ABSTRACT

Low power and high speed digital circuits are basic needs for any of digital circuit. Multiplexer is a basic circuit for any digital circuit. In this paper, different techniques of multiplexer designs like Complementary CMOS, Transmission gate, Pass transistor logic, Dual Pass transistor logic styles and Gate Diffusion Input has been introduced and their comparison on the basis of power, delay and Area (number of transistor) is done. A low power Multiplexer has been introduced which consumes least power as compare to above mentioned logic but have more delay as compare to other ,On the basis of these analyses it is concluded that proposed transmission gate multiplexer is better technique for designing an low power low area Multiplexer design with high speed

Keywords: CMOS, DPTL, PT, TG, GDI

CHAPTER 1

INTRODUCTION

1. INTRODUCTION

The increasing demand for low-power very large scale integration (VLSI) can be addressed at different design levels, such as the architectural, circuit, layout, and the process technology level. At the circuit design level, considerable potential for power savings exists by means of proper choice of a logic style for implementing Combinational circuits. This is because all the important parameters governing power dissipation, switching capacitance, transition activity, and short-circuit currents are strongly influenced by the chosen logic style. Depending on the application, the kind of circuit to be implemented, and the design technique used, different performance aspects become important, disallowing the formulation of universal rules for optimal logic styles [2].

This paper analyzes 4-to-1 multiplexer using complementary CMOS, Transmission gate, Pass transistor logic, Dual Pass transistor logic styles and Gate Diffusion Input. These implementations are compared based on the basis of transistor count, power dissipation, and delay A multiplexer is a device that selects one of several analog or digital input signals and forwards the selected input into a single line. A multiplexer of inputs has n select lines, which are used to select which input line to send to the output that is why it is also called a data selector. Multiplexer can also be used to implement any combinational circuit. So by simplifying design of multiplexer, design of many combinational circuits can be simplified [5]. Fig.1 and fig.2 show the block diagram and truth table for 4-to-1 multiplexer given below [5].

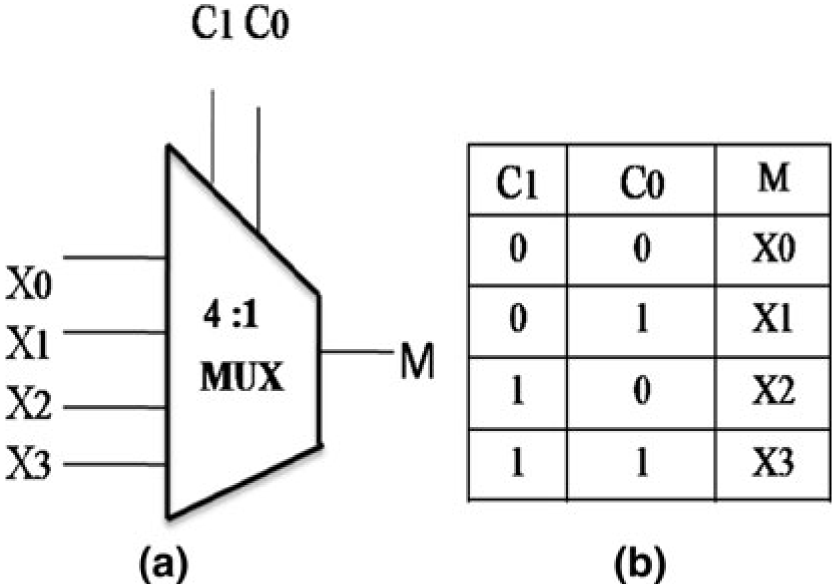
Multiplexer

It quite often happens, in the design of large-scale digital systems, that a single line is required to carry two or more different digital signals. Of course, only one signal at a time can be placed on the one line. What is required is a device that will allow us to select, at different instants, the signal we wish to place on this common line. Such a circuit is referred to as Multiplexer. The graphical symbol and truth table of 4:1 MUX are shown in Fig. 1a, b, respectively. A multiplexer performs the function of selecting the input on any one of 'n' input lines and feeding this input to one output line. Multiplexers are used as one method of reducing the number of integrated circuit packages required by a particular circuit design. This in turn reduces the cost of the system.

$$\text{Output}=X0C0'C1'+X1C0'C1+X2C0C1'+X3C0C1$$

Assume that we have four lines, X0, X1, X2 and X3, which are to be multiplexed on a single line, Output (M). The four input lines are also known as the Data Inputs. Since there are four inputs, we will need two additional inputs to multiplexer, known as the Select Inputs, to select which of the X inputs is to appear at the output, called as select lines C0 and C1. The gate implementation of a 4:1 MUX is shown in Fig. 2. Equation 1 is given for 4:1 MUX. CMOS transmission gate logic based 4:1 MUX CMOS transmission gate logic This section describes the purpose and basic operation of a transmission gate. The transmission gate can be used to quickly isolate multiple signals with a minimal investment in board area and with a negligible degradation in the characteristics of those critical signals.

A transmission gate is defined as an electronic element that will selectively block or pass a signal level from the input to the output. The solid-state-switch is comprised of parallel connection of a PMOS transistor and NMOS transistor. The control gates are biased in a complementary manner so that both transistors are either on or off. When the voltage on node A is a Logic 1, the complementary Logic 0 is applied to node active-low A, allowing both transistors to conduct and pass the signal at IN to OUT. When the voltage on node active-low A is a Logic 0, the complementary Logic 1 is applied to node A, turning both transistors off and forcing a high-impedance condition on both the IN and OUT nodes. The schematic diagram (Fig. 3) includes the arbitrary labels for IN and OUT, as the circuit will operate in an identical manner if those labels were reversed. This design provides true bidirectional connectivity without degradation of the input signal. The transmission gate graphical symbol and truth table are shown in Fig. 3. General MUX design concept This circuit is transistor level architecture of 4:1 MUX Fig. 1 4:1 MUX: graphical symbol (a), truth table (b)



C1	C0	M
0	0	X0
0	1	X1
1	0	X2
1	1	X3

Fig. 1.1 4:1 MUX: graphical symbol (a), truth table (b)

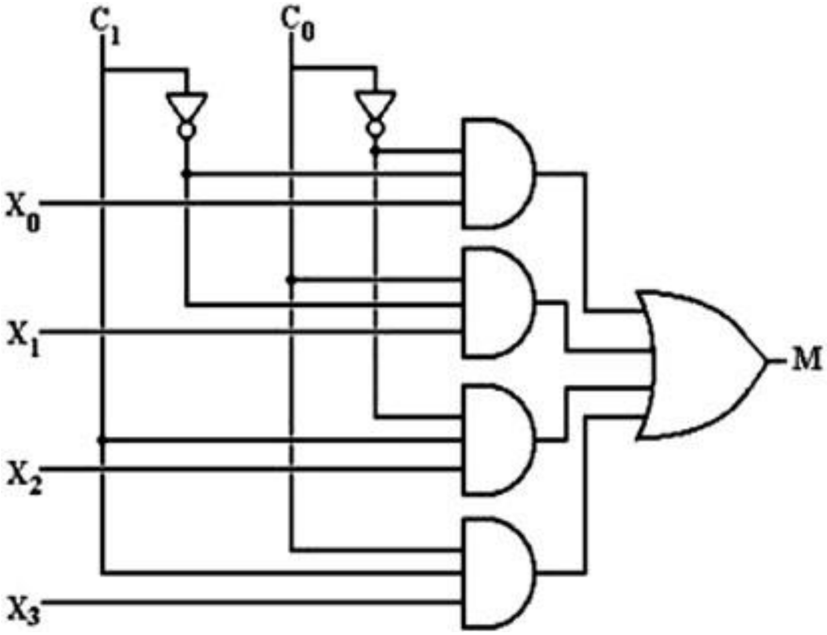


Fig. 1.2 Gate implementation of a 4:1 Multiplexer

select lines C0 and C1. The gate implementation of a 4:1 MUX is shown in Fig. 2. Equation 1 is given for 4:1 MUX.

CHAPTER 2

LITERATURE SURVEY

“Design and analysis of 2:1 multiplexer circuit for high performance,”

A multiplexer is a unidirectional device and used in any application in which data must be switched from multiple sources to a destination. The low power circuits have become a top priority in modern VLSI design. This paper presents the power consumption comparisons and delay of various designs of 2:1 Multiplexer. The various logic styles such as Differential Cascode Voltage Switch Logic (DCVSL), Modified Differential Cascode voltage switch logic (MDCVSL), CMOS logic and Pseudo NMOS Logic, these designs are analyzed using the Tanner EDA tool. The Pseudo NMOS Logic design demonstrates its superiority against other styles of 2:1 multiplexer design in terms of power consumption.

“Layout design and simulation of CMOS multiplexer,”

Multiplexer circuit is important device that have application in many field of Engineering. The research area of VLSI is to reduce area and complexity of the design. The purpose of this paper is to design 2 to 1 multiplexer with the help of CMOS logic to reduce area and complexity of the circuit. The different design methodologies are adopted in this paper to reduce the size, area and complexity of the multiplexer. This work evaluates 45nm technology. At the end, design methodologies are analyzed and optimize area of multiplexer is purposed.

“Design of Power efficient MUX using dual gate FinFET technology”

This paper presents the design and analysis of a 2:1 multiplexer. The conventional circuit of 2:1 multiplexer(MUX) is used for the calculation of different parameters like power 37 consumption, noise, delay, leakage power, etc. The multiplexer designed in this paper is suitable for low-power applications and works on very low supply voltage. Multiplexer is a digital circuit, it consists of 2N input and has n select line which are used to select the input line to transmit to the output. The multiplexer are used to expand the measure of information that can be sent over the system of a sure measure of time and bandwidth. Multiplexer comprises of multiple input signals and gives a single output switch. In this paper, a novel FinFET technique is used for the reduction of leakage power. The parameters of the conventional circuit and FinFET are compared and the performance of the multiplexer circuit is increased. The proposed multiplexer works on supply voltage of 0.7V. The design and simulation of FinFET based 2:1 multiplexer is done by using 45nm technology at cadence virtuoso version 6.1 platform.

“Simulation and analysis of 2:1 multiplexer circuits at 90nm technology”

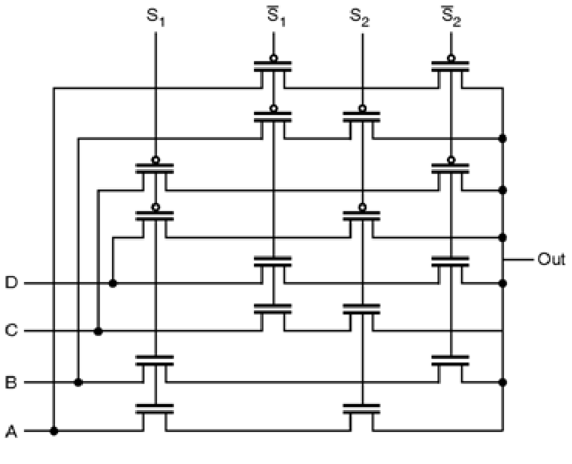
A multiplexer, sometimes referred to as a "MUX", is a device that selects between a numbers of input signals. It is a unidirectional device and used in any application in which data must be switched from multiple sources to a destination. This paper represents the simulation of different 2:1 MUX configurations and their comparative analysis on different parameters such as Power Supply Voltage, Operating Frequency, Temperature, Load Capacitance and Area Efficiency etc. All the simulations have been carried out on BSIM 3V3 90nm technology at Tanner EDA tool.

