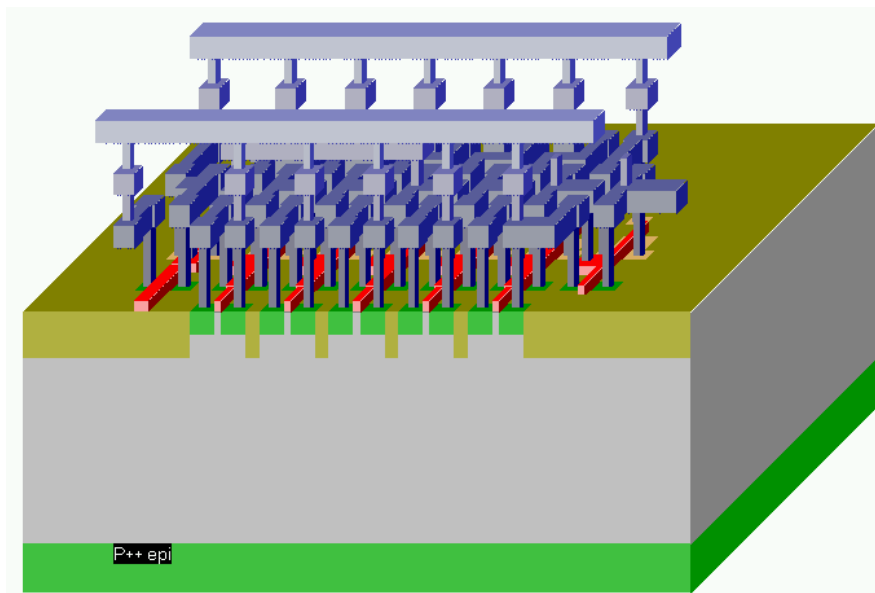


Microwind & Dsch User's Manual

Version 2.7



November 2003

Etienne Sicard

<http://www.microwind.org>

About the author

ETIENNE SICARD was born in Paris, France, in June 1961. He received a B.S degree in 1984 and a PhD in Electrical Engineering in 1987 both from the University of Toulouse. He was granted a Monbusho scholarship and stayed 18 months at the University of Osaka, Japan. Previously a professor of electronics in the department of physics, at the University of Balearic Islands, Spain, E. Sicard is currently a professor at the INSA Electronic Engineering School of Toulouse. His research interests include several aspects of design of integrated circuits including crosstalk fault tolerance, and electromagnetic compatibility of integrated circuits. Etienne SICARD is the author of several commercial software in the field of microelectronics and sound processing.

Copyright

© Copyright 1997-2004 by INSA

Address

Etienne Sicard
INSA-DGEI, 135, Av de Ranguueil
31077 TOULOUSE Cedex 4, FRANCE
Tel : +33.561.55.98.42, Fax: +33.561.55.98.00
e-mail: etienne.sicard@insa-tlse.fr
home page: <http://intrade.insa-tlse.fr/~etienne>

Web information

<http://www.microwind.org>

Acknowledgements

Special thanks are due to technical contributors to the Dsch and Microwind software, to numerous professors, students and engineers who patiently debugged the software, gave valuable comments and suggestions. I would like to thank several people at INSA, especially Sonia Bendhia and Chen Xi, Jean-Pierre Schoellkopf, Fabrizio Battaglia at ST-Microelectronics, and Bernard Courtois, Tima Grenoble, for their help, support and contributions. Also, I would like to thank Marie-Agnes Detourbe for having carefully reviewed the manuscript. This software is dedicated to John Uyemura, from Georgia Tech, Usa, who was an unconditional fan of Microwind.

ISBN 2-87649-046-3

Table of Contents

1	Introduction & Installation.....	7
	INSTALLATION.....	8
2	Technology Scale Down.....	9
2.1	Evolution of Microprocessors and Memories.....	9
2.2	Frequency Improvements.....	10
2.3	Increased Layers.....	11
2.4	Design Trends.....	12
2.5	Exercises.....	12
3	The MOS device.....	13
3.1	Logic Levels.....	13
3.2	The MOS as a switch.....	13
3.3	Logic Simulation of the MOS.....	14
3.4	MOS layout.....	14
3.5	Vertical aspect of the MOS.....	16
3.6	Static Mos Characteristics.....	16
3.7	Dynamic MOS behavior.....	17
3.8	Analog Simulation.....	18
3.9	Layout considerations.....	19
3.10	The MOS Model 1.....	20
3.11	The MOS Model 3.....	21
3.12	The BSIM4 MOS Model.....	23
3.13	Low leakage MOS.....	25
3.14	High voltage MOS.....	26
3.15	Temperature effects on the MOS.....	27
3.16	The PMOS Transistor.....	29
3.17	Process Variations.....	30
3.18	The Transmission Gate.....	31
4	The Inverter.....	33
4.1	The Logic Inverter.....	33
4.2	THE CMOS INVERTER.....	34
4.3	MANUAL LAYOUT OF THE INVERTER.....	35
4.4	Connection between Devices.....	35
4.5	Useful Editing Tools.....	36
4.6	Metal-to-poly.....	37
4.7	Supply Connections.....	38
4.8	Process steps to build the Inverter.....	39
4.9	Inverter Simulation.....	40
4.10	3-STATE INVERTER.....	42
4.11	Exercises.....	42
5	Basic Gates.....	43
5.1	Introduction.....	43
5.2	The Nand Gate.....	43
5.3	The AND gate.....	45

5.4	The 3-Input OR Gate	46
5.5	The XOR Gate	47
5.6	Complex Gates.....	48
5.7	Multiplexor	51
5.8	8 to 1 Multiplexor	52
5.9	Interconnects and Vias.....	52
5.10	Exercises.....	55
6	<i>Arithmetics</i>.....	56
6.1	Unsigned Integer format	56
6.2	Creating Arithmetic Circuits From Logic Design	56
6.3	Half-Adder Gate	57
6.4	Full-Adder Gate.....	59
6.5	Full-Adder Symbol in DSCH	60
6.6	Full-Adder Layout	61
6.7	Four-Bit Adder	62
6.8	Comparator	64
6.9	Arithmetic and Logic Unit.....	65
6.10	Model of the PIC 16f84	68
7	<i>Latches</i>.....	70
7.1	Basic Latch	70
7.2	RS Latch	70
7.3	D Latch	73
7.4	Edge Trigged Latch	75
7.5	Counter	78
8	<i>Memory Circuits</i>.....	79
8.1	The world of Memory.....	79
8.2	RAM Memory	80
8.3	RAM Array	84
8.4	Row Selection Circuit.....	85
8.5	Column Selection Circuit	86
8.6	A Complete 64 bit SRAM	87
8.7	Dynamic RAM Memory	88
8.8	EEPROM	91
8.8.1	Double-gate MOS Charge.....	92
8.8.2	Double-gate MOS Discharge	93
8.9	Flash Memories	95
8.10	Ferroelectric RAM memories	96
8.11	Memory Interface	99
8.12	EXERCISES	101
9	<i>Analog Cells</i>.....	102
9.1	Resistor	102
9.2	Capacitor.....	104
9.2.1	Diode Capacitor.....	105
9.3	Mos Capacitor.....	106
9.4	Poly-Poly2 Capacitor.....	107
9.5	Inter-Metal Capacitor.....	108
9.6	Diode-connected MOS	108
9.7	Voltage Reference	110
9.8	Current mirror.....	112
9.9	Amplifier	114
9.10	Simple Differential Amplifier.....	117

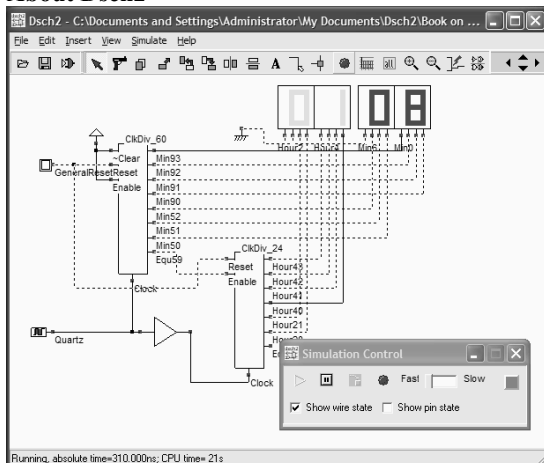
9.11	Push-Pull Amplifier	119
9.12	Exercises	121
10	<i>Radio Frequency Circuits</i>	123
10.1	On-Chip Inductors	123
10.2	Inductor Design in Microwind.....	124
10.3	Power Amplifier	126
10.4	Oscillator	129
10.5	Voltage Controlled Oscillator.....	132
10.6	Phase-lock-loop	134
10.6.1	Phase Detector	136
10.6.2	Filter.....	137
10.6.3	Voltage controlled oscillator for PLL	138
10.6.4	Complete Phase Lock Loop	138
10.7	Gilbert Mixer	140
10.8	Exercises.....	142
11	<i>Converters</i>	143
11.1	Introduction	143
11.2	Digital-Analog Converters architectures	143
11.2.1	Resistor string converter	144
11.2.2	R-2R ladder converter.....	147
11.3	Sample and Hold circuits.....	148
11.4	Analog-Digital Converters architectures	149
11.4.1	The Flash converter Principles.....	150
11.4.2	Low speed ADC Converters	152
11.5	Exercises.....	153
12	<i>Input/Output Interfacing</i>	154
12.1	Power Supply.....	154
12.2	The Bonding Pad	155
12.3	The Pad ring.....	155
12.4	The supply rails	156
12.5	Input Structures.....	157
12.6	High voltage MOS.....	160
12.7	Input pad with Schmitt Trigger.....	161
12.8	Digital Output Structures.....	164
12.8.1	Level shifter	165
12.9	CORE/PAD LIMITATION	166
12.10	I/O Pad description using Ibis.....	168
13	<i>Design Rules</i>	170
13.1	Select a Design Rule File.....	170
13.2	Start Microwind with a specific design Rule File.....	170
13.3	Lambda Units.....	170
13.4	N-Well	171
13.5	Diffusion.....	171
13.6	Polysilicon	172
13.7	2 nd Polysilicon Design Rules	172
13.8	MOS option	172
13.9	Contact.....	173
13.10	Metal 1	173

13.11	Via	173
13.12	Metal 2	173
13.13	Via 2	173
13.14	Metal 3	174
13.15	Via 3	174
13.16	Metal 4	174
13.17	Via 4	174
13.18	Metal 5	174
13.19	Via 5	174
13.20	Metal 6	175
13.21	Pads	175
13.22	Electrical Extraction Principles	175
13.23	Node Capacitance extraction	176
13.24	SURFACE CAPACITANCE	176
13.25	INTER-LAYER CROSSTALK CAPACITANCE	177
13.26	LATERAL CROSSTALK CAPACITANCE	177
13.27	Parameters for Vertical Aspect of the Technology	177
13.28	Resistance Extraction	178
13.29	Dielectrics	179
13.30	Simulation Parameters	179
13.31	Models Level1 and Level3 for analog simulation	180
13.32	BSIM4 Model for analog simulation	181
13.33	Technology files for DSCH2	182
13.34	Design Rule File	182
14	<i>Microwind2 Menus</i>	184
14.1	FILE MENU	184
14.2	VIEW MENU	184
14.3	EDIT MENU	184
14.4	SIMULATE MENU	185
14.5	COMPILE MENU	185
14.6	ANALYSIS MENU	185
14.7	PALETTE	185
14.8	NAVIGATOR WINDOW	186
14.9	LIST OF ICONS	187
14.10	Microwind2 Simulation menu	188
14.11	Dsch2 Menus	189
14.12	Edit Menu	189
14.13	Insert Menu	189
14.14	View Menu	190
14.15	Simulate Menu	190
14.16	Symbol Palette	190
14.17	List of Files	191
14.18	List of Measurement Files	192
14.19	Measurement file example	192
15	<i>References</i>	194

1 Introduction & Installation

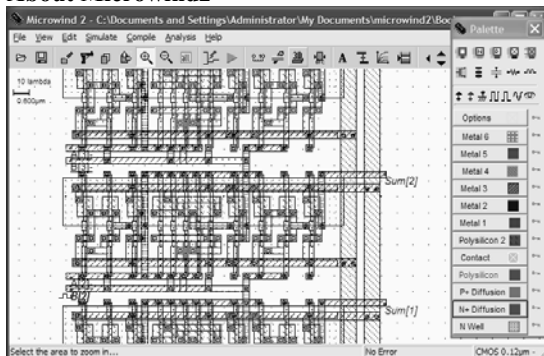
The present manual introduces the design and simulation of CMOS integrated circuits, in an attractive way thanks to user-friendly PC tools Dsch2 and Microwind2.

About Dsch2



The DSCH2 program is a logic editor and simulator. DSCH2 is used to validate the architecture of the logic circuit before the microelectronics design is started. DSCH2 provides a user-friendly environment for hierarchical logic design, and fast simulation with delay analysis, which allows the design and validation of complex logic structures. Some techniques for low power design are described in the manual. DSCH also features the symbols, models and assembly support for 8051 and 18f64. DSCH also includes an interface to SPICE.

About Microwind2



The MICROWIND2 program allows the student to design and simulate an integrated circuit at physical description level. The package contains a library of common logic and analog ICs to view and simulate. MICROWIND2 includes all the commands for a mask editor as well as original tools never gathered before in a single module (2D and 3D process view, VERILOG compiler, tutorial on MOS devices). You can gain access to *Circuit Simulation* by pressing one single key. The electric extraction of your circuit is automatically performed and the analog simulator produces voltage and current curves immediately.

The chapters of this manual have been summarized below. Chapter 2 describes the technology scale down and the major improvements given by deep sub-micron technologies. Chapter 3 is dedicated to the presentation of the single MOS device, with details on the device modeling, simulation at logic and layout levels. Chapter 4 presents the CMOS Inverter, the 2D and 3D views, the comparative design in micron and deep-submicron technologies. Chapter 5 concerns the basic logic gates (AND, OR, XOR, complex gates), Chapter 6 the arithmetic functions (Adder, comparator, multiplier, ALU). The latches and memories are detailed in Chapter 7.

As for Chapter 8, analog cells are presented, including voltage references, current mirrors, operational amplifiers and phase lock loops. Chapter 9 concerns analog-to-digital and digital to analog converter principles. Chapter 10 is dedicated to radio-frequency circuit. The input/output interfacing principles are illustrated in Chapter 11.

The detailed explanation of the design rules is in Chapter 12. Electrical rules are described in chapter 13. The program operation and the details of all commands are given in the help files of the programs.

INSTALLATION

From The web

Connect to page <http://www.microwind.org>

Click "Introduction to microelectronics"

- ◆ Click "Download MICROWIND2 (ZIP file)". In your PC, create manually a directory (Suggested : c:\program files\microwind2). Store the ZIP file in this directory.
- ◆ Extract all files in the selected directory
- ◆ *Test: double-click MICROWIND2.EXE. Click "File ->Load", select "CMOS.msk". Click "Simulate".*

- ◆ Click "Download DSCH2 (ZIP file)". In your PC, create manually a directory (Suggested : c:\program files\dsch2). Store the ZIP file in this directory.
- ◆ Extract all files in the selected directory.
- ◆ *Test: double click in DSCH2.EXE. Load "base.sch". Click "Simulate".*

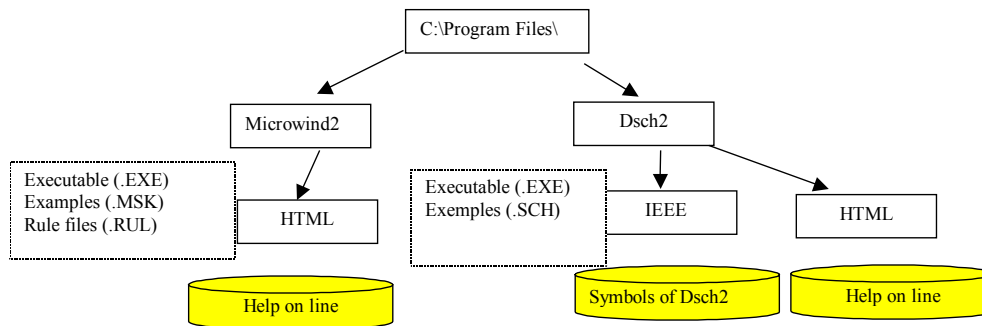


Figure 1: The architecture of Microwind and Dsch

Once installed, two directories are created, one for Microwind, one for Dsch. In each directory, a sub-directory called `html` contains help files.

2 Technology Scale Down

The evolution of integrated circuit (IC) fabrication techniques is a unique fact in the history of modern industry. The improvements in terms of speed, density and cost have keep constant for more than 30 years. By the end of 2004, "System-on-Chips" with about 300,000,000 transistors will be fabricated on a single piece of silicon no larger than 2x2 cm. In this chapter, we present some information illustrating the technology scale down.

2.1 Evolution of Microprocessors and Memories

Figure 2-1 describes the evolution of Intel ® microprocessors, figure 2-2 describes the evolution of memory size during the last decades. In figure 3, it is shown that industry has started to produce ICs in deep submicron technology starting 1995. Research has always kept around 5 years ahead mass production.

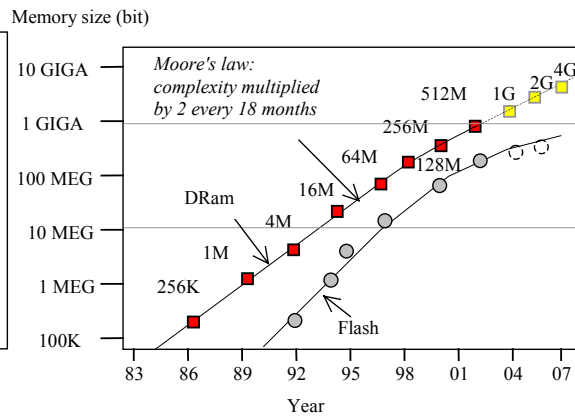
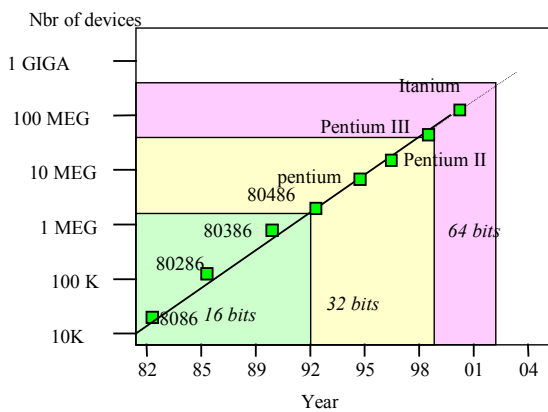


Fig.2-1: Evolution of microprocessors

Fig.2-2: Evolution of memories

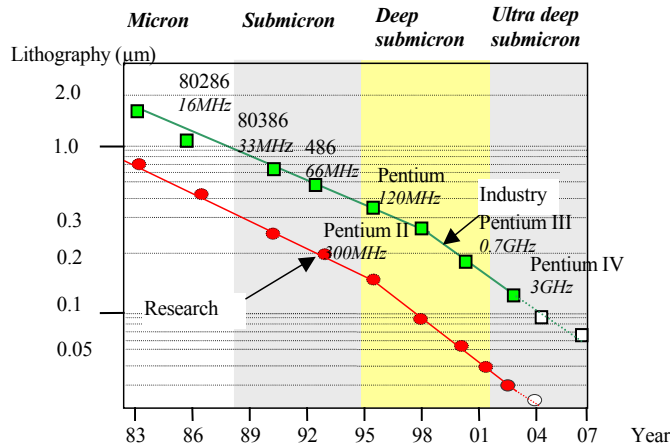


Fig. 2-3: Evolution of lithography

2.2 Frequency Improvements

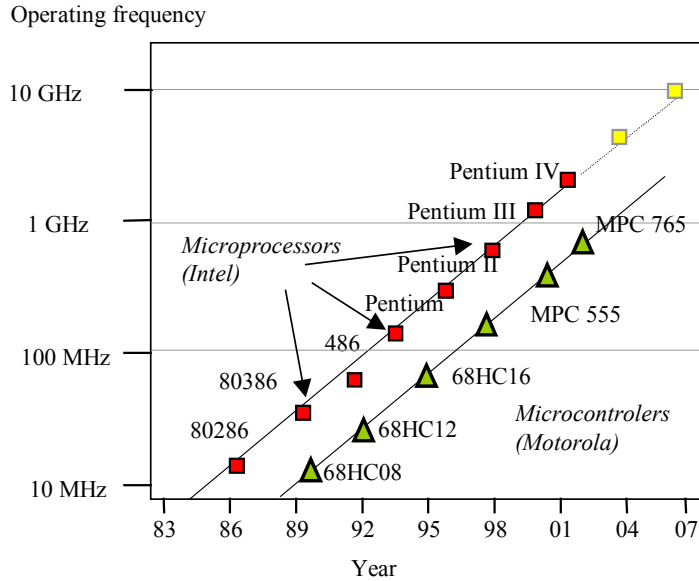
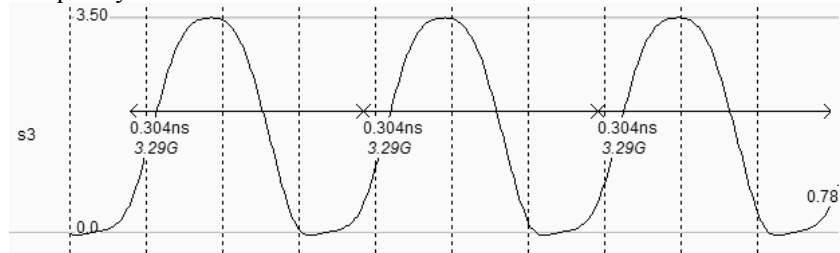


Fig. 2-4: Reduced device features and increased interconnect layers

Figure 2-4 illustrates the main improvements in terms of clock frequency for Intel microprocessors and Motorola micro-controllers. An illustration of this trend towards higher frequencies is given below (Figure 5), with a ring oscillator made from 3 inverters, simulated with MICROWIND2 using 0.35 μ m and 0.12 μ m technologies. Although the supply voltage has been cut by more than half (VDD is 3.3V in 0.35 μ m, 1.2 in 0.12 μ m), the gain in frequency is around six.

