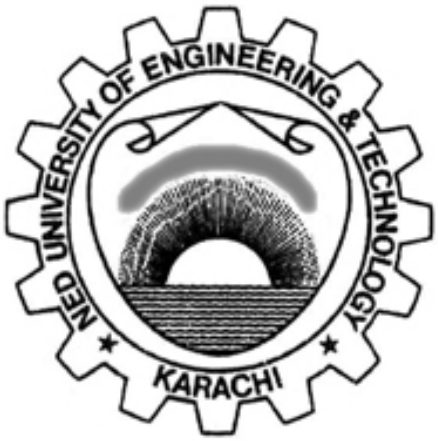
LABORATORY WORKBOOK for the course

EL-407 VLSI SYSTEMS DESIGN



Name : _____

Roll No. : _____

Batch : _____

Year : _____

Dept. :_____

Department of Electronic Engineering
N.E.D. University of Engineering & Technology

LABORATORY WORKBOOK for the course

EL-407 VLSI SYSTEMS DESIGN

Prepared by

Abdur Rahim Quershi (Assistant Professor)

Naveera Sami (Lecturer)

Reviewed by:

Hashim Raza Khan (Assistant Professor)

Approved by:

Board of Studies of Department Of Electronics Engineering

CONTENTS

Lab Session 01

Objective

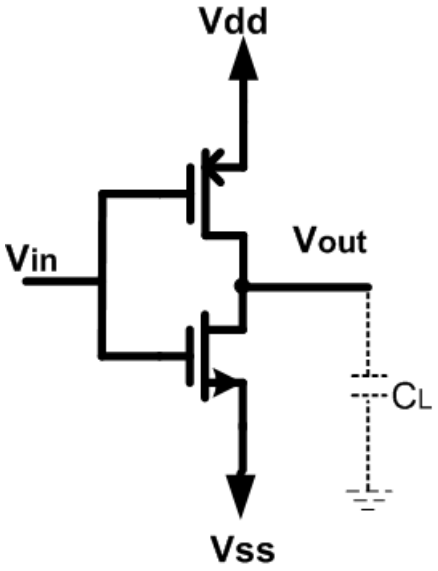
To design and verify the layout of a CMOS inverter.

- There must be a single rail of Vdd as well as for Vss.
- Width of PMOS is twice (or 2.5 times) as compared to NMOS transistor.
- Also draw stick diagram and mention Euler's path.

Equipment Required

- Microwind/Mentor Graphics software installed PCs

Schematic



Schematic of CMOS Inverter

Theory

CMOS logic based function consists of a pair of NMOS and PMOS transistors. Two different transistors have to be placed in Layout so it will be a good exercise for learning structure of both transistors. The network consisting of all PMOS transistors, known as P-network, is responsible for determining rise time of output. While, the network consisting of all NMOS transistors, known as N-network, is responsible for determining fall time of output waveform.

Procedure

Follow the instructions given during Lab to complete task successfully.

Results

Q. What will be the effect on rise and fall time of output waveform, when:

1) $(\frac{W}{L})_p = (\frac{W}{L})_n$
2) $(\frac{W}{L})_p = 2(\frac{W}{L})_n$

Q. Attached the Layout of CMOS inverter.

Lab Session 02

Objective

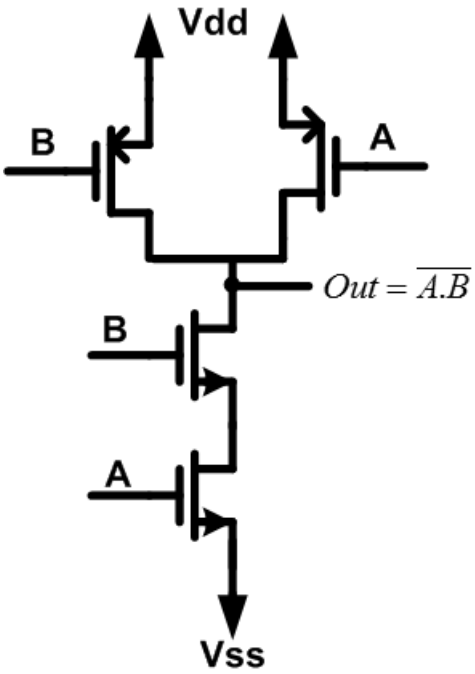
To design and verify the layout of a CMOS NAND gate.

- There should be continuous layer of n+ and p+ diffusion.
- There must be a single rail of Vdd as well as for Vss.
- Width of PMOS is twice (or 2.5 times) as compared to NMOS transistor and length of all transistors should be same.
- Also draw stick diagram and mention Euler's path.

Equipment Required

- Microwind/Mentor Graphics software installed PCs

Schematic



Schematic of $F=\overline{(A.B)}$

Theory

CMOS logic based function consists of a pair of NMOS and PMOS transistors. Two different transistors have to be placed in Layout so it will be a good exercise for learning structure of both transistors. The network consisting of all PMOS transistors, known as P-network, is responsible

for determining rise time of output. While, the network consisting of all NMOS transistors, known as N-network, is responsible for determining fall time of output waveform.

There are two inputs involved i.e., A and B. So, there should be optimum gate ordering in layout. A simple method for finding the optimum gate ordering is the Euler-path method.

Procedure

Follow the instructions given during Lab to complete task successfully.

Results

Attached the Layout of CMOS logic NAND gate, mentioning all diffusion layers.

Lab Session 03

Objective

To design and verify the layout of given logic function on CMOS logic.

- There should be continuous layer of n+ and p+ diffusion.
- There must be a single rail of Vdd as well as for Vss.
- Width of PMOS is twice (or 2.5 times) as compared to NMOS transistor and length of all transistors should be same.
- Also draw stick diagram and mention Euler's path.

Equipment Required

- Microwind/Mentor Graphics software installed PCs

Schematic

Schematic of given function (showing Euler's path)

Theory

CMOS logic based function consists of a pair of NMOS and PMOS transistors. Two different transistors have to be placed in Layout so it will be a good exercise for learning structure of both transistors. The network consisting of all PMOS transistors, known as P-network, is responsible for determining rise time of output. While, the network consisting of all NMOS transistors, known as N-network, is responsible for determining fall time of output waveform.

For multiple inputs, there should be optimum gate ordering in layout. A simple method for finding the optimum gate ordering is the Euler-path method.

Procedure

Follow the instructions given during Lab to complete task successfully.

Results

Attached layout of a given function.

Lab Session 04

Objective

To design and verify the layout of a given logic function on CMOS logic.

- There should be continuous layer of n+ and p+ diffusion.
- There must be a single rail of Vdd as well as for Vss.
- Width of PMOS is twice (or 2.5 times) as compared to NMOS transistor and length of all transistors should be same.
- Also draw stick diagram and mention Euler's path.