CHAPTER 3

EXISTING LOGICS

3.1. Conventional CMOS

In conventional or complementary CMOS logic gates are made up of an pmos pull-up and a nmos pull down logic network. CMOS logic style has an advantage of robustness against voltage scaling and transistor sizing .It has high noise margins and operates reliably at low voltages. Connection of input signals to transistor gates only, facilitates the usage and characterization of logic cells. The complementary transistor pair makes the layout of CMOS gates efficient and straightforward. The major disadvantage of CMOS is substantial number of large PMOS transistors which results in high input loads.



4 : 1 MUX using CMOS logic

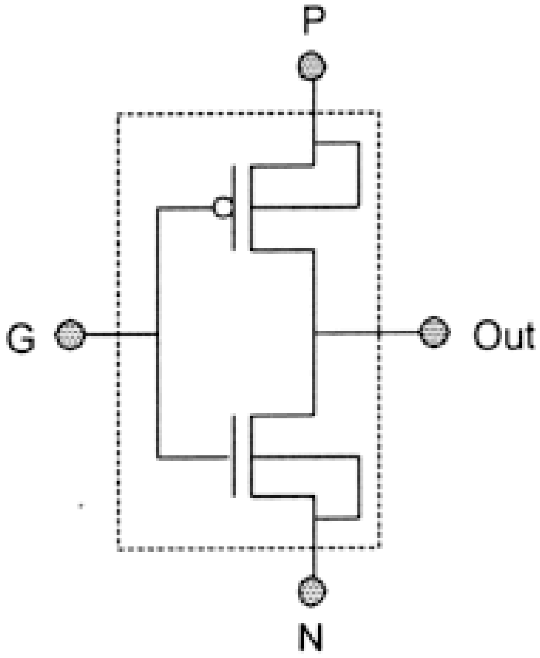
3.1 4:1 MUX USING CMOS LOGIC

3.1.1 DRAWBACKS

- Transistor count is more
- Power decipation is more
- Occupies more space

3.2. Gate Diffusion Input (GDI)

Apart from Conventional CMOS design, another alternative low power and area efficient technique is GDI technique. A basic GDI cell consists of four terminals- D (common diffusion of both transistors), N (outer diffusion node of nMOS transistor), P (outer diffusion node of pMOS), G (common gate input to both pMOS and nMOS transistors). Depending on the circuit structure and its mode of operation P, D and N can be used as either inputs or outputs.



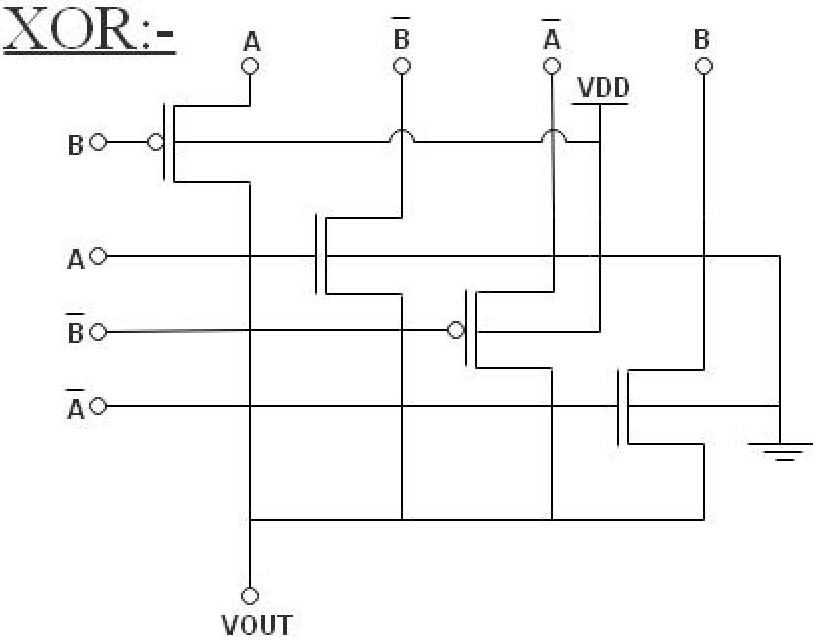
3.2 STRUCTURE OF GDI

3.2.1 DRAWBACKS

- Speed is low
- Power dissipation is more

3.3. Dual Pass Transistor Logic (DPTL)

The powerful configuration in CMOS technology is Dual Pass Transistor Logic (DPTL). Regardless of input signal-swing variation, DPTL buffers have the ability to generate standard CMOS levels. A basic DPTL structure consists of pMOS and nMOS transistors connected in parallel. Dual logic function in DPTL is generated by exchanging NMOS and PMOS, VDD and GND.



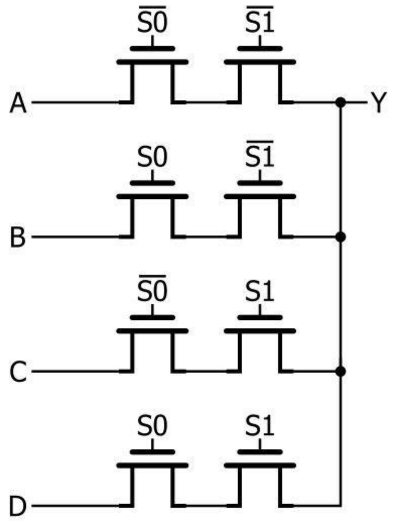
3.3 STRUCTURE OF DPTL

3.3.1. DRAWBACKS

- Speed is low
- Power dissipation is more

3.4. Pass-Transistor Logic style (PTL)

The pass-transistor logic reduces the number of transistors required to implement logic by allowing the primary inputs to drive gate terminals as well as source drain terminals. The advantage is that one pass-transistor network (either NMOS or PMOS) is sufficient to perform the logic operation [7, 8]. Several pass-transistor logic styles such as NMOS Pass Transistor Logic, CMOS Transmission gate, and pass transistor logic(PTL) are considered to implement4-to-1multiplexer [3, 5, 6]. Among all these NMOS Multiplexer is optimal. It uses two NMOS transistors and these two-pass transistors at the input select which signal to propagate. The logic levels will be deteriorated by the pass transistor. The threshold voltage of both pass-transistors should be identical for accurate operation. Figures in the simulation section represent design of 4-to-1 multiplexer using several logic styles.



3.4 STRUCTURE OF PTL

3.4.1 DRAWBACKS

- Power dissipation is more
- Occupies more space

CHAPTER 4

PROPOSED LOGIC

4.1 TRANSMISSION GATE

The analogue switch is a solid-state semiconductor switch that controls the transmission path of analogue signals. The open and closed operations of the switch positions are usually controlled by some digital logic network, with standard analogue switches available in many styles and configurations.

For example, single or dual normally open (NO) or normally closed (NC), single-pole single-throw (SPST), single-pole, double-throw (SPDT) configurations etc, in much the same way as for conventional electromechanical relays and contacts.

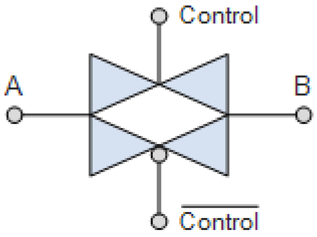
The switching and routing of digital and analogue signals (both voltage and current) can easily be done using mechanical relays and their contacts, but these can be slow and costly. The obvious choice is to use much faster acting solid state electronic switches which use metal oxide semiconductor (MOS) analogue gates to route the signal currents from their input to their output, with the well-known CMOS 4016B bilateral switch being the most common example.

MOS technology uses both NMOS and PMOS devices to perform the logical switching functions, thus allowing a digital computer or logic circuit to control the operation of these analogue switches. CMOS devices where both NMOS and PMOS transistors are fabricated into the same gate circuit, can pass (closed-condition) or block (open-condition) an analogue or digital signal, depending on the digital logic level that controls it.

The type of solid-state switch which allows for a signal or data transfer in both directions is called a **Transmission Gate**, or **TG**. But first lets consider the operation of a Field Effect Transistor, or FET as a basic analogue switch.

In this logic style N and P devices with sources and drains connected in parallel. Vg is the control signal for the N device, Vgc (complement of Vg) is the control signal for the P device. So When Vg is high (at Vdd) and Vgc is therefore low (at Gnd), the NFET and PFET are both ON [4]. (Depending upon the devices' source potentials, one may be ON more strongly than the other.) The switch is therefore CLOSED and Vout will be the same logic level as Vin. When Vg is low (at Gnd) and Vgc is high (at Vdd), both devices are OFF. The switch is therefore OPEN and Vout will be independent of Vin. A 4-to-1 multiplexer can be implemented using transistors by this logic style.

Transmission Gate Truth Table

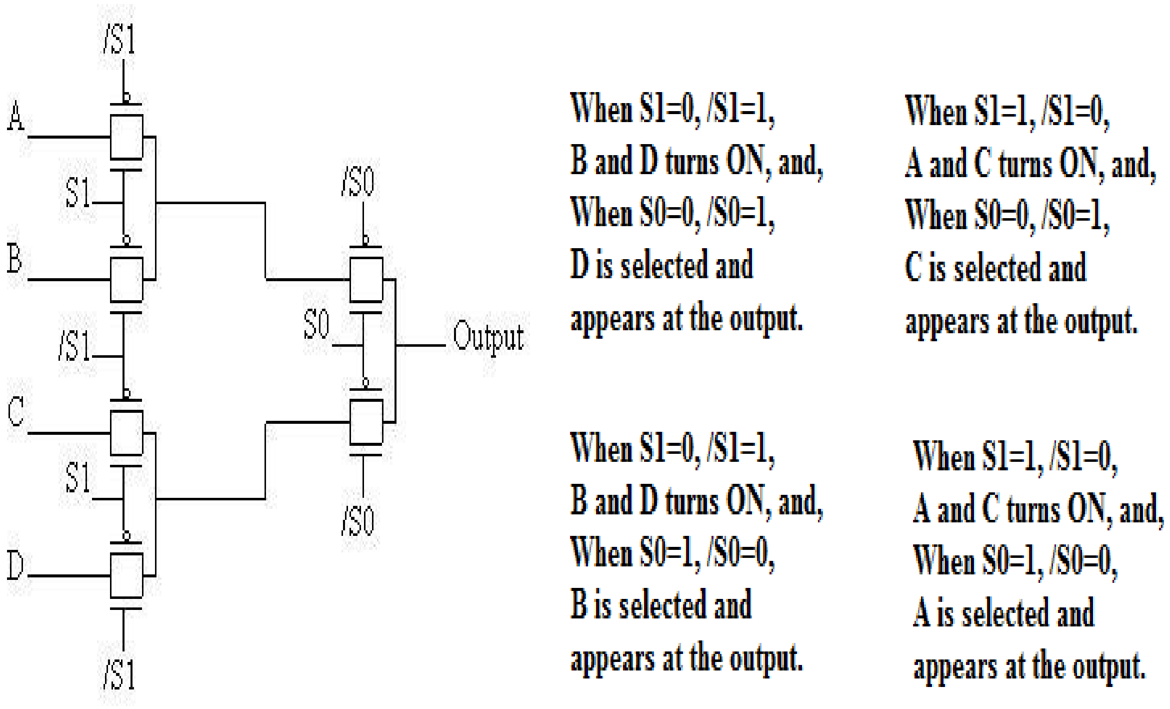
Symbol	Truth Table		
 Transmission Gate	Control	A	B
	1	0	0
	1	1	1
	0	0	Hi-Z
	0	1	Hi-Z
Boolean Expression **B = A.Control**	Read as A **AND** Cont. gives B		

4.1 Transmission gate symbol and its logic

We can see from the above truth table, that the output at B relies not only the logic level of the input A, but also on the logic level present on the control input. Thus the logic level value of B is defined as both A AND Control giving us the boolean expression for a transmission gate of: B = A.Control

4.2 MUX USING TRANSMISSION GATE (TG) LOGIC STYLE

4:1 mux using Transmission gate type is formed when PMOS is connected in front of NMOS and it works as a switch. NMOS devices pass a strong zero but a weak 1, while PMOS pass a strong one on the other hand, a weak 0[11]. In TGL neither transistor is connected to VDD or GND. The transmission gate takes the finest properties from both type of transistor by inserting NMOS in front of PMOS device. Four transmission gates square measure are used to create an MUX structure



When S1=0, /S1=1,
B and D turns ON, and,
When S0=0, /S0=1,
D is selected and
appears at the output.