0.35 μ m



0.12 μ m

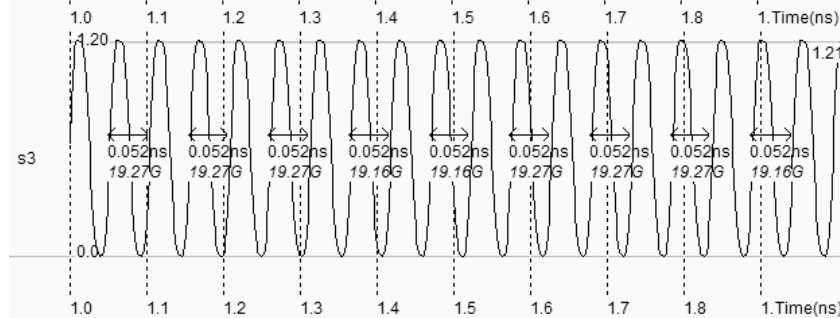


Fig. 2-5: Improvement in speed thanks to deep submicron technology

HOW TO SIMULATE

- ① Start Microwind2. By default the software is configured with 0.12µm technology. Click **File → Open**.
- ② Select INV3. Click **Simulate→ Start Simulation**. The oscillation on the bottom of figure 2-5 appears.
- ③ Click **Close**.
- ④ Click **File → Select Foundry**. Click `cmos35.rul`.
- ⑤ Run again the simulation. Observe the change of VDD and the slow down of the oscillating frequency.

2.3 Increased Layers

The table below lists a set of key parameters, and their evolution with the technology. Worth of interest is the increased number of metal interconnects the reduction of the power supply VDD and the reduction of the gate oxide down to atomic scale values. Notice also the slow decrease of the threshold voltage of the MOS device and the increasing number of input/output pads available on a single die.

Lithography	Year	Metal layers	Core supply (V)	Core Oxide (nm)	Chip size (mm)	Input/output pads	Microwind2 rule file
1.2µm	1986	2	5.0	25	5x5	250	Cmos12.rul
0.7µm	1988	2	5.0	20	7x7	350	Cmos08.rul
0.5µm	1992	3	3.3	12	10x10	600	Cmos06.rul
0.35µm	1994	5	3.3	7	15x15	800	Cmos035.rul
0.25µm	1996	6	2.5	5	17x17	1000	Cmos025.rul
0.18µm	1998	6	1.8	3	20x20	1500	Cmos018.rul
0.12µm	2001	6-8	1.2	2	22x20	1800	Cmos012.rul
90nm	2003	6-10	1.0	1.8	25x20	2000	Cmos90n.rul
65nm	2005	6-12	0.8	1.6	25x20	3000	Cmos70n.rul

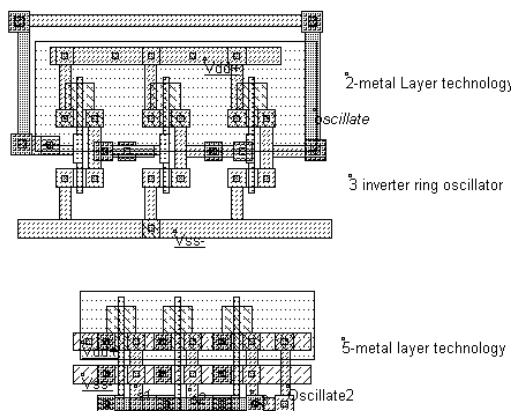


Fig. 2-6: A three-inverter ring oscillator routed with 2-metal layers and 5-metal layers technologies

As can be noticed, the number of metal layers used for interconnects has been continuously increasing in the course of the past ten years. More layers for routing means a more efficient use of the silicon surface, as for printed circuit boards. Active areas, i.e MOS devices can be placed closer from each other if many routing layers are provided (Figure 2-6).

HOW TO SIMULATE

- ① Start Microwind2. By default the software is configured with 0.12µm technology. Click **File →Open**.
- ② Select INV3. Click **Simulate→ Process section in 2D**.
- ③ Draw a line representing the location for 2D-process view. The 2D view appears. Click **OK**.

- ④ Click **Simulate** → **Start Simulation**. Observe the oscillator frequency.
- ⑤ Click **File** → **Select Foundry**. Change the technology (For example `cmos035.rul`)
- ⑥ Ask again for the 2D view. Observe the change in the process aspect.
- ⑦ Ask again for analog simulation. Observe the change in frequency and voltage supply.

2.4 Design Trends

Originally, integrated circuits were designed at layout level, with the help of logic design tools, to achieve design complexities of around 10,000 transistors. The Microwind layout tool works at the lowest level of design, while DSCH operates at logic level.

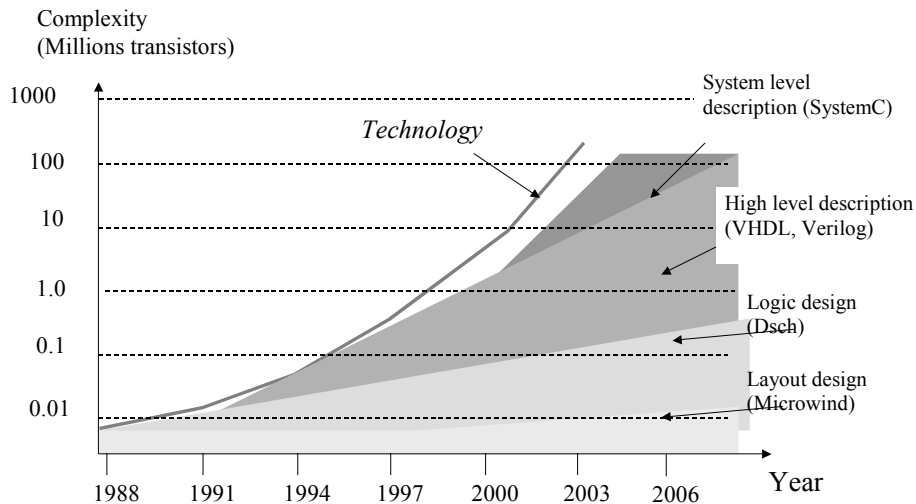


Figure 2-7: The evolution of integrated circuit design techniques, from layout level to system level

2.5 Exercises





1. Plot the frequency improvement versus the technology for the CMOS_{xx} technology family, using the 3-inverter ring oscillator. Can you guess the performances of the 35nm technology?
2. Does the 3-inverter frequency performance represent the microprocessor frequency correctly? Use data of figure 1-3 to build your answer.
3. From the 2D comparative aspect of 0.35 μm and 0.12 μm technologies, what may be the rising problems of using multiple metallization layers?

3 The MOS device

This chapter presents the CMOS transistor, its layout, static characteristics and dynamic characteristics. The vertical aspect of the device and the three dimensional sketch of the fabrication are also described.

3.1 Logic Levels

Three logic levels 0,1 and X are defined as follows:

Logical value	Voltage	Name	Symbol in DSCH	Symbol in Microwind
0	0.0V	VSS	 (Green in logic simulation)	 (Green in analog simulation)
1	1.2V in cmos 0.12µm	VDD	 (Red in logic simulation)	 (Red in analog simulation)
X	Undefined	X	(Gray in simulation)	(Gray in simulation)

3.2 The MOS as a switch

The MOS transistor is basically a switch. When used in logic cell design, it can be *on* or *off*. When *on*, a current can flow between drain and source. When *off*, no current flow between drain and source. The MOS is turned on or off depending on the gate voltage. In CMOS technology, both n-channel (or nMOS) and p-channel MOS (or pMOS) devices exist. The nMOS and pMOS symbols are reported below. The symbols for the ground voltage source (0 or VSS) and the supply (1 or VDD) are also reported in figure 3-1.

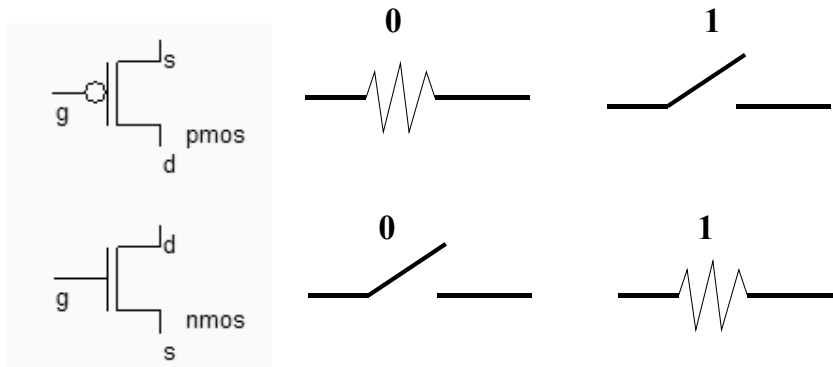


Fig. 3-1: the MOS symbol and switch

The n-channel MOS device requires a logic value 1 (or a supply VDD) to be on. In contrary, the p-channel MOS device requires a logic value 0 to be on. When the MSO device is on, the link between the source and

drain is equivalent to a resistance. The order of range of this 'on' resistance is 100Ω - $5K\Omega$. The 'off' resistance is considered infinite at first order, as its value is several $M\Omega$.

3.3 Logic Simulation of the MOS

At logic level, the MOS is considered as a simple switch. Moreover, the logic switch is unidirectional, meaning that the logic signal always flows from the source to the drain. This major restriction has no physical background. In reality, the current may flow both ways. The reason why the logic MOS device enables the signal to propagate only from source to drain is purely a software implementation problem. In the logic simulator of DSCH2, an arrow indicates whether or not the current flows, and its direction (Figure 3.2). When the device is OFF, the drain keeps its last logic value, thus acting as an elementary memory. Notice that you cannot pass any logic information from the drain to the source. Such a circuit would fail.

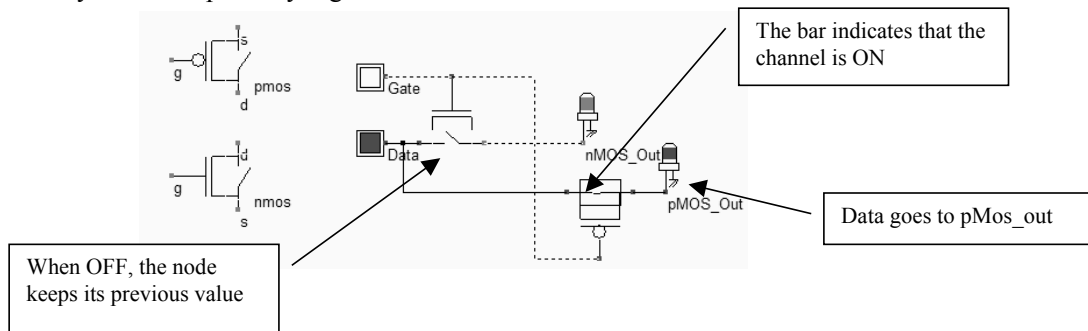


Fig. 3-2: the logic simulation of the MOS device (*MosExplain.SCH*)

3.4 MOS layout

We use MICROWIND2 to draw the MOS layout and simulate its behavior. Go to the directory in which the software has been copied (By default `microwind2`). Double-click on the MicroWind2 icon.

The MICROWIND2 display window includes four main windows: the main menu, the layout display window, the icon menu and the layer palette. The layout window features a grid, scaled in lambda (λ) units. The lambda unit is fixed to half of the minimum available lithography of the technology. The default technology is a CMOS 6-metal layers $0.12\mu\text{m}$ technology, consequently lambda is $0.06\mu\text{m}$ (60nm).

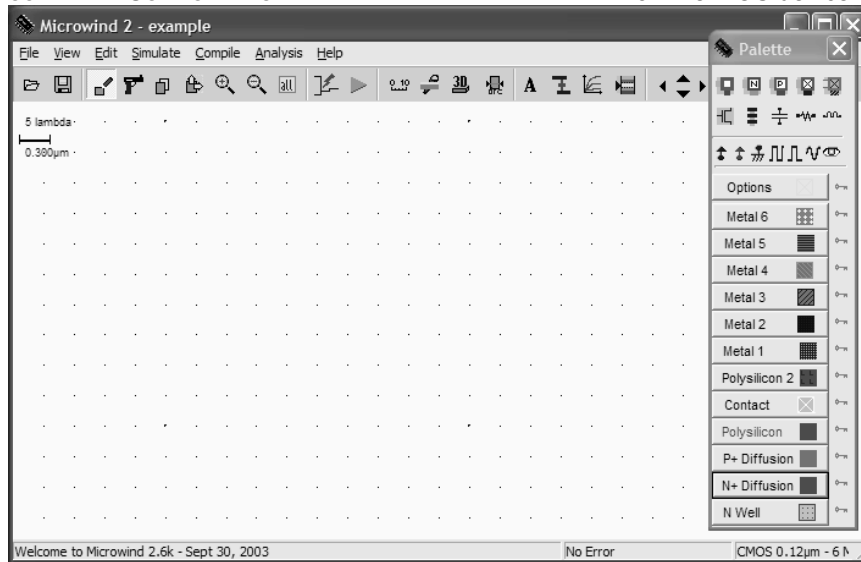


Fig. 3-3 The MICROWIND2 window as it appears at the initialization stage..

The palette is located in the lower right corner of the screen. A red color indicates the current layer. Initially the selected layer in the palette is polysilicon. By using the following procedure, you can create a manual design of the n-channel MOS.

- ❶ Fix the first corner of the box with the mouse. While keeping the mouse button pressed, move the mouse to the opposite corner of the box. Release the button. This creates a box in polysilicon layer as shown in Figure 3-4. The box width should not be inferior to 2λ , which is the minimum width of the polysilicon box.
- ❷ Change the current layer into N+ diffusion by a click on the palette of the Diffusion N+ button. Make sure that the red layer is now the N+ Diffusion. Draw a n-diffusion box at the bottom of the drawing as in Figure 3-4. N-diffusion boxes are represented in green. The intersection between diffusion and polysilicon creates the channel of the nMOS device.

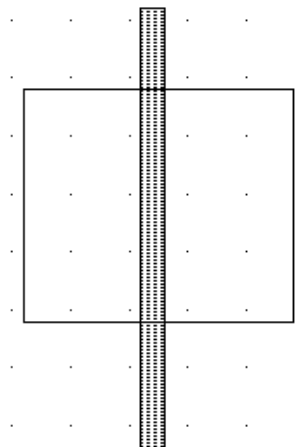


Fig. 3-4. Creating the N-channel MOS transistor

3.5 Vertical aspect of the MOS



Click on this icon to access *process simulation* (Command **Simulate** → **Process section in 2D**). The cross-section is given by a click of the mouse at the first point and the release of the mouse at the second point.

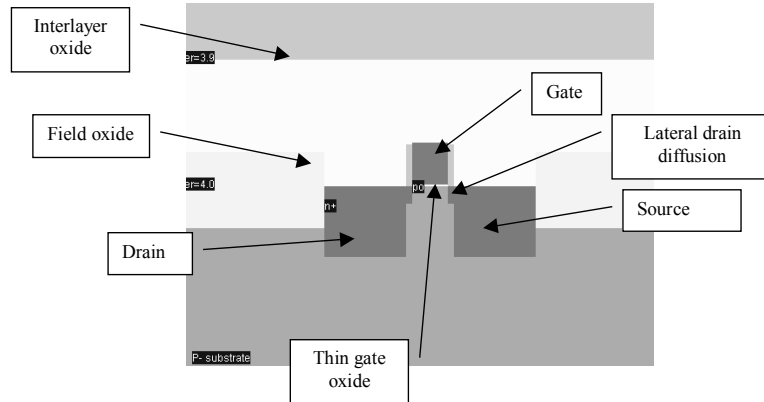


Fig. 3-5. The cross-section of the nMOS devices.

In the example of Figure 3-5, three nodes appear in the cross-section of the n-channel MOS device: the gate (red), the left diffusion called *source* (green) and the right diffusion called *drain* (green), over a substrate (gray). A thin oxide called the gate oxide isolates the gate. Various steps of oxidation have led to stacked oxides on the top of the gate.

The physical properties of the source and of the drain are exactly the same. Theoretically, the source is the origin of channel impurities. In the case of this nMOS device, the channel impurities are the electrons. Therefore, the source is the diffusion area with the lowest voltage. The polysilicon gate floats over the channel, and splits the diffusion into 2 zones, the source and the drain. The gate controls the current flow from the drain to the source, both ways. A high voltage on the gate attracts electrons below the gate, creates an electron channel and enables current to flow. A low voltage disables the channel.

3.6 Static Mos Characteristics



Click on the *MOS characteristics* icon. The screen shown in Figure 2-6 appears. It represents the I_d/V_d static characteristics of the nMOS device.

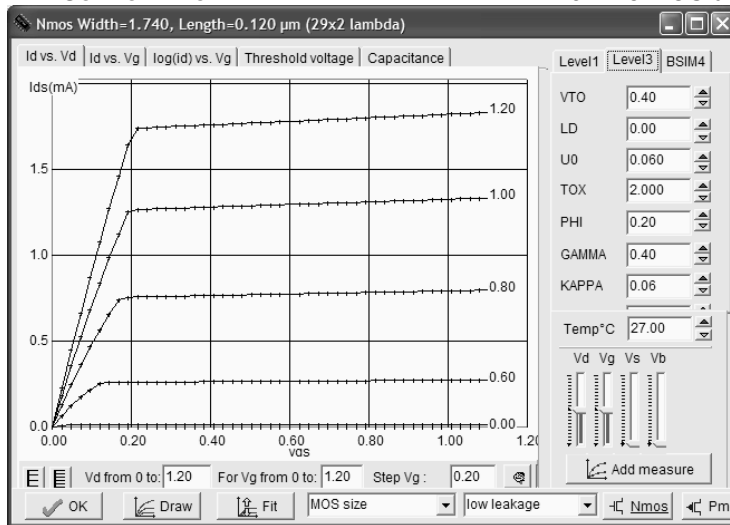
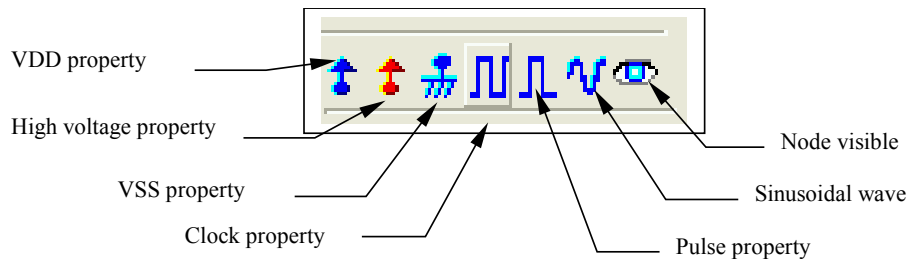


Fig. 3-6. N-Channel MOS characteristics.

The MOS size (width and length of the channel situated at the intersection of the polysilicon gate and the diffusion) has a strong influence on the value of the current. In Figure 3-6, the MOS width is 1.74 μ m and the length is 0.12 μ m. A high gate voltage ($V_g = 1.2V$) corresponds to the highest I_d/V_d curve. For $V_g=0$, no current flows. You may change the voltage values of V_d , V_g , V_s by using the voltage cursors situated on the right side of the window. A maximum current around 1.5mA is obtained for $V_g=1.2V$, $V_d=1.2V$, with $V_s=0.0$. The MOS parameters correspond to SPICE Level 3. A tutorial on MOS model parameters is proposed later in this chapter.

3.7 Dynamic MOS behavior

This paragraph concerns the dynamic simulation of the MOS to exhibit its switching properties. The most convenient way to operate the MOS is to apply a clock to the gate, another to the source and to observe the drain. The summary of available properties that can be added to the layout is reported below.



- 1 Apply a clock to the gate. Click on the *Clock* icon and then, click on the polysilicon gate. The clock menu appears again. Change the name into V_{gate} and click on **OK** to apply a clock with 0.5ns period (0.225ns at 0, 25ps rise, 0.225ns at 1, 25ps fall).

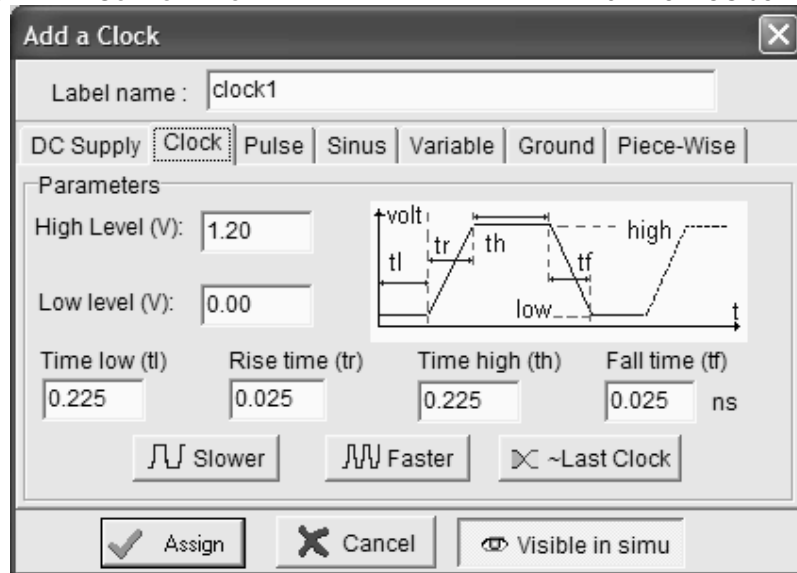


Fig. 3-7. The clock menu.

- ② Apply a clock to the drain. Click on the Clock icon, click on the left diffusion. The *Clock* menu appears. Change the name into *vdrain* and click on **OK**. A default clock with 1ns period is generated. The *Clock* property is sent to the node and appears at the right hand side of the desired location with the name *vdrain*.
- ③ Watch the output: Click on the *Visible* icon and then, click on the right diffusion. Click **OK**. The *Visible* property is then sent to the node. The associated text *s1* is in italic, meaning that the waveform of this node will appear at the next simulation.

Always save **BEFORE** any simulation. The analog simulation algorithm may cause run-time errors leading to a loss of layout information. Click on **File** → **Save as**. A new window appears, into which you enter the design name. Type for example *myMOS*. Then click on **Save**. The design is saved under that filename.

3.8 Analog Simulation

Click on **Simulate** → **Start Simulation**. The timing diagrams of the nMOS device appear, as shown in Figure 3-8.