Equipment Required

- Microwind/Mentor Graphics software installed PCs

Schematic

Schematic of given function (showing Euler's path)

Theory

CMOS logic based function consists of a pair of NMOS and PMOS transistors. Two different transistors have to be placed in Layout so it will be a good exercise for learning structure of both transistors. The network consisting of all PMOS transistors, known as P-network, is responsible for determining rise time of output. While, the network consisting of all NMOS transistors, known as N-network, is responsible for determining fall time of output waveform.

For multiple inputs, there should be optimum gate ordering in layout. A simple method for finding the optimum gate ordering is the Euler-path method.

Procedure

Follow the instructions given during Lab to complete task successfully.

Results

Attached the layout of a given function.

Lab Session 05

Objective

1) Getting familiar with Verilog HDL for digital design.

2) To simulate and verify the verilog code on ModelSim Software.

Equipment Required

- Modelsim software installed PCs
-

Introduction

Traditionally, digital design was done by the schematic entry. This has been replaced today by the use of Hardware Description Language (HDL).

In electronics, HDL is the language used for formal description of electronic circuits.HDL offers several advantages over traditional design techniques such as efficient and convenient way of designing, simulation and synthesis of large electronic circuits containing larger number of electronic components and devices, efficient verification of design in initial phase of development.

Two popular HDLs are Verilog and VHDL. The HDL used in our lab will be Verilog.

Hardware and Software Used In Lab

Hardware

- Xilinx Spartan III kit
- Xilinx Virtex 5 kit
- Altera cyclone II kit

Software

- Quartus II
- Model Sim

Modes of Programming

There are two modes of programming into FPGA.

- JTAG

- AS (active serial)

Modes can be selected manually by a two way sliding button on the board by keeping its position on RUN or PROG respectively. The only advantage of AS mode is to keep your program saved into the flash memory of FPGA. In this mode your program will not be lost even after turning off the device and you may be able to access it again by powering up it again.

For JTAG, set up the switch on the board to RUN position.

Introduction to Verilog HDL

Verilog is a hardware description language used to model electronic systems. Verilog HDL is one of the HDLs used by the integrated circuits (IC) designers. The other one is VHDL (VHSIC-Very High Speed Integrated Circuit Hardware Description Language).

Verilog uses four levels of abstraction to describe the designs.

1. The switch level
2. The gate or structural level
3. The data flow level
4. the behavioral or procedural level

The Switch Level

It includes MOS transistors modeled as switches.

The Gate or Structural Level

At this level, gates primitives are called to design the logic. It is not synthesizable.

The Data Flow Level

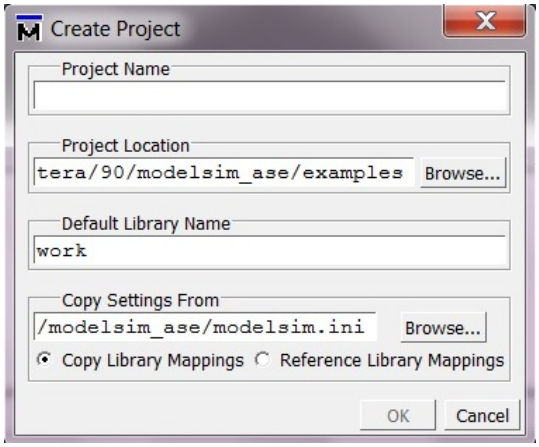
At this level, continuous assignments are used by using the keyword ‘assign’. Synthesizer is required. It is also called Data Flow Models. This level allows using every type of operators.

The Behavioral or Procedural Level

At this level, you just have to define the behavior. It is user friendly. It uses procedural blocks (blocking and non blocking) hence, called procedural level.

Procedure

1. Open the ModelSim software.
2. Create a new project by **File => New => Project** from the Main window.
3. A "**Create Project**" window appears as shown in figure below. Select a suitable name for your project; leave the Default Library Name to work.



4. After project name, an **Add items to the Project** dialog pops out as shown in figure below.

5. From the "Add items to the Project" dialog click on **Create a new file**. If you have closed the "Add items to the Project" dialog, then select **Project => Add to Project => New File** from the main window.
6. A **Create Project File** dialog pops out. Select an appropriate file name for the file you want to add (the name of file must be same as you write in Step 4); choose **Verilog** as the add file as type option and **Top level** as the Folder option (see figure below) and then click on OK.
7. On the workspace section of the **Main Window**, double-click on the file you have just created (VLSI.v in our case).
8. Type verilog code of the given task in the new window.
9. Save your code.
10. In workspace window do right click on project name (i.e. VLSI) select **Compile => Compile All**. A message " Compile of VLSI.v was successful" will appear in message window

11. For simulating the design click on **Simulation => Start Simulation** in main window, simulation environment will appears as shown in figure below.

12. Click on the (+) sign next to the **work** library. You should see the name of the entity of the code that we have just compiled **"VLSI"** select your desired file.

13. Locate the signals window and select the signals that you want to monitor for simulation. For this example of AND gate, select all signals as shown figure below.

14. Drag the above signals by selecting all then right click and select **Add => to Wave** => **Selected items** to the wave window.

15. Now we are ready to simulate our design. For this purpose we will change the values of inputs (i.e. a and b in above example of AND gate) by right click on input and select **Force** and write either '0' or '1' in value box and repeat same step for changing the value of other inputs.

16. Click **Run** button in main window tool bar and can see the changes in the both the wave and objects windows.

Results & Observations

Design the modules using all basic gates: AND, OR, XOR, NOR, NAND and XNOR gate and verify the results

Lab Session 06

Objective

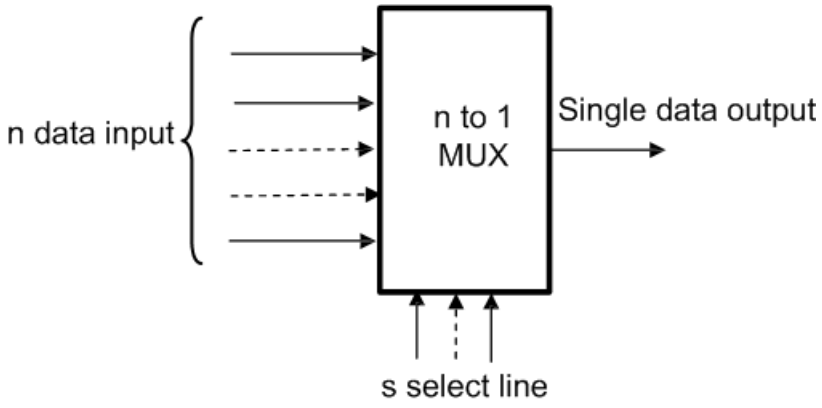
- To understand 4 to 1 MUX working principle
- To understand ModelSim Software for Development of Verilog HDL Codes.
- To implement and Test 4 to 1 MUX on Verilog HDL by
    - Gate Level Modeling
    - Data Flow Modeling
    - Behavioral Modeling