When S1=1, /S1=0,
A and C turns ON, and,
When S0=0, /S0=1,
C is selected and
appears at the output.

When S1=0, /S1=1,
B and D turns ON, and,
When S0=1, /S0=0,
B is selected and
appears at the output.

When S1=1, /S1=0,
A and C turns ON, and,
When S0=1, /S0=0,
A is selected and
appears at the output.

4.2 . 4:1 mux using transmission gate logic

4.3 COMPARISION TABLE

S. no	**Technique**	**Transistor count**	**Speed**	**Power Dissipation**	**used Area**
1	CMOS	26	HIGH	10.511x10-8	MORE
2	GDI	6	LOW	6.153 x10-8	LOW
3	DPTL	6	LOW	8.907 x10-8	LOW
4	PT	10	HIGH	8.657 x10-8	MORE
5	**TG**	**14**	**HIGH**	**5.158 x10-8**	**LOW**

4.3 COMPARISION TABLE

CHAPTER 5

SOFTWARE USED: TANNER EDA

5.1 TANNER EDA TOOL

Tanner eda is a set of gear for the layout of integrated circuits. That equipment will let you input schematics, carry out spice simulations, and do bodily design (i.e., chip layout), and carry out design rule assessments and format as opposed to schematic exams.

There are 3 tools which can be used for this method: S-edit - a schematic capture tool

T-spice - the spice simulation engine incorporated with s-edit

L-edit - the physical design tool

S-edit is a schematic access tool that is used to record circuits that may be driven forward into a format of an integrated circuit. It also presents the capacity to carry out spice simulations of the circuits the use of a simulation engine called t-spice. T-spice can be setup and invoked from with in S-edit.

Begin a brand new design & setup libraries

a) Begin s-edit: - Begin – all packages – tanner eda – tanner gear v12.6 – s-edit v12.6

b) Begin a new layout: The usage of the pull down menus, create a new layout: - File – new - new design

A conversation will appear soliciting for a design name and vicinity. when you deliver the name, edit Will create a folder of that name in the listing that you offer a good way to include all Of the design documents. You must give a descriptive call that represents each simulation You may be running.

- enter the name “hw03_NMOS_iv_part1” and skim to your

“eele414_vlsi_fall2011tanner_projects” directory

- Click “good enough”

c) Create a brand new cellular a “cellular” is a design element. a cellular can contain more than one perspectives which includes schematics and Three Symbols. Cells can be instantiated in other cells. When appearing a simulation, we are able to generally name the cell “pinnacle”. When we are checking out a circuit, for instance an inverter, the inverter could have its own cell that incorporates a schematic of the gadgets and a symbol. The inverter cellular is instantiated in the top mobile that consists of best factors together with voltage resources and probes which might be simplest used for simulation. this lets in us to split the cells which are truly going to be carried out at the die as opposed to cells which might be most effective used for simulation.

The use of the pull down menus, create a brand new cell view: - Cell – new view: - input the cell name “top”. Make certain the layout name is “hw03_NMOS_iv_part1” and Click on ok. You may go away the interface and think about names “view0”. A clean schematic page will appear. It is a superb idea to store this right now.

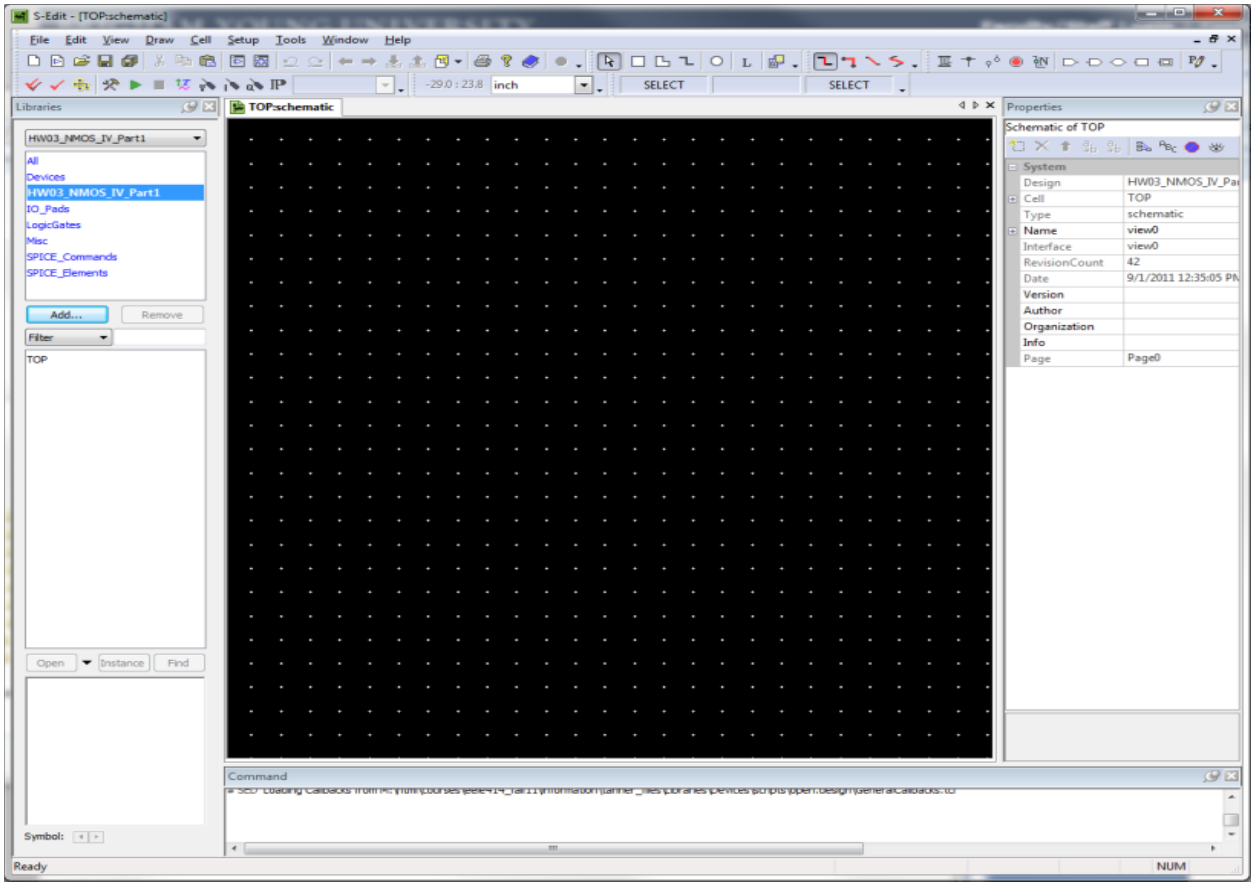
d) Enter the image libraries: First, you want to encompass a library which contains the symbols for all simple circuitFactors together with resistors, NMOS, capacitors, etc… the libraries for all the fundamental Symbols are within the tanner_libraries.zip document you downloaded and unzipped.

- On the left side of the s-edit display you’ll see a libraries window, click on on the “upload”

Button.

- browse to “librariesallall.tanner” and click on “ok”

You should see a set of libraries appear:



HW03_NMOS_IV_Part1

All
Devices
HW03_NMOS_IV_Part1
IO_Pads
LogicGates
Misc
SPICE_Commands
SPICE_Elements

TOP

Schematic of TOP

System	
Design	HW03_NMOS_IV_Pa
Cell	TOP
Type	schematic
Name	view0
Interface	view0
RevisionCount	42
Date	9/1/2011 12:35:05 PN
Version	
Author	
Organization	
Info	
Page	Page0

```
# SED: Loading Callbacks from M:\[illegible]\courses\eele414_fall11\information\tanner_files\Libraries\Devices\scripts\open.design\GeneralCallbacks.tcl
```

Fig 5.1.a.Appearance of Libraries

e) Setup the spice models for the generic_025 package.

The libraries that you simply added have symbols for NMOS and PMOS transistors.

However, all non-linear additives which includes mos transistors require a model to explain

Their conduct. if you actually input an NMOS image on your schematic, spice will not

Recognize what to do on account that each NMOS transistor fabricated in a one-of-a-kind generation behaves in a different way.

In this situation, we are able to use a transistor generation known as “generic_025”, which

Represents a widespread, zero.25um CMOS process. You’ll want to setup the spice fashions

For this process in s-edit. When you try this, while you input an NMOS or PMOS

Transistor, you could then companion the 0.25um version to that image.

The usage of the pull down menus, setup the spice fashions:

- Setup – spice simulation

- In the dialog that looks, you must highlight “standard” at the left.

- On the right, click in the “library files” area. this is wherein you will specify

Any spice models you may be using on your simulations. browse & pick

“generic_025_kit generic_025_spice_models_level1.lib”

- On the proper, click inside the “spice document call” area. This is in which you specify the

Name and vicinity of the spice net list output. Browse to your design listing

“eele414_vlsi_fall2011tanner_projectshw03_NMOS_iv_part1” and input

The filename “top.sp”.

