<= a xor b;
        c(5)<= a xnor b;
        c(6)<= not a;
end dataflow;
```

TEST BENCH(VHDL):

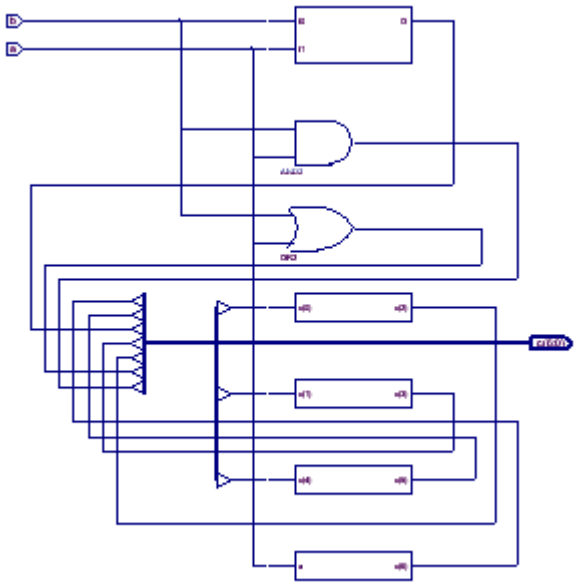
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
```

```
USE ieee.numeric_std.ALL;

ENTITY logicgatestest_vhd IS
END logicgatestest_vhd;


ARCHITECTURE testbench OF logicgatestest_vhd IS
	COMPONENT logicgates
	PORT(
		a : IN std_logic;
		b : IN std_logic;
		c : OUT std_logic_vector(6 downto 0)
		);
	END COMPONENT;

	SIGNAL a :  std_logic := '0';
	SIGNAL b :  std_logic := '0';

	SIGNAL c :  std_logic_vector(6 downto 0);

BEGIN

	uut: logicgates PORT MAP(
		a => a,
		b => b,
		c => c
	);

	tb : PROCESS
	BEGIN

	a<='0'; b<='0';		wait for 100 ps;
	a<='0'; b<='1';		wait for 100 ps;
	a<='1'; b<='0';		wait for 100 ps;
	a<='1'; b<='1';		wait for 100 ps;

	END PROCESS;

END testbench;
```

Simulation output:



Synthesis RTL Schematic:



Synthesis report:

```
=========================================================================
*                Final Report                *
=========================================================================
Final Results
RTL Top Level Output File Name     : logicgates.ngr
Top Level Output File Name         : logicgates
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : NO

Design Statistics
# IOs                              : 9
```

```
Cell Usage :
# BELS                          : 7
#      INV                      : 1
#      LUT2                     : 6
# IO Buffers                    : 9
#      IBUF                     : 2
#      OBUF                     : 7
=========================================================================

Device utilization summary:
---------------------------

Selected Device : 3s400tq144-5

 Number of Slices:                3  out of   3584     0%
 Number of 4 input LUTs:          6  out of   7168     0%
 Number of bonded IOBs:           9  out of     97     9%
=========================================================================

TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
No clock signals found in this design

Timing Summary:
---------------
Speed Grade: -5

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 7.985ns

Timing Detail:
--------------
All values displayed in nanoseconds (ns)

=========================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 13 / 7
-------------------------------------------------------------------------
Delay:               7.985ns (Levels of Logic = 3)
  Source:            a (PAD)
  Destination:       c<5> (PAD)

  Data Path: a to c<5>
                        Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
```

```
    IBUF:I->O             7   0.715   1.201  a_IBUF (a_IBUF)
    LUT2:I0->O            1   0.479   0.681  _n00021 (c_0_OBUF)
    OBUF:I->O             4.909         c_0_OBUF (c<0>)
    ----------------------------------------
    Total                 7.985ns (6.103ns logic, 1.882ns route)
                          (76.4% logic, 23.6% route)

========================================================================
CPU : 3.03 / 3.27 s | Elapsed : 3.00 / 4.00 s
```

Place and root report:

```
Constraints file: logicgates.pcf.
Loading device for application Rf_Device from file '3s400.nph' in environment
C:/Xilinx.
   "logicgates" is an NCD, version 3.1, device xc3s400, package tq144, speed -5
Initializing temperature to 85.000 Celsius. (default - Range: 0.000 to 85.000
Celsius)
Initializing voltage to 1.140 Volts. (default - Range: 1.140 to 1.260 Volts)
Device speed data version:  "ADVANCED 1.35 2005-01-22".
Device Utilization Summary:
 Number of External IOBs           9 out of 97     9%
 Number of LOCed IOBs              0 out of 9      0%
 Number of Slices                  3 out of 3584   1%
 Number of SLICEMs                 0 out of 1792   0%
Overall effort level (-ol):   Standard (set by user)
Placer effort level (-pl):    Standard (set by user)
Placer cost table entry (-t): 1
Router effort level (-rl):    Standard (set by user)
Starting Placer
Phase 1.1
Phase 1.1 (Checksum:989697) REAL time: 0 secs
Phase 2.31
Phase 2.31 (Checksum:1312cfe) REAL time: 0 secs
Phase 3.2
Phase 3.2 (Checksum:1c9c37d) REAL time: 0 secs
Phase 4.3
Phase 4.3 (Checksum:26259fc) REAL time: 0 secs
Phase 5.5
Phase 5.5 (Checksum:2faf07b) REAL time: 0 secs
Phase 6.8
Phase 6.8 (Checksum:98a327) REAL time: 0 secs
Phase 7.5
Phase 7.5 (Checksum:42c1d79) REAL time: 0 secs
Phase 8.18
Phase 8.18 (Checksum:4c4b3f8) REAL time: 0 secs
Phase 9.5
Phase 9.5 (Checksum:55d4a77) REAL time: 0 secs
Writing design to file logicgates.ncd
Total REAL time to Placer completion: 0 secs
Total CPU time to Placer completion: 0 secs
```

```
Starting Router
Phase 1: 19 unrouted;       REAL time: 0 secs
Phase 2: 19 unrouted;       REAL time: 0 secs
Phase 3: 1 unrouted;       REAL time: 0 secs
Phase 4: 0 unrouted;       REAL time: 0 secs
Total REAL time to Router completion: 0 secs
Total CPU time to Router completion: 0 secs
Generating "PAR" statistics.
INFO:Par:340 -
   The Delay report will not be generated when running non-timing driven PAR
   with effort level Standard or Medium. If a delay report is required please do
   one of the following:  1) use effort level High, 2) use the following
   environment variable "XIL_PAR_GENERATE_DLY_REPORT", 3) create Timing
   constraints for the design.
Generating Pad Report.
All signals are completely routed.
Total REAL time to PAR completion: 1 secs
Total CPU time to PAR completion: 1 secs
Peak Memory Usage:  76 MB
Placement: Completed - No errors found.
Routing: Completed - No errors found.
Number of error messages: 0
Number of warning messages: 0
Number of info messages: 1
Writing design to file logicgates.ncd
PAR done!
```

RESULT:

Ex. No:	ADDERS AND SUBTRACTORS
Date:	

AIM:

To develop the source code for adders and subtractors by using VHDL and obtain the simulation, synthesis, place and route and implement into FPGA.