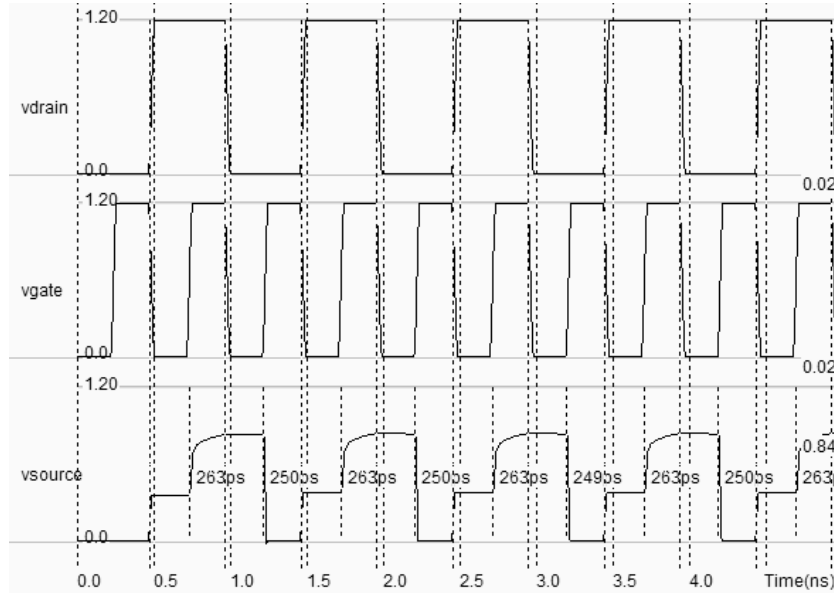


Fig. 3-8. Analog simulation of the MOS device.

When the gate is at zero, no channel exists so the node *vsource* is disconnected from the drain. When the gate is on, the source copies the drain. It can be observed that the nMOS device drives well at zero but poorly at the high voltage. The highest value of *vsource* is around 0.85V, that is VDD minus the threshold voltage. This means that the n-channel MOS device do not drives well logic signal 1, as summarized in figure 3-9. Click on **More** in order to perform more simulations. Click on **Close** to return to the editor.

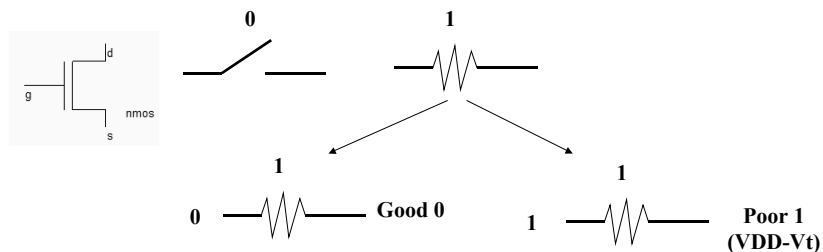


Fig. 3-9. The nMOS device behavior summary

3.9 Layout considerations

The safest way to create a MOS device is to use the MOS generator. In the palette, click the MOS generator icon. A window appears as reported below. The programmable parameters are the MOS width, length, the number of gates in parallel and the type of device (n-channel or p-channel). By default metal interconnects and contacts are added to the drain and source of the MOS. You may add a supplementary metal2 interconnect on the top of metal 1 for drain and source.

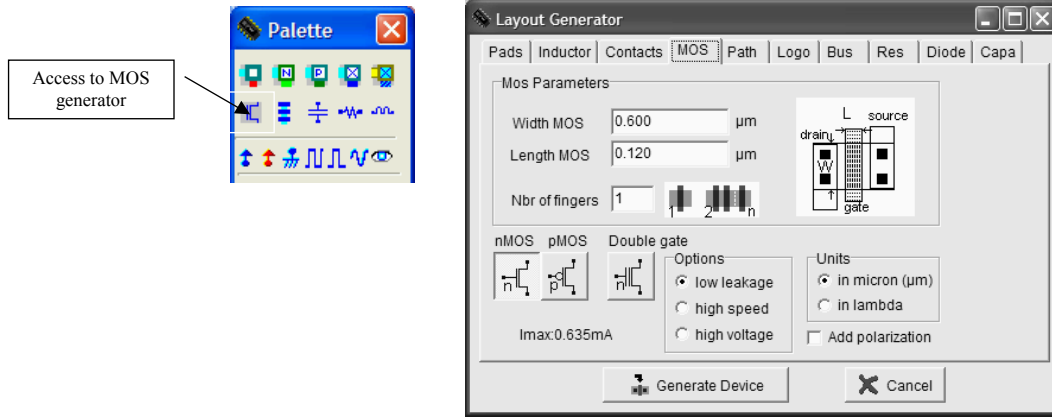


Fig. 3-10. The MOS generator

3.10 The MOS Model 1

For the evaluation of the current I_{ds} between the drain and the source as a function of V_d, V_g and V_s , you may use the old but nevertheless simple MODEL 1 described below.

Mode	Condition	Expression for the current I_{ds}
CUT-OFF	$V_{gs} < 0$	$I_{ds} = 0$
LINEAR	$V_{ds} < V_{gs} - V_t$	$I_{ds} = U_0 \frac{\epsilon_0 \epsilon_r}{TOX} \cdot \frac{W}{L} (V_{gs} - vt)^2$
SATURATED	$V_{ds} > V_{gs} - V_t$	$I_{ds} = U_0 \frac{\epsilon_0 \epsilon_r}{TOX} \cdot \frac{W}{L} (V_{gs} - vt)^2$

With:

$$vt = VTO + GAMMA(\sqrt{(PHI - vb)} - \sqrt{PHI})$$


$\epsilon_0 = 8.85 \cdot 10^{-12}$ F/m is the absolute permittivity

ϵ_r = relative permittivity, equal to 3.9 in the case of SiO2 (no unit)

Mos Model 1 parameters			
Parameter	Definition	Typical Value 0.12μm	
		NMOS	PMOS
VTO	Theshold voltage	0.4V	-0.4V
U0	Carrier mobility	0.06m ² /V-s	0.02m ² /V-s
TOX	Gate oxide thickness	2nm	2nm
PHI	Surface potential at strong inversion	0.3V	0.3V
GAMMA	Bulk threshold parameter	0.4 V ^{0.5}	0.4 V ^{0.5}
W	MOS channel width	1μm	1μm
L	MOS channel length	0.12μm	0.12μm

Table 3-1: Parameters of MOS level 1 implemented into Microwind2

Let us compare the simulation and the measurement, for a 10x10μm device.

- ❶ Click **Simulate** → **Mos** characteristics (Or the icon )
- ❷ Click **Add Measure**.
- ❸ Select the data file **Ne10x0,12.MES**. The “N” means an n-channel MOS device. The “e” corresponds to a test chip fabricated in 0.12µm technology. The values 10x0,12 means Width=10µm, Length=0.12µm.
- ❹ Select “Level 1” in the parameter list to compare LEVEL1 simulated characteristics with the measurements.

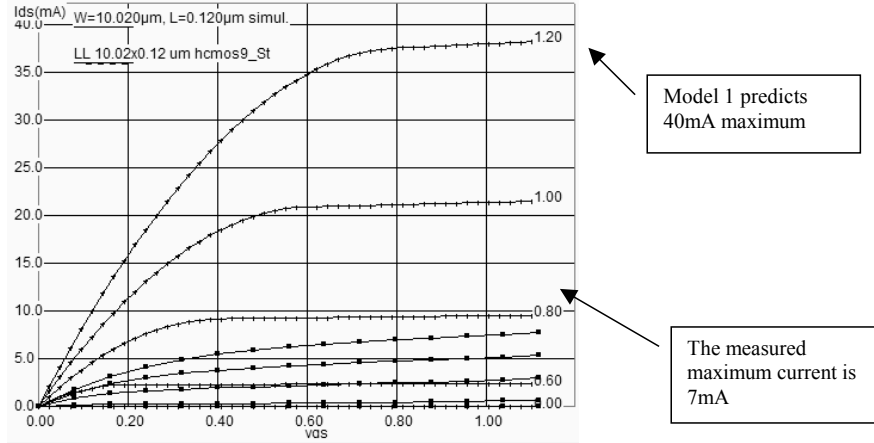


Fig. 3-11: The model 1 predict a current 4 times higher than the measurement

When dealing with sub-micron technology, the model 1 is more than 4 times too optimistic regarding current prediction, compared to real-case measurements, as shown above for a 10x0,12µm n-channel MOS.

3.11 The MOS Model 3

For the evaluation of the current I_{ds} as a function of V_d, V_g and V_s between drain and source, we commonly use the following equations, close from the SPICE model 3 formulations. The formulations are derived from the model 1 and take into account a set of physical limitations in a semi-empirical way.

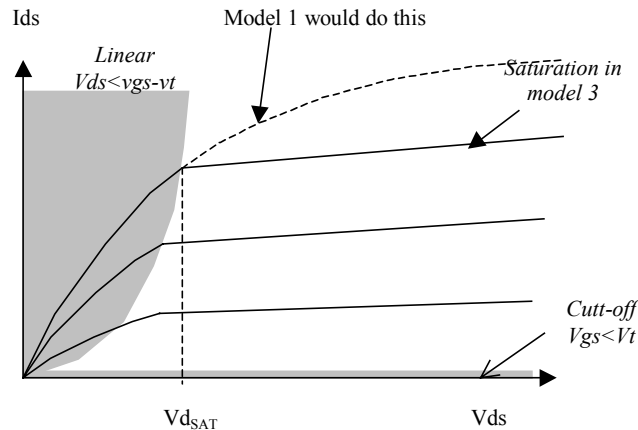


Fig. 3-12: Introduction of the saturation voltage V_{dSAT} which truncates the equations issued from model 1

One of the most important change is the introduction of V_{dSAT} , a saturation voltage from which the current saturates and do not rise as the LEVEL1 model would do (Figure 3-12). This saturation effect is significant for small channel length. The main LEVEL3 equations are listed below.

CUT-OFF MODE. $V_{gs} < 0$

$$I_{ds} = 0$$

NORMAL MODE. $V_{gs} > V_{on}$

$$I_{ds} = K_{eff} \frac{W}{L_{eff}} (1 + KAPPA \cdot V_{ds}) V_{de} \left((V_{gs} - V_{th}) - \frac{V_{de}}{2} \right)$$

with

$$V_{on} = 1.2V_{th}$$

$$V_{th} = V_{TO} + GAMMA(\sqrt{PHI - V_{bs}} - \sqrt{PHI})$$

$$V_{de} = \min(V_{ds}, V_{dsat})$$

$$V_{dsat} = V_c + V_{sat} - \sqrt{V_c^2 + V_{sat}^2}$$

$$V_{dsat} = V_{gs} - V_{th}$$

$$V_c = V_{MAX} \frac{L_{eff}}{0.06}$$

$$L_{eff} = L - 2LD$$

The sub-threshold mode corresponds to $V_{gs} < V_{TO}$. The model 3 includes a specific set of equations based on an exponential decrease of I_{ds} , as compared to $I_{ds} = 0$ predicted by model 1.

$$I_{ds} = I_{ds}(V_{on}, V_{ds}) \exp\left(\frac{q(V_{gs} - V_{on})}{nkT}\right)$$

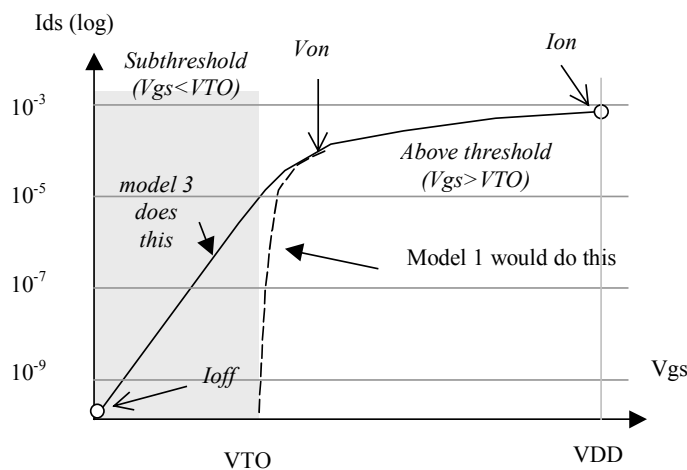


Figure 3-13: Introduction of an exponential law to model the sub-threshold behavior of the current

Parameter	Definition	Typical Value
		0.12μm
		NMOS
		pMOS

VTO	Theshold voltage of a long channel device, at zero Vbs.	0.4V	-0.4V
U0	Carrier mobility	0.06 m ² /V.s	0.025 m ² /V.s
TOX	Gate oxide thickness	3 nm	3 nm
PHI	Surface potential at strong inversion	0.3V	0.3V
LD	Lateral diffusion into channel	0.01μm	0.01μm
GAMMA	Bulk threshold parameter	0.4 V ^{0.5}	0.4 V ^{0.5}
KAPPA	Saturation field factor	0.01 V ⁻¹	0.01 V ⁻¹
VMAX	Maximum drift velocity	150Km/s	100Km/s
THETA	Mobility degradation factor	0.3 V ⁻¹	0.3 V ⁻¹
NSS	Subthreshold factor	0.07 V ⁻¹	0.07 V ⁻¹
W	MOS channel width	0.5-20μm	0.5-40μm
L	MOS channel length	0.12μm	0.12μm

The curve shown in Figure 3-14 shows the effects of VTO, U0 and GAMMA. Act on VTO cursors in order to shift the curves right or left, U0 to adjust the slope, and GAMMA to fit the spacing between curves.

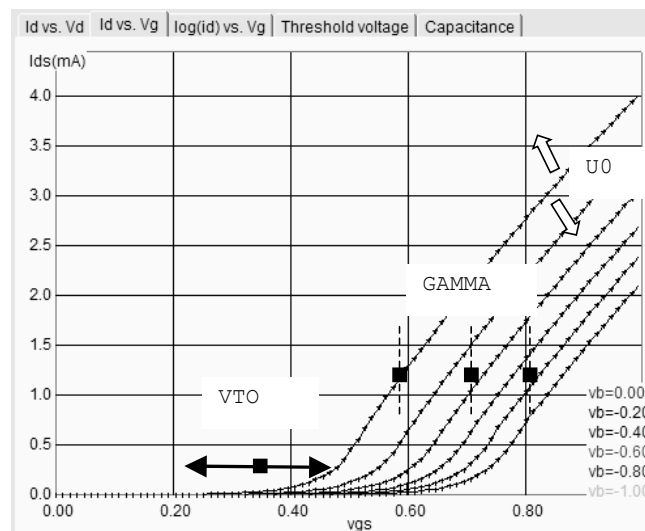


Fig. 3-14. Effect of KP, VTO and GAMMA on the Id/Vd curve

3.12 The BSIM4 MOS Model

A family of models has been developed at the University of Berkeley for the accurate simulation of sub-micron technology. The Berkeley Short-channel IGFET Model (BSIM) exist in several version (BSIM1, BSIM2, BSIM3). The BSIM3v3 version, promoted by the Electronic Industries Alliance (EIA) is an industry standard for deep-submicron device simulation.

A new MOS model, called BSIM4, has been introduced in 2000. A simplified version of this model is supported by Microwind2, and recommended for ultra-deep submicron technology simulation. BSIM4 still considers the operating regions described in MOS level 3 (linear for low V_{ds} , saturated for high V_{ds} , subthreshold for $V_{gs} < V_t$), but provides a perfect continuity between these regions. BSIM4 introduces a new region where the impact ionization effect is dominant.

The number of parameters specified in the official release of BSIM4 is as high as 300. A significant portion of these parameters is unused in our implementation. We concentrate on the most significant parameters, for educational purpose. The set of parameters is reduced to around 20, shown in the right part of figure 3-15.

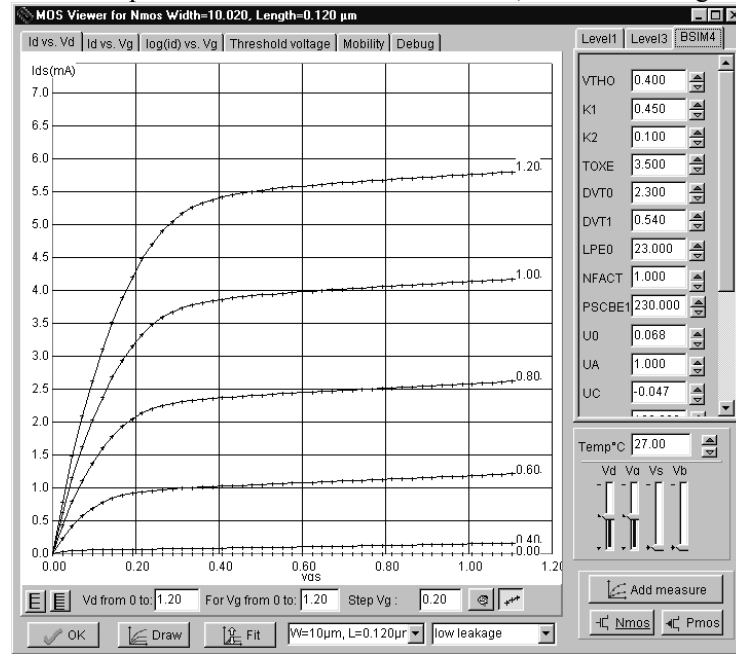


Fig.3-15: Implementation of BSIM4 within Microwind2

The general equation of the threshold voltage is presented below.

$$v_{th} = V_{TH0} + K1 \cdot \sqrt{(\Phi_s - V_{bs}) - \sqrt{\Phi_s}} - K2 \cdot V_{bs} + \Delta V_{t_{SCE}} + \Delta V_{t_{NULD}} + \Delta V_{t_{DIBL}}$$

where V_{TH0} is the long channel threshold voltage at $V_{bs}=0$ (Around 0.5V), $K1$ is the first order body bias coefficient ($0.5 \text{ V}^{1/2}$), Φ_s is the surface potential, V_{bs} is the bulk-source voltage, $K2$ is the second order body bias coefficient, $\Delta V_{t_{SCE}}$ is the short channel effect on V_t , $\Delta V_{t_{NULD}}$ is the non-uniform lateral doping effect, and $\Delta V_{t_{DIBL}}$ is the drain-induced barrier lowering effect of short channel on V_t .

Concerning the formulations for mobility of channel carriers, the generic parameter is $U0$, the mobility of electrons and holes. The effective mobility μ_{eff} is reduced due to several effects: the bulk polarization, and the gate voltage. The equation implemented in Microwind2 is the most recent mobility model proposed in BSIM4.

$$\mu_{eff} = \frac{U0}{1 + (UA + UC \cdot V_{bseff}) \left(\frac{V_{gsteff} + 2(V_{TH0} - V_{fb} - \phi_s)}{TOXE} \right)^{EU}}$$

where

- $U0$ is the low field mobility, in $\text{m}^2/\text{V}\cdot\text{s}$. Its default value is around 0.06 for n-channel MOS and 0.025 for p-channel MOS.
- UA is the first order mobility degradation coefficient, in m/V . Its default value is around 10^{-15} .
- UC is the body-effect coefficient of mobility degradation, in m/V^2 . Its default value is -0.045×10^{-15} .

- VFB is the flat band voltage, in V.
- TOXE is the oxide thickness, in m. A typical value for TOXE in 0.12μm is 3nm (3.10⁻⁹m).
- EU is a coefficient equal to 1.67 for n-channel MOS, and 1.0 for p-channel MOS.

The current I_{ds} is computed using one single equation, as described below.

$$I_{ds0} = \frac{W_{eff}}{L_{eff}} \mu_{eff} \frac{\epsilon_r \epsilon_0}{TOXE} V_{gsteff} \left(1 - \frac{A_{bulk} V_{dseff}}{(2V_{gsteff} + 4.vt)} \right) \frac{V_{dseff}}{\left(1 + \frac{V_{dseff}}{\epsilon_{sat} L_{eff}} \right)}$$

Parameter	Description	NMOS value in 0.12μm	NMOS value in 0.12μm
VTH0	Long channel threshold voltage at Vbs = 0V	0.3V	0.3V
VFB	Flat-band voltage	-0.9	-0.9
K1	First-order body bias coefficient	0.45 V ^{1/2}	0.45 V ^{1/2}
K2	Second-order body bias coefficient	0.1	0.1
LPE0	Lateral non-uniform doping parameter at Vbs = 0	2.3 [°] -10	2.3 [°] -10
DVT0	First coefficient of short-channel effect on threshold voltage	2.2	2.2
DVT1	Second coefficient of short-channel effect on Vth	0.53	0.53
ETA0	Drain induced barrier lowering coefficient	0.08	0.08
NFACTOR	Sub-threshold turn-on swing factor. Controls the exponential increase of current with Vgs.	1	1
U0	Low-field mobility	0.060 m ² /Vs	0.025 m ² /Vs
UA	Coefficient of first-order mobility degradation due to vertical field	11.0e-15 m/V	11.0e-15 m/V
UC	Coefficient of mobility degradation due to body-bias effect	-0.04650e-15 V ⁻¹	-0.04650e-15 V ⁻¹
VSAT	Saturation velocity	8.0e4 m/s	8.0e4 m/s
WINT	Channel-width offset parameter	0.01 [°] -6μm	0.01 [°] -6μm
LINT	Channel-length offset parameter	0.01 [°] -6μm	0.01 [°] -6μm
PSCBE1	First substrate current induced body-effect mobility reduction	4.24e8 V/m	4.24e8 V/m
PSCBE2	Second substrate current induced body-effect mobility reduction	4.24e8 V/m	4.24e8 V/m
KT1	Temperature coefficient of the threshold voltage.	-0.1V	-0.1V
UTE	Temperature coefficient for the zero-field mobility U0.	-1.5	-1.5
VOFF	Offset voltage in subthreshold region.	-0.08V	-0.08V
PCLM	Parameter for channel length modulation	1.2	1.2

3.13 Low leakage MOS

A new kind of MOS device has been introduced in deep submicron technologies, starting the 0.18μm CMOS process generation. The new MOS, called high speed MOS (HS) is available as well as the normal one, recalled Low leakage MOS (LL). The main objective is to propose two types of devices, one which reduces significantly the leakage current (LL version), that is the small current I_{off} that flows from between drain and source with a gate voltage 0 (Supposed to be no current in first order approximation). On the figure below, the low leakage MOS device (right side) has an I_{off} current reduced by a factor 100, thanks to a higher threshold voltage (0.4V rather than 0.3V).