- At the right, click inside the “simulations outcomes record name” field. this is where

The consequences of the simulation could be written. This record is what the waveform

Viewer will search for when you pass to plot your consequences. Browse to your design

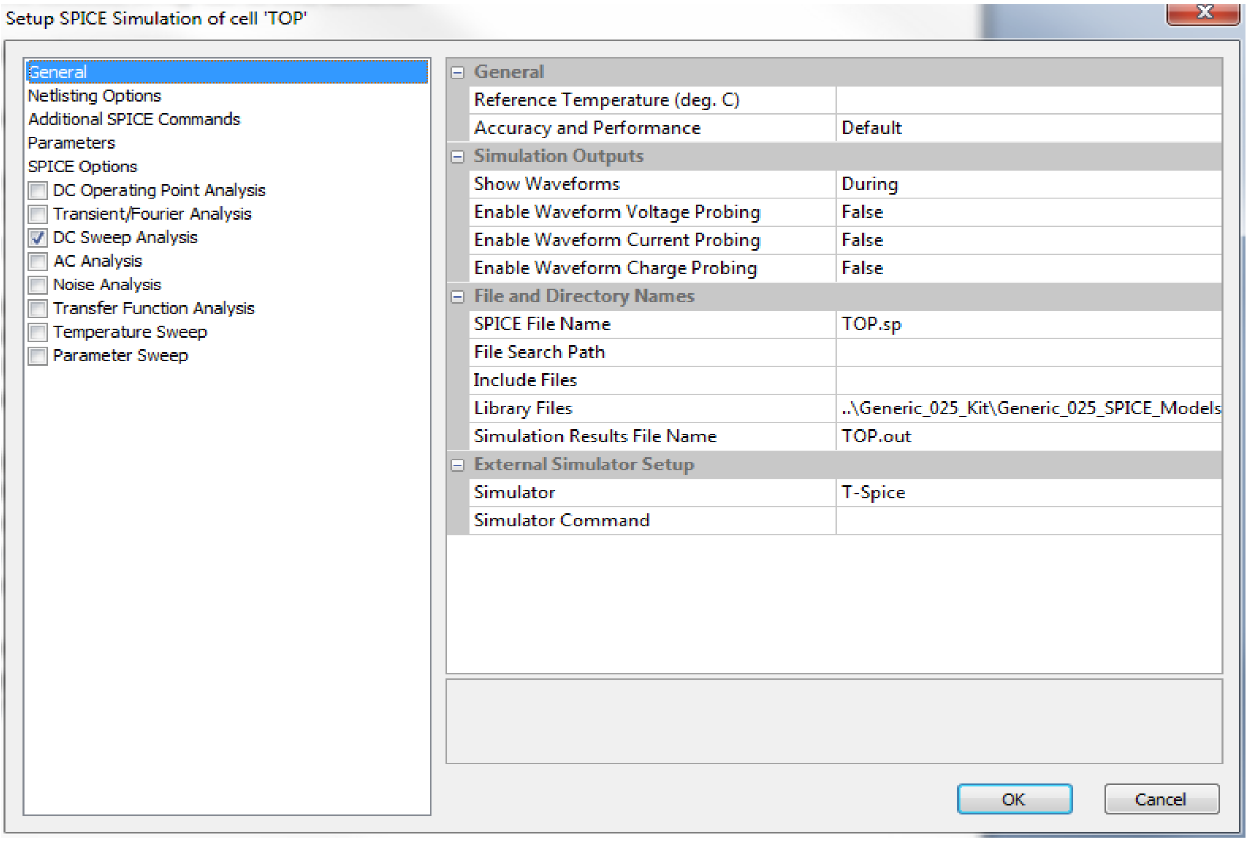
Listing “eele414_vlsi_fall2011tanner_projectshw03_NMOS_iv_part1”

And input the filename “top. Out”.

- Before you could go out this window, you may need to choose an analysis type. we

Will setup the information of the evaluation later, however for now, simply check the “dc sweep

Evaluation” and click “ok” to close the setup window.



Setup SPICE Simulation of cell 'TOP'

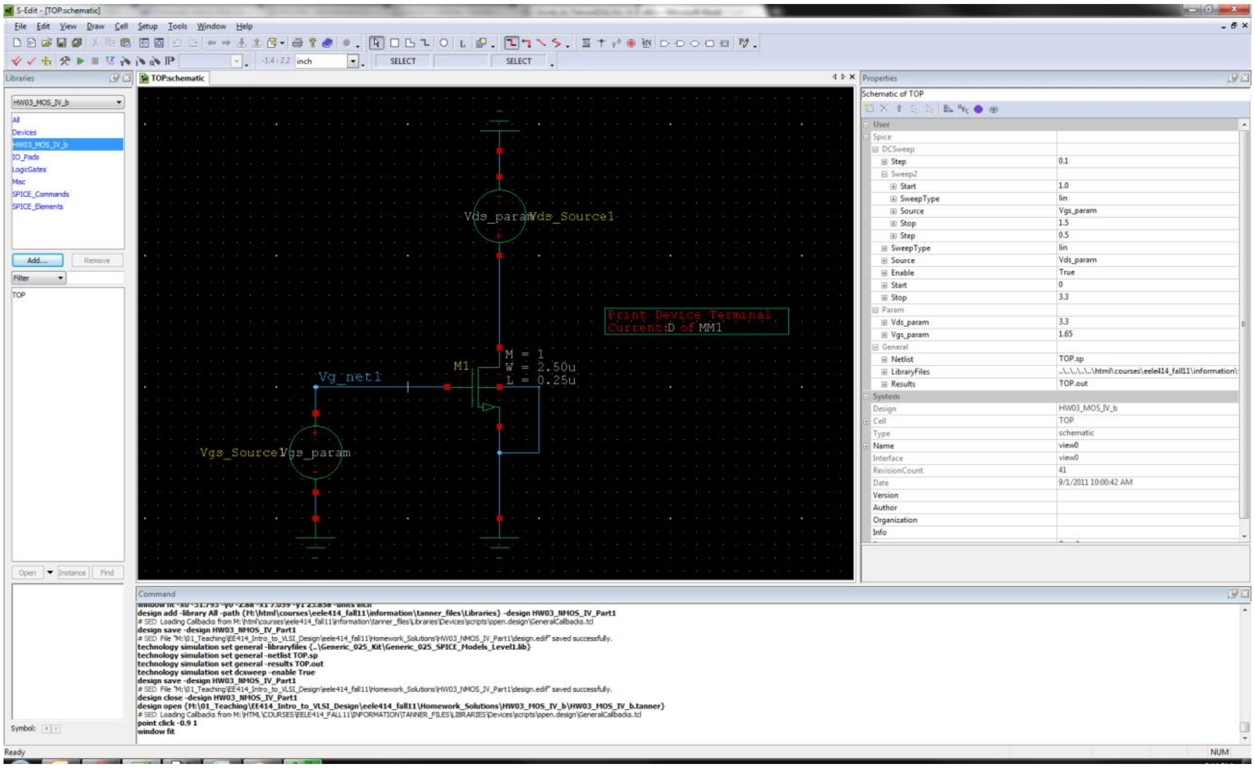
- General
- Netlisting Options
- Additional SPICE Commands
- Parameters
- SPICE Options
- □ DC Operating Point Analysis
- □ Transient/Fourier Analysis
- ☑ DC Sweep Analysis
- □ AC Analysis
- □ Noise Analysis
- □ Transfer Function Analysis
- □ Temperature Sweep
- □ Parameter Sweep

General	
Reference Temperature (deg. C)	
Accuracy and Performance	Default
Simulation Outputs	
Show Waveforms	During
Enable Waveform Voltage Probing	False
Enable Waveform Current Probing	False
Enable Waveform Charge Probing	False
File and Directory Names	
SPICE File Name	TOP.sp
File Search Path	
Include Files	
Library Files	..\Generic_025_Kit\Generic_025_SPICE_Models
Simulation Results File Name	TOP.out
External Simulator Setup	
Simulator	T-Spice
Simulator Command	

Fig 5.2 .Set Up Window

component 3: enter the schematic to simulate the iv behavior of an NMOS transistor

we can be entering the following circuit.



Schematic of TOP

User	
Spice	
DCSweep	
Step	0.1
Sweep2	
Start	1.0
SweepType	lin
Source	Vgs_param
Stop	1.5
Step	0.5
SweepType	lin
Source	Vds_param
Enable	True
Start	0
Stop	3.3
Param	
Vds_param	3.3
Vgs_param	1.65
General	
Netlist	TOP.sp
LibraryFiles	..\..\..\..\..\html\courses\eele414_fall11\information\
Results	TOP.out
System	
Design	HW03_MOS_IV_b
Cell	TOP
Type	schematic
Name	view0
Interface	view0
RevisionCount	41
Date	9/1/2011 10:00:42 AM
Version	
Author	
Organization	
Info	

```
window fit -x0 -31.795 -y0 -2.88 -x1 7.059 -y1 23.858 -units inch
design add -library All -path {M:\html\courses\eele414_fall11\information\tanner_files\Libraries} -design HW03_NMOS_IV_Part1
# SED Loading Callbacks from M:\html\courses\eele414_fall11\information\tanner_files\Libraries\Devices\scripts\open.design\GeneralCallbacks.tcl
design save -design HW03_NMOS_IV_Part1
# SED File "M:\01_Teaching\EE414_Intro_to_VLSI_Design\eele414_fall11\Homework_Solutions\HW03_NMOS_IV_Part1\design.edif" saved successfully.
technology simulation set general -libraryfiles {..\Generic_025_Kit\Generic_025_SPICE_Models_Level1.lib}
technology simulation set general -netlist TOP.sp
technology simulation set general -results TOP.out
technology simulation set dcsweep -enable True
design save -design HW03_NMOS_IV_Part1
# SED File "M:\01_Teaching\EE414_Intro_to_VLSI_Design\eele414_fall11\Homework_Solutions\HW03_NMOS_IV_Part1\design.edif" saved successfully.
design close -design HW03_NMOS_IV_Part1
design open {M:\01_Teaching\EE414_Intro_to_VLSI_Design\eele414_fall11\Homework_Solutions\HW03_MOS_IV_b\HW03_MOS_IV_b.tanner}
# SED Loading Callbacks from M:\HTML\COURSES\EELE414_FALL11\INFORMATION\TANNER_FILES\LIBRARIES\Devices\scripts\open.design\GeneralCallbacks.tcl
point click -0.9 1
window fit
```