ALGORITM:

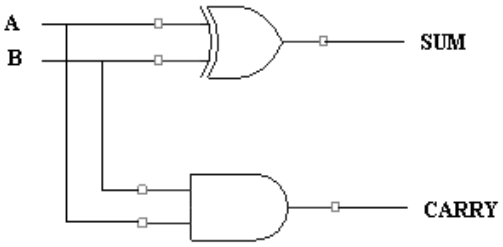
Step1: Define the specifications and initialize the design.
Step2: Declare the name of the entity and architecture by using VHDL source code.
Step3: Write the source code in VERILOG.
Step4: Check the syntax and debug the errors if found, obtain the synthesis report.
Step5: Verify the output by simulating the source code.
Step6: Write all possible combinations of input using the test bench.
Step7: Obtain the place and route report.

LOGIC DIAGRAM:

BASIC ADDERS & SUBTRACTORS:

HALF ADDER:

LOGIC DIAGRAM:



TRUTH TABLE:

A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

VHDL SOURCE CODE:

```
--Design       : HALF ADDER
--Description  : To implement HALF ADDER
--Author       :
--Reg no       :
--Version      :
```

Dataflow Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity hadd is
   Port ( a : in std_logic;
          b : in std_logic;
          sum : out std_logic;
```

```
        carry : out std_logic);
end hadd;
architecture dataflow of hadd is
begin
sum <= a xor b;
carry <= a and b;
end dataflow;
```

Behavioral Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity haddbehavioral is
   Port ( a : in std_logic;
        b : in std_logic;
        sum : out std_logic;
        carry : out std_logic);
end haddbehavioral;
architecture Behavioral of haddbehavioral is
begin
p1:process (a,b)
begin
sum<= a xor b;
carry<= a and b;
end process p1;
end Behavioral;
```

Structural Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity haddstructural is
   Port ( a : in std_logic;
        b : in std_logic;
        sum : out std_logic;
        carry : out std_logic);
end haddstructural;

architecture structural of haddstructural is
component xor2
port(x,y:in std_logic;
    z:out std_logic);
                    end component;
component and2
port(l,m:in std_logic;
      n:out std_logic);
                    end component;
begin
x1: xor2 port map (a,b,sum);
```

```
a1: and2 port map (a,b,carry);
end structural;
```

and2 component source code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity and2 is
   Port ( l : in std_logic;
          m : in std_logic;
          n : out std_logic);
end and2;
architecture dataflow of and2 is
begin
n<= l and  m;
end dataflow;
```

xor2 component source code:

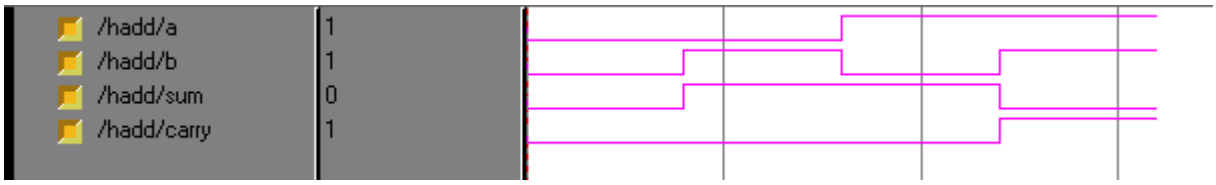
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity xor2 is
   Port ( x : in std_logic;
          y : in std_logic;
          z : out std_logic);
end xor2;
architecture dataflow of xor2 is
begin
z<= x xor y;
end dataflow;
```

TEST BENCH(VHDL):

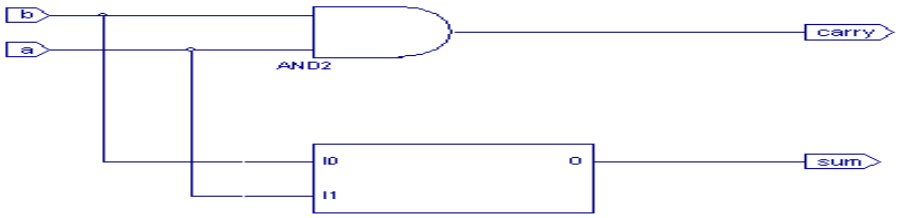
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;

ENTITY test_bench_vhd IS
END test_bench_vhd;

ARCHITECTURE behavior OF test_bench_vhd IS

        COMPONENT hadd
        PORT(
                a : IN std_logic;
                b : IN std_logic;
                sum : OUT std_logic;
                carry : OUT std_logic
```

```
        );
    END COMPONENT;

    --Inputs
    SIGNAL a :  std_logic := '0';
    SIGNAL b :  std_logic := '0';

    --Outputs
    SIGNAL sum :  std_logic;
    SIGNAL carry :  std_logic;

BEGIN

    -- Instantiate the Unit Under Test (UUT)
    uut: hadd PORT MAP(
        a => a,
        b => b,
        sum => sum,
        carry => carry
    );

    tb : PROCESS
    BEGIN

        a<='0'; b<='0'; wait for 100 ps;
        a<='0'; b<='1'; wait for 100 ps;
        a<='1'; b<='0'; wait for 100 ps;
        a<='1'; b<='1'; wait for 100 ps;

    END PROCESS;

END;
```

Simulation output:



Synthesis RTL Schematic:

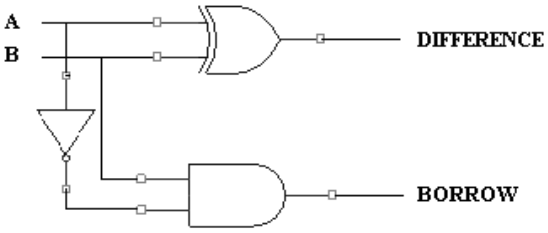


Synthesis report:

```
================================================================================
*                      Final Report                      *
================================================================================
Device utilization summary:
---------------------------
Selected Device : 3s400tq144-5

 Number of Slices:                        1  out of   3584     0%
 Number of 4 input LUTs:                  2  out of   7168     0%
 Number of bonded IOBs:                   4  out of     97     4%
================================================================================
TIMING REPORT
Clock Information:
------------------
No clock signals found in this design

Timing Summary:
---------------
Speed Grade: -5
   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 7.824ns
```

HALF SUBSTRACTOR:

LOGIC DIAGRAM:



TRUTH TABLE

A	B	DIFFERENCE	BORROW
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

VHDL SOURCE CODE:

Dataflow Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity hsub_dataflow is
   Port ( a : in std_logic;
          b : in std_logic;
          diff : out std_logic;
          borrow : out std_logic);
end hsub_dataflow;
architecture dataflow of hsub_dataflow is
signal s1:std_logic;
begin
s1<=not a;
diff <= a xor b;
```

```
borrow <= s1 and b;
end dataflow;
```

Behavioral Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity hsub_behv is
   Port ( a : in std_logic;
        b : in std_logic;
        diff : out std_logic;
        borrow : out std_logic);
end hsub_behv;
architecture Behavioral of hsub_behv is
begin
process(a,b)
variable  s1:std_logic;
begin
s1:= not a;
diff<=a xor b;
borrow<=s1and b;
end process;
end Behavioral;
```

Structural Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity hsub_structural is
   Port ( a : in std_logic;
        b : in std_logic;
        diff : out std_logic;
        borrow : out std_logic);
end hsub_structural;
architecture structural of hsub_structural is
component xor2
port(x,y:in std_logic;
    z:out std_logic);
end component;
component and2
port(l,m:in std_logic;
    n:out std_logic);
end component;
component not1
port(x:in std_logic;
    z:out std_logic);
end component;
signal s1:std_logic;
begin
x1:xor2 port map (a,b,diff);
```