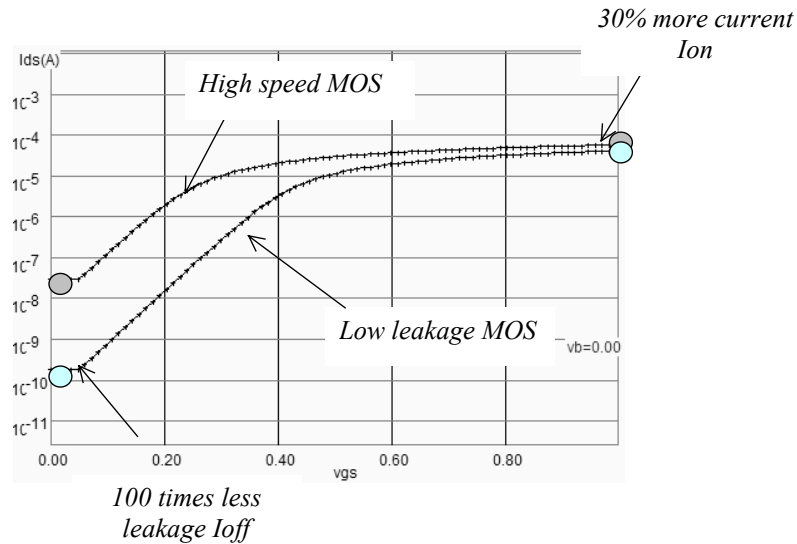


Fig. 3-16: Low leakage MOS for lower I_{off} current

The main drawback of the Low leakage MOS device is a 30% reduction of the I_{on} current, leading to a slower switching. High speed MOS devices should be used in the case of fast operation linked to critical nodes, while low leakage MOS should be placed whenever possible, for all nodes where a maximum switching speed is not required.

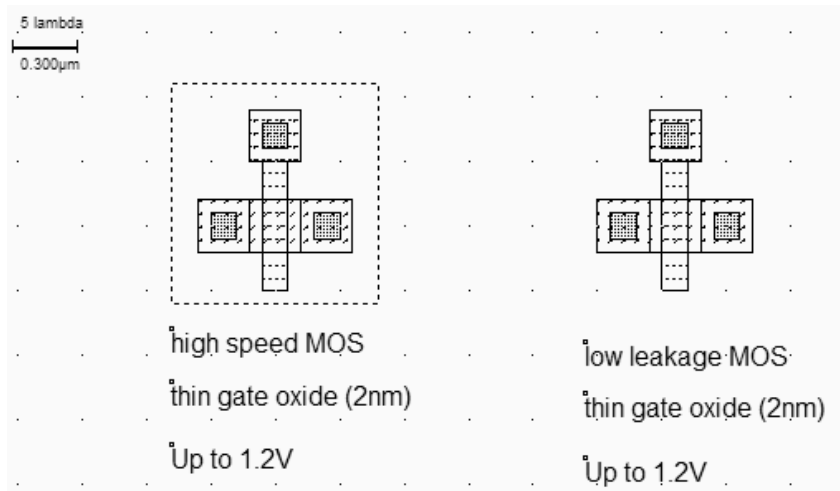


Fig. 3-17: High speed and Low leakage MOS layout. The only difference is the option layer to setup the high speed option

3.14 High voltage MOS

Integrated circuits with low voltage internal supply and high voltage I/O interface are getting common in deep sub-micron technology. The internal logic of the integrated circuit operates at very low voltage (Typically 1.0V in 0.12µm), while the I/O devices operate in standard voltages (2.5, 3.3 or 5V). The

input/output structures work at high voltage thanks to specific MOS devices with thick oxide, while the internal devices work at low voltage with optimum performances.

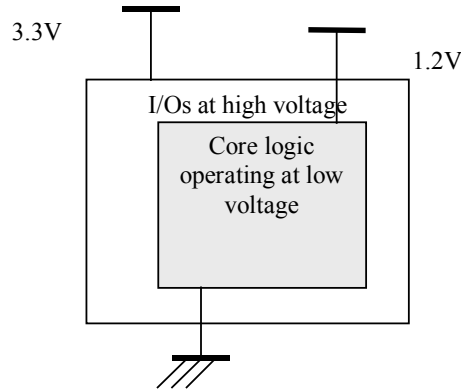


Fig. 3-18: Interfacing low voltage logic signals with high voltage I/Os requires specific circuits operating in high voltage mode.

For I/Os operating at high voltage, specific MOS devices called "High voltage MOS" are used. We cannot use high-speed or low leakage devices as their oxide is too small. A 2.5V voltage would damage the gate oxide of a high-speed MOS in 0.12µm technology. The high voltage MOS is built using a thick oxide, two to three times thicker than the low voltage MOS, to handle high voltages as required by the I/O interfaces.

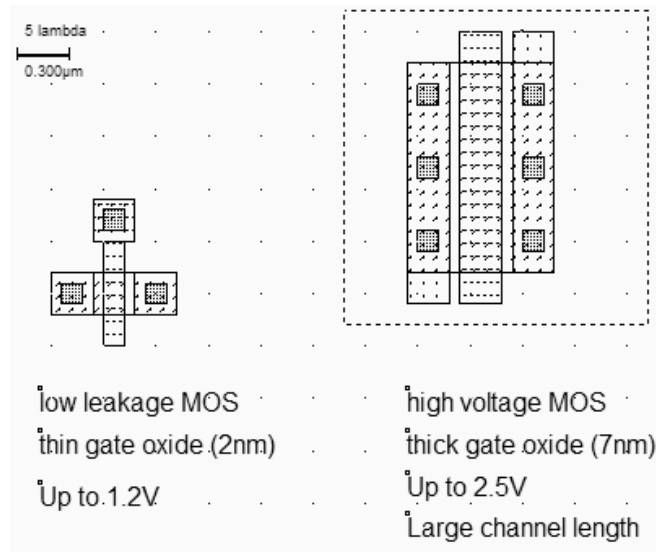


Fig. 3-19: low leakage and high voltage MOS. The high voltage MOS has a minimum length longer than for the other MOS. The gate oxide is also thicker to handle high voltage operation.

3.15 Temperature effects on the MOS

Three main parameters are concerned by the sensitivity to temperature: the threshold voltage V_{TO} , the mobility μ_0 and the slope in sub-threshold mode. Both V_{TO} and μ_0 decrease when the temperature increases. The modeling of the temperature effect in BSIM4 is as follows. In Microwind2, T_{NOM} is fixed to

300°K, equivalent to 27°C. U_{TE} is negative, and set to -1.8 in 0.12µm CMOS technology, while K_{T1} is set to -0.06 by default.

$$U_0 = U_{0(T=27)} \left(\frac{T + 273}{T_{NOM}} \right)^{U_{TE}} \tag{3-45}$$

$$V_T = V_{T0(T=27)} + K_{T1} \left(\frac{T + 273}{T_{NOM}} - 1 \right) \tag{3-46}$$

A higher temperature leads to a reduced mobility, as U_{TE} is negative. Consequently, at a higher temperature, the current I_{ds} is lowered.

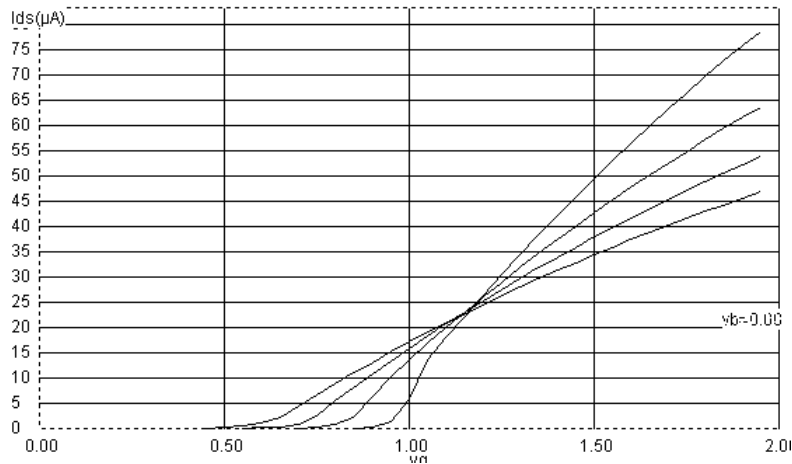


Fig. 3-20. Effect of temperature on the MOS device characteristics

To obtain the curve of figure 3-20, proceed as follows:



- ❶ Click the icon MOS characteristics
- ❷ Select one MOS in the design or click anywhere
- ❸ Select the curve I_d/V_g



- ❹ Enable the screen memory mode by a click on this icon.



Change the temperature. The change in the slope is shown. You may reduce the number of I_d curves by putting a 0.0 in the field **For Vb from 0 to**.

In Microwind2, you can also get access to temperature using the command **Simulate**→**Simulation Parameters**. The screen below appears. The temperature is given in °C.

3.16 The PMOS Transistor

The p-channel transistor simulation features the same functions as the n-channel device, but with opposite voltage control of the gate. For the nMOS, the channel is created with a logic 1 on the gate. For the pMOS, the channel is created for a logic 0 on the gate. Load the file `pmos.msk` and click the icon **MOS characteristics**. The p-channel MOS simulation appears, as shown in Figure 3-21. Note that the pMOS gives approximately half of the maximum current given by the nMOS with the same device size. The highest current is obtained with the lowest possible gate voltage, that is 0.

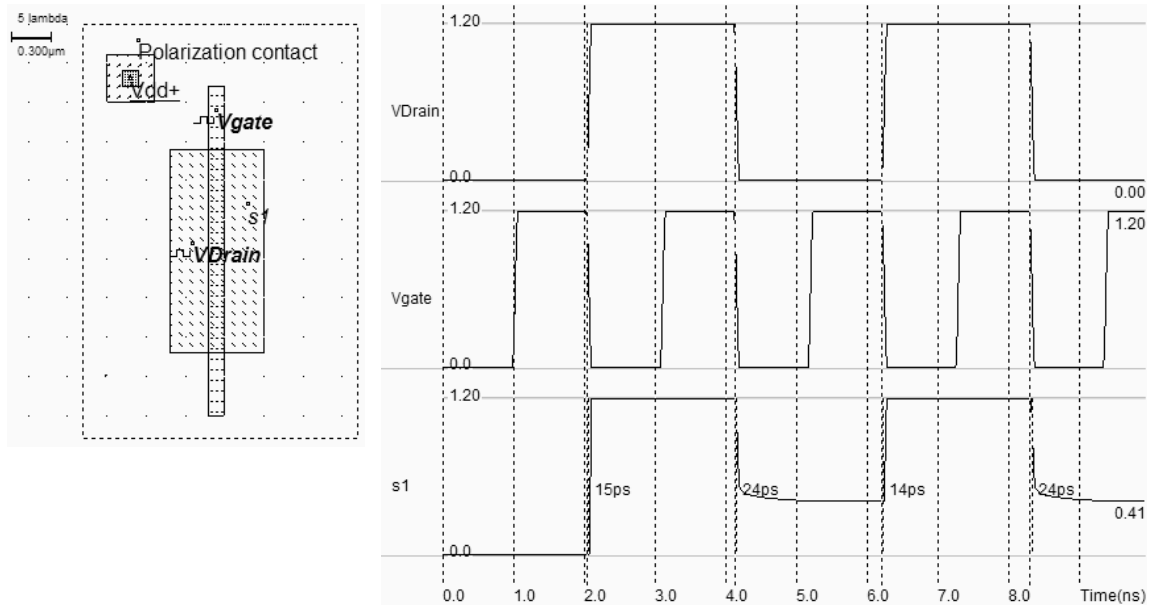


Fig. 3-21. Layout and simulation of the p-channel MOS (`pMOS.MSK`)

From the simulation of figure 3-21, we see that the pMOS device is able to pass well the logic level 1. But the logic level 0 is transformed into a positive voltage, equal to the threshold voltage of the MOS device. The summary of the p-channel MOS performances is reported in figure 3-20.

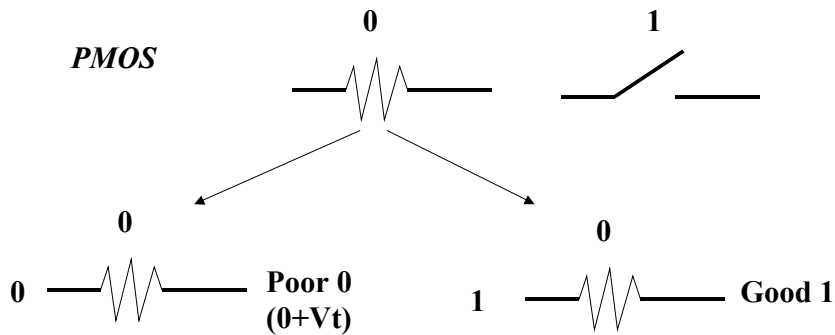


Fig. 3-22. Summary of the performances of a pMOS device

3.17 Process Variations

The simulated results should not be considered as absolute values. Due to unavoidable process variations during the hundreds of chemical steps for the fabrication of the integrated circuit, the MOS characteristics are never exactly identical from one device to another, and from one die to another. It is very common to measure 5% to 20% electrical difference within the same die. In figure 3-23, although both devices have been designed with a drawn 2 lambda, the result is a 0.11µm length for the MOS situated on the left side, and 0.13µm for the MOS situated on the right side.

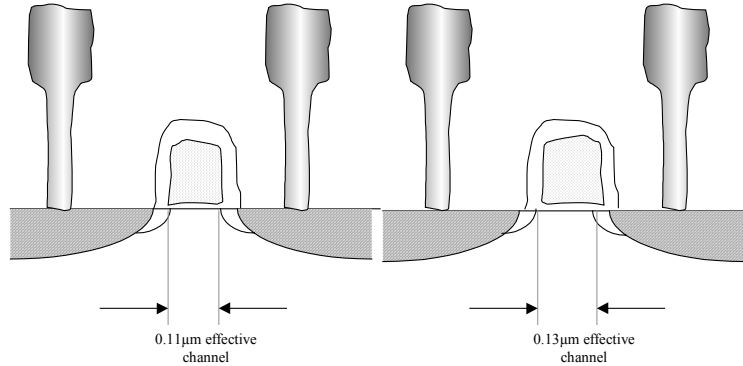


Fig. 3-23: The same MOS device may be fabricated with an important effective channel variation

The menu **Simulate**→**Simulation parameters** gives a simple access to minimum/typical/maximum parameter sets. The approach has consisted in altering two main parameters: the threshold voltage (20% random variation, Gaussian distribution) and the mobility (20% random variation). All other parameters are supposed to be constant.

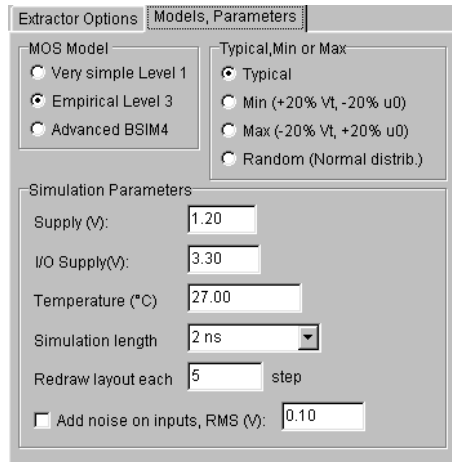


Fig. 3-24: Access to minimum, typical, maximum model parameters or Random simulation

A comparative simulation of the I_d/V_d curve in typical, maximum and minimum scenarios shows a very large variation of performances (Figure 3-25). The user may automatically switch from one parameter set to another by a press of a key ("M" for maximum, "m" for minimum, "t" for typical).

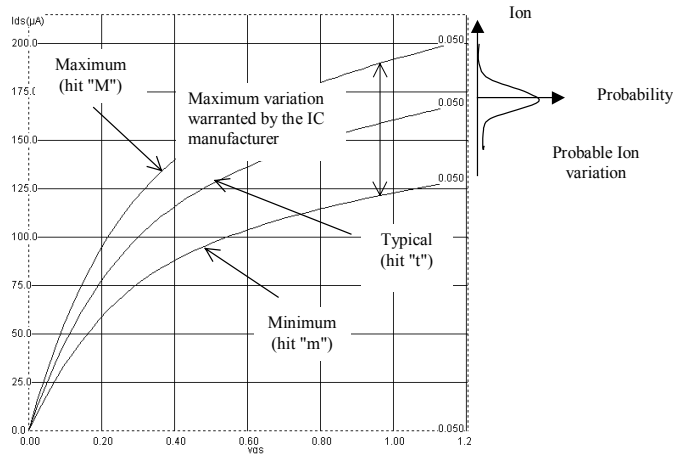


Fig. 3-25: The MOS I_d/v_d curve in Min, Typ, Max modes.

3.18 The Transmission Gate

Both NMOS devices and PMOS devices exhibit poor performances when transmitting one particular logic information. The nMOS degrades the logic level 1, the pMOS degrades the logic level 0. Thus, a perfect pass gate can be constructed from the combination of nMOS and pMOS devices working in a complementary way, leading to improved switching performances. Such a circuit, presented in figure 3-26, is called the transmission gate. In DSCH2, the symbol may be found in the **Advance** menu in the palette. The transmission gate includes one inverter, one nMOS and one pMOS.

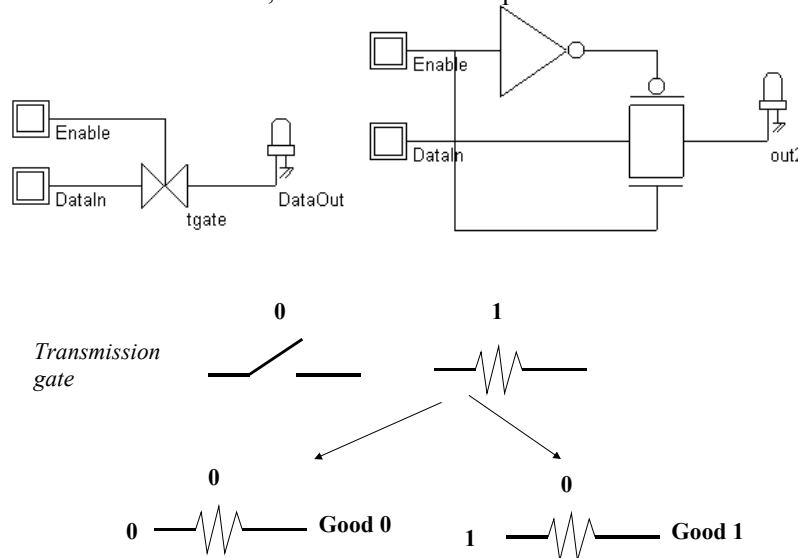


Fig. 3-26. Schematic diagram of the transmission gate (Tgate.SCH)

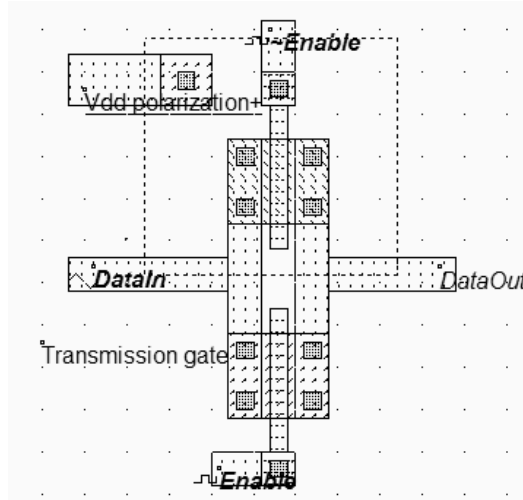


Fig. 3-27. Layout of the transmission gate (TGATE.MSK)

The layout of the transmission gate is reported in figure 3-27. The n-channel MOS is situated on the bottom the p-channel MOS on the top. Notice that the gate controls are not connected, as $\sim Enable$ is the opposite of $Enable$. The operation of the transmission gate is illustrated in figure 3-24. A sinusoidal wave with a frequency of 2GHz is assigned to $DataIn$. With a zero on $Enable$ (And a 1 on $\sim Enable$), the switch is off, and no signal is transferred. When $Enable$ is asserted, the sinusoidal wave appears nearly identical to the output.

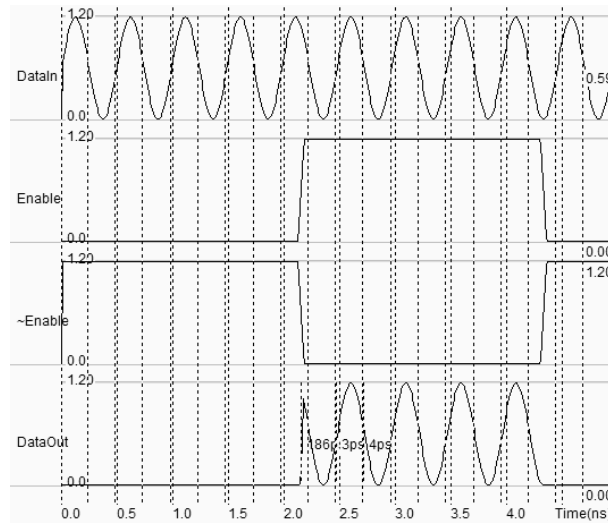


Fig. 3-24. Simulation of the transmission gate (TGATE.MSK)

4 The Inverter

This chapter describes the CMOS inverter at logic level, using the logic editor and simulator DSCH2, and at layout level, using the tool MICROWIND2.

4.1 The Logic Inverter

In this section, an inverter circuit is loaded and simulated. Click **File**→ **Open** in the main menu. Select `INV.SCH` in the list. In this circuit are one button situated on the left side of the design, the inverter and a led. Click **Simulate**→ **Start simulation** in the main menu.

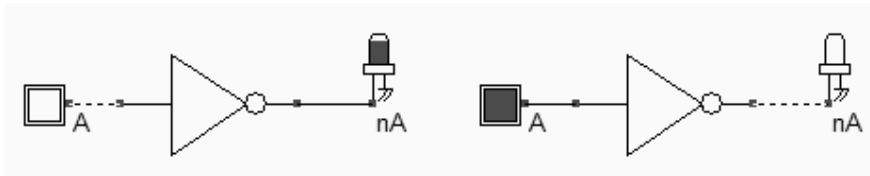


Fig. 4.1: The schematic diagram including one single inverter (`Inverter.SCH`)

Now, click inside the buttons situated on the left part of the diagram. The result is displayed on the leds. The red value indicates logic 1, the black value means a logic 0. Click the button **Stop simulation** shown in the picture below. You are back to the editor.