Equipment Required

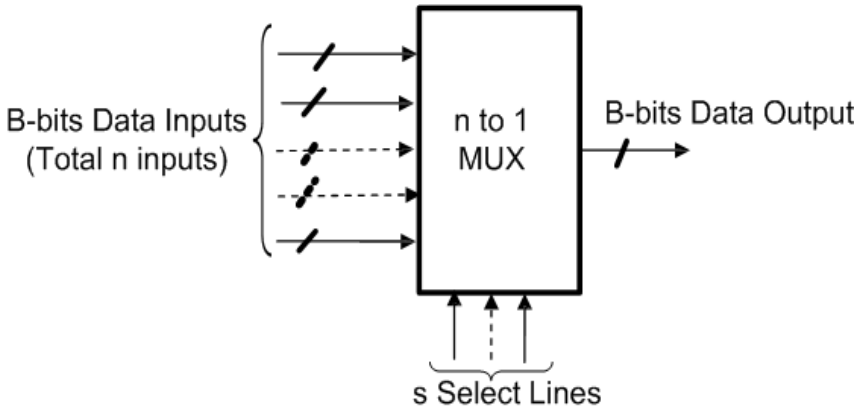
- Modelsim Installed PCs

Theory

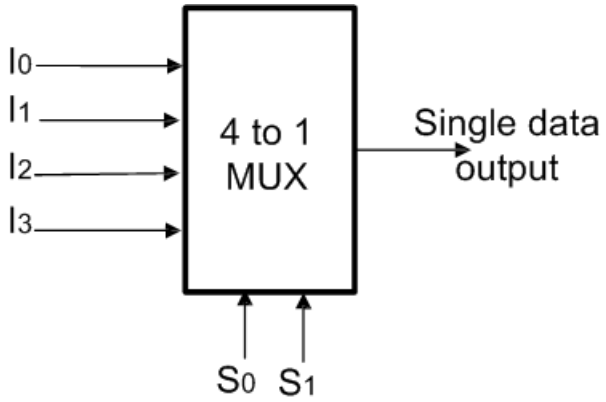
A multiplexer (MUX) is a digital switch which connects data from one of "n" inputs to a single output. A number of "Select Inputs" determine which data input is connected to the output. The Block Diagram of MUX with "n" data inputs and "s" select lines is shown in figure below:



MUX acts like a digitally controlled multi-position switch where the binary code applied to the select inputs controls the input source that will be switched on to the output. At any given point of time only one input gets selected and is connected to output, based on the select input signal. Input can be single bit or multi bits in nature. Following figure shows n to 1 MUX, handling "B" bits of each input and select them to "B" bits output.



A 4 to 1 MUX is shown in figure below. There are four input lines, I0 to I3, and two selections lines, S0 and S1, are decoded to select a particular input to appear at output.



The truth table for the 4:1 MUX is given as:

S1	S0	Output
0	0	I0
0	1	I1
1	0	I2
1	1	I3

Procedure

1. Understand Gate level modeling.
2. Create new Modelsim project for writing the code.
3. Open new Verilog file and write code for Multiplexer in it.
4. Include Verilog file in your project and compile your project.
5. Simulate your project and verify results.
6. Understand Data flow modeling.
7. Repeat steps 3 to 5.
8. Understand Behavioral Modeling.
9. Repeat steps 3 to 5.

Results

Attach the verified results.

Lab Session 07

Objective

To understand Quartus-II Software for Development of Verilog HDL codes.

- To implement and test Verilog HDL code of a given function.
- To test program on ALTERA DE2 board.

Equipment Required

- Quartus II software installed PCs
- ALTERA DE2 Board

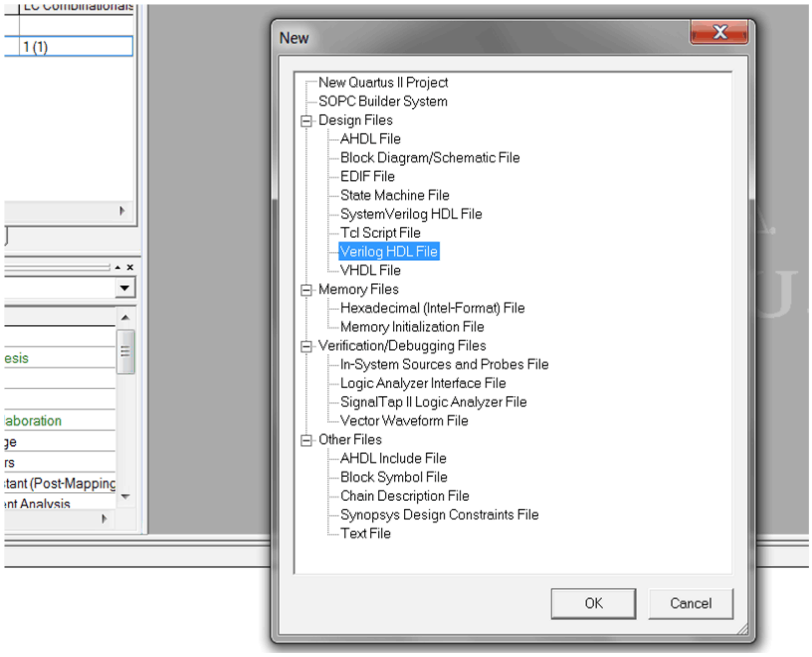
Procedure

Connect USB-blaster cable and power adaptor with ALTERA DE2 board.