Fig 5.3 .Schematic Of Inverter

a) enter the NMOS transistor

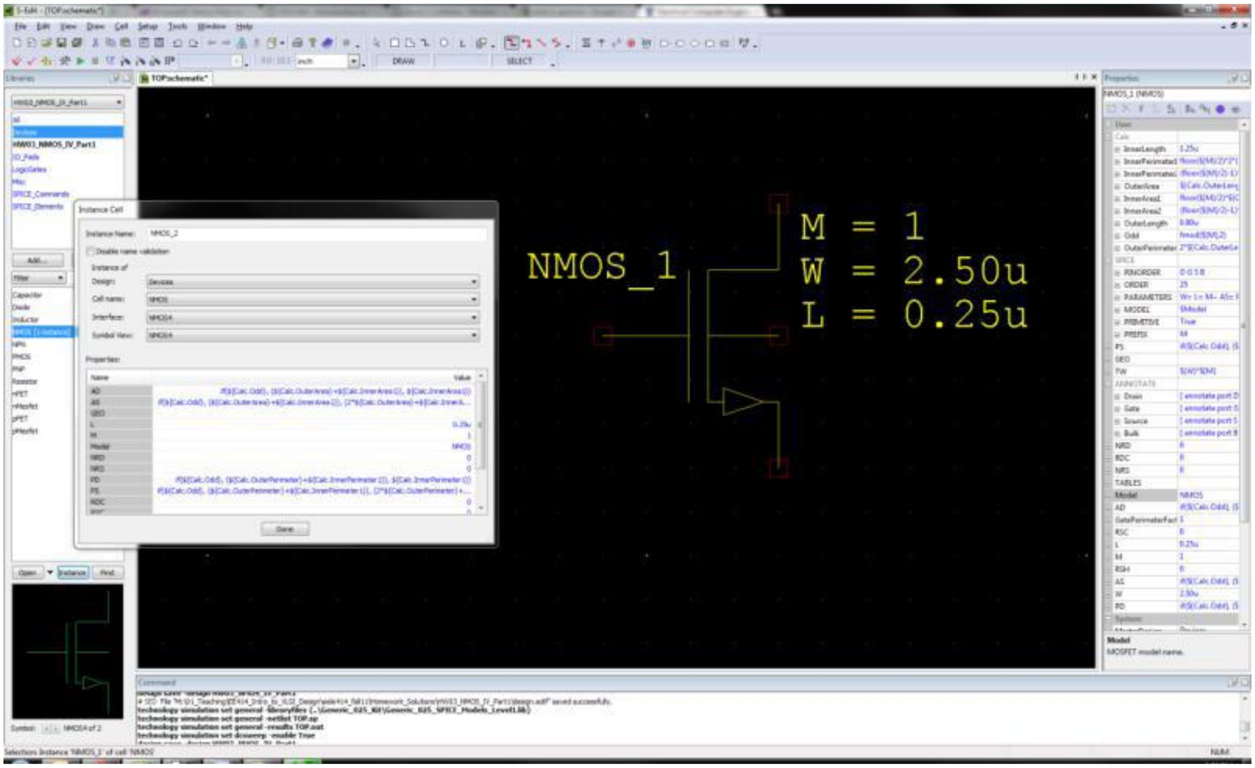
- At the left, click on “gadgets” within the upper window. This will show all the symbols To be had in this group. You ought to see all of the additives that you could put into effect on a CMOS included circuit.

- On the bottom left window, click on once on “NMOS”. You ought to see the image of the NMOS transistor display up inside the symbol viewer window at the lowest.

- To area the NMOS, you'll click at the “instance” button. Things manifest while You click in this button. First, a dialog will seem so that it will assist you to setup the Parameters for the NMOS. Second, the symbol will connect in your mouse. We will region The NMOS in the schematic first after which set its properties later. This is a less difficult way to

Input the device. Click within the schematic window to drop an instance of the NMOS. Hit

The “esc” button to quit the insert-mode.



Instance Name: NMOS_2

Instance of
Design: Devices
Cell name: NMOS
Interface: NMOS4
Symbol View: NMOS4

Name	Value
L	0.25u
M	1
Model	NMOS
NRD	0
NRS	0
RDC	0

```
technology simulation set general -netlist TOP.sp
technology simulation set general -results TOP.out
technology simulation set dcsweep -enable True
```

Model
MOSFET model name.

Fig 5.4 NMOS Schematic

The NMOS is currently in the representation.

A be aware on zooming:

- [Home] = zoom in shape

- [-] = zoom out

- [=] = zoom in

- The scroll wheel also zooms in/out.

- To setup the NMOS, click on the NMOS symbol. You will see the homes of the device on the left. We need to setup the following:

- Model: enter “NMOS”. This model is located in the generic_025 library you brought

- Call: m1. The spice designation for mos transistors is to have the name start with an “m”. S-edit mechanically appends an m to the call is the final call could be “mm1” within the top’s document. However it's far accurate exercise to call all mos transistors with m’s.

- W set to two.5u. This is the default.

- L set to 0.25u. This is the default.

b) Input a dc source for vgs

- The usage of the identical manner you used for the NMOS image, enter “spice_elements: voltagesource”. This is a typical voltage source image this is configured as a dc, tarn, pwl, and so forth.. in its properties conversation.

- click on the voltage source and enter the following:

- Masterinterface: dc (this is the default however this is how you will change it to

Something else.

- Name: vgs_source (it is a good idea to use descriptive names)

- v this is in which you may set the dc voltage (i.e., 4v, 5v).

but, for this example we can use a parameter as a substitute of a hardcoded fee. we are able to enter a parameter name right here and then set up the parameter later. input “vgs_param” for the value of v. while performing a dc sweep, you should use parameters for the sweep.

c) enter a dc source for vds

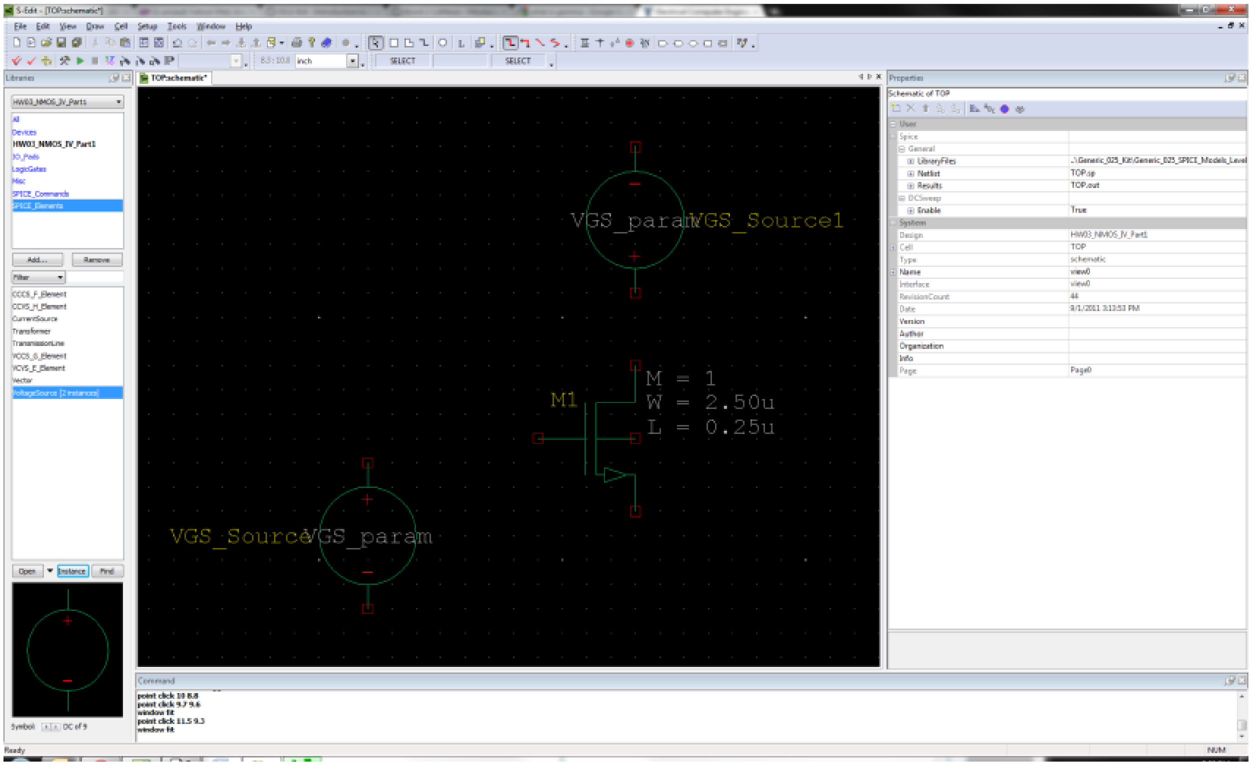
- the usage of the identical method as above, enter a dc source for vgs with the following:

- masterinterface: dc (that is the default but that is how you will alternate it to something else.

- call: vds_source

- v “vds_param

function the sources as inside the following parent:



Schematic of TOP

User	
Spice	
General	
LibraryFiles	..\Generic_025_Kit\Generic_025_SPICE_Models_Level
Netlist	TOP.sp
Results	TOP.out
DCSweep	
Enable	True
System	
Design	HW03_NMOS_IV_Part1
Cell	TOP
Type	schematic
Name	view0
Interface	view0
RevisionCount	44
Date	9/1/2011 3:13:53 PM
Version	
Author	
Organization	
Info	
Page	Page0

Command

```
point click 10 8.8
point click 9.7 9.6
window fit
point click 11.5 9.3
window fit
```

Fig 5.5 Volatge Source Schematic

A be aware on zooming:

- hold down alt-m to transport a issue. even as preserving those buttons

down, click on and drag the components.

- to rotate, click at the tool and click on the [r] button.

d) input grounds

- the usage of the identical manner as above, input 3 grounds from misc:gnd

e) Enter Wires

- You can arrive wires by clicking on the “wire” icon at the top

Enter wires through clicking on a image node after which dragging. input corners through clicking as soon as in which you want to show.

- you could label nets the usage of the “net label” icon on the pinnacle.

f) Input a present day probe to monitor ids

- enter the spice_commands:printcurrent factor. this doesn’t connect to something.

You just vicinity it anywhere after which tell it what current to display in its residences dialog.

- in its houses conversation, setup:

Terminal: d (this is the drain of the NMOS)

Device: mm1 (this is the name of the tool. word that we known as it m1,

However s-edit mechanically appends every other m to the name. You may

Most effective see this once you run the net list.

Evaluation: dc (very essential to choose this!!!!)

Component 4: setup the parameters to be able to be used all through the dc sweep analysis

When we entered the vgs and vds sources, we set their values to “vgs_param” and “vds_param”.

we now need to setup these parameters.

The use of the pull down menus:

- Setup – spice simulations

- at the left, click on “parameters”

- at the right, click on on the “upload parameters” button (it is within the top right nook subsequent to the pink x)

Input: name: vgs_param

Value: 1v

- On the right, click on at the “add parameters” button

Input: name: vds_param

Value: 2.5v

We are able to overwrite those values during our sweep, however the parameters want to exist first.

Element five: setup the spice dc sweep analysis

The usage of the pull down menus:

- Setup – spice simulations

- On the left, click on “dc sweep analysis”

- At the proper, input the following for supply (this is what is going to be swept)

Source or parameter name: vds_param

Begin price: 0

Prevent value: 2.five

Step: 0.1

Sweep type: lin

- On the right, input the following for supply (that is what is going to be swept)

Supply or parameter name: vgs_param

Start value: 1

Forestall price: 1.five

Step: zero.5

Sweep type: line

Notice: the first parameter you setup in this dialog will be plotted at the independent axis.