```
a1:and2 port map (s1,b,borrow);
n1:not1 port map (a,s1);
end structural;
```

and2 component source code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity and2 is
   Port ( l : in std_logic;
          m: in std_logic;
          n: out std_logic);
end and2;
architecture dataflow of and2 is
begin
n<= l and m;
end dataflow;
```

xor2 component source code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity xor2 is
   Port ( a : in std_logic;
          b : in std_logic;
          z : out std_logic);
end xor2;
architecture dataflow of xor2 is
begin
z<= a xor b;
end dataflow;
```

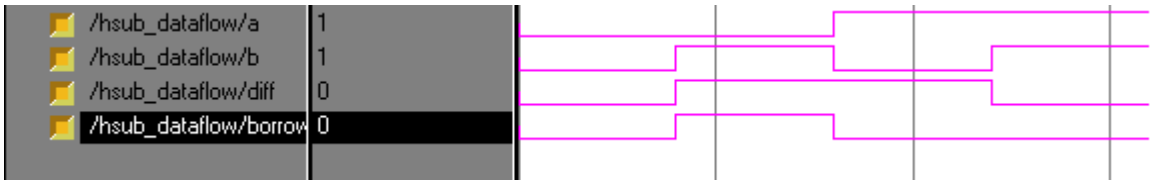
not1 component source code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity not1 is
   Port ( x : in std_logic;
          z : out std_logic);
end not1;
architecture dataflow of not1 is
begin
z<= not x;
end dataflow;
```

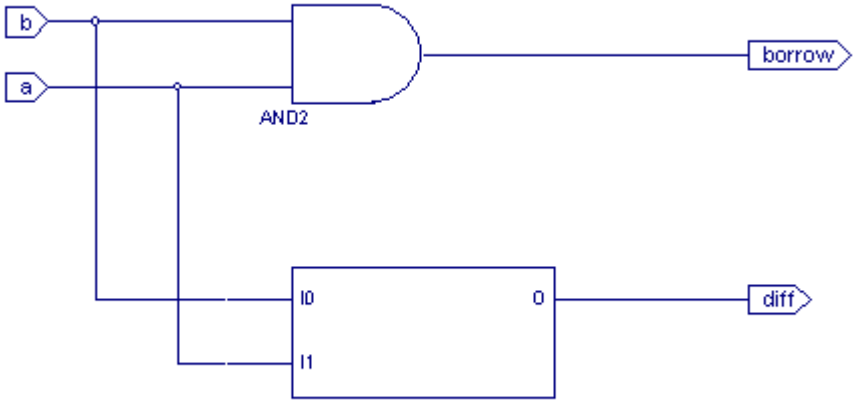
TEST BENCH(VHDL):

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;

ENTITY test_vhd IS
END test_vhd;

ARCHITECTURE behavior OF test_vhd IS

	-- Component Declaration for the Unit Under Test (UUT)
	COMPONENT hsub_dataflow
	PORT(
		a : IN std_logic;
		b : IN std_logic;
		diff : OUT std_logic;
		borrow : OUT std_logic
		);
	END COMPONENT;

	--Inputs
	SIGNAL a :  std_logic := '0';
	SIGNAL b :  std_logic := '0';

	--Outputs
	SIGNAL diff :  std_logic;
	SIGNAL borrow :  std_logic;

BEGIN

	-- Instantiate the Unit Under Test (UUT)
	uut: hsub_dataflow PORT MAP(
		a => a,
		b => b,
		diff => diff,
		borrow => borrow
	);

	tb : PROCESS
	BEGIN

		a<='0'; b<='0'; wait for 100 ps;
		  a<='0'; b<='1'; wait for 100 ps;
		a<='1'; b<='0'; wait for 100 ps;
		  a<='1'; b<='1'; wait for 100 ps;

	END PROCESS;

END;
```

Simulation output:



Synthesis RTL Schematic:



Synthesis report:

```
================================================================================
*                      Final Report                      *
================================================================================
Device utilization summary:
---------------------------

Selected Device : 3s400tq144-5

 Number of Slices:                  1  out of   3584    0%
 Number of 4 input LUTs:            2  out of   7168    0%
 Number of bonded IOBs:             4  out of     97    4%


================================================================================
TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
```

No clock signals found in this design

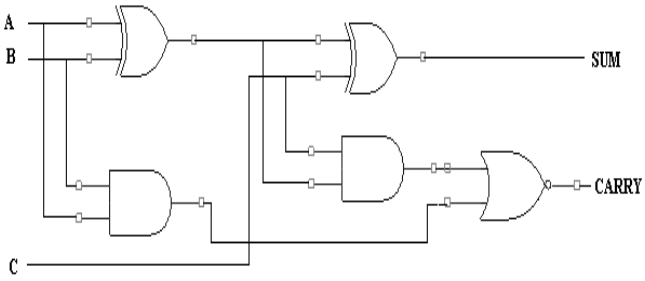
Timing Summary:

Speed Grade: -5

Minimum period: No path found
Minimum input arrival time before clock: No path found
Maximum output required time after clock: No path found
Maximum combinational path delay: 7.824ns

FULL ADDER:

LOGIC DIAGRAM:



TRUTH TABLE:

A	B	C	SUM	CARRY
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

VHDL SOURCE CODE:
Dataflow Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fadd_dataflow is
   Port ( a : in std_logic;
          b : in std_logic;
          c : in std_logic;
          sum : out std_logic;
          carry : out std_logic);
end fadd_dataflow;
architecture dataflow of fadd_dataflow is
signal s1,s2,s3,s4:std_logic;
begin
s1<= a xor b;
s2<= a and b;
s3<= b and c;
s4<= c and a;
```

```
sum<= s1 xor c;
carry<= s2 or s3 or s4;
end dataflow;
```

Behavioral Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fadd_behv is
   Port ( a : in std_logic;
        b : in std_logic;
        c : in std_logic;
        sum : out std_logic;
        carry : out std_logic);
end fadd_behv;
architecture Behavioral of fadd_behv is
begin
p1:process(a,b,c)
variable s1,s2,s3,s4:std_logic;
begin
s1:= a xor b;
s2:= a and b;
s3:= b and c;
s4:= c and a;
sum<= s1 xor c;
carry<= s2 or s3 or s4;
end process p1;
end Behavioral;
```

Structural Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fadd_structural is
   Port ( a : in std_logic;
        b : in std_logic;
        c : in std_logic;
        sum : out std_logic;
        carry : out std_logic);
end fadd_structural;
architecture structural of fadd_structural is
component xor2
port(x,y:in std_logic;
    z:out std_logic);
            end component;
component and2
port(l,m:in std_logic;
      n:out std_logic);
           end component;
component or3
```

```
port(d,e,f:in std_logic;
     g:out std_logic);
                    end component;
signal s1,s2,s3,s4:std_logic;
begin
x1: xor2 port map (a,b,s1);
x2: xor2 port map (s1,c,sum);
a1: and2 port map (a,b,s2);
a2: and2 port map (b,c,s3);
a3: and2 port map (c,a,s4);
o1: or3 port map (s2,s3,s4,carry);
end structural;
```