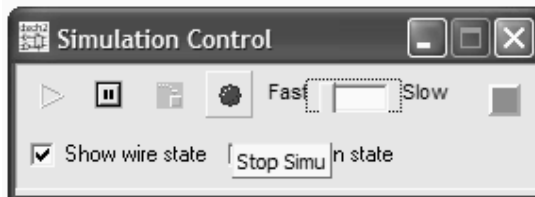
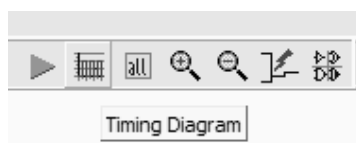


Fig. 4.2: The button **Stop Simulation**



Click the **chronogram** icon to get access to the chronograms of the previous simulation (Figure 4-3). As seen in the waveform, the value of the output is the logic opposite of that of the input.

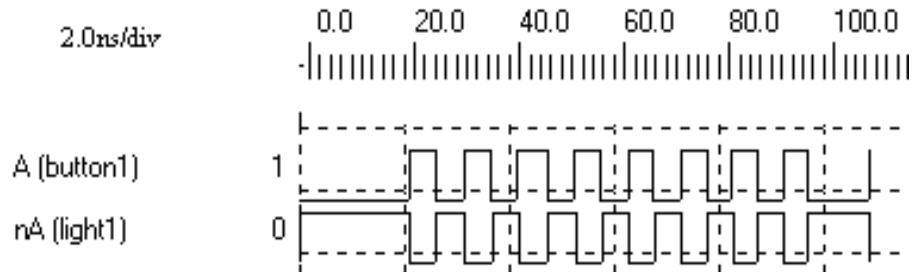


Fig. 4-3 Chronograms of the inverter simulation (CmosInv.SCH)

Double click on the INV symbol, the symbol properties window is activated. In this window appears the VERILOG description (left side) and the list of pins (right side). A set of drawing options is also reported in the same window. Notice the gate delay (0.03ns in the default technology), the fanout that represents the number of cells connected to the output pin (1 cell connected), and the wire delay due to this cell connection (An extra 0.140ns delay).

4.2 THE CMOS INVERTER

The CMOS inverter design is detailed in the figure below. Here the p-channel MOS and the n-channel MOS transistors function as switches. When the input signal is logic 0 (Fig. 4-3 left), the nMOS is switched off while PMOS passes VDD through the output. When the input signal is logic 1 (Fig. 4-3 right), the pMOS is switched off while the nMOS passes VSS to the output.

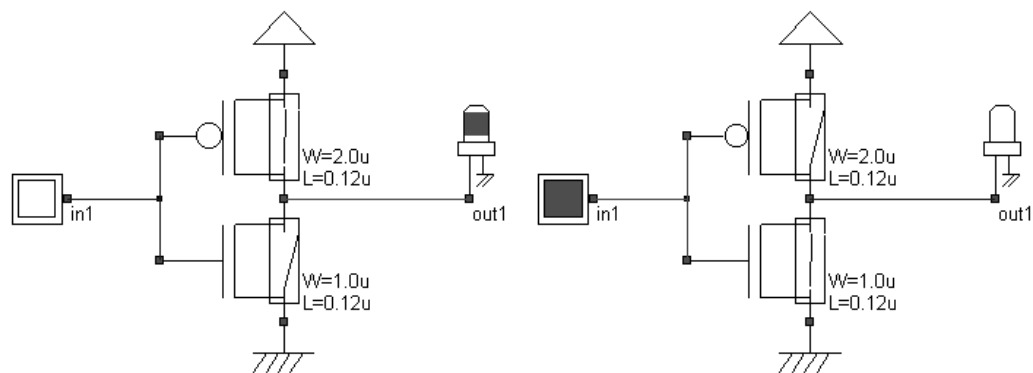


Fig. 4-3: The MOS Inverter (File CmosInv.sch)

The fanout corresponds to the number of gates connected to the inverter output. Physically, a large fanout means a large number of connections, that is a large load capacitance. If we simulate an inverter loaded with one single output, the switching delay is small. Now, if we load the inverter by several outputs, the delay and the power consumption are increased. The power consumption linearly increases with the load capacitance. This is mainly due to the current needed to charge and discharge that capacitance.

4.3 MANUAL LAYOUT OF THE INVERTER

In this paragraph, the procedure to create manually the layout of a CMOS inverter is described. Click the icon **MOS generator** on the palette. The following window appears. By default the proposed length is the minimum length available in the technology (2 lambda), and the width is 10 lambda. In 0.12µm technology, where lambda is 0.06µm, the corresponding size is 0.12µm for the length and 0.6µm for the width. Simply click **Generate Device**, and click on the middle of the screen to fix the MOS device.

Click again the icon **MOS generator** on the palette. Change the type of device by a tick on **p-channel**, and click **Generate Device**. Click on the top of the nMOS to fix the pMOS device. The result is displayed in figure 4-4.

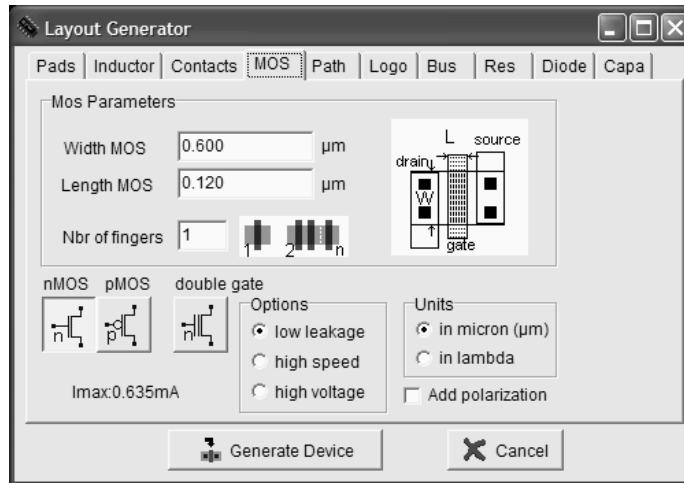


Fig. 4-4. Selecting the nMOS device

4.4 Connection between Devices

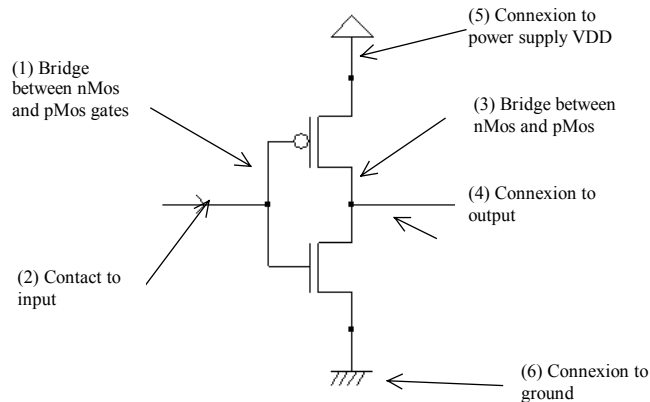


Fig. 4-5 Connections required to build the inverter (CmosInv.SCH)

Within CMOS cells, metal and polysilicon are used as interconnects for signals. Metal is a much better conductor than polysilicon. Consequently, polysilicon is only used to interconnect gates, such as the bridge (1) between pMOS and nMOS gates, as described in the schematic diagram of figure 4-5. Polysilicon is rarely used for long interconnects, except if a huge resistance value is expected.

In the layout shown in figure 4-5, the polysilicon bridge links the gate of the n-channel MOS with the gate of the p-channel MOS device. The polysilicon serves as the gate control and the bridge between MOS gates.

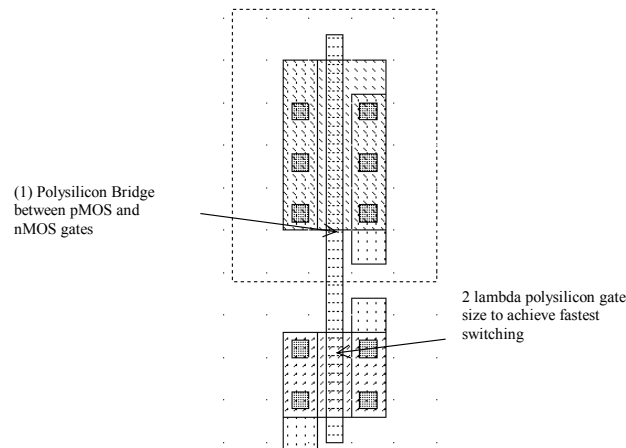


Fig. 4-5 Polysilicon bridge between nMOS and pMOS devices (InvSteps.MSK)

4.5 Useful Editing Tools

The following commands may help you in the layout design and verification processes.






Command	Icon/Short cut	Menu	Description
UNDO	CTRL+U	Edit menu	Cancels the last editing operation
DELETE	 CTRL+X	Edit menu	Erases some layout included in the given area or pointed by the mouse.
STRETCH		Edit menu	Changes the size of one box, or moves the layout included in the given area.
COPY	 CTRL+C	Edit Menu	Copies the layout included in the given area.
VIEW ELECTRICAL NODE	 CTRL+N	View Menu	Verifies the electrical net connections.
2D CROSS-SECTION		Simulate Menu	Shows the aspect of the circuit in vertical cross-section.

Table 4-1: A set of useful editing tools

4.6 Metal-to-poly

As polysilicon is a poor conductor, metal is preferred to interconnect signals and supplies. Consequently, the input connection of the inverter is made with metal. Metal and polysilicon are separated by an oxide which prevents electrical connections. Therefore, a box of metal drawn across a box of polysilicon does not allow an electrical connection (Figure 4-6). To build an electrical connection, a physical contact is needed. The corresponding layer is called "contact". You may insert a metal-to-polysilicon contact in the layout using a direct macro situated in the palette.

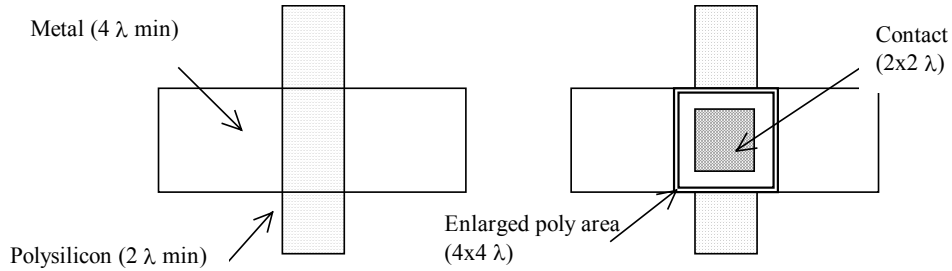


Fig. 4-6 Physical contact between metal and polysilicon

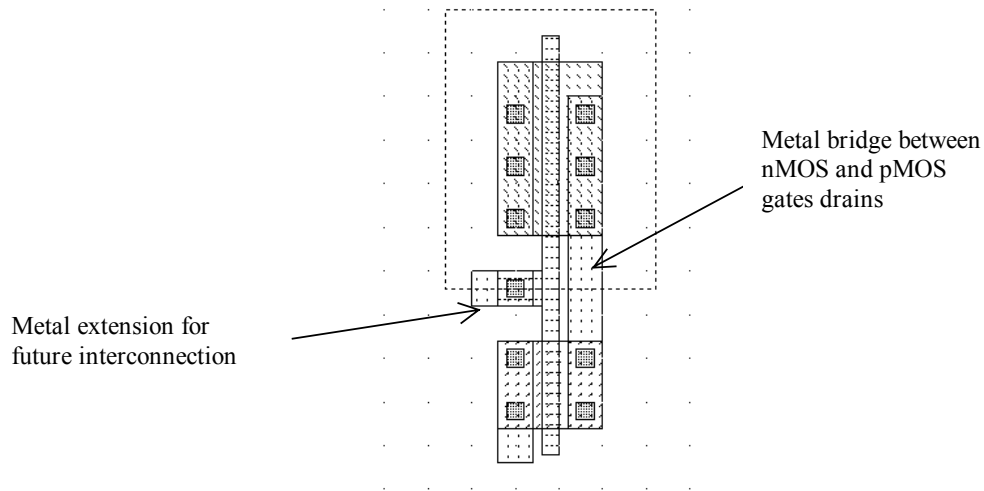


Fig. 4-7 Adding a poly contact, poly and metal bridges to construct the CMOS inverter (InvSteps.MSK)

The *Process Simulator* shows the vertical aspect of the layout, as when fabrication has been completed. This feature is a significant aid to understand the circuit structure and the way layers are stacked on top of each other. A click of the mouse on the left side of the n-channel device layout and the release of the mouse at the right side give the cross-section reported in figure 4-8.

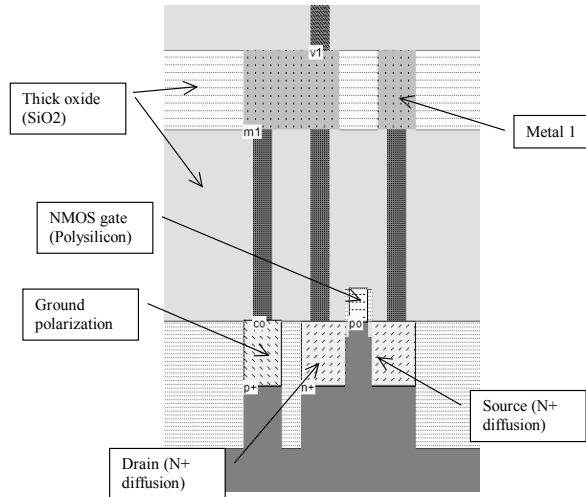


Fig.4-8 The 2D process section of the inverter circuit near the nMOS device (InvSteps.MSK)

4.7 Supply Connections

The next design step consists in adding supply connections, that is the positive supply VDD and the ground supply VSS. In figure 4-9, we use the metal2 layer (Second level of metallization) to create horizontal supply connections. Notice that the metal connections have a large width. This is because a strong current may flow within these supply interconnects. Enlarging the supply metal lines reduces the resistance and avoids electrical overstress.

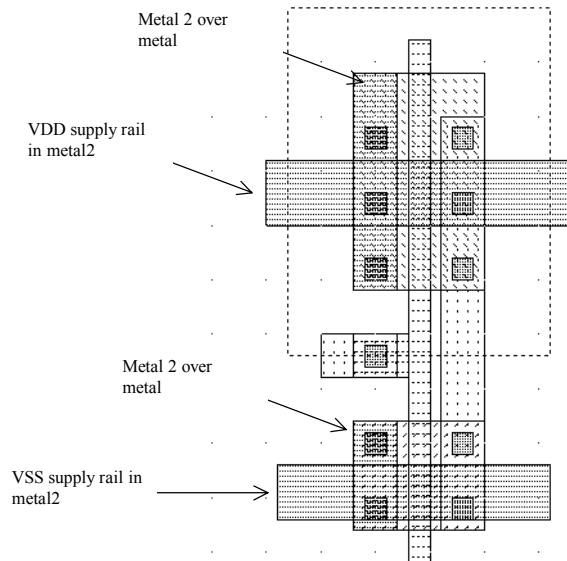


Fig.4-9 Adding metal2 supply lines and the appropriate vias (InvSteps.MSK)

The simplest way to build the physical connection is to add a metal/Metal2 contact that may be found in the palette. The connection is created by a plug called "via" between metal2 and metal layers.

The final layout design step consists in adding polarization contacts. These contacts convey the VSS and VDD voltage supply close to the bulk regions of the device. Remember that the n-well region should always be polarized to a high voltage to avoid short-circuit between VDD and VSS. Adding the VDD polarization in the n-well region is a very strict rule.

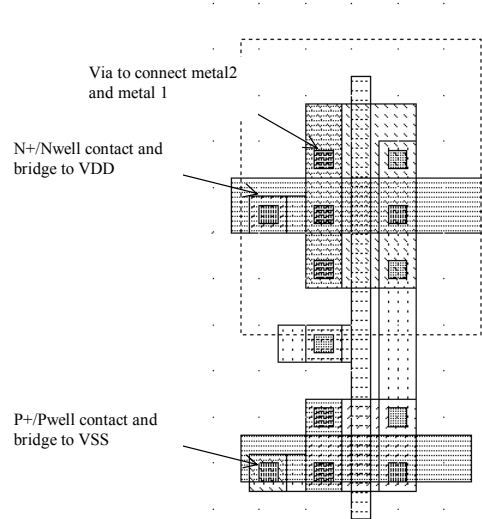


Fig.4-10 Adding polarization contacts

4.8 Process steps to build the Inverter

At that point, it might be interesting to illustrate the steps of fabrication as they would sequence in a foundry. Microwind includes a 3D process viewer for that purpose. Click **Simulate** → **Process steps in 3D**. The simulation of the CMOS fabrication process is performed, step by step by a click on **Next Step**. On figure 4-11, the picture on the left represents the nMOS device, pMOS device, common polysilicon gate and contacts. The picture on the right represents the same portion of layout with the metal layers stacked on top of the active devices.

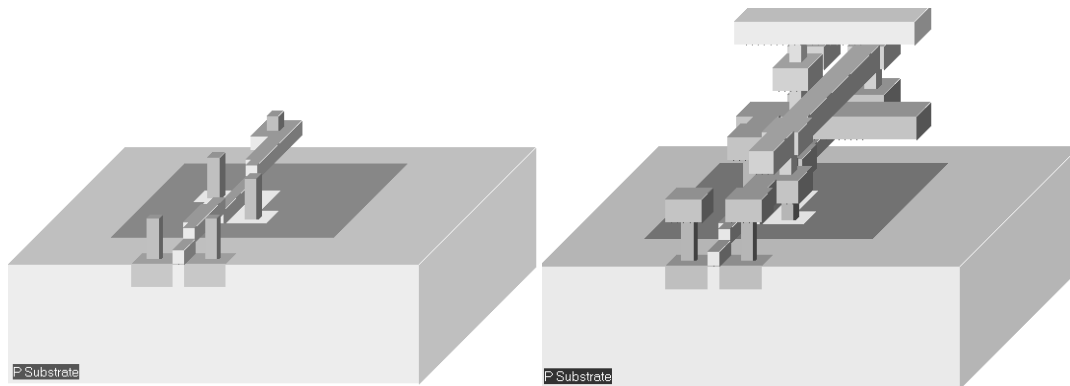


Fig.4-11 The step-by-step fabrication of the Inverter circuit (InvSteps.MSK)

4.9 Inverter Simulation

The inverter simulation is conducted as follows. Firstly, a VDD supply source (1.2V) is fixed to the upper metal2 supply line, and a VSS supply source (0.0V) is fixed to the lower metal2 supply line. The properties are located in the palette menu. Simply click the desired property, and click on the desired location in the layout. Add a clock on the inverter input node (The default node name *clock1* has been changed into *Vin*) and a visible property on the output node *Vout*.

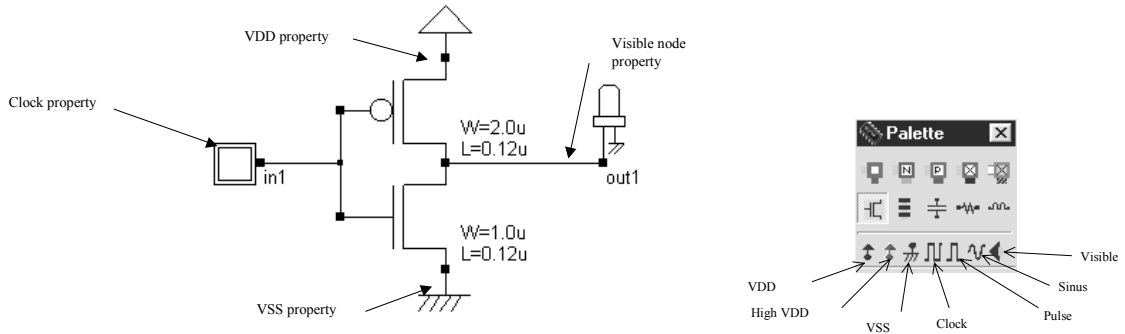


Fig.4-12 Adding simulation properties (InvSteps.MSK)

The command **Simulate** → **Run Simulation** gives access to the analog simulation. Select the simulation mode **Voltage vs. Time**. The analog simulation of the circuit is performed. The time domain waveform, proposed by default, details the evolution of the voltages *in1* and *out1* versus time. This mode is also called transient simulation, as shown in figure 4-13.

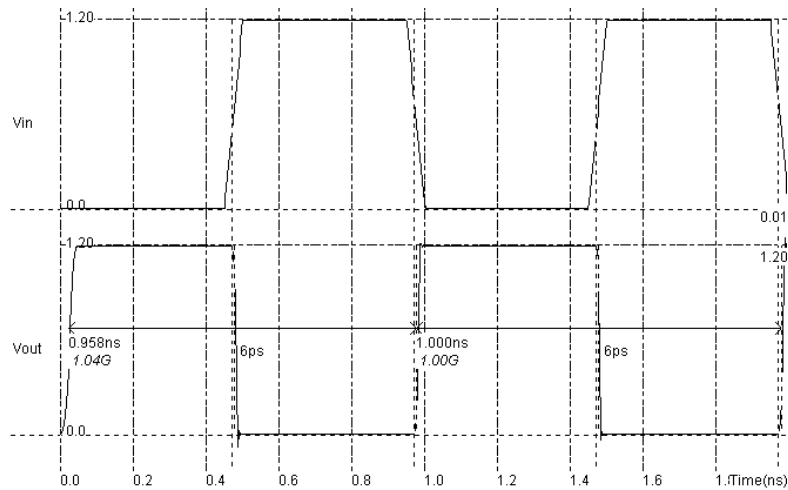


Fig.4-13 Transient simulation of the CMOS inverter (InvSteps.MSK)

The truth-table is verified as follows. A logic zero corresponds to a zero voltage and a logic 1 to a 1.20V. When the input rises to 1, the output falls to 0, with a 6 Pico-second delay ($6 \cdot 10^{-12}$ second).

The power consumption occurs briefly during transitions of the output, either from 0 to 1 or from 1 to 0 (Fig. 4-14). The simulation contains the supply currents in the upper window, and all voltage waveforms in the lower window. The current consumption is important only during a very short period corresponding to the charge or discharge of the output node. Without any switching activity, the current is almost equal to zero.