Part 6: simulate the design

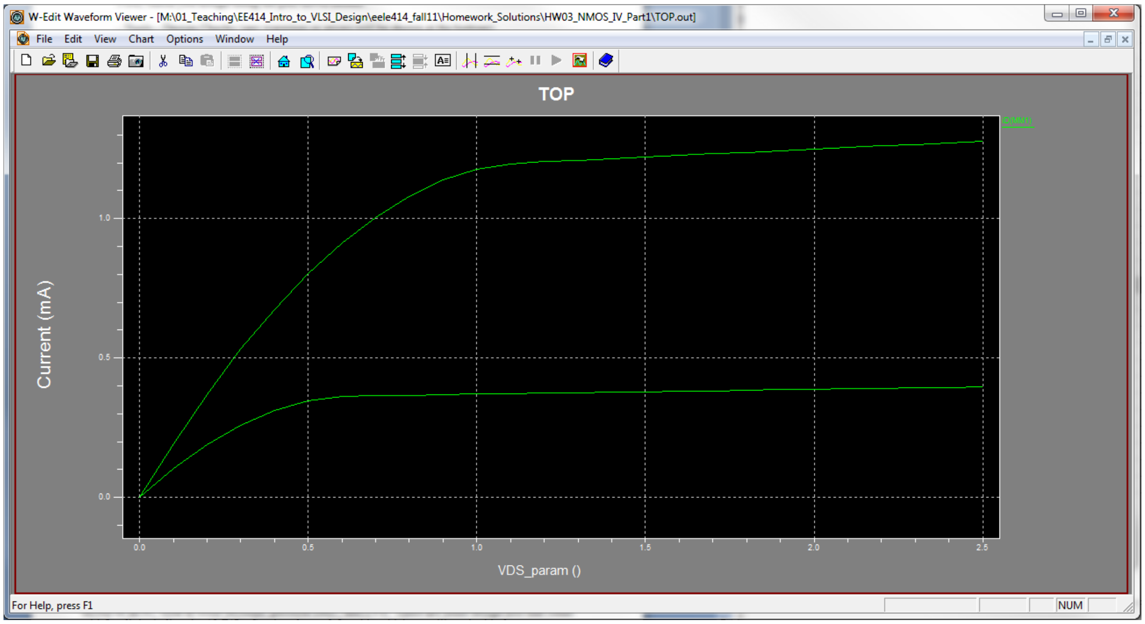
a) First, take a look at you design using the pull down menus:

- Tools – design checks (any warnings or mistakes might be proven at the bottom)

b) Simulate your layout:

- Clock on the green arrow to begin the simulator:

The t-spice window will appear. if the whole lot is adequate, the waveform viewer can even seem. if the entirety worked, your waveforms need to look like this:



W-Edit Waveform Viewer - [M:\01_Teaching\EE414_Intro_to_VLSI_Design\eele414_fall11\Homework_Solutions\HW03_NMOS_IV_Part1\TOP.out]

Fig 5.6. Simulation Output of NMOS

c) view the netlist:

- within the t-spice window, right click on the file at the lowest and select “display netlist” this will deliver up the top.sp netlist that became created and used by the spice engine. This is a superb area to appearance when you get mistakes. this is the text primarily based description of what you entered in s-edit.

d) view the waveform:

- if the windows viewer did now not mechanically appear, you may click on on the record within the tspice window andand pick “display waveform”.

instance: brief analysis of a CMOS inverter & image introduction

part 1: start a brand new layout, setup libraries & setup simulation

a) in this situation, we are able to create a CMOS inverter and simulate its brief reaction. we

will create an inverter layout that contains a image and then instantiate it in every other

schematic to stimulate the circuit.

symbols are handled by means of including every other view to a layout. we can begin by using growing a layout referred to as “inverter” and then create a schematic view. this schematic will incorporate an NMOS and PMOS stressed as an inverter. we will upload “ports” for the enter, output, vdd, and vss. we will then add a symbol view to this layout. the image will incorporate the inverter form and the corresponding pins for input, output, vdd, and vss.

we can then create a separate schematic known as top with the intention to be used to check the inverter. we are able to instantiate the inverter symbol in top. we most effective want to position objects into the inverter design that can be fabricated. top will include the ideal voltage assets to provide the input waveform, the strength resources, and a ridicule load. on this way, when we go into bodily design (i.e., layout), we only power ahead the remoted circuits.

- start s-edit

- create a brand new design called:

“eele414_vlsi_fall2011tanner_projectshw04_inv_transient_part1”

- add the tanner_projectslibrariesallall.tanner library to the library list at the left

- create a brand new cell called “top” the usage of the pull down menus

- cellular - new view

call = pinnacle

view kind = schematic

- setup the simulation the use of the pull down menus:

- setup – spice simulation

- highlight the overall tab of the setup spice window and set the following:

spice file name: hw04_inv_transient_part1top.sp

library files: ..generic_025_kitgeneric_025_spice_models_level1.lib

simulation outcomes report name: document name: hw04_inv_transient_part1top.out

- take a look at the “brief/fourier evaluation” box on the left and set the following:

prevent time = 2ns

most time step = 10ps

- click “ok”

component 2: create the inverter

a) create a new schematic view the use of the pull-down menus:

- cell - new view

call = inverter

view type = schematic

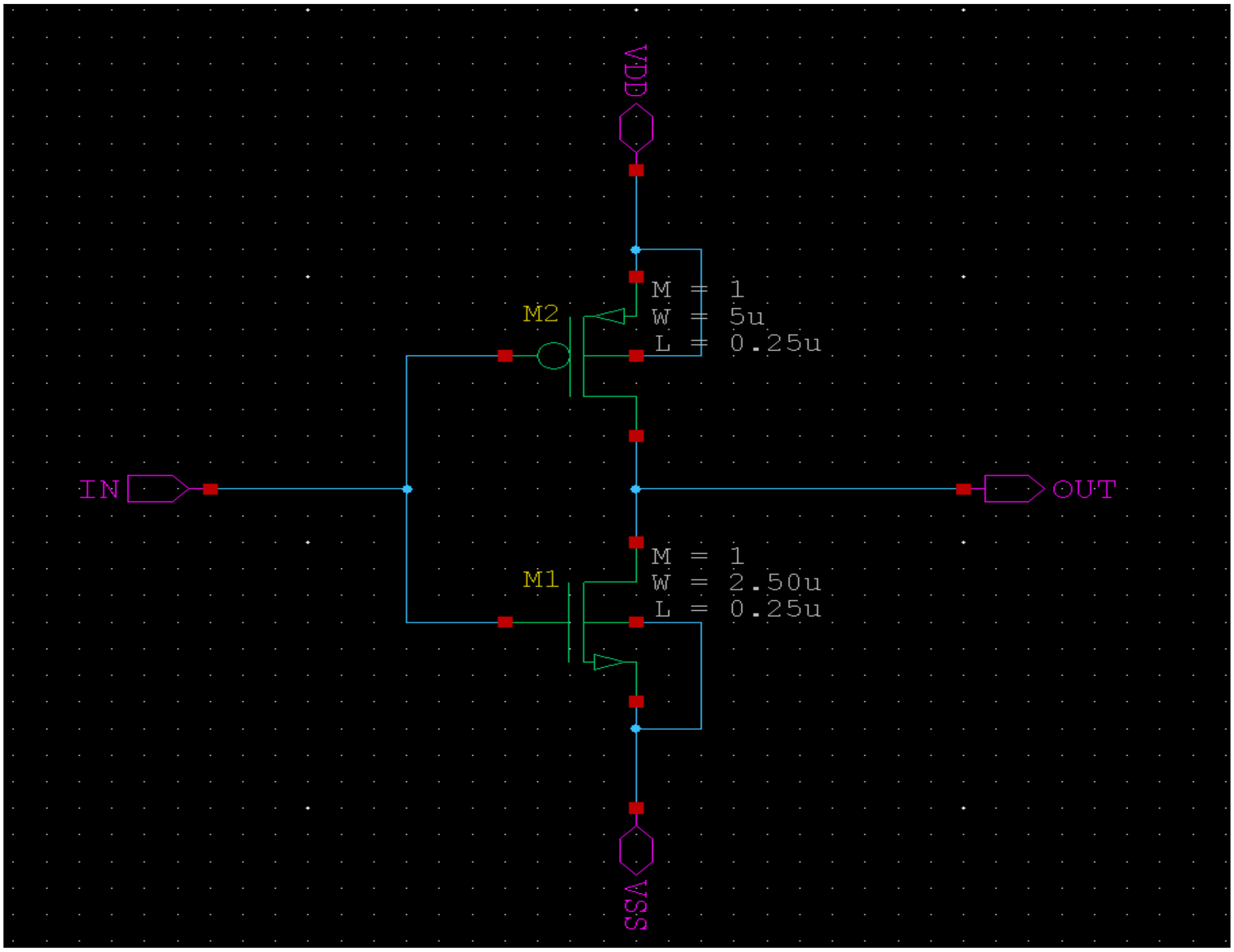


Fig:5.7 Schematic of Inverter

b) enter the inverter schematic:

Entering the NMOS:

Name = M1

L = 0.25u

W = 2.5u

Model =NMOS

- Entering the PMOS:

Name = M2

L = 0.25u

W = 5.0u

Model =PMOS

- Entering the Ports:

Ports are entered using the icons on the top of the S-edit window.

Enter the following:

In Port: Name it “IN”

Out Port: Name it “OUT”

In/Out Port: Name it “VDD”

In/Out Port: Name it “VSS”

wire up the inverter

enter twine connections as proven within the preceding determine.

c) export a spice netlist

exporting a spice netlist is a good idea on the way to verify that you have entered the schematic successfully. additionally, this netlist will be used later when acting a “format versus schematic (lvs)” check. we need to export a netlist at the inverter schematic cell level so that a netlist of just the inverter exists for lvs. while we behavior the simulation of this inverter, we can create a pinnacle degree schematic that will have a netlist containing best voltage sources. this netlist can’t be used for lvs because it includes additives that gained’t be fabricated.

with the schematic open, use the pull down menus to carry out:

- report – export – export spice.

- browse on your design listing and supply the file call “inverter.spc”.
- click “adequate”

In case you open the Inverter.spc with a text editor, you will see the following:

d) create the inverter image

Symbols can either be created manually by using developing a new image view or routinely

by means of s-edit. we can use the automated symbol generation. this can create a brand new symbol

view from the schematic, create the ports for the symbol, and make a symbol form. even as

the form of the image is hardly ever what we in the long run want, it will do plenty of the work for us.