and2 component source code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity and2 is
   Port ( l : in std_logic;
        m : in std_logic;
        n : out std_logic);
end and2;
architecture dataflow of and2 is
begin
n<=l and m;
end dataflow;
```

or3 component source code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity or3 is
   Port ( d : in std_logic;
        e : in std_logic;
        f : in std_logic;
        g: out std_logic);
end or3;
architecture dataflow of or3 is
begin
g<= d or e or f;
end dataflow;
```

xor2 component source code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity xor2 is
   Port ( x : in std_logic;
```

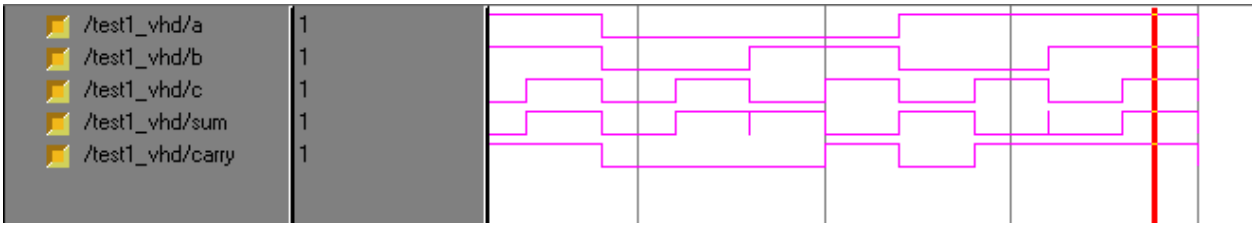
```
        yb : in std_logic;
        z : out std_logic);
end xor2;
architecture dataflow of xor2 is
begin
z<= x xor y;
end dataflow;
```

TEST BENCH(VHDL):

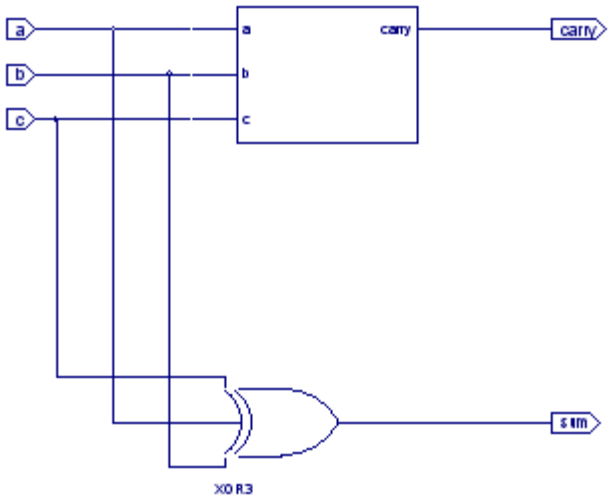
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;

ENTITY test1_vhd IS
END test1_vhd;

ARCHITECTURE behavior OF test1_vhd IS

        -- Component Declaration for the Unit Under Test (UUT)
        COMPONENT fadd_dataflow
        PORT(
                a : IN std_logic;
                b : IN std_logic;
                c : IN std_logic;
                sum : OUT std_logic;
                carry : OUT std_logic
                );
        END COMPONENT;

        --Inputs
        SIGNAL a :  std_logic := '0';
        SIGNAL b :  std_logic := '0';
        SIGNAL c :  std_logic := '0';

        --Outputs
        SIGNAL sum :  std_logic;
        SIGNAL carry :  std_logic;

BEGIN
        -- Instantiate the Unit Under Test (UUT)
        uut: fadd_dataflow PORT MAP(
                a => a,
                b => b,
                c => c,
                sum => sum,
                carry => carry
        );

        tb : PROCESS
        BEGIN

  a<='0'; b<='0'; c<='0';       wait for 100 ps;
```

```
  a<='0'; b<='0'; c<='1';   wait for 100 ps;
  a<='0'; b<='1'; c<='0';       wait for 100 ps;
  a<='0'; b<='1'; c<='1';       wait for 100 ps;
  a<='1'; b<='0'; c<='0';       wait for 100 ps;
  a<='1'; b<='0'; c<='1';       wait for 100 ps;
  a<='1'; b<='1'; c<='0';       wait for 100 ps;
  a<='1'; b<='1'; c<='1';       wait for 100 ps;
 END PROCESS;
END;
```

Simulation output:



Synthesis RTL Schematic:



Synthesis report:

```
================================================================================
*                      Final Report                      *
================================================================================
Device utilization summary:
---------------------------

Selected Device : 3s400tq144-5

 Number of Slices:                  1  out of   3584    0%
 Number of 4 input LUTs:            2  out of   7168    0%
 Number of bonded IOBs:             5  out of     97    5%


================================================================================
TIMING REPORT
```

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

Clock Information:

No clock signals found in this design

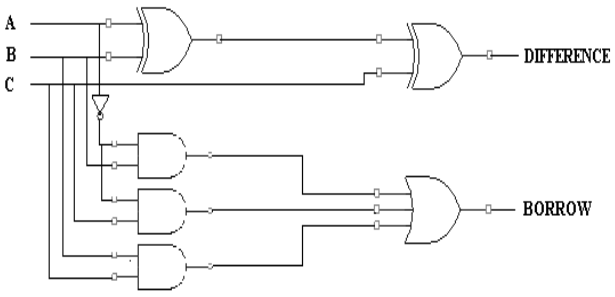
Timing Summary:

Speed Grade: -5

Minimum period: No path found
Minimum input arrival time before clock: No path found
Maximum output required time after clock: No path found
Maximum combinational path delay: 7.824ns

FULL SUBSTRACTOR:

LOGIC DIAGRAM:



TRUTH TABLE:

A	B	C	DIFFERENCE	BORROW
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

VHDL SOURCE CODE:

Dataflow Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fsub_dataflow is
   Port ( a : in std_logic;
        b : in std_logic;
        c : in std_logic;
        diff : out std_logic;
        borrow : out std_logic);
end fsub_dataflow;
architecture dataflow of fsub_dataflow is
signal s1,s2,s3,s4,s5:std_logic;
begin
s1r<= a xor b;
diff<=s1 xor c;
```

```
s2<=not a;
s3<=s2 and b;
s4<=b and c;
s5<=s2 and c;
borrow<=s3 or s4 or s5;
end dataflow;
```

Behavioral Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fsub_behv is
   Port ( a : in std_logic;
          b : in std_logic;
          c : in std_logic;
          diff : out std_logic;
          borrow : out std_logic);
end fsub_behv;
architecture Behavioral of fsub_behv is
begin
process(a,b,c)
variable s1,s2,s3,s4,s5:std_logic;
begin
s1:= a xor b;
s2:=not a;
s3:=s2 and b;
s4:=b and c;
diff<=s1  xor c;
borrow<=s3 or s4 or s5;
end process;
end Behavioral;
```