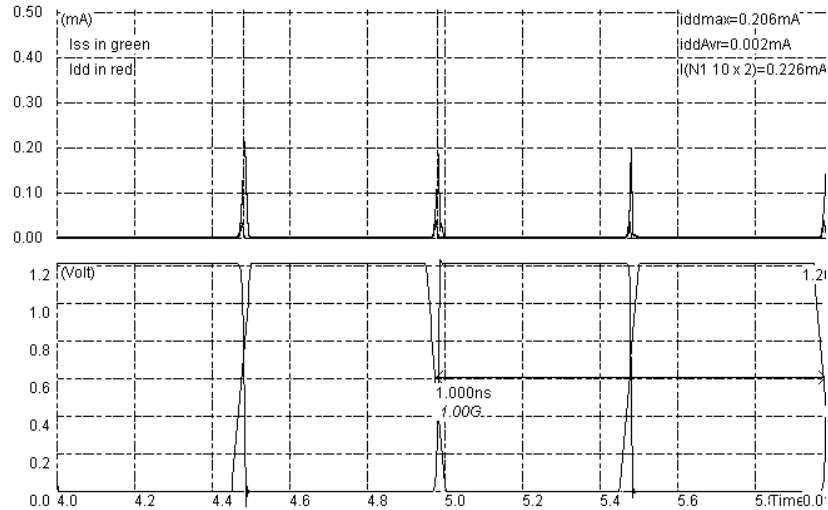


Fig. 4-14: Simulation of the current peaks appearing between V_{DD} and V_{SS} in the CMOS inverter at each output transition (InvSteps.MSK)

Three main factors contribute to the power consumption P : the load capacitance C , the supply voltage V_{DD} and the clock frequency f . For a CMOS inverter, this relation is usually represented by the first-order approximation below. The equation below shows a linear dependence of the power consumption P with the total capacitance C and the operating frequency f . The power consumption is also proportional to the square of the supply voltage V_{DD} .

$$P = \frac{1}{2} \eta \cdot C \cdot V_{DD}^2 \cdot f$$

Where:

- k : technological factor (close to 1)
- C : Output load capacitance (Farad)
- V_{DD} : supply voltage (V)
- f : Clock frequency (Hz)
- η : switching activity factor (Between 0 and 1)

4.10 3-STATE INVERTER

Until now all the symbols produced the value logic '0' and logic '1'. However, if two outputs are connected together, as the left circuit shown below, it will provoke a circuit error. In order to avoid such conflicts, specific symbols are used, featuring the possibility to remain in a 'high impedance' state.

The 3-state symbol used below is *Bufif1*, and it consists of the logic buffer and an enable control. There also exists a 3-state inverter *Notif1*. The output remains in 'high impedance' as long as the enable 'En' is set to level '0'. The truth table of the 3-state inverter is reported below.

NOTIF1		
In	En	Out
x	0	High impedance
0	1	1
1	1	0

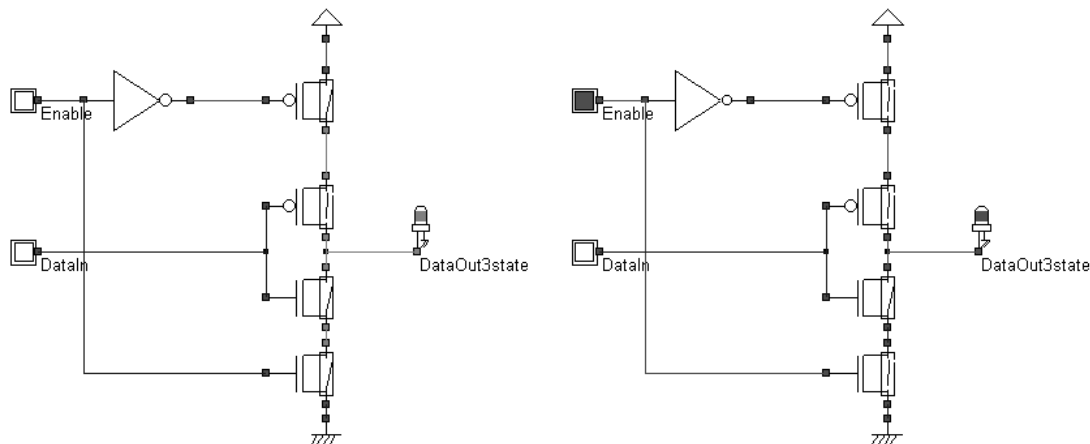


Figure 4-15: Simulation of the 3-state inverter (*CmosInv3State.SCH*)

4.11 Exercises

Create the layout and compare the static characteristics of the three following inverters: $W_n < W_p$, $W_n = W_p$, $W_n > W_p$. Which one seems to be well balanced? Justify your answer.

Using Microwind in $0.12\mu\text{m}$, draw an inverter ($W_{\text{PMOS}}=2\mu\text{m}$, $W_{\text{NMOS}}=1\mu\text{m}$) connected to a 100fF capacitor. Find R_{ON} and R_{OFF} of each transistor and calculate the delays. Compare their value to the simulated results.

5 Basic Gates

5.1 Introduction

Table 5-1 gives the corresponding symbol to each basic gate as it appears in the logic editor window as well as the logic description. In this description, the symbol & refers to the logical AND, | to Or, ~to INVERT, and ^ to XOR.








Name	Logic symbol	Logic equation
INVERTER		$Out = \sim in;$
AND		$Out = a \& b;$
NAND		$Out = \sim (a \& b);$
OR		$Out = (a b);$
NOR		$Out = \sim (a b);$
XOR		$Out = a \wedge b;$
XNOR		$Out = \sim (a \wedge b);$

Table 5-1. The list of basic gates

5.2 The Nand Gate

The truth-table and logic symbol of the NAND gate with 2 inputs are shown below. In DSCH, select the NAND symbol in the palette, add two buttons and one lamp as shown above. Add interconnects if necessary to link the button and lamps to the cell pins. Verify the logic behavior of the cell.

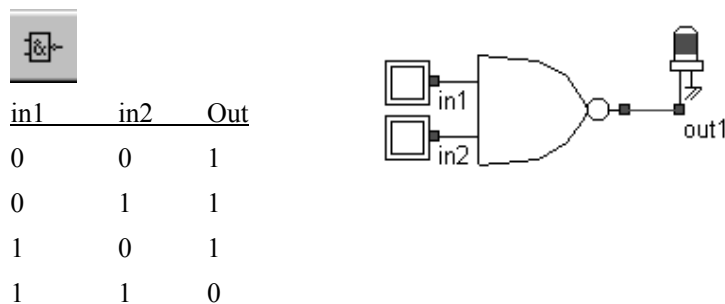


Fig. 5-1. The truth table and symbol of the NAND gate

In CMOS design, the NAND gate consists of two nMOS in series connected to two pMOS in parallel. The schematic diagram of the NAND cell is reported below. The nMOS in series tie the output to the ground for one single combination $A=1, B=1$.

For the three other combinations, the nMOS path is cut, but a least one pMOS ties the output to the supply VDD. Notice that both nMOS and pMOS devices are used in their best regime: the nMOS devices pass “0”, the pMOS pass “1”.

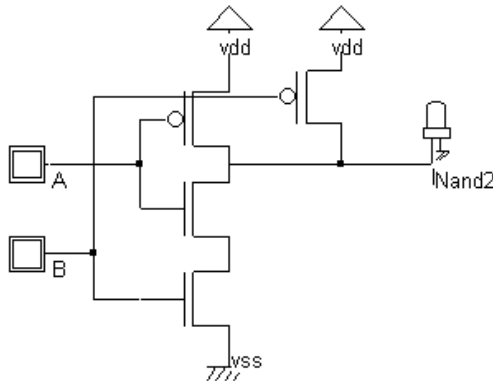
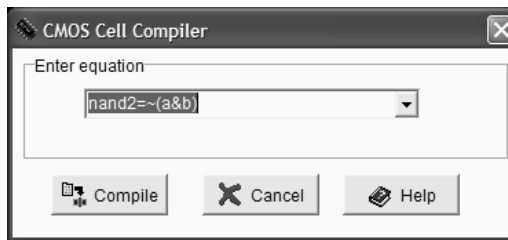
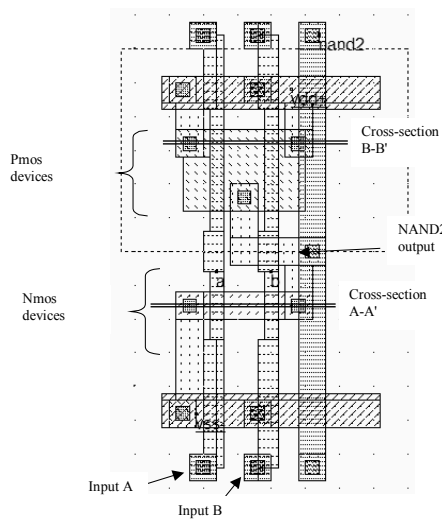


Fig. 5-2. The truth table and schematic diagram of the CMOS NAND gate design (NandCmos.SCH)

You may load the NAND gate design using the command **File → Read→NAND.MSK**. You may also draw the NAND gate manually as for the inverter gate. An alternative solution is to compile directly the NAND gate into layout with Microwind2. In this case, complete the following procedure:



In Microwind2, click on **Compile→Compile One Line**. Select the line corresponding to the 2-input NAND description as shown above. The input and output names can be by the user modified.



Click **Compile**. The result is reported above.

The compiler has fixed the position of VDD power supply and the ground VSS. The texts *A*, *B*, and *S* have also been fixed to the layout.

Default clocks are assigned to inputs *A* and *B*.

Fig. 5-3. A NAND cell created by the CMOS compiler.

The 2D-process viewer is a useful tool to display the two nMOS in series and the two pMOS in parallel. Select the corresponding icon and draw an horizontal line in the layout in the middle of the nMOS channels. The figure below appears. In fig. 5-4, the output is connected to the VSS supply only if A=1 and B=1.

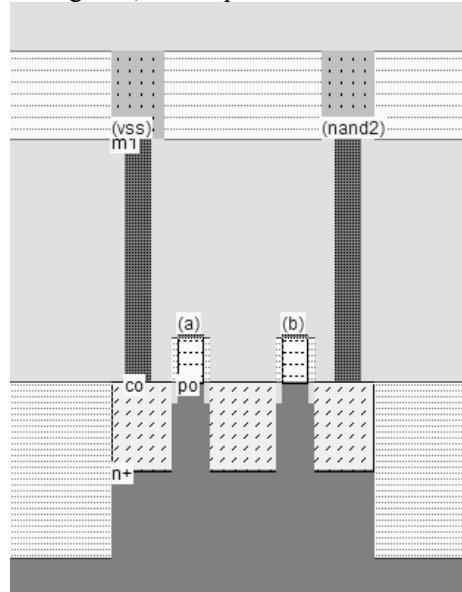
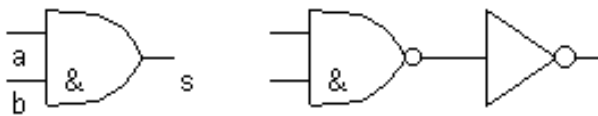


Fig. 5-4. The nMOS devices in serial in the NAND gate

The cell architecture has been optimized for easy supply and input/output routing. The supply bars have the property to connect naturally to the neighboring cells, so that specific effort for supply routing is not required. The input/output nodes are routed on the top and the bottom of the active parts, with a regular spacing to ease automatic channel routing between cells.

5.3 The AND gate

As can be seen in the schematic diagram and in the compiled results, the AND gate is the sum of a NAND2 gate and an inverter. The layout ready to simulate can be found in the file AND2.MSK. In CMOS, the negative gates (NAND, NOR, INV) are faster and simpler than the non-negative gates (AND, OR, Buffer). The cell delay observed in the figure 5-5 are significantly higher than for the NAND2 gate alone, due to the inverter stage delay.



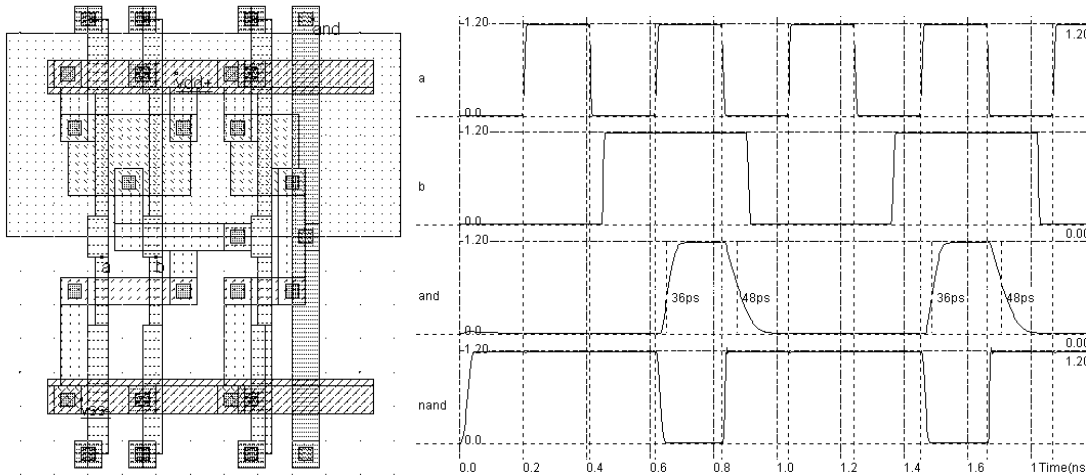


Fig. 5-5: Layout and simulation of the AND gate

5.4 The 3-Input OR Gate

The truth-table and the schematic diagram of the three-input OR gate are shown in Figure 5-6. You may use the DSCH2 logic editor to design a schematic diagram based on the OR gate, generate a Verilog description, and compile the text file in Microwind2. The OR gate is the sum of a NOR3 gate and an inverter. Figure 5-7 shows two implementations of the 2-input OR gate. The cell on the right is more compact thanks to an arrangement of the transistors.

OR 3 Inputs			
A	B	C	Or3
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

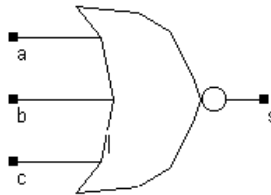


Fig. 5-6. The truth table and symbol of the OR3 gate

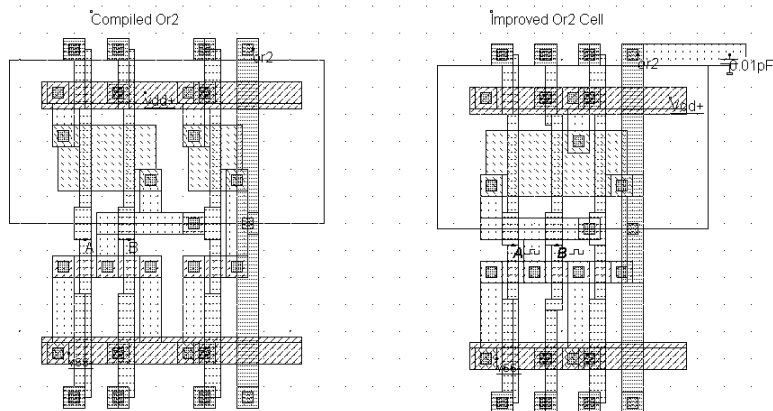
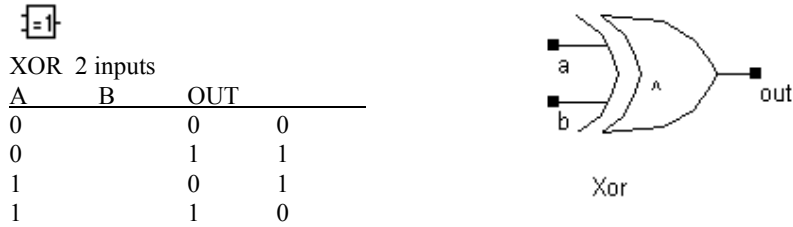


Fig. 5-7. Two versions of the OR2 gate (OR2.MSK)

5.5 The XOR Gate



The truth-table and the schematic diagram of the CMOS XOR gate are shown above. There exist many possibilities for implementing the XOR function into CMOS. The least efficient design, but the most forward, consists in building the XOR logic circuit from its Boolean equation.

The proposed solution consists of a transmission-gate implementation of the XOR operator. The truth table of the XOR can be read as follow: *IF B=0, OUT=A, IF B=1, OUT = Inv(A)*. The principle of the circuit presented below is to enable the A signal to flow to node NI if B=1 and to enable the *Inv(A)* signal to flow to node NI if B=0. The node OUT inverts NI, so that we can find the XOR operator. Notice that the nMOS and pMOS devices situated in the middle of the gate serve as pass transistors.

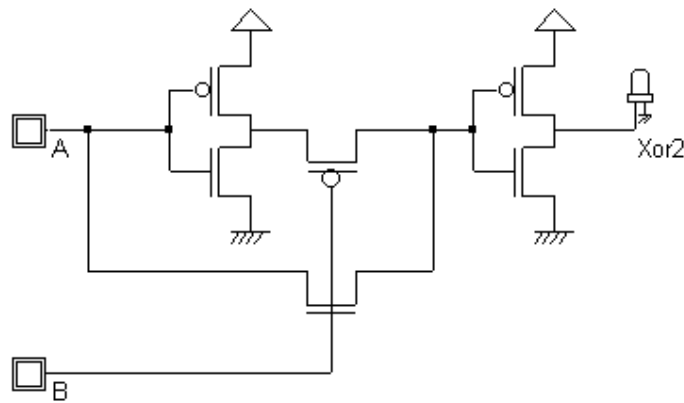


Fig. 5-8. The schematic diagram of the XOR gate (XORCmos.SCH)

You may use DSCH2 to create the cell, generate the Verilog description and compile the resulting text. In Microwind2, the Verilog compiler is able to construct the XOR cell as reported in Figure 5-9. You may add a visible property to the intermediate node which serves as an input of the second inverter. See how the signal, called *internal*, is altered by *Vtn* (when the nMOS is ON) and *Vtp* (when the pMOS is ON). Fortunately, the inverter regenerates the signal.

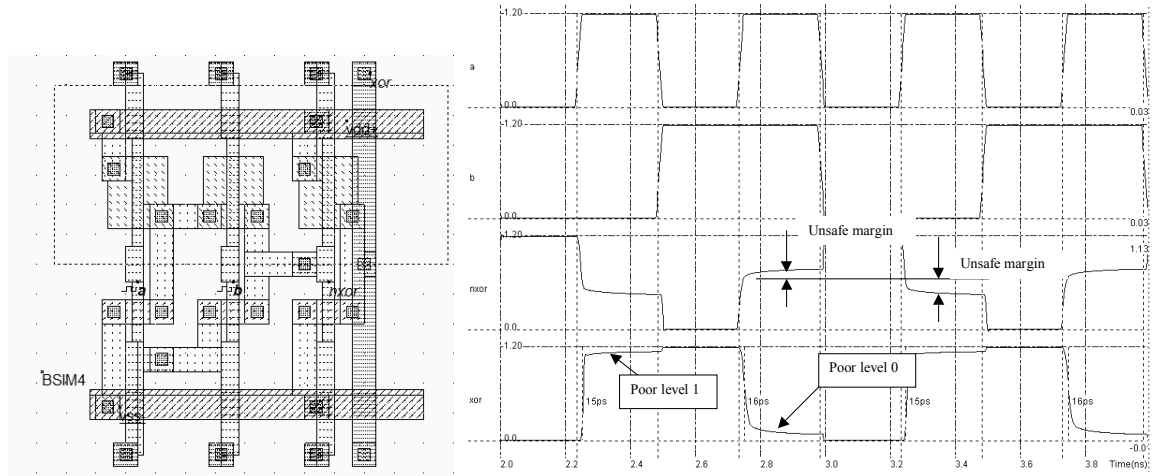


Fig. 5-9. Layout and simulation of the XOR gate (XOR.MSK).

5.6 Complex Gates

The complex gate design technique applies for any combination of operators **AND** and **OR**. The technique produces compact cells with higher performances in terms of spacing and speed than conventional logic circuits. To illustrate the concept of complex gates, let us take the example of the following Boolean equation:

$$F = A \& (B | C)$$

Bit operation	Verilog symbol used in complex gates
Not	~
And	&
Or	

The logic circuit corresponding to this equation is reported below. The circuit is built using a 2-input NOR and a 2-input AND cell, that is 10 transistors and three delay stages.

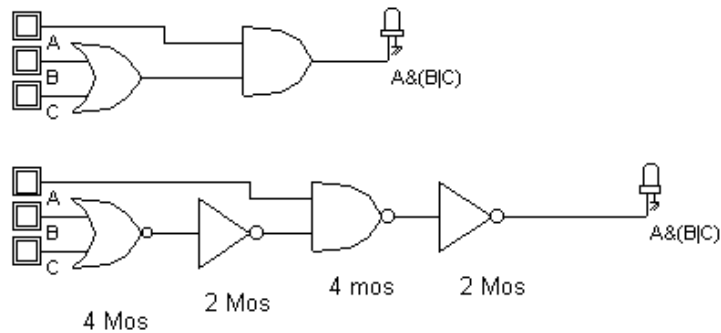


Fig. 5-10: The conventional schematic diagram of the function $F = A \& (B | C)$

A much more compact exists in this case (Figure 5-11), consisting in the following steps:

1. For the nMOS network, translate the AND operator '&' into nMOS in series, and the OR operator '|' into nMOS in parallel.

$$F_n = A \text{ series } (B \text{ parallel } C)$$

2. For the pMOS network, translate the AND operator '&' into pMOS in parallel, and the OR operator '|' into pMOS in series.

$$F_p = A \text{ parallel } (B \text{ series } C) \tag{Equ. 6-8}$$

3. If the function is non-inverting, as for F, the inverter is mandatory.

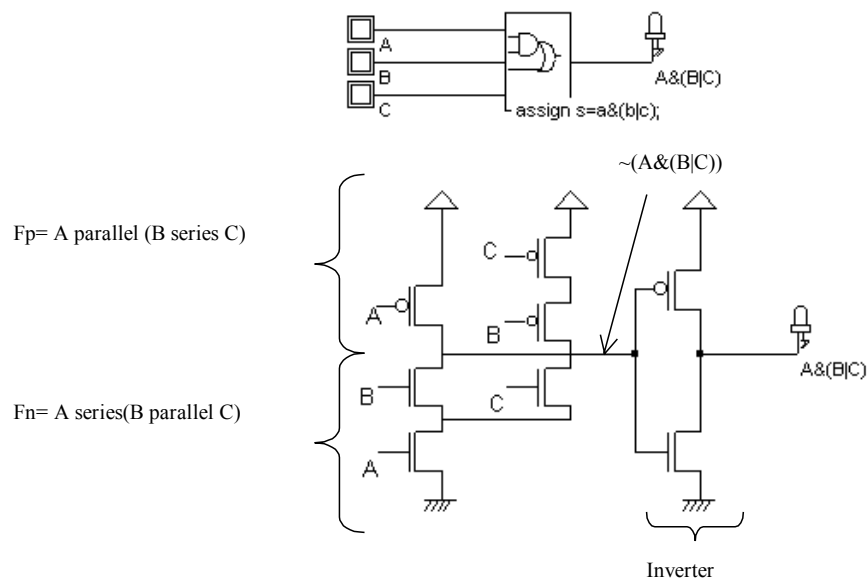


Fig. 5-11: The complex gate implementation of the function $F=A\&(B|C)$

Microwind2 is able to generate the CMOS layout corresponding to any description based on the operators AND and OR, using the command **Compile** → **Compile one line**. Using the keyboard, enter the cell equation, or modify the items proposed in the list of examples. In the one-line equation, the first parameter is the output name. In the present case that name is s. The sign '=' is obligatory. The '~' sign corresponds to the operation NOT and can be used only right after the '=' sign. The parenthesis '(' ') are used to build the function, where '&' is the AND operator, '|' is the OR operator, and '^' is the XOR operator.