- with the schematic view open, use the pull down menus to create the image view:

mobile – update image

a brand new window will give you a rectangular symbol and four ports with the identical names you entered inside the inverter:schematic view (i.e., in, out, vdd, vss). you should edit the shapes until you have got created a image that seems like an inverter:

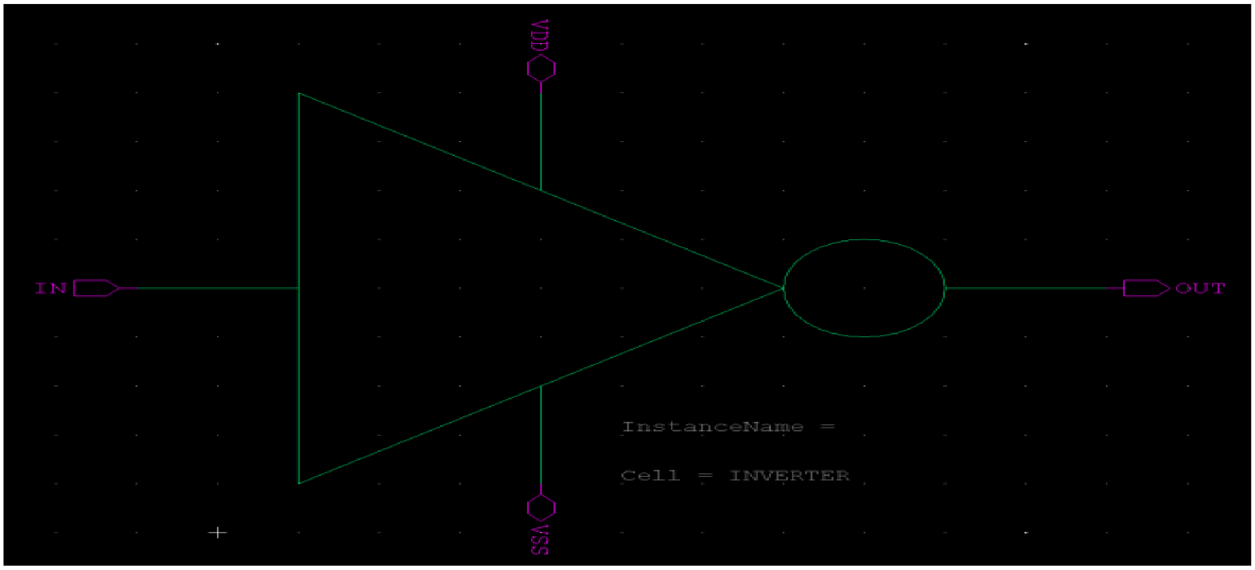


Fig 5.8 . Inverter with in,out,vdd and gnd

a notice on drawing:

The “path” icon will positioned you into a mode in which you can draw strains that aren't wires.

The “circle” icon will can help you input the inversion bubble.

The ports may be moved via keeping down “alt-m”

The ports may be turned around through choosing and pressing the “r” button

Take into account to keep

part 3: create the top schematic to check the inverter

a) instantiate the inverter in the pinnacle schematic

open the pinnacle schematic view the usage of the pull-down menus:

- mobile – open view:

mobile name: top

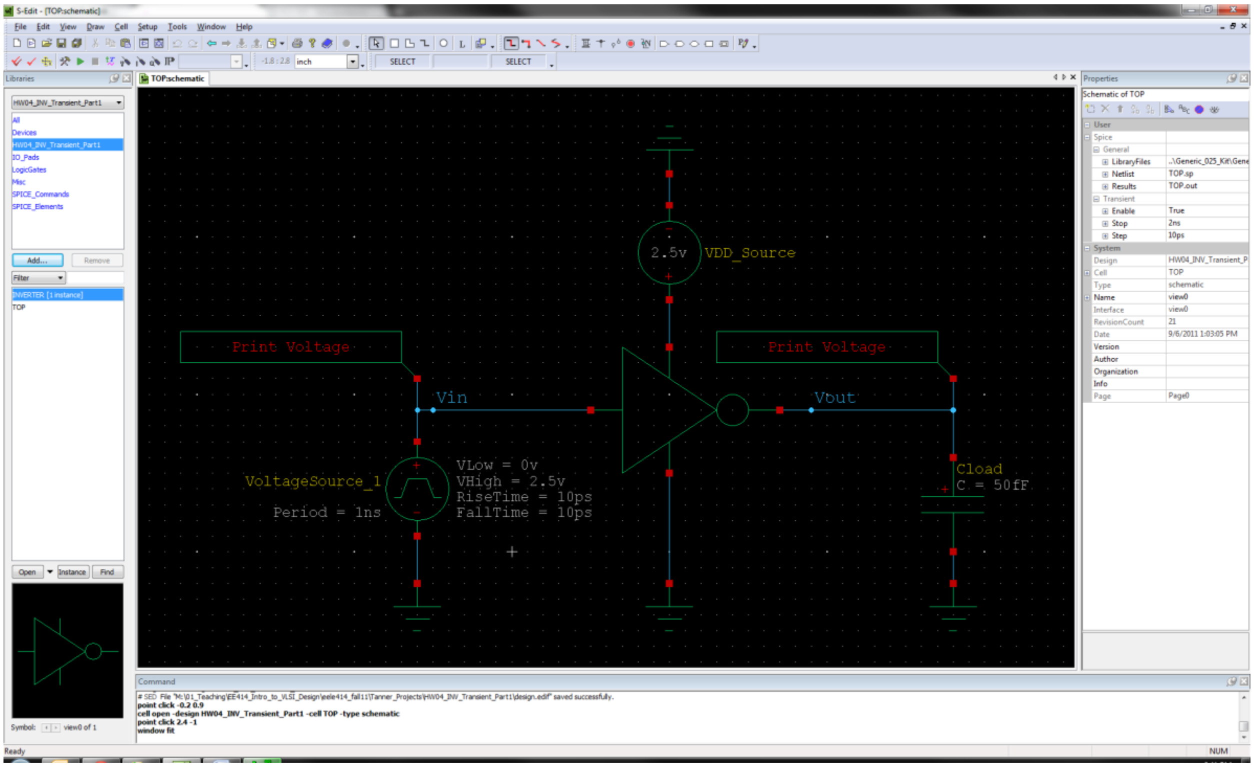
view type: schematic

within the library windows at the left of the window, spotlight your "hw04_inv_transient_part1" library. in the decrease left window, you'll see your two cells "top" and "inverter".

- click on "inverter" and you'll see your symbol display up inside the image viewer.

- click on the "example" button and place your image within the pinnacle schematic

b) enter the following circuit to be able to energy and stimulate your inverter:



Properties

Schematic of TOP

User	
Spice	
General	
LibraryFiles	..\Generic_025_Kit\Gene
Netlist	TOP.sp
Results	TOP.out
Transient	
Enable	True
Stop	2ns
Step	10ps
System	
Design	HW04_INV_Transient_P
Cell	TOP
Type	schematic
Name	view0
Interface	view0
RevisionCount	21
Date	9/6/2011 1:03:05 PM
Version	
Author	
Organization	
Info	
Page	Page0

Command

```
# SED File "M:\01_Teaching\EE414_Intro_to_VLSI_Design\eele414_fall11\Tanner_Projects\HW04_INV_Transient_Part1\design.edif" saved successfully.
point click -0.2 0.9
cell open -design HW04_INV_Transient_Part1 -cell TOP -type schematic
point click 2.4 -1
window fit
```

Fig 5.9 Inverter Schematic

- input the pulse voltage supply. all voltage assets are the same factor inside the spice_elements library. the default is dc, but this may be changed to another type of source inside the homes dialog.

name = vin_source

masterinterface = pulse

duration = 1ns

pulsewidth = zero.5ns

vhigh = 2.5v

vlow = 0v

risetime = 10ps

falltime = 10ps

- enter a load capacitor from the gadgets library.

name = cload

c =50ff

- enter a dc source for vdd

name = vdd_source

masterinterface = dc

v = 2.5v

- enter the grounds from the misc library

- input wire connections and name them vin and vout

- enter a voltage probe for both vin and vout

component 4: simulate the design

a) first, test you layout the use of the pull down menus:

- gear – design checks (any warnings or errors could be shown at the bottom)

b) simulate your layout:

- clock on the inexperienced arrow to start the simulator:

the t-spice window will seem. if the whole thing is adequate, the waveform viewer can even appear. if the entirety labored, your waveforms ought to seem like this:

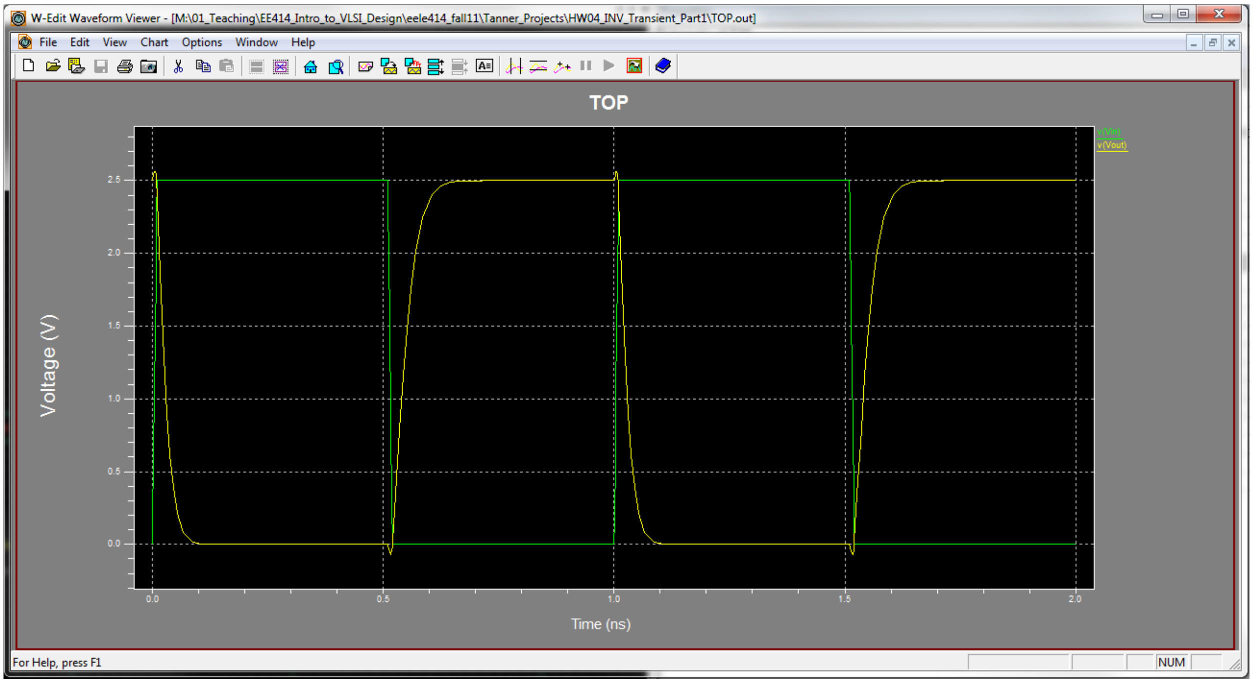

W-Edit Waveform Viewer - [M:\01_Teaching\EE414_Intro_to_VLSI_Design\eele414_fall11\Tanner_Projects\HW04_INV_Transient_Part1\TOP.out]
File
Edit
View
Chart
Options
Window
Help
TOP
v(Vin)
v(Vout)
Voltage (V)
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
Time (ns)
For Help, press F1
NUM

Fig 5.10. Output Simulation Of Inverter

CHAPTER 6

INTRODUCTION TO VLSI

6.1 OBJECTIVE:

Very-large-scale integration (VLSI) is the process of creating integrated circuits by combining thousands of transistor-based circuits into a single chip. VLSI began in the 1970s when complex semiconductor and communication technologies were being developed. The microprocessor is a VLSI device. The term is no longer as common as it once was, as chips have increased in complexity into the hundreds of millions of transistors.

6.2 OVERVIEW:

The first semiconductor chips held one transistor each. Subsequent advances added more and more transistors, and, as a consequence, more individual functions or systems were integrated over time. The first integrated circuits held only a few devices, perhaps as many as ten diodes, transistors, resistors and capacitors, making it possible to fabricate one or more logic gates on a single device. Now known retrospectively as "small-scale integration" (SSI), improvements in technique led to devices with hundreds of logic gates, known as large-scale integration (LSI), i.e. systems with at least a thousand logic gates. Current technology has moved far past this mark and today's microprocessors have many millions of gates and hundreds of millions of individual transistors.