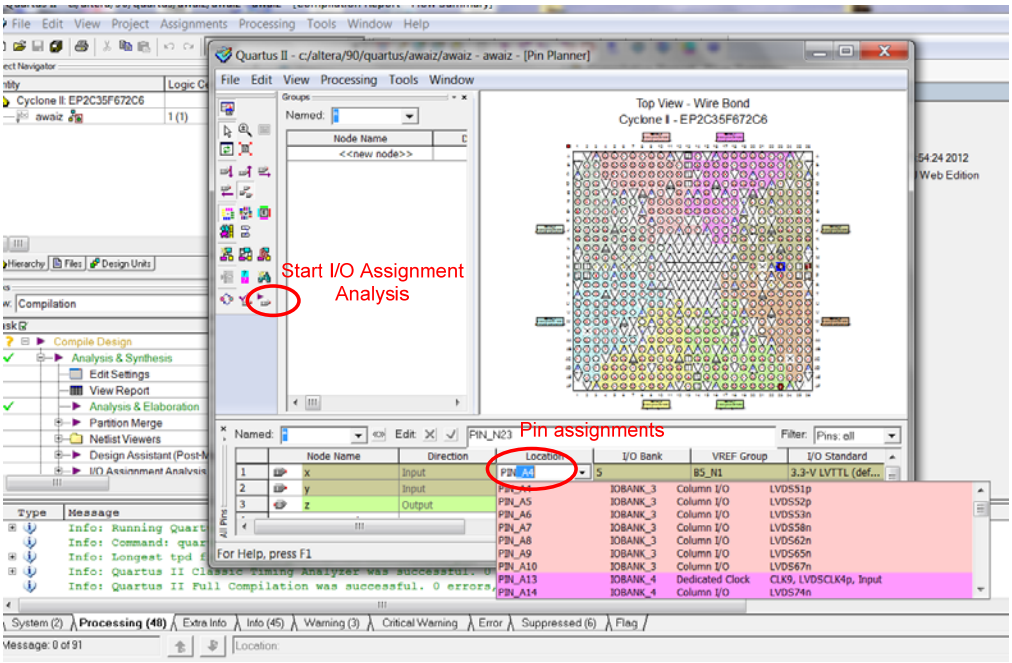
1. Open the Quartus II software.
2. Create a new project by selecting "**Create a New Project (New Project Wizard)**" as shown in Figure below.

3. Select a suitable name for your new directory (or you can use the existing one) and also the name of the project and click on **next** option.

4. After creating new directory and project, create a new file by selecting **File** => **New** and select **Verilog HDL File** type as shown in Figure below.



5. A command window will appear. Write your program and save it with the same name as given in module command. (Make sure that file should be saved in the same project directory mentioned in step **2**)
6. Now compile your program by selecting **Processing** => **start compilation.**
7. After completion of compilation a message will appear "**full compilation was successful**".
8. To verify your verilog code on ALTERA board, assign suitable pins/switches/LEDs to your input/output terminals by **Assignments** => **Pin Planner.** A pin planner window will appear.



9. Assign switches and LEDs to all input and output terminals respectively and start I/O assignment analysis as shown in above figure. The location of ALTERA DE2 board can be selected from "**ALTERA DE2 user manual**".

10. After I/O assignment analysis, now code is ready to be dumped in ALTERA DE2 board. Select **Tools** => **Programmer** and after selection of USB blaster option select **Start**. A 100% completion message will appear, when program is completely dump.

11. Now, you can test your program on ALTERA DE2 board.

Results and Observations

Lab Session 08

Objective

Program FPGA with SR latch using Verilog HDL and getting the overview of Flip flops.

Equipment required

- Xilinx Spartan III, Virtex 5 kit and Altera cyclone II kit.
- Quartus II installed PCs.

Introduction

Latches are level sensitive storage elements; the action of data storage is dependent on the level (value) of the input clock (or enable) signal. Flip flops are edge sensitive storage elements; the action of data storage is synchronized to either a rising or falling edge of a signal.

SR latch (set reset) latches can be used by the implementation of nor gates or nand gates. We will program active low input SR latch cross coupled with nand gates in the lab session.

Schematic Diagram of SR latch

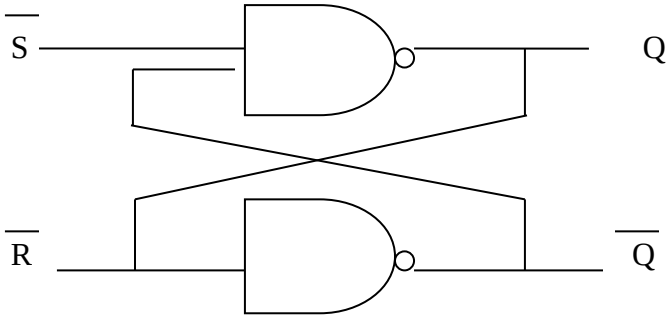


Fig 8.1

Characteristic Table of active low input SR Latch with Cross Coupled Nand Gates

S'	R'	Q_{next}	Q'_{next}
0	0	1	1
0	1	1	0
1	0	0	1
1	1	0	Q'

Table 8.1

Note:

- First stage in the table is indeterminate or invalid.
- Second stage is called set.
- Third stage is called set.
- Fourth stage is called hold.

Exercise

Attach the printout of SR latch observed values and make a table.

Lab Session 09

Objective

Design and compile 2-to-4 line decoder and program FPGA.

Equipment required

- Altera cyclone II kit.
- Model Sim/Quartus II installed PCs.

Introduction

The decoder is an integrated circuit that receives the input at its pins and then decodes the combination at the input and shows the unique output word in which only one bit is asserted. A decoder has n inputs and 2^n output lines. These are widely used in communication and video transmission circuits.

Truth table

The output pins are active low.

Enable	In 1	In 0	Out 3	Out 2	Out 1	Out 0
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	1	1	0
1	0	1	1	1	0	1
1	1	0	1	0	1	1
1	1	1	0	1	1	1

Table 9.1

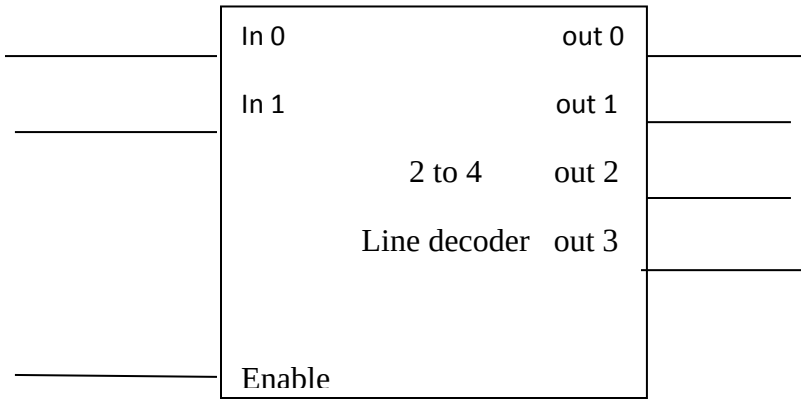


Fig 9.1

Exercise: Apply different combination of inputs and verify results and attach print out.

Lab Session 10

Objective

Getting started with the concept of Instantiation in Verilog HDL.

- Design and compile 3-to-8 line decoder using instantiation and program FPGA.

Equipment required

- Altera cyclone II kit.
- Model Sim/Quartus II installed PCs.

Introduction

Instantiation

A module provides a template from which you can create actual objects. When a module is invoked, Verilog creates a unique object from the template. Each object has its own name, variables, parameters and I/O interface. The process of creating objects from a module template is called instantiation.