Structural Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fsub_structural is
   Port ( a : in std_logic;
          b : in std_logic;
          c : in std_logic;
          diff : out std_logic;
          borrow : out std_logic);
end fsub_structural;
architecture structural of fsub_structural is
component xor2
port(x,y:in std_logic;
    z:out std_logic);
end component;
```

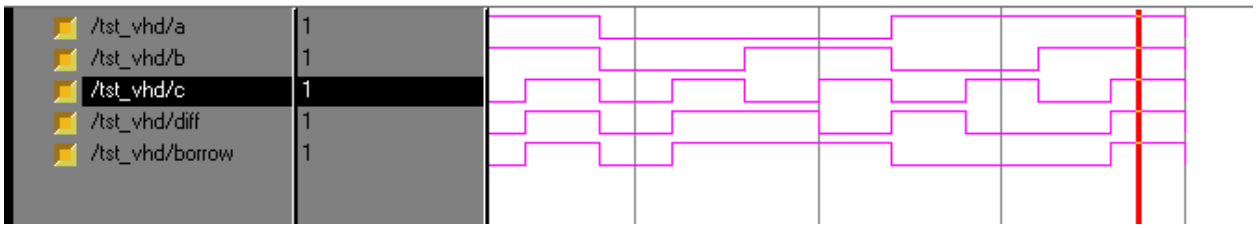
```
component and2
port(l,m:in std_logic;
     n:out std_logic);
end component;
component not1
port(x:in std_logic;zz
    z:out std_logic);
end component;
component or3
port(d,e,f:in std_logic;
    g:out std_logic);
end component;
signal s1,s2,s3,s4,s5:std_logic;
begin
x1:xor2 port map (a,b,s1);
x2:xor2 port map (s1,c,diff);
n1:not1 port map (a,s2);
a1:and2 port map (s2,b,s3);
a2:and2 port map (b,c,s4);
a3:and2 port map (c,s2,s5);
o1:or3 port map (s3,s4,s5,borrow);
end structural;
```

TEST BENCH(VHDL):

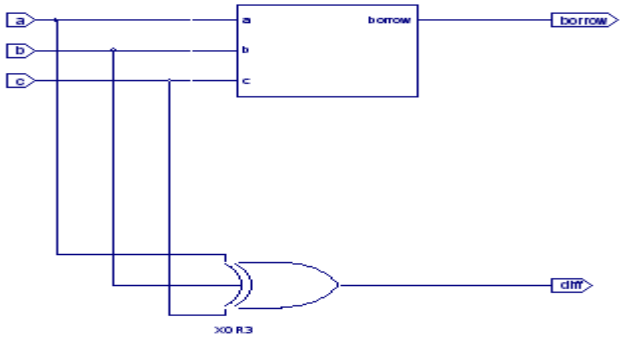
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;

ENTITY tst_vhd IS
END tst_vhd;

ARCHITECTURE behavior OF tst_vhd IS

	-- Component Declaration for the Unit Under Test (UUT)
	COMPONENT fsub_dataflow
	PORT(
		a : IN std_logic;
		b : IN std_logic;
		c : IN std_logic;
		diff : OUT std_logic;
		borrow : OUT std_logic
		);
	END COMPONENT;

	--Inputs2
	SIGNAL a :  std_logic := '0';
	SIGNAL b :  std_logic := '0';
	SIGNAL c :  std_logic := '0';

	--Outputs
	SIGNAL diff :  std_logic;
	SIGNAL borrow :  std_logic;
```

```
BEGIN

        -- Instantiate the Unit Under Test (UUT)
        uut: fsub_dataflow PORT MAP(
                a => a,
                b => b,
                c => c,
                diff => diff,
                borrow => borrow
        );

        tb : PROCESS
        BEGIN
  a<='0'; b<='0'; c<='0';       wait for 100 ps;
  a<='0'; b<='0'; c<='1';      wait for 100 ps;
  a<='0'; b<='1'; c<='0';       wait for 100 ps;
  a<='0'; b<='1'; c<='1';       wait for 100 ps;
  a<='1'; b<='0'; c<='0';       wait for 100 ps;
  a<='1'; b<='0'; c<='1';       wait for 100 ps;
  a<='1'; b<='1'; c<='0';       wait for 100 ps;
  a<='1'; b<='1'; c<='1';       wait for 100 ps;
  END PROCESS;
END;
```

Simulation output:



Synthesis RTL Schematic:

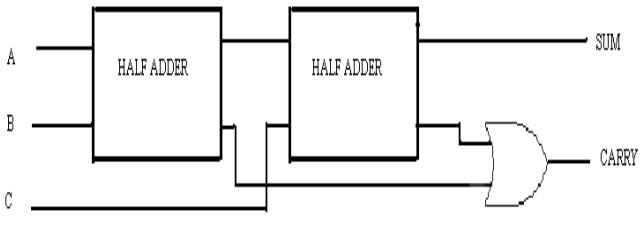


Synthesis report:

```
=========================================================================
*                    Final Report                     *
=========================================================================
Device utilization summary:
---------------------------

Selected Device : 3s400tq144-5

 Number of Slices:                  1  out of   3584     0%
 Number of 4 input LUTs:            2  out of   7168     0%
 Number of bonded IOBs:             5  out of     97     5%


=========================================================================
TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
No clock signals found in this design

Timing Summary:
---------------
Speed Grade: -5

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 7.824ns
```

FULL ADDER USING TWO HALF ADDERS:

LOGIC DIAGRAM:



TRUTH TABLE:

A	B	C	SUM	CARRY
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

VHDL SOURCE CODE:

Structural Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fadd2 is
   Port ( a : in std_logic;
          b : in std_logic;
          c : in std_logic;
          sum : out std_logic;
          carry : out std_logic);
end fadd2;
architecture structural of fadd2 is
component hadd
port(a,b:in std_logic;
sum,carry:out std_logic);
end component;
component or2
port(a,b:in std_logic;
    z:out std_logic);
end component;
signal p,q,r:std_logic;
begin
h1:hadd port map (a,b,p,q);
h2:hadd port map (p,c,sum,r);
o1:or2 port map (r,q,carry);
end structural;
```

hadd component source code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity hadd is
   Port ( a : in std_logic;
          b : in std_logic;
          sum : out std_logic;
          carry : out std_logic);
end hadd;
architecture dataflow of hadd is
begin
sum <= a xor b;
carry <= a and b;
end dataflow;
```

or2 component source code:

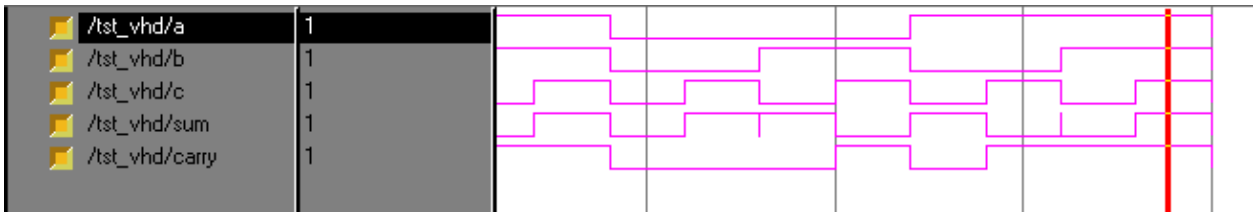
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity or2 is
   Port ( a : in std_logic;
        b : in std_logic;
        z : out std_logic);
end or2;
architecture dataflow of or2 is
begin
z<= a or b;
end dataflow;
```

TEST BENCH(VHDL):

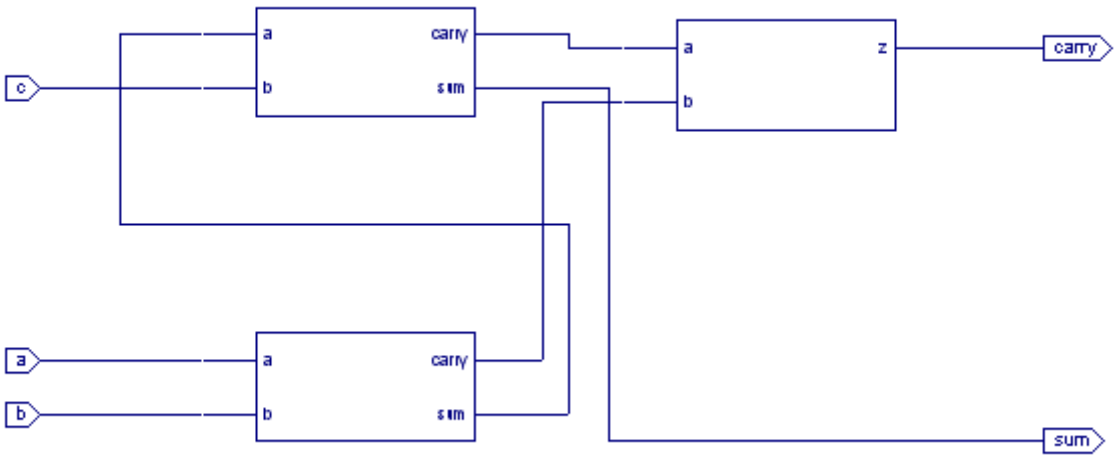
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;

ENTITY tst_vhd IS
END tst_vhd;

ARCHITECTURE behavior OF tst_vhd IS

        -- Component Declaration for the Unit Under Test (UUT)
        COMPONENT fadd2
        PORT(
                a : IN std_logic;
                b : IN std_logic;
                c : IN std_logic;
                sum : OUT std_logic;
                carry : OUT std_logic
                );
        END COMPONENT;

        --Inputs
        SIGNAL a :  std_logic := '0';
        SIGNAL b :  std_logic := '0';
        SIGNAL c :  std_logic := '0';

        --Outputs
        SIGNAL sum :  std_logic;
        SIGNAL carry :  std_logic;

BEGIN

        -- Instantiate the Unit Under Test (UUT)
        uut: fadd2 PORT MAP(
                a => a,
                b => b,
                c => c,
```

```
                sum => sum,
                carry => carry
        );

        tb : PROCESS
        BEGIN
 a<='0'; b<='0'; c<='0';       wait for 100 ps;
 a<='0'; b<='0'; c<='1';      wait for 100 ps;
 a<='0'; b<='1'; c<='0';       wait for 100 ps;
 a<='0'; b<='1'; c<='1';       wait for 100 ps;
 a<='1'; b<='0'; c<='0';       wait for 100 ps;
 a<='1'; b<='0'; c<='1';       wait for 100 ps;
 a<='1'; b<='1'; c<='0';       wait for 100 ps;
 a<='1'; b<='1'; c<='1';       wait for 100 ps;
END PROCESS;
END;
```

Simulation output:



Synthesis RTL Schematic:

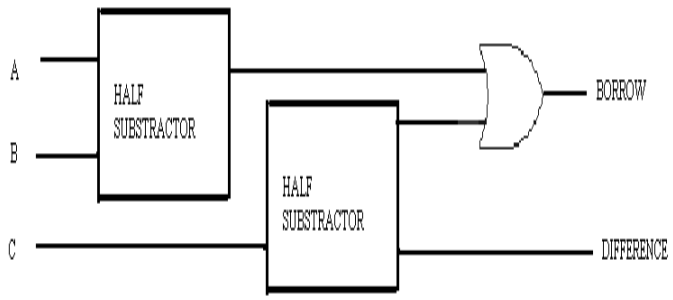


Synthesis report:

```
=========================================================================
*                      Final Report                     *
=========================================================================
Device utilization summary:
---------------------------

Selected Device : 3s400tq144-5

 Number of Slices:                  1  out of   3584     0%
 Number of 4 input LUTs:            2  out of   7168     0%
 Number of bonded IOBs:             5  out of     97     5%


=========================================================================
TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
No clock signals found in this design

Timing Summary:
---------------
Speed Grade: -5

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 7.824ns
```

FULL SUBTRACTOR USING TWO HALF SUBTRACTORS:

LOGIC DIAGRAM:



TRUTH TABLE:

A	B	C	DIFFERENCE	BORROW
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

VHDL SOURCE CODE:

Structural Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fsub2 is
   Port ( a : in std_logic;
        b : in std_logic;
        c : in std_logic;
        diff : out std_logic;
        borrow : out std_logic);
end fsub2;
architecture structural of fsub2 is
component hsub_dataflow
port(a,b:in std_logic;
diff,borrow:out std_logic);
end component;
component or2
port(a,b:in std_logic;
    z:out std_logic);
end component;
signal p,q,r:std_logic;
begin
h1:hsub_dataflow port map (a,b,p,q);
h2:hsub_dataflow port map (p,c,diff,r);
o1:or2 port map (r,q,borrow);
end structural;
```

hsub component source code:

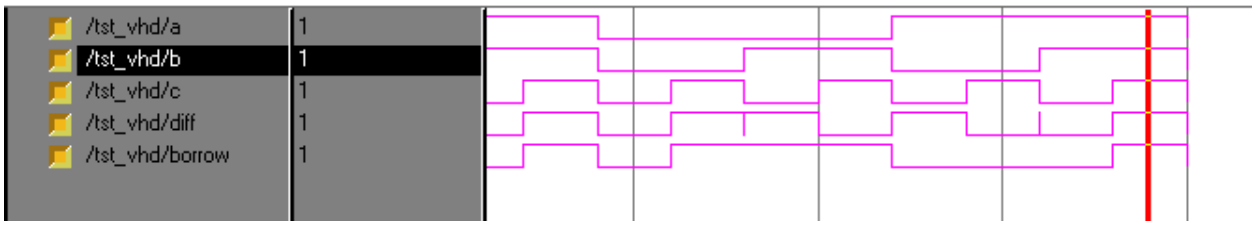
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity hsub_dataflow is
   Port ( a : in std_logic;
        b : in std_logic;
        diff : out std_logic;
        borrow : out std_logic);
end hsub_dataflow;
architecture dataflow of hsub_dataflow is
begin
diff <= a xor b;
borrow <= not a and b;
end dataflow;
```

TEST BENCH(VHDL):

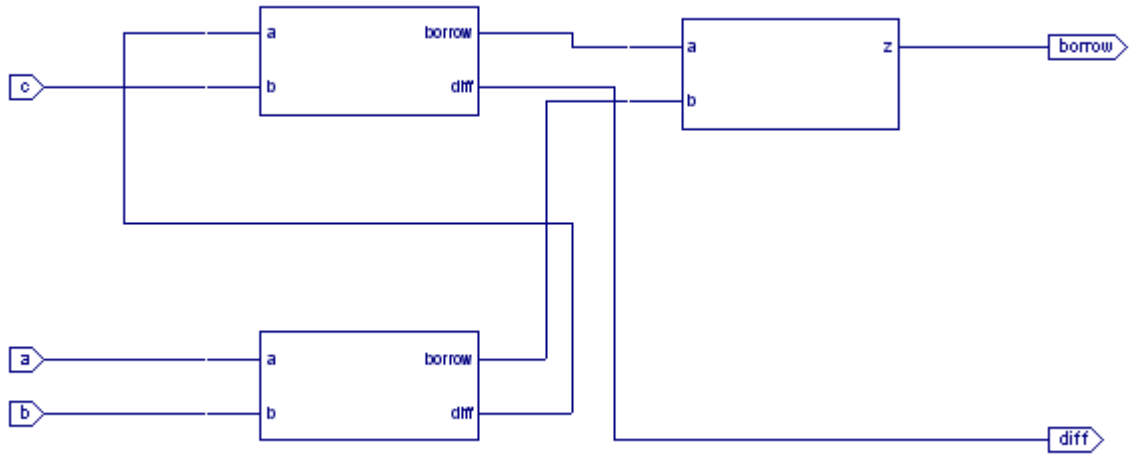
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;
```