At one time, there was an effort to name and calibrate various levels of large scale integration above VLSI. Terms like Ultra-large-scale Integration (ULSI) were used. But the huge number of gates and transistors available on common devices has rendered such fine distinctions moot.

Terms suggesting greater than VLSI levels of integration are no longer in widespread use. Even VLSI is now somewhat quaint, given the common assumption that all microprocessors are VLSI or better.

As of early 2008, billion-transistor processors are commercially available, an example of which is Intel's Montecito Itanium chip. This is expected to become more commonplace as semiconductor fabrication moves from the current generation of 65 nm processes to the next 45 nm generations (while experiencing new challenges such as increased variation across process corners). Another notable example is NVIDIA’s 280 series GPU

This microprocessor is unique in the fact that its 1.4 Billion transistor count, capable of a teraflop of performance, is almost entirely dedicated to logic (Itanium's transistor count is largely due to the 24MB L3 cache). Current designs, as opposed to the earliest devices, use extensive design automation and automated logic synthesis to lay out the transistors, enabling higher levels of complexity in the resulting logic functionality. Certain high-performance logic blocks like the SRAM cell, however, are still designed by hand to ensure the highest efficiency (sometimes by bending or breaking established design rules to obtain the last bit of performance by trading stability).

6.2.1 WHAT IS VLSI?

VLSI stands for "Very Large Scale Integration". This is the field which involves packing more and more logic devices into smaller and smaller areas.

➢simply we say Integrated circuit is many transistors on one chip.

➢ Design/manufacturing of extremely small, complex circuitry using modified semiconductor material

➢ Integrated circuit (IC) may contain millions of transistors, each a few mm in size

- Applications wide ranging: most electronic logic devices 5.3 History of Scale Integration:

6.3 HISTORY OF SCALE INTEGRATION:

- Late 40s Transistor invented at Bell Labs
- Late 50s First IC (JK-FF by Jack Kirby at TI)
- Early 60s Small Scale Integration (SSI)
- 10s of transistors on a chip
- Late 60s Medium Scale Integration (MSI)
- 100s of transistors on a chip
- Early 70s Large Scale Integration (LSI)
- 1000s of transistor on a chip
- Early 80s VLSI 10,000s of transistors on a chip (later 100,000s & now 1,000,000s)
- Ultra LSI is sometimes used for 1,000,000s
- SSI – Small-Scale Integration (0-102)
- MSI – Medium-Scale Integration (102-104)
- LSI – Large-Scale Integration (103-105)
- VLSI – Very Large-Scale Integration (105-107)
- ULSI – Ultra Large-Scale Integration (>=107)

6.4 ADVANTAGES OF ICS OVER DISCRETE COMPONENTS:

While we will concentrate on integrated circuits, the properties of integrated circuits-what we can and cannot efficiently put in an integrated circuit-largely determine the architecture of the entire system. Integrated circuits improve system characteristics in several critical ways. ICs have three key advantages over digital circuits built from discrete components:

- **Size:** Integrated circuits are much smaller-both transistors and wires are shrunk to micrometer sizes, compared to the millimeter or centimeter scales of discrete components. Small size leads to advantages in speed and power consumption, since smaller components have smaller parasitic resistances, capacitances, and inductances.

➢ **Speed:** Signals can be switched between logic 0 and logic 1 much quicker within a chip than they can between chips. Communication within a chip can occur hundreds of times faster than communication between chips on a printed circuit board. The high speed of circuit's on-chip is due to their small size-smaller components and wires have smaller parasitic capacitances to slow down the signal.

➢ **Power Consumption:** Logic operations within a chip also take much less power. Once again, lower power consumption is largely due to the small size of circuits on the chip-smaller parasitic capacitances and resistances require less power to drive them.

6.5 VLSI AND SYSTEMS:

These advantages of integrated circuits translate into advantages at the system level:

➢smaller physical size. Smallness is often an advantage in itself-consider portable televisions or handheld cellular telephones.

➢ Lower power consumption. Replacing a handful of standard parts with a single chip reduces total power consumption. Reducing power consumption has a ripple effect on the rest of the system: a smaller, cheaper power supply can be used; since less power consumption means less heat, a fan may no longer be necessary; a simpler cabinet with less shielding for electromagnetic shielding may be feasible, too.

➢ Reduced cost. Reducing the number of components, the power supply requirements, cabinet costs, and so on, will inevitably reduce system cost. The ripple effect of integration is such that the cost of a system built from custom ICs can be less, even though the individual ICs cost more than the standard parts they replace.

- Understanding why integrated circuit technology has such profound influence on

The design of digital systems requires understanding both the technology of IC manufacturing and the economics of ICs and digital systems

APPLICATIONS:

- Electronic system in cars.
- Digital electronics control VCRs
- Transaction processing system, ATM
- Personal computers and Workstations
- Medical electronic systems.
- Etc….

6.6 APPLICATIONS OF VLSI:

Electronic systems now perform a wide variety of tasks in daily life. Electronic systems in some cases have replaced mechanisms that operated mechanically, hydraulically, or by other means; electronics are usually smaller, more flexible, and easier to service. In other cases electronic systems have created totally new applications. Electronic systems perform a variety of tasks some of them visible, some more hidden:

Personal entertainment systems such as portable MP3 players and DVD players perform sophisticated algorithms with remarkably little energy.

Electronic systems in cars operate stereo systems and displays; they also control fuel injection systems, adjust suspensions to varying terrain, and perform the control functions required for anti-lock braking (ABS) systems.

Digital electronics compress and decompress video, even at high-definition data rates, on-the-fly in consumer electronics.

Low-cost terminals for Web browsing still require sophisticated electronics, despite their dedicated function

Personal computers and workstations provide word-processing, financial analysis, and games. Computers include both central processing units (CPUs) and special-purpose hardware for disk access, faster screen display, etc.

Medical electronic systems measure bodily functions and perform complex processing algorithms to warn about unusual conditions. The availability of these complex systems, far from overwhelming consumers, only creates demand for even more complex systems.

The growing sophistication of applications continually pushes the design and manufacturing of integrated circuits and electronic systems to new levels of complexity. And perhaps the most amazing characteristic of this collection of systems is its variety-as systems become more complex, we build not a few general-purpose computers but an ever wider range of special-purpose systems. Our ability to do so isa testament to our growing mastery of both integrated circuit manufacturing and design, but the increasing demands of customers continue to test the limits of design and manufacturing

6.7 ASIC:

An Application-Specific Integrated Circuit (ASIC) is an integrated circuit (IC) customized for a particular use, rather than intended for general-purpose use. For example, a chip designed solely to run a cell phone is an ASIC. Intermediate between ASICs and industry standard integrated circuits, like the 7400 or the 4000 series, are application specific standard products (ASSPs).

As feature sizes have shrunk and design tools improved over the years, the maximum complexity (and hence functionality) possible in an ASIC has grown from 5,000 gates to over 100 million. Modern ASICs often include entire 32-bit processors, memory blocks including ROM, RAM, EEPROM, Flash and other large building blocks. Such an ASIC is often termed a SOC (system-on-a-chip). Designers of digital ASICs use a hardware description language (HDL), such as Verilog or VHDL, to describe the functionality of ASICs.

Field-programmable gate arrays (FPGA) are the modern-day technology for building a breadboard or prototype from standard parts; programmable logic blocks and programmable interconnects allow the same FPGA to be used in many different applications. For smaller designs and/or lower production volumes, FPGAs may be more cost effective than an ASIC design even in production.

➢ An application-specific integrated circuit (ASIC) is an integrated circuit (IC) customized for a particular use, rather than intended for general-purpose use

. ➢ A Structured ASIC falls between an FPGA and a Standard Cell-based ASIC

➢ Structured ASIC’s are used mainly for mid-volume level design. The design task for structured ASIC’s is to map the circuit into a fixed arrangement of known sells.

CHAPTER 7

APPLICATIONS

APPLICATIONS OF MULTIPLEXERS

A Multiplexer is used in various applications wherein multiple data can be transmitted using a single line.

COMMUNICATION SYSTEM

A Multiplexer is used in communication systems, which has a transmission system and also a communication network. A Multiplexer is used to increase the efficiency of the communication system by allowing the transmission of data, such as audio & video data from different channels via cables and single lines.

COMPUTER MEMORY

A Multiplexer is used in computer memory to keep up a vast amount of memory in the computers, and also to decrease the number of copper lines necessary to connect the memory to other parts of the computer.

Telephone Network

A multiplexer is used in telephone networks to integrate the multiple audio signals on a single line of transmission.

Transmission from the Computer System of a Satellite

A Multiplexer is used to transmit the data signals from the computer system of a satellite to the ground system by using a GSM communication.

CHAPTER 8

CONCLUSION

In this project, the digital circuit 4X1 mux was implemented by t low power technique Transmission Gate. The results were simulated using tanner EDA and comparison has been done for different parameters like power dissipation, speed, area and transistor count. The results concluded that as compared to other proposed techniques, CMOS has more power dissipation and transistor count. These advantages of proposed techniques **TRANSMISSION GATE** make them more efficient and convenient to be used in digital circuits

REFERENCES

[1] Neil H. E. Weste and David Money Harris, "CMOS VLSI Design A Circuits and System Perspective", Pearson Education (Asia) Pvt. Ltd.2000.

[2] J. M. Rabaey, A. Chandrakasan, and B. Nicolic, Digital Integrated Circuits: A Design Perspective, 2nd ed. Upper Saddle River,NJ:
Prentice-Hall, 2003

[3] M. Padmaja, V.N.V. Satya Prakash, "Design of multiplexer in multiple logic styles for low power VLSI", International Journal of Computer Trends and Technology – volume 3 Issue 3 – 2012.

[4] Gurjeet Kaur, Gurmohan Singh, "Analysis of low power CMOS current comparison domino logic circuits in ultra deep submicron technologies," International Journal of Computer Applications (0975 – 8887), February 2014, Volume 88 –No.7.

[5] Purnima Kumari Sharma, Neeraj Kumar Singh, "Power comparison of single and dual rail 2:1 MUX design at different levels of technology," IEEE Conference, 2014, pp.671-775.

[6] S. Abirami, M. Arul Kumar, E.Abinaya, J.Sowmaya, "Design and analysis of 2:1 multiplexer circuit for high performance," International Journal of Electrical and Electronics Engineers, Vol. No. 7, Issue No. 01, Jan-June,2015

[7] Priti Gupta , Rajesh Mehra, "Layout design and simulation of CMOS multiplexer," International Journal of Scientific Research Engineering &Technology(IJSRET),14-15March,2015

[8] Mohit Vyas, Soumya Kanti Manna, Shyam Akashe, "Design of Power efficient MUX using dual gate FinFET technology," IEEE Conference, 2016.

[9] Ila Gupta, Nehra Arora, Prof.B.P.Singh, "Simulation and analysis of 2:1 multiplexer circuits at 90nm technology", International Journal of Modern Engineering Research (IJMER) – volume 1 Issue.2,pp-642-646