The decoder is an integrated circuit that receives the input at its pins and then decodes the combination at the input and shows the unique output word in which only one bit is asserted. A decoder has n inputs and 2^n output lines.

The decoders are used to extract opcode from instruction in a computer. These are widely used in communication and video transmission circuits.

By using the two instances of 2 to 4 line decoder designed in the previous lab session, the 3 to 8 line decoder is designed.

Truth table

In 2	In 1	In 0	En 1	En 0	Out7	Out6	Out5	Out4	Out3	Out2	Out1	Out0
0	0	0	0	1	1	1	1	1	1	1	1	0
0	0	1	0	1	1	1	1	1	1	1	0	1
0	1	0	0	1	1	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1	0	1	1	1	1
1	0	1	1	0	1	1	0	1	1	1	1	1
1	1	0	1	0	1	0	1	1	1	1	1	1
1	1	1	1	0	0	1	1	1	1	1	1	1

Table 10.1

Note:

- The enable input is active high.
- The output (out) is active low.
- The two 2 to 4 line decoders are cascaded.

Block diagram

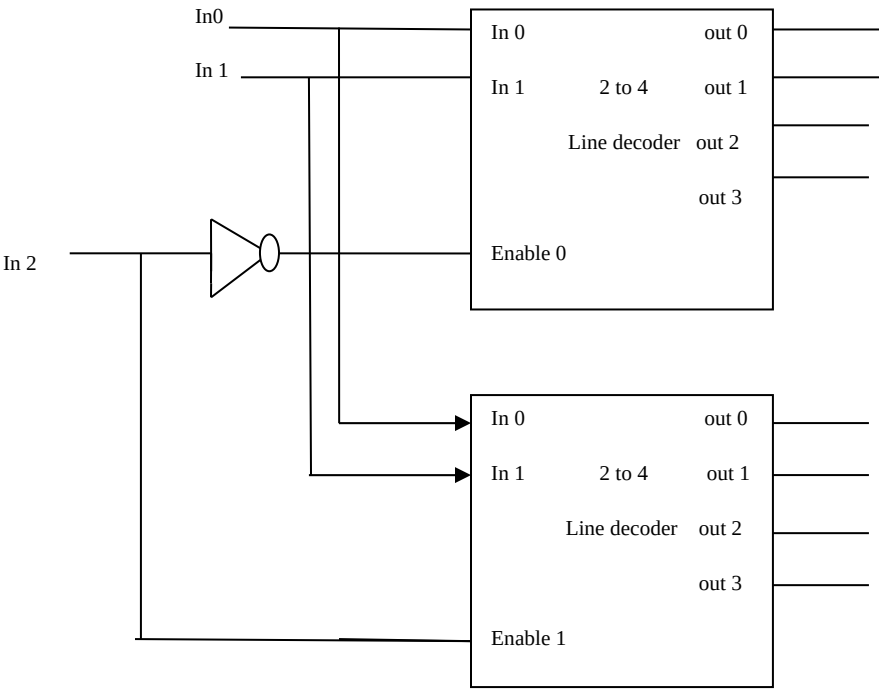


Fig 10.1

Exercise: Apply different combination of inputs and verify results and attach print out.

Lab Session 11

Objective

To understand the Mentor Graphic Software for schematic and Layout design for electronic circuit, using standard and Generic MGC Libraries of Mentor Graphics.

Equipment Required

- Linux and Mentor Graphics Installed PCs.

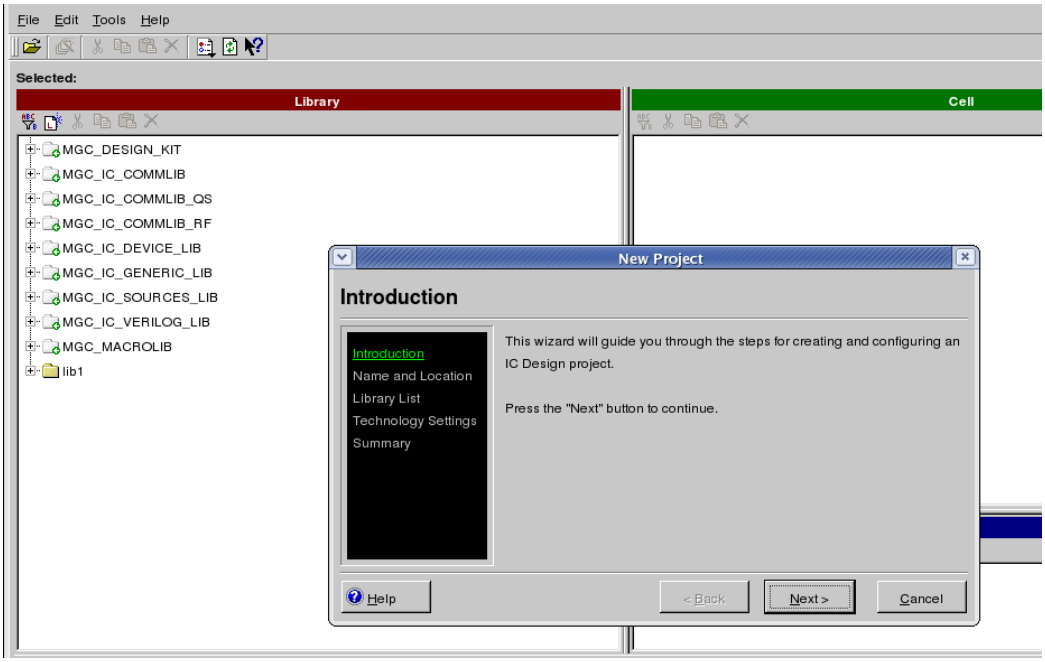
Procedure

1. Run the linux environment by open the virtual machine.
2. To run mentor graphics open the terminal by right click and type following commands:

    - Source adk_da_ic (**Enter**)
    - Cd /usr/mentor-graphic/caliber/2008.2_rhelx86linux/icflow_home/bin (**Enter**)
    - ./icstudio (**Enter**)