```
ENTITY tst_vhd IS
END tst_vhd;

ARCHITECTURE behavior OF tst_vhd IS

	-- Component Declaration for the Unit Under Test (UUT)
	COMPONENT fsub2
	PORT(
		a : IN std_logic;
		b : IN std_logic;
		c : IN std_logic;
		diff : OUT std_logic;
		borrow : OUT std_logic
		);
	END COMPONENT;

	--Inputs
	SIGNAL a :  std_logic := '0';
	SIGNAL b :  std_logic := '0';
	SIGNAL c :  std_logic := '0';

	--Outputs
	SIGNAL diff :  std_logic;
	SIGNAL borrow :  std_logic;

BEGIN

	-- Instantiate the Unit Under Test (UUT)
	uut: fsub2 PORT MAP(
		a => a,
		b => b,
		c => c,
		diff => diff,
		borrow => borrow
	);

	tb : PROCESS
	BEGIN
	a<='0'; b<='0'; c<='0';		wait for 100 ps;
  a<='0'; b<='0'; c<='1';   wait for 100 ps;
  a<='0'; b<='1'; c<='0';	wait for 100 ps;
  a<='0'; b<='1'; c<='1';	wait for 100 ps;
  a<='1'; b<='0'; c<='0';	wait for 100 ps;
  a<='1'; b<='0'; c<='1';	wait for 100 ps;
  a<='1'; b<='1'; c<='0';	wait for 100 ps;
  a<='1'; b<='1'; c<='1';	wait for 100 ps;
	END PROCESS;
END;
```

Simulation output:



Synthesis RTL Schematic:



Synthesis report:

```
================================================================================
*                      Final Report                      *
================================================================================
Device utilization summary:
---------------------------

Selected Device : 3s400tq144-5

 Number of Slices:                  1  out of   3584     0%
 Number of 4 input LUTs:            2  out of   7168     0%
 Number of bonded IOBs:             5  out of     97     5%

================================================================================
TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
```

No clock signals found in this design

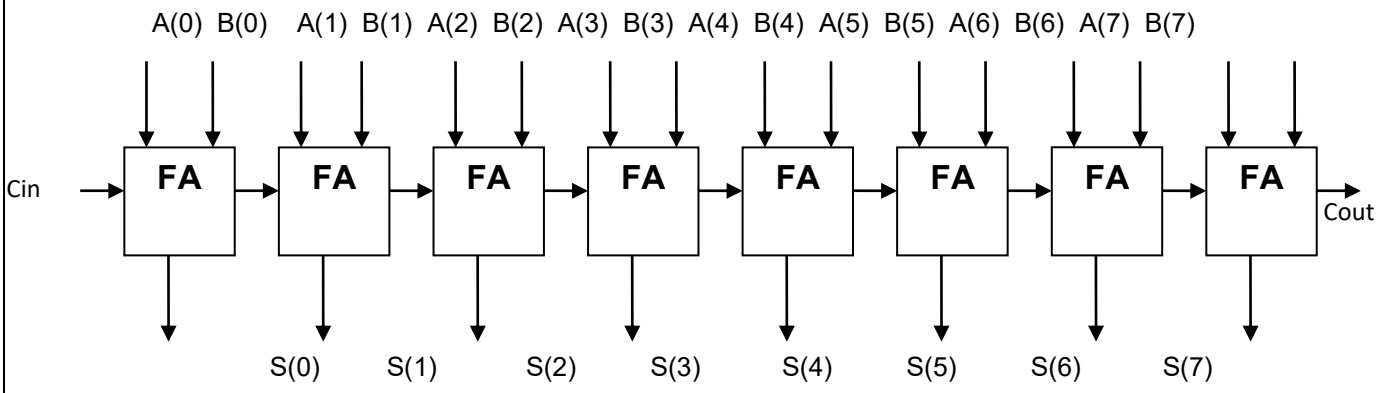
Timing Summary:

Speed Grade: -5

Minimum period: No path found
Minimum input arrival time before clock: No path found
Maximum output required time after clock: No path found
Maximum combinational path delay: 7.824ns

RIPPLE CARRY ADDER(Binary adder):

LOGIC DIAGRAM:



VHDL SOURCE CODE:

```
--Design      : RIPPLE CARRY ADDER
--Description : To implement RIPPLE CARRY ADDER
--Author      :
--Reg no      :
--Version     :
```

Structural Modeling:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity rca is
   Port ( a : in std_logic_vector(3 downto 0);
        b : in std_logic_vector(3 downto 0);
        c : in std_logic;
        s : out std_logic_vector(3 downto 0);
        cout : out std_logic);
end rca;
```

```
architecture structural of rca is
component fadd_behv
port(a,b,c:in std_logic;
sum,carry:out std_logic);
end component;
signal c0,c1,c2:std_logic;
begin
f1:fadd_behv port map (a(0),b(0),c,s(0),c0);
f2:fadd_behv port map (a(1),b(1),c0,s(1),c1);
f3:fadd_behv port map (a(2),b(2),c1,s(2),c2);
f4:fadd_behv port map (a(3),b(3),c2,s(3),cout);
end structural;
```

FULL ADDER SOURCE CODE:

Dataflow Modeling:

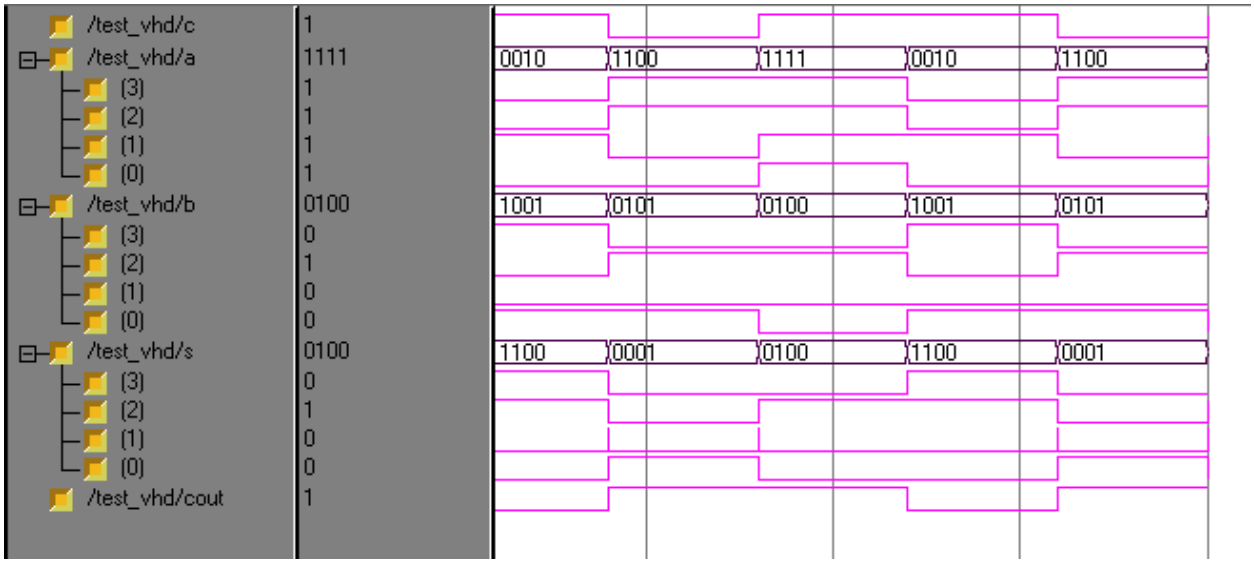
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fadd_dataflow is
   Port ( a : in std_logic;
          b : in std_logic;
          c : in std_logic;
          sum : out std_logic;
          carry : out std_logic);
end fadd_dataflow;
architecture dataflow of fadd_dataflow is
signal s1,s2,s3,s4:std_logic;
begin
s1<= a xor b;
s2<= a and b;
s3<= b and c;
s4<= c and a;
sum<= s1 xor c;
carry<= s2 or s3 or s4;
end dataflow;
```

TEST BENCH(VHDL):

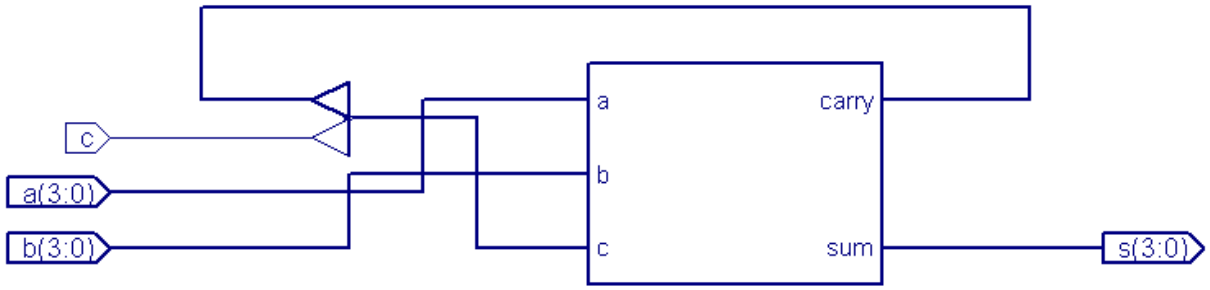
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;
ENTITY test_vhd IS
END test_vhd;
ARCHITECTURE behavior OF test_vhd IS
        -- Component Declaration for the Unit Under Test (UUT)
        COMPONENT rca
        PORT(
                a : IN std_logic_vector(3 downto 0);
                b : IN std_logic_vector(3 downto 0);
                c : IN std_logic;
```

```
                s : OUT std_logic_vector(3 downto 0);
                cout : OUT std_logic
                );
        END COMPONENT;
        --Inputs
        SIGNAL c :  std_logic := '0';
        SIGNAL a :  std_logic_vector(3 downto 0) := (others=>'0');
        SIGNAL b :  std_logic_vector(3 downto 0) := (others=>'0');
        --Outputs
        SIGNAL s :  std_logic_vector(3 downto 0);
        SIGNAL cout :  std_logic;
BEGIN
        -- Instantiate the Unit Under Test (UUT)
        uut: rca PORT MAP(
                a => a,
                b => b,
                c => c,
                s => s,
                cout => cout
        );
        tb : PROCESS
        BEGIN
        a<="0010";b<="1001";c<='1'; wait for 200 ps;
        a<="1100";b<="0101";c<='0'; wait for 200 ps;
        a<="1111";b<="0100";c<='1'; wait for 200 ps;
        END PROCESS;
END;
```

Simulation output:



Synthesis RTL Schematic:



Synthesis report:

```
=========================================================================
*                  Final Report                     *
=========================================================================

Device utilization summary:
---------------------------
Selected Device :                3s400tq144-5
 Number of Slices:               5  out of   3584    0%
 Number of 4 input LUTs:         9  out of   7168    0%
 Number of bonded IOBs:          14  out of    97    14%

=========================================================================

TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
No clock signals found in this design
Timing Summary:
---------------

Speed Grade: -5

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 11.747ns
```

RESULT:

VIVA QUESTIONS:

1. Why does the present VLSI circuits use MOSFETs instead of BJTs?

Compared to BJTs, MOSFETs can be made very small as they occupy very small silicon area on IC chip and are relatively simple in terms of manufacturing. Moreover digital and memory ICs can be implemented with circuits that use only MOSFETs i.e. no resistors, diodes, etc.

2. What are the various regions of operation of MOSFET? How are those regions used?

MOSFET has three regions of operation: the cut-off region, the triode region, and the saturation region. The cut-off region and the triode region are used to operate as switch. The saturation region is used to operate as amplifier.

3. What is threshold voltage?

The value of voltage between Gate and Source i.e. VGS at which a sufficient number of mobile electrons accumulate in the channel region to form a conducting channel is called threshold voltage (Vt is positive for NMOS and negative for PMOS).

4. What does it mean "the channel is pinched off"?

For a MOSFET when VGS is greater than Vt, a channel is induced. As we increase VDS current starts flowing from Drain to Source (triode region). When we further increase VDS, till the voltage between gate and channel at the drain end to become Vt, i.e. VGS – VDS = Vt, the channel depth at Drain end decreases almost to zero, and the channel is said to be pinched off. This is where a MOSFET enters saturation region.

5. Explain the three regions of operation of a MOSFET.

Cut-off region: When VGS < Vt, no channel is induced and the MOSFET will be in cut-off region. No current flows. Triode region: When VGS ≥ Vt, a channel will be induced and current starts flowing if VDS > 0. MOSFET will be in triode region as long as VDS < VGS – Vt.

Saturation region: When VGS ≥ Vt, and VDS ≥ VGS – Vt, the channel will be in saturation mode, where the current value saturates. There will be little or no effect on MOSFET when VDS is further increased.

6. What is channel-length modulation?

In practice, when VDS is further increased beyond saturation point, it does has some effect on the characteristics of the MOSFET. When VDS is increased the channel pinch-off point starts moving away from the Drain and towards the Source. Due to which the effective channel length decreases, and this phenomenon is called as Channel Length Modulation.

7. Explain depletion region.

When a positive voltage is applied across Gate, it causes the free holes (positive charge) to be repelled from the region of substrate under the Gate (the channel region). When these holes are pushed down the substrate they leave behind a carrier-depletion region.

8. What is body effect?

Usually, in an integrated circuit there will be several MOSFETs and in order to maintain cut-off condition for all MOSFETs the body substrate is connected to the most negative power supply (in case of PMOS most positive power supply). Which causes a reverse bias voltage between source and body that effects the transistor operation, by widening the depletion region. The widened depletion region will result in the reduction of channel depth. To restore the channel depth to its normal depth the VGS has to be increased. This is effectively seen as change in the threshold voltage – Vt. This effect, which is caused by applying some voltage to body is known as body effect.

9. Give various factors on which threshold voltage depends.

As discussed in the above question, the Vt depends on the voltage connected to the Body terminal. It also depends on the temperature, the magnitude of Vt decreases by about 2mV for every 1oC rise in temperature.