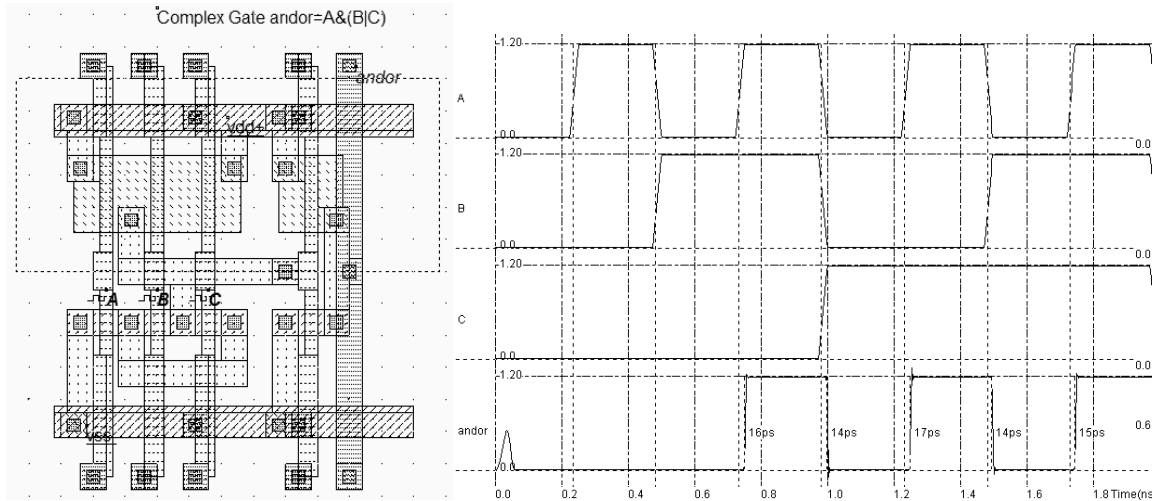


Fig. 5-12. A compiled complex gate and its analog simulation (ComplexABC.MSK)

Specific symbols exist to handle complex gate description in DSCH. The location of these symbols (3-input complex gate and 5-input complex gate) is shown in figure 5-13. At a double click inside the symbol, the menu shown in figure 5-14 appears. You must describe the logical function that links the output *s* to the inputs *a,b,c*. The syntax corresponds to the examples proposed in the previous paragraph (~for NOT, & for AND and | for OR). By using this behavioral description approach instead of building the function with basic cells, the switching performances of the gate are improved. Furthermore, the complex gate can be directly compiled into a compact layout using Microwind.

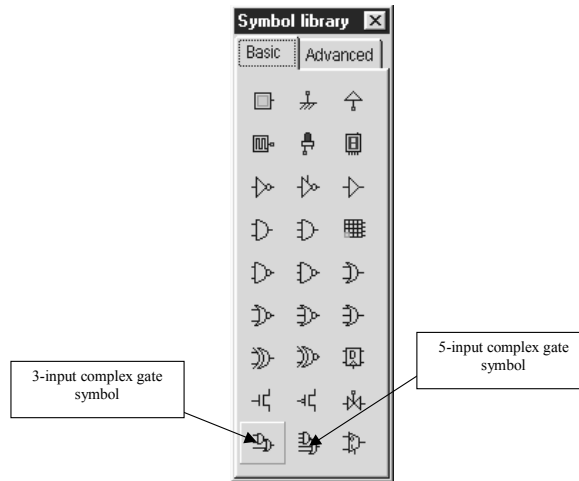


Fig. 5-13. Specific symbols proposed for complex gate description at logic level

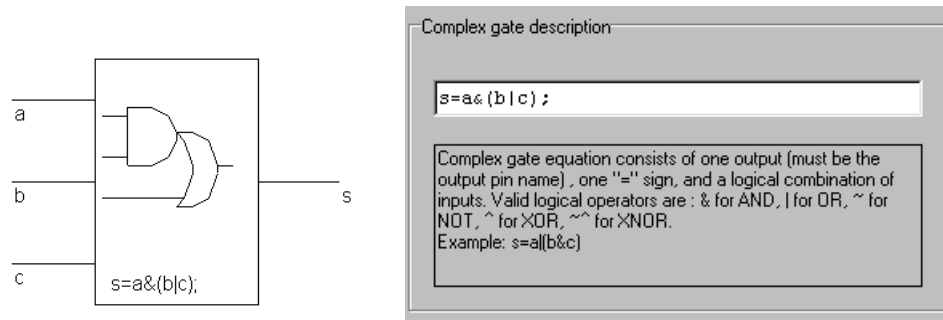


Fig. 5-14. The complex gate symbol and its logic description

5.7 Multiplexor

Multiplexing means transmitting a large amount of information through a smaller number of connections. A digital multiplexor is a circuit that selects binary information from one of many input logic signals and directs it to a single input line. The main component of the multiplexor is a basic cell called the transmission gate. The transmission gate let a signal flow if *Enable* is asserted. Remember that the n-channel MOS is only good for low signals, and the p-channel MOS is only good for high signals. To pass logic signals well, both a n-channel device and a p-channel device must be used.

Sel	In0	In1	f
0	x	0	0
0	x	1	1
1	0	x	0
1	1	x	1

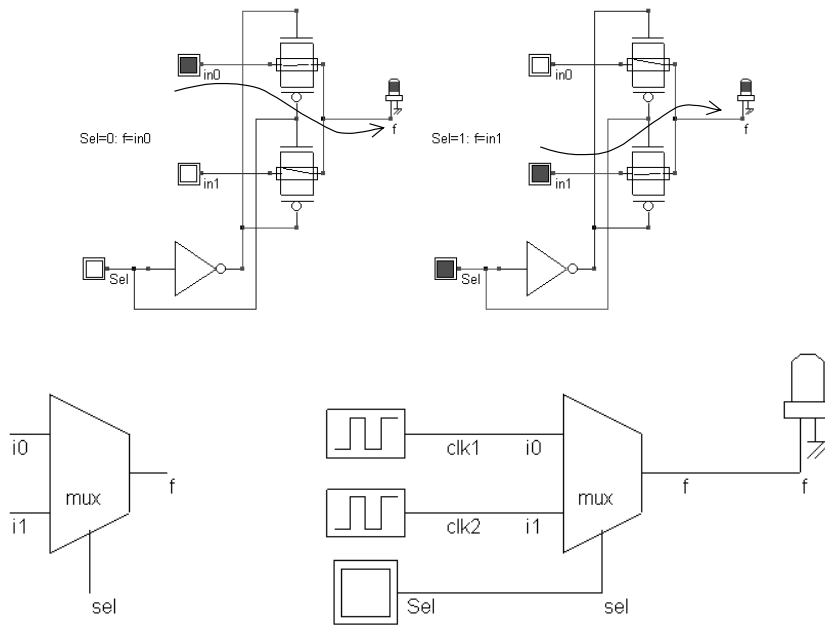


Fig. 5-15. The transmission gate used as a multiplexor (MUX.SCH)

In DSCH2, a transmission gate symbol exists (Figure 5-15). It includes the nMOS, pMOS and inverter cells. Concerning the layout, the channel length is usually the minimum length available in the technology, and the width is set large, in order to reduce the parasitic 'on' resistance of the gate.

5.8 8 to 1 Multiplexor

The multiplexor is a very useful function and has a multitude of application. The selection of a particular input line is controlled by a set of selection lines. Normally, there are 2^n input lines and n selection lines whose bit combinations determine which input is selected. Figure 5-16 shows the transmission gate implementation of the 8 to 1 multiplexor.

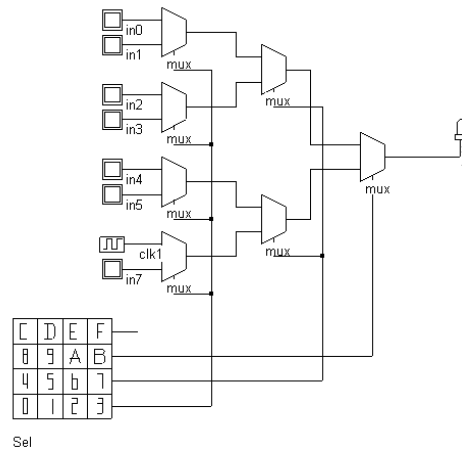


Fig. 5-16: 8 to 1 multiplexing based on transmission gates (Mux8to1.sch)

5.9 Interconnects and Vias

Up to 6 metal layers are available for signal connection and supply purpose. A significant gap exists between the $0.7\mu\text{m}$ 2-metal layer technology and the $0.12\mu\text{m}$ technology in terms of interconnect efficiency. Firstly, the contact size is 6 lambda in $0.7\mu\text{m}$ technology, and only 4 lambda in $0.12\mu\text{m}$. This features a significant reduction of device connection to metal and metal2, as shown in figure 5-17. Notice that a MOS device generated using $0.7\mu\text{m}$ design rules is still compatible with $0.12\mu\text{m}$ technology. But a MOS device generated using $0.12\mu\text{m}$ design rules would violate several rules if checked in $0.7\mu\text{m}$ technology.

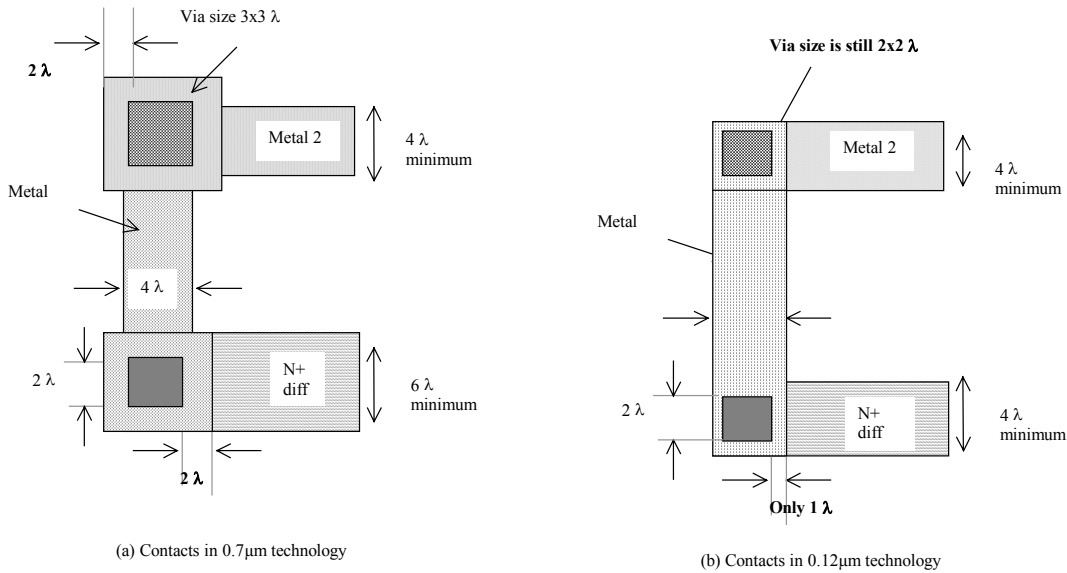


Figure 5-17: Contacts in 0.7µm technology require more area than in 0.12µm technology

Secondly, the stacking of contacts is not allowed in micro technologies. This means that a contact from poly to metal2 requires a significant silicon area (Figure 5-18a) as contacts must be drawn in a separate location. In deep-submicron technology (Starting 0.35µm and below), stacked contacts are allowed (Figure 5-18b).

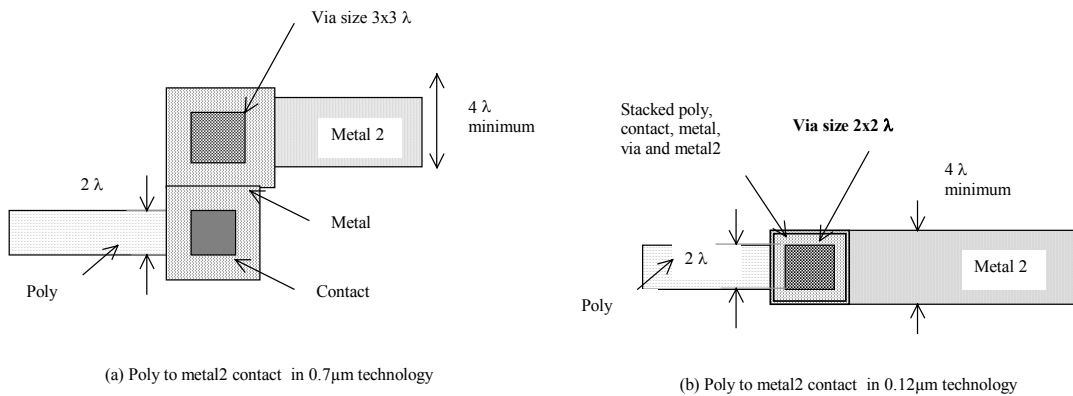


Figure 5-18: Stacked vias are allowed in 0.12µm technology, which saves a significant amount of silicon area compared to 0.7µm design style.

Metal layers are labeled according to the order in which they are fabricated, from the lower level 1 (metal 1) to the upper level (metal 6 in 0.12µm). Each layer is embedded into a silicon oxide (SiO₂) which isolates layers from each other. The connection material between diffusion and metal is called "contact". The same layer is also used to connect poly to metal, or poly2 to metal. The connection material between metal and metal2 is called "via". By extension, the material that connects metal2 to metal3 is "via2", metal3 to metal4 "via3", etc..

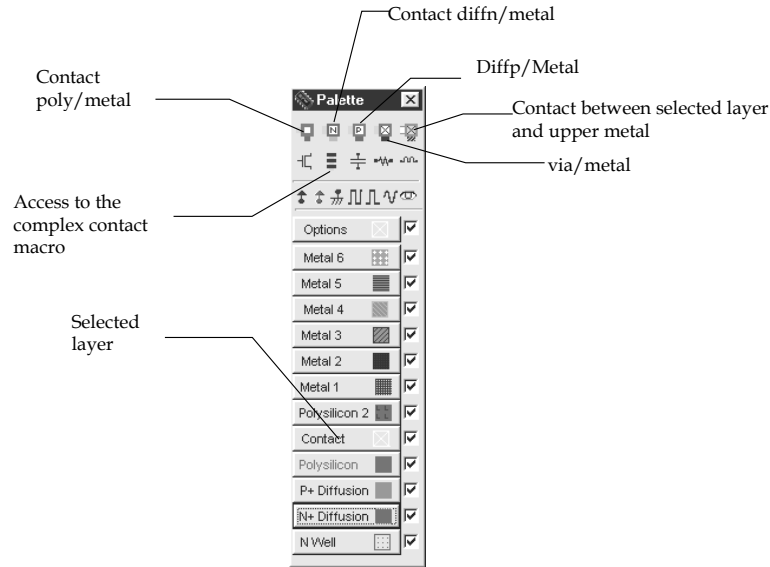


Figure 5-19: Access to basic contact macros

In Microwind, specific macros are accessible to ease the addition of contacts in the layout. These macros may be found in the palette, as shown in figure 5-19. As an example, you may instantiate a design-error free poly/metal contact by a click on the upper left corner icon in the palette. You may obtain the same result by drawing one box of poly (4x4 lambda), one box of metal (4x4 lambda) and one box of contact (2x2 lambda), according to design rules.

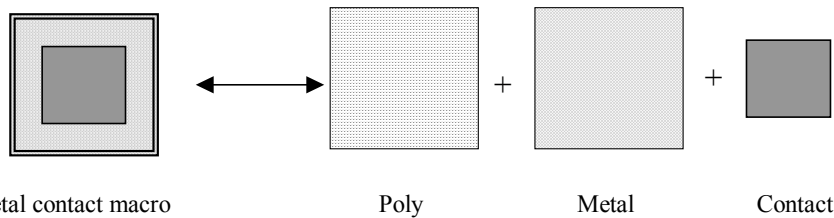


Figure 5-20: Access to basic contact macros

	<p>Additionally, an access to complex stacked contacts is proposed thanks to the icon "complex contacts" situated in the palette, second row, second column. The screen reported in figure 5-7 appears. By default you create a contact from poly to metal1, and from metal1 to metal2. Change the tick to build more complex stacked contacts.</p>
	<p>A convenient command exists in Microwind to add the appropriate contact between two layers. Let us imagine that we need to connect two signals, one routed in polysilicon and an other in metal3. Rather than invoking the complex macro command, we may just select the icon "connect layers". As a result a stack of contacts is inserted at the desired location to connect the lower layer to the upper layer. An illustration of this command is shown in figure 5-9.</p>

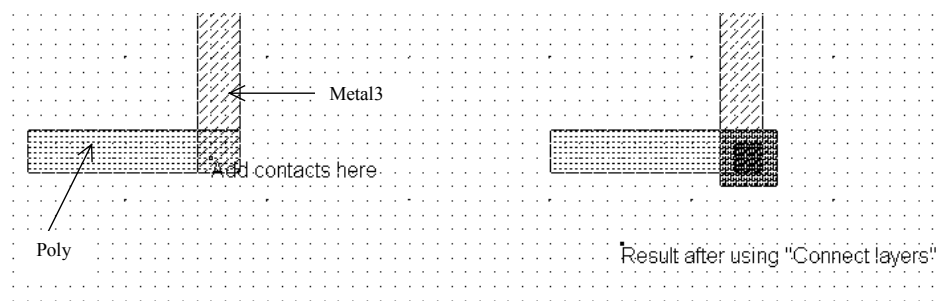


Figure 5-21: The command "Connect layers" inserts the appropriate stacked contacts to build the connection between the desired layers (ConnectLayers.MSK)

5.10 Exercises

Design a CMOS 3-input XOR using CMOS And-OR-Invert logic.

Design the following function using CMOS And-OR-Invert logic and try to find continuous diffusions.

$$F = \sim(A \& B | c \& (A | B))$$

Using Microwind, compare the switching point voltage of a 3-input NOR gate (minimum size MOS) to the switching point voltage of a 3-input NAND gate (minimum size MOS). Which one is closer to the ideal and why?

6 Arithmetics

This chapter introduces basic concepts concerning the design of arithmetic gates. The adder circuit is presented, with its corresponding layout created manually and automatically. Then the comparator, multiplier and the arithmetic and logic unit are also discussed. This chapter also includes details on a student project concerning the design of binary-to-decimal addition and display.

6.1 Unsigned Integer format

The two classes of data formats are the integer and real numbers. The integer type is separated into two formats: unsigned format and signed format. The real numbers are also sub-divided into fixed point and floating point descriptions. Each data is coded in 8,16 or 32 bits. We consider here unsigned integers, as described in figure 6-1.

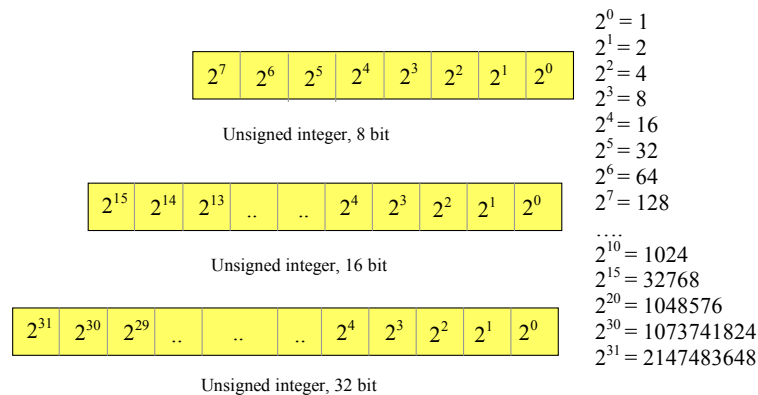


Fig. 6-1. Unsigned integer format

6.2 Creating Arithmetic Circuits From Logic Design

When the design starts to be complex, the manual layout is very difficult to conduct, and an automatic approach is preferred, at the price of a less compact design and more rigid implementation methodology. The steps from logic design to layout validation are reported in figure 6-2. The schematic diagram constructed using DSCH is validated first at logic level.

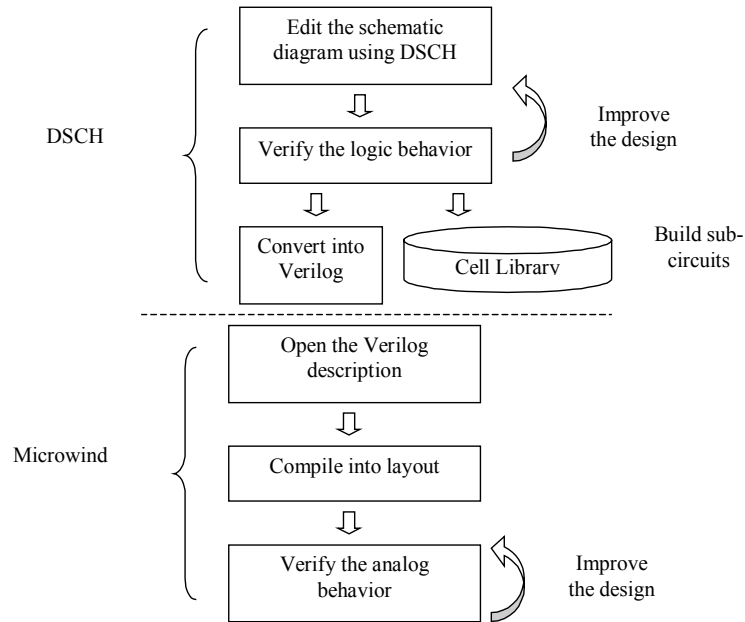


Figure 6-2: Design flow from logic design to layout implementation

A specific command in DSCH creates the Verilog description of the logic design, including the list of primitives and some stimulation information. This Verilog text file is understood by Microwind to construct the corresponding layout, with respect to the desired design rules. The supply properties and most stimulation information are added to the layout automatically, according to the logic simulation. Finally, the analog simulation permits to validate the initial design and verify its switching and power consumption performances.

6.3 Half-Adder Gate

The Half-Adder gate truth-table and schematic diagram are shown in Figure 6-3. The SUM function is made with an XOR gate, the Carry function is a simple AND gate.

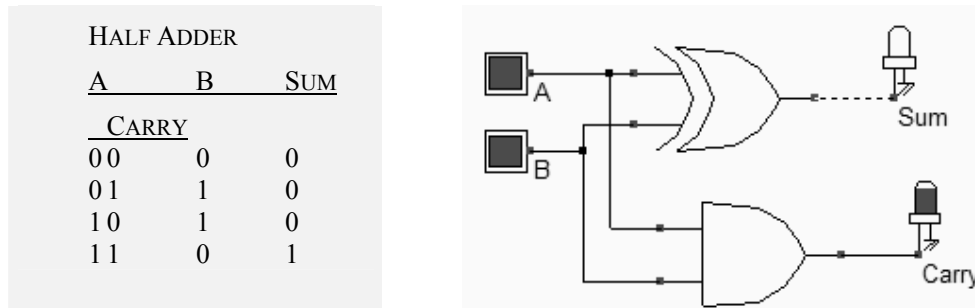


Fig. 6-3. Truth table and schematic diagram of the half-adder gate (HADD.MSK).

FULL CUSTOM LAYOUT	You may create the layout of the half-adder fully by hand in order to create a compact design. Use the polysilicon and metal1 layers for short connections only, because of the high resistance of these materials. Use Poly/Metal, Diff/Metal contact macros situated in the upper part of the Palette menu to link the layers together.
LAYOUT LIBRARY	Load the layout design of the Half-Adder using File → Open and loading the file HADD.MSK.

VERILOG COMPILING. Use DSCH2 to create the schematic diagram of the half-adder. Verify the circuit with buttons and lamps. Save the design under the name hadd.sch using the command **File → Save As**. Generate the Verilog text by using the command **File → Make Verilog File**. In Microwind2, click on the command **Compile → Compile Verilog File**. Select the text file hadd.txt.

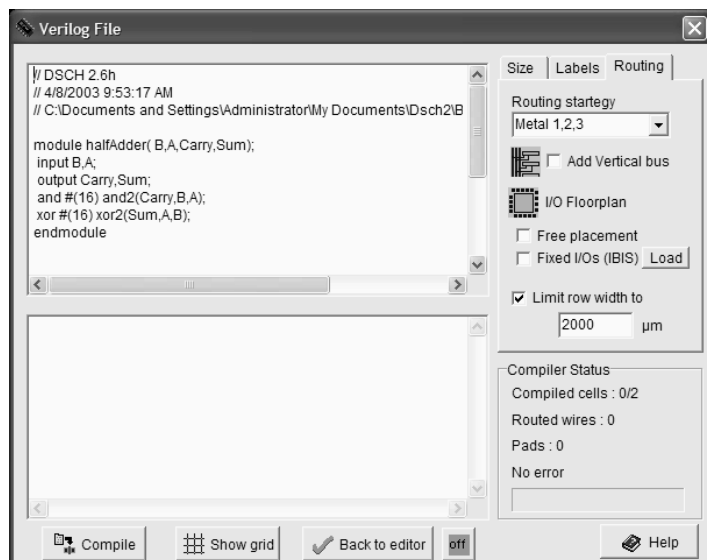


Fig. 6-4 Compile the Verilog text into layout

Click **Compile**. When the compiling is complete, the resulting layout appears shown below. The XOR gate is routed on the left and the AND gate is routed on the right. Now, click on **Simulate → Start Simulation**. The timing diagrams of figure 6-4 appear and you should verify the truth table of the half-adder. Click on **Close** to return to the editor

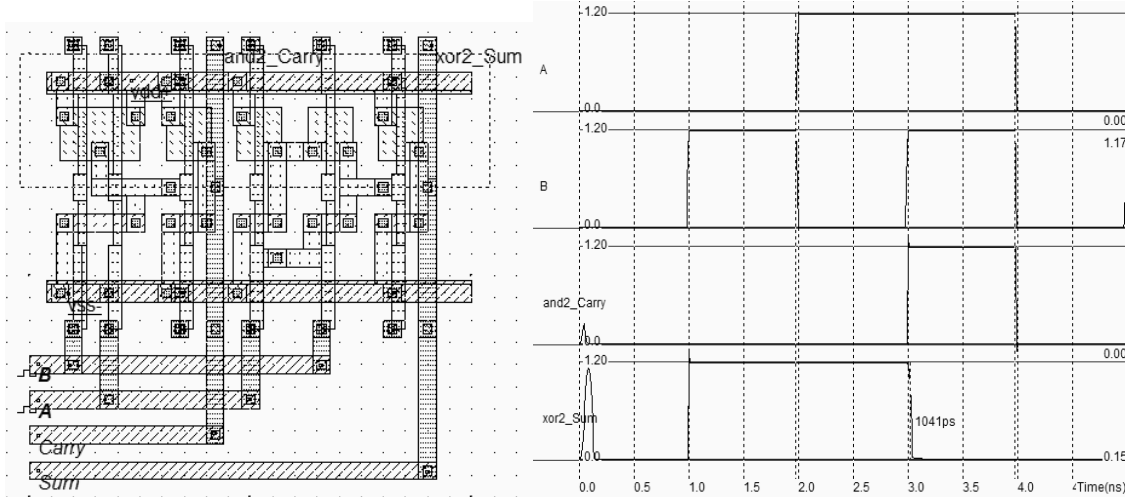


Fig. 6-5. Compiling and simulation of the half-adder gate (Hadd.MSK)

6.4 Full-Adder Gate

The truth table and schematic diagram for the full-adder are shown in Figure 6-6. The SUM is made with two XOR gates and the CARRY is a combination of NAND gates, as shown below. The most straightforward implementation of the CARRY cell is $AB+BC+AC$. The weakness of such a circuit is the use of positive logic gates, leading to multiple stages. A more efficient circuit consists in the same function but with inverting gates.

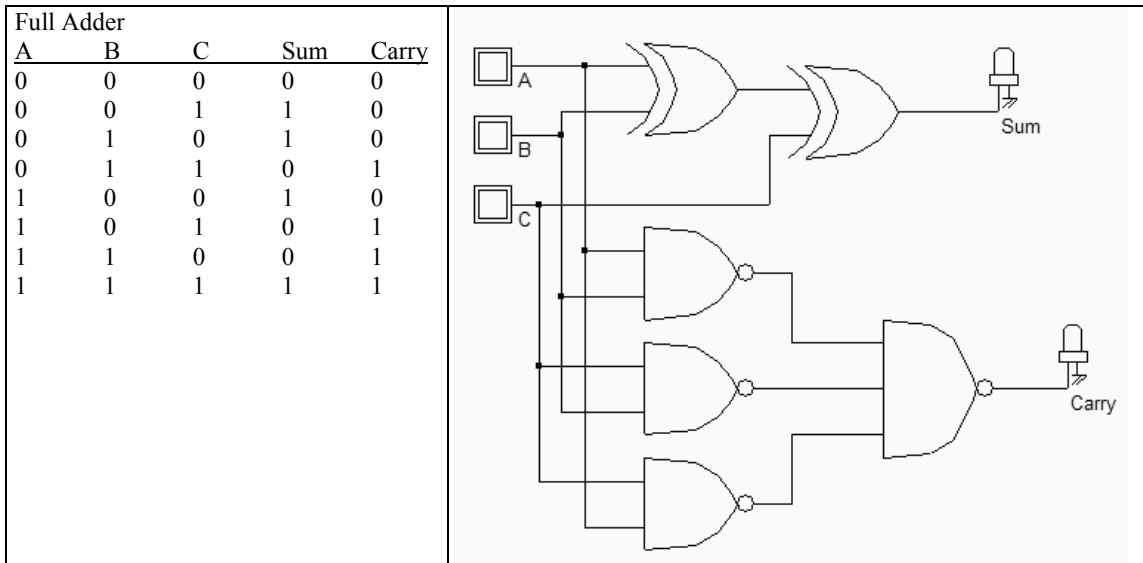


Fig. 6-6. The truth table and schematic diagram of a full-adder(FADD.SCH)

6.5 Full-Adder Symbol in DSCH

When invoking **File** → **Schema to new symbol**, the screen of figure 6-7 appears. Simply click **OK**. The symbol of the full-adder is created, with the name *FullAdder.sym* in the current directory. Meanwhile, the Verilog file **fullAdder.txt** is generated, which contents is reported in the left part of the window (Item **Verilog**).

We see that the XOR gates are declared as primitives while the complex gate is declared using the **Assign** command, as a combination of AND (&) and OR (|) operators. If we used AND and OR primitives instead, the layout compiler would implement the function in a series of AND and OR CMOS gates, losing the benefits of complex gate approach in terms of cell density and switching speed.

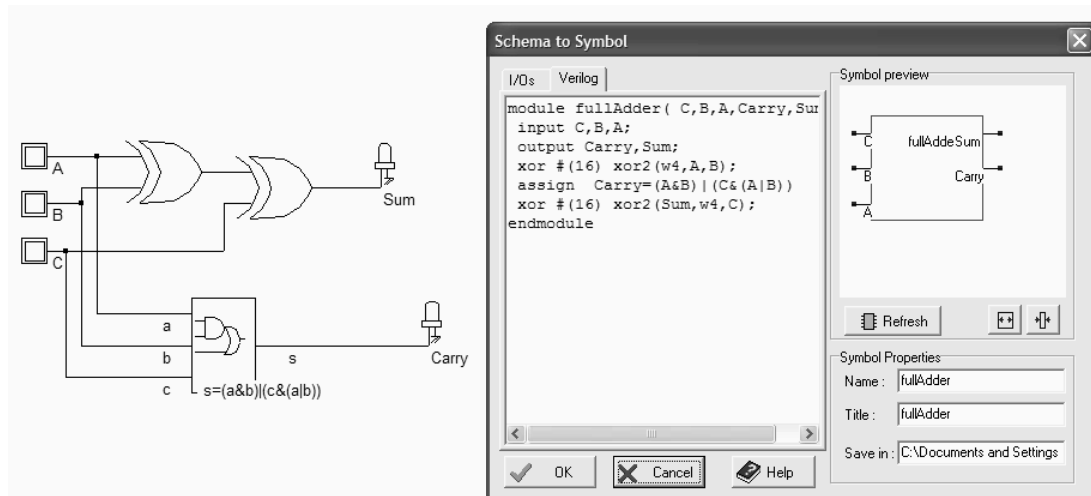


Fig. 6-7 Verilog description of the full adder (FullAdder.SYM)

Use the command **Insert** → **User Symbol** to include this symbol into a new circuit. For example, the circuit **FaddTest** includes the hierarchical symbol and verifies its behavior. Three clocks with 20,40 and 80ns are declared as inputs. Using such clocks is of particular importance to scan all possible combination of inputs, by following line by line the truth table. What we observe in the chronograms of figure 6-8 is the addition of three numbers, which follows the requested result given in the initial truth table. The addition of *A*, *B* and *C* appears in the output *sum*. For example, at time $t=70\text{ns}$, $A=B=C=1$, the output is 3 after a little delay.

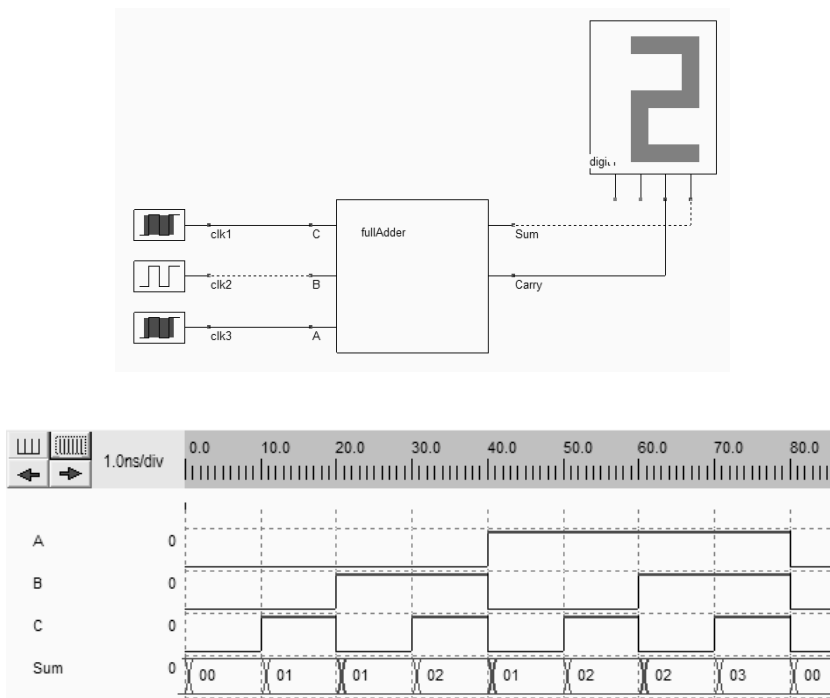


Fig. 6-8. Testing the new Adder symbol using clocks (FaddTest.SCH)

6.6 Full-Adder Layout

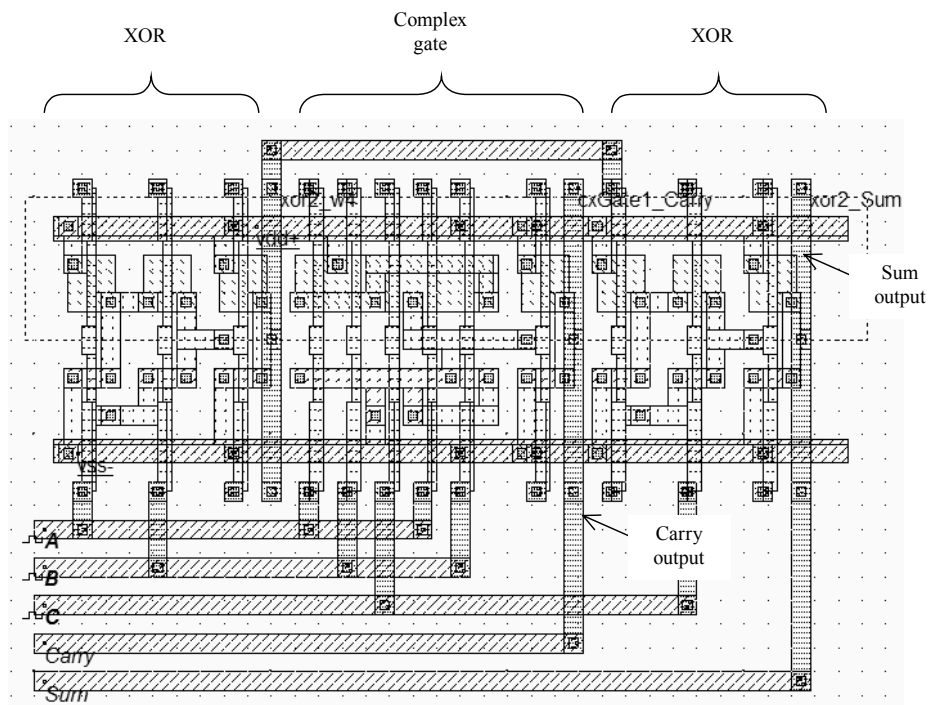


Fig. 6-9. The full adder compiled from the Verilog description (FullAdder.MSK)

You may create the layout of the full-adder by hand in order to create a compact design. Notice that the AND/OR combination of cells may be replaced by a complex gate. Alternatively, you may use DSCH2 to create the schematic diagram of the full-adder and compile it directly into layout. In the compiled version of the adder shown in figure 6-9, the XOR gate is routed on the left and the AND gate is routed on the right. Click on **Simulate**→**Start Simulation** and verify the truth table of the full-adder.

6.7 Four-Bit Adder

The four-bit adder circuit includes adders in serial to perform the arithmetic addition. The result of each stage propagates to the next one, from the right to left (Figure 6-10).

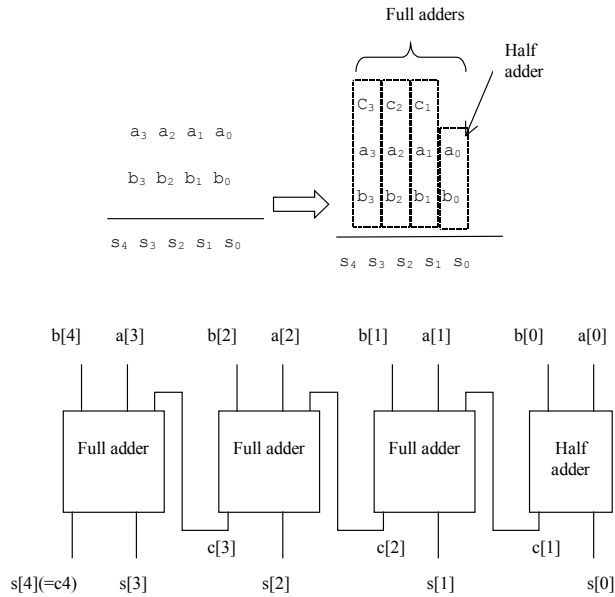


Fig. 6-10. Structure of the 4-bit ripple-carry adder

The circuit of figure 6-11 allows a four-bit addition between two numbers A3,A2,A1,A0 and B3,B2,B1,B0. Insert the user-defined Fadd.sym symbol using the command **Insert** → **User Symbol**. In DSCH2, the A and B numbers are generated by keyboard symbols, as reported below. Also notice the hexadecimal displays.

The two displays are connected to the identical data, but are configured in different mode: hexadecimal format for the right-most display, and integer mode for the left-most display. To change the display mode, double click inside the symbol, and change the format in the symbol property window. The default option is the hexadecimal format. The unsigned integer format is used in the schematic diagram of figure 6-11 for the left display. Integer and fixed point display formats are used for arithmetic circuits.

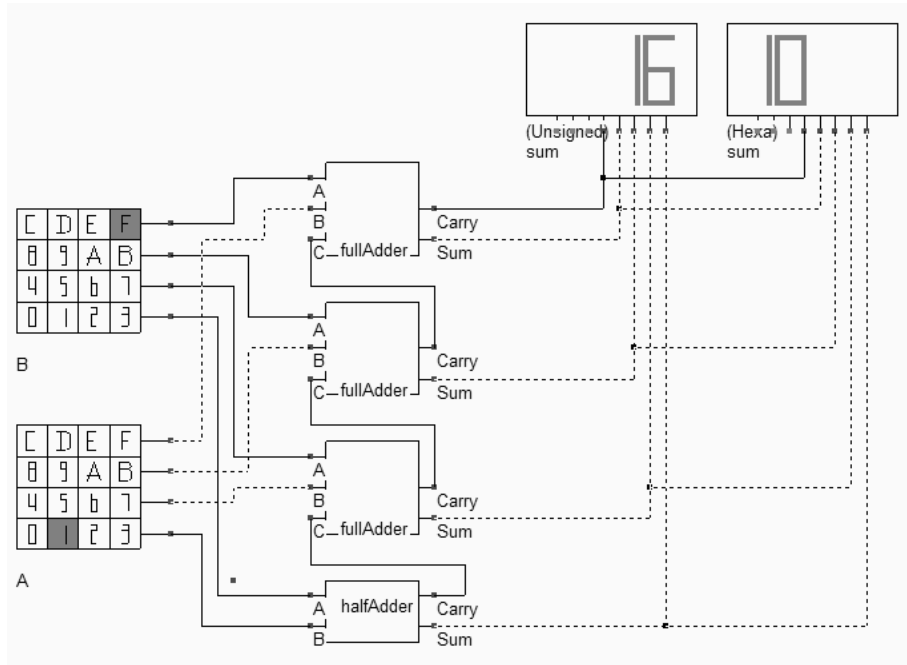


Fig. 6-11. Schematic diagram of the four-bit adder and some examples of results (Add4.SCH).

Figure 6-12 details the four-bit adder layout based on the full-custom cell design, with the corresponding simulation. In Microwind2, the command **Edit** → **Duplicate X,Y** has been used to duplicate the full-adder layout vertically.

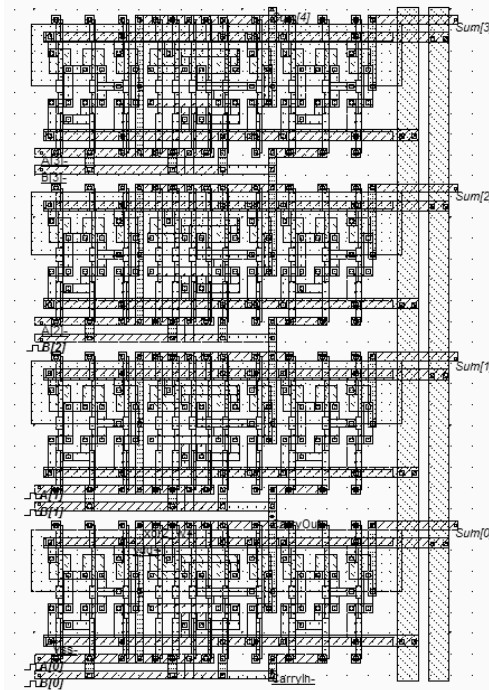


Fig. 6-12. Manual design of the four-bit adder (ADD4.MSK).