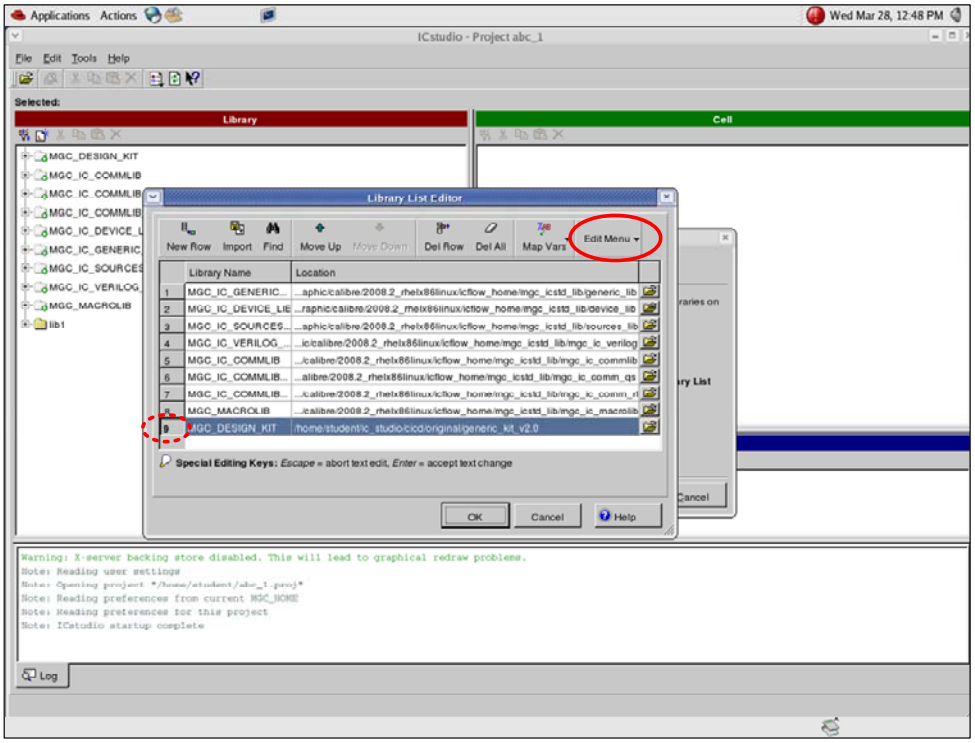
    Finally the Mentor Graphics environment will run.

3. Now create a new project from File =>New=>Project.



4. Give the proper name and location of project and press **Next**. The location of the project should be **/home/student.**
5. Then '**Open Library List Editor'** and include generic and standard libraries.
6. Go to **Edit Menu** and "**Add standard MGC Libraries**". After that eight libraries will add in **Library List Editor.**
7. Now add the generic Kit through "**Add MGS Design Kit**" from **Edit Menu** of **Library List Editor**, from path (write path here)

_____

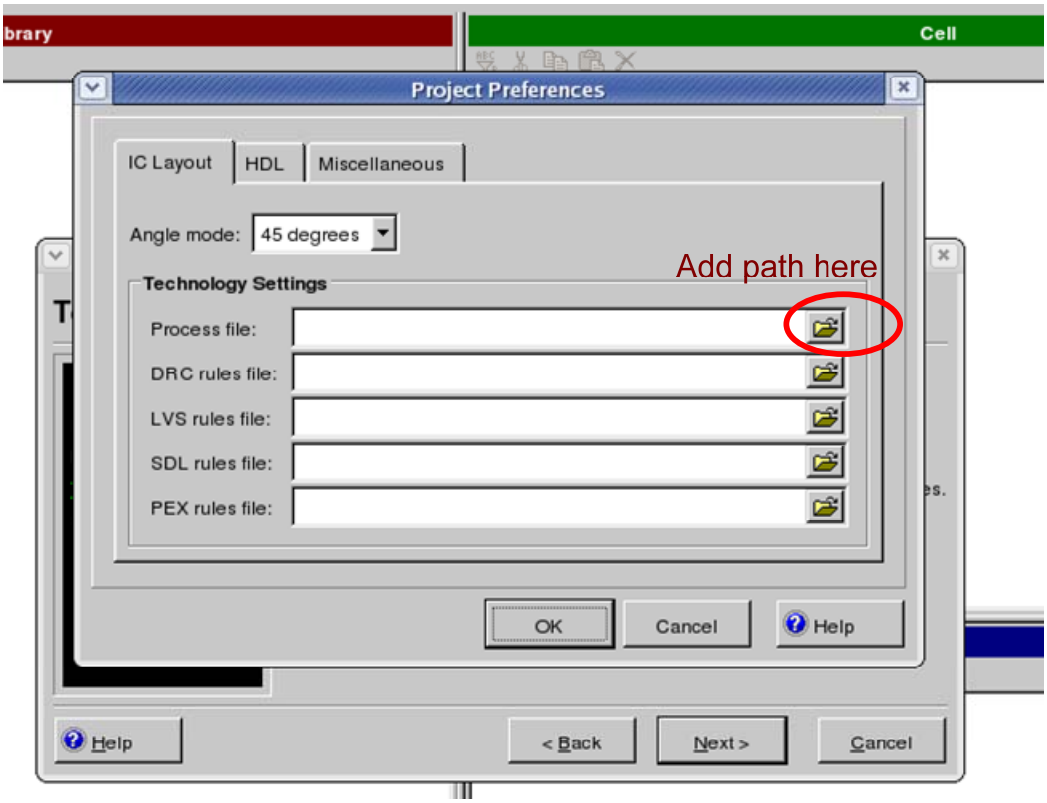


8. After step 7, there should be total eight (8) libraries will appear in **Library List Editor** wizard. Now press **OK** and then **Next**.
9. Now in **Technology Settings** "open Setting Editor" and include all technology file of your MGC Design Kit.
10. Here include **Process file , DRC rules file, LVS rules file, SDL rules file** and **PEX rules file** and then **Next** and **Finish**.

    Write path for following rules files.

    Process file:

    _____



DRC rules file:

_____

LVS rules file:

_____

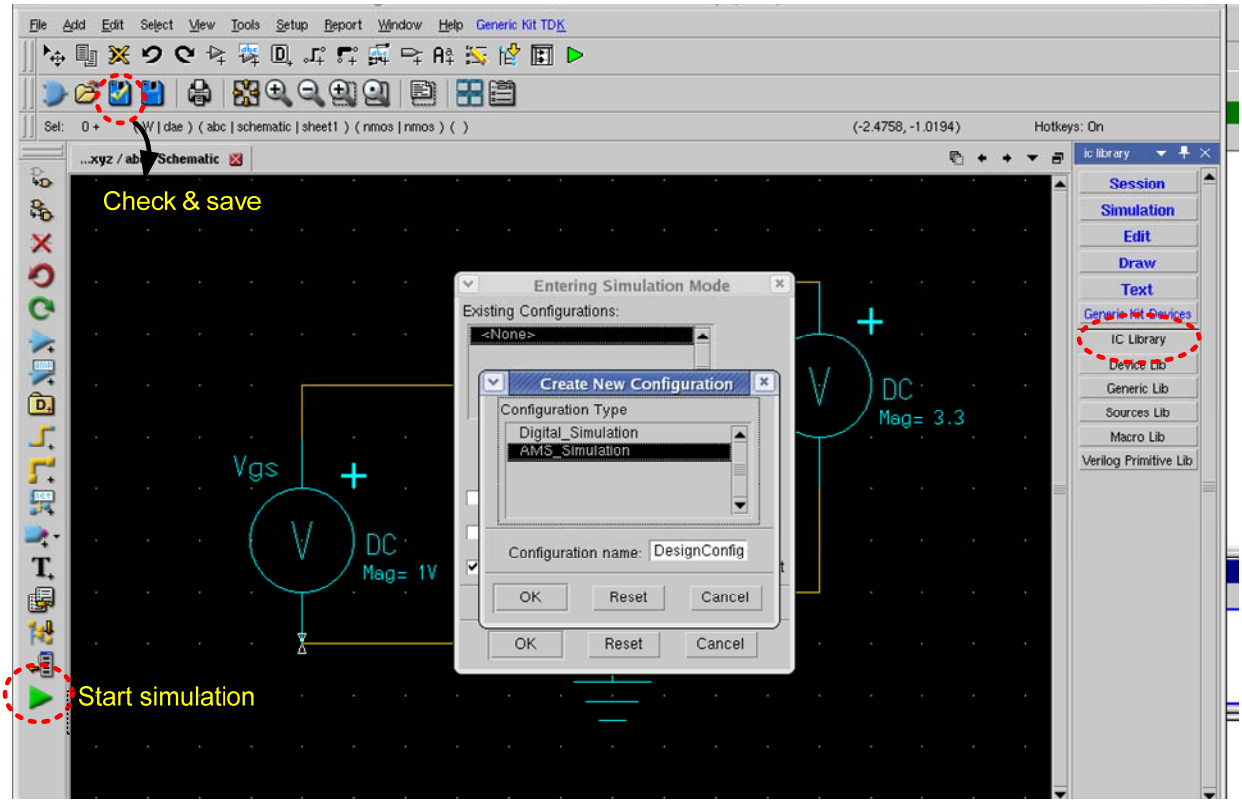
SDL rules file:

_____

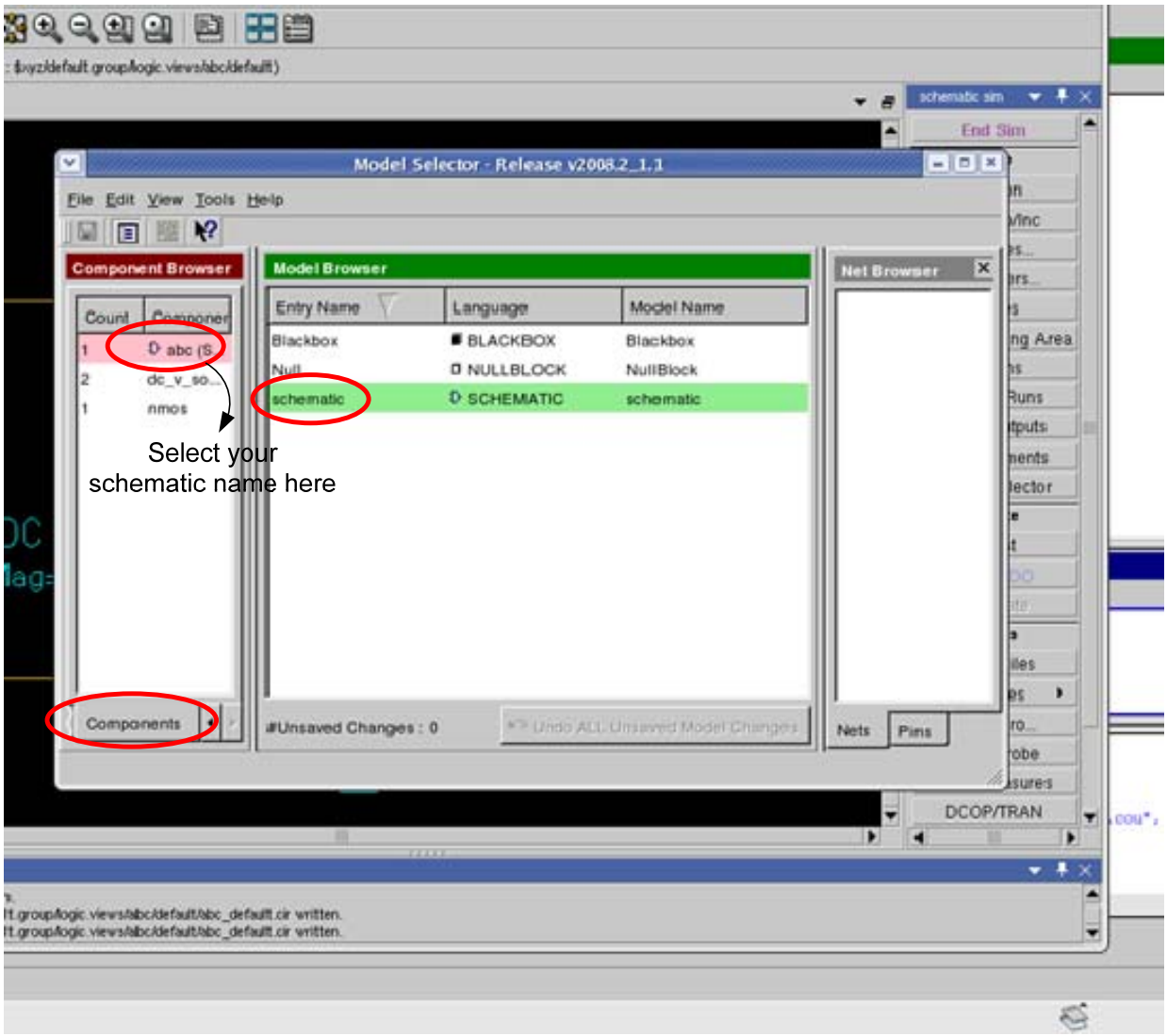
PEX rules file:

_____

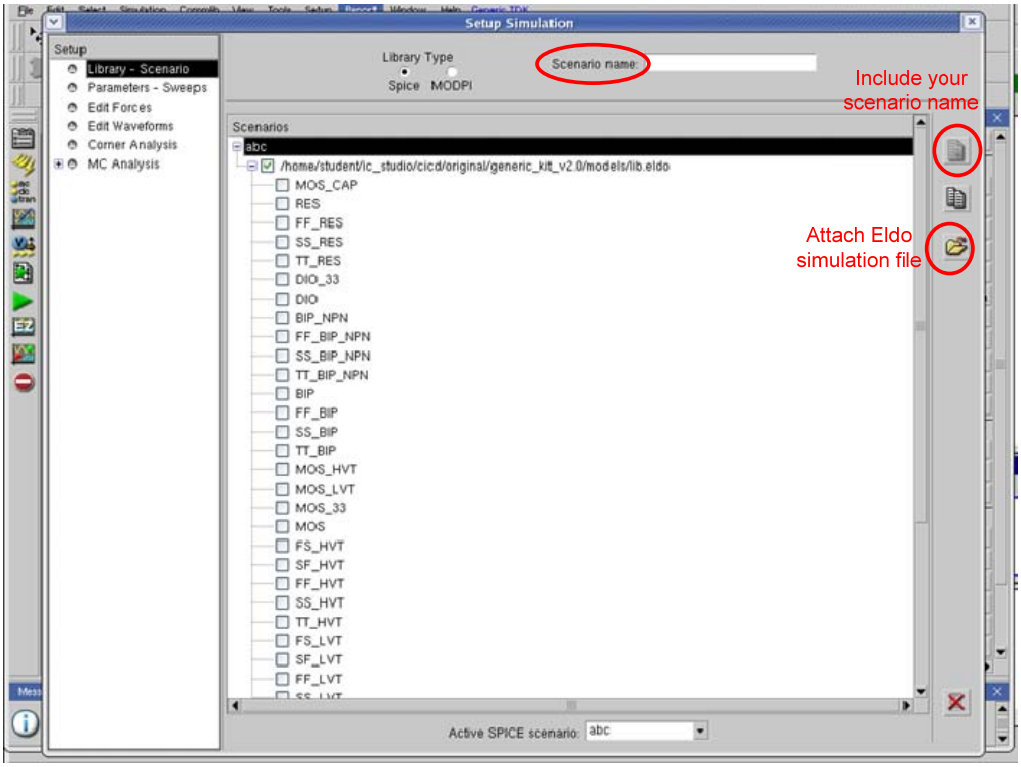
11. Now **File => New Library** and create new Library.
12. Now create a new schematic with suitable name in same created library (in step 11) by right click on library and select **New Cell View** and select Schematic in **View Type**.
13. Schematic window will appear. Draw your circuit on Schematic level and selecting components and sources from suitable libraries.
14. When schematic will complete, "**Check and Save**" your design. The message will appear. If there is no error you can move further for simulations.
15. Now run the simulation a window will pop up. Select **New configuration** and select **AMS_Simulation**.



16. Select “Eldo” simulation by **Setup Sim Session =>Simulator/View => Eldo.**
17. Now select and appropriate environment for your simulation “*View Waveforms After Simulation*” from “**Setup Sim Session**” and select “**Environment**” .
18. Now select “**Model Selector**” and window will appear. Select **Component** and **Schematic** as shown in Fig. below.



19. Now select which type of analysis you want to run by select "**ac dc Trans**" and any simulation. For example in case of **DC** analysis.
20. Check DC analysis and then select **Setup.** Select any source and Sweep parameters (Start point, End point and step size) of selected source, then press OK.
21. Now select "**Setup Output**", a window will appears as shown in Fig. below.



22. Now you can run your simulations and by drag and drop option you can view waveforms.

Results and Observation

Lab Session 12

Objective

To determine the behavior of MOS transistor using Mentor Graphics by analyzing its

- I_D v/s V_{DS} curve
- I_D v/s V_{GS} curve
- Early Effect

Equipment Required

- Linux and Mentor Graphics Installed PCs.

Theory

EARLY EFFECT:

When the MOS operated in the saturation region, practical MOS shows some dependence of the drain current on the drain source voltage at a constant V_{GS}. That dependency is almost linear with a slope equal 1/ro. When extrapolated the curves, the characteristics lines meet at a point on the V_{DS} axis at $V_{DS} = V_A$. The voltage V_A is called the early voltage. This phenomenon is generated because of the reduction in the effective channel width due to the increase in the reverse-bias voltage on the drain-bulk junction.

Procedure

Follow the instruction given in Lab session 09 and during the Lab to complete task successfully.

Results and Observations

Draw/attach the aforementioned curves.

Lab Session 13

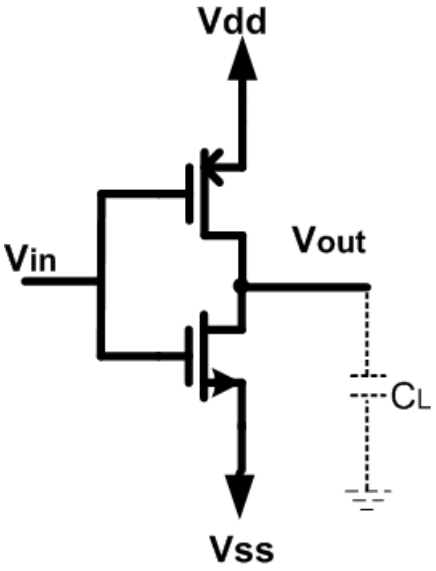
Objective

To determine the behavior of a given circuit by modeling and simulation on Mentor Graphics Software.

Equipment Required

- Linux and Mentor Graphics Installed PCs.

Theory



An inverter circuit outputs a voltage representing the opposite logic-level to its input. Inverters can be constructed using a single NMOS transistor or a single PMOS transistor coupled with a resistor. Since, this 'resistive-drain' approach uses only a single type of transistor, it can be fabricated at low cost. However, because current flows through the resistor in one of the two states, the resistive-drain configuration is disadvantaged for power consumption and processing speed. Alternately, inverters can be constructed using two complimentary transistors in a CMOS configuration. This configuration greatly reduces power consumption since one of the transistors is always off in both logic states. Processing speed can also be improved due to the relatively low resistance compared to the NMOS-only or PMOS-only type devices. Inverters can also be constructed with Bipolar Junction Transistors (BJT) in either a resistor-transistor logic (RTL) or a transistor-transistor logic (TTL) configuration.

Procedure

Follow the instruction given in Lab session 09 and during the Lab to complete task successfully.

Results and Observations

Draw/attach the voltage transfer curve.

Lab Session 14

Objective

To develop a self selected / assigned Lab project based on previous lab tasks.

- To select Lab project after discussion with course/Lab teacher.
- In case of Verilog project, simulation should be on Quartus-II and Modelsim software. DE2 Board will be used for hardware verification.

Equipment Required

- Linux and Mentor Graphics/Microwind/Quartus II and Modelsim Installed PCs.
- ALTERA DE2 Boards (in case of Verilog Project).

Theory

All students should select different projects related to their interests or assigned by course teacher. Students can perform these tasks in groups or individual. Maximum number of students in a group is restricted to two persons. Remaining matters/discussion related to project will discuss in Lab.

Procedure

- Identify selected/assigned project and the scope of the project.
- In case of Verilog project, develop code and test simulation results on Modelsim and verify hardware implementation on DE2 Boards.

Results and Observations

Submit a small report on project with workbook or appear in demo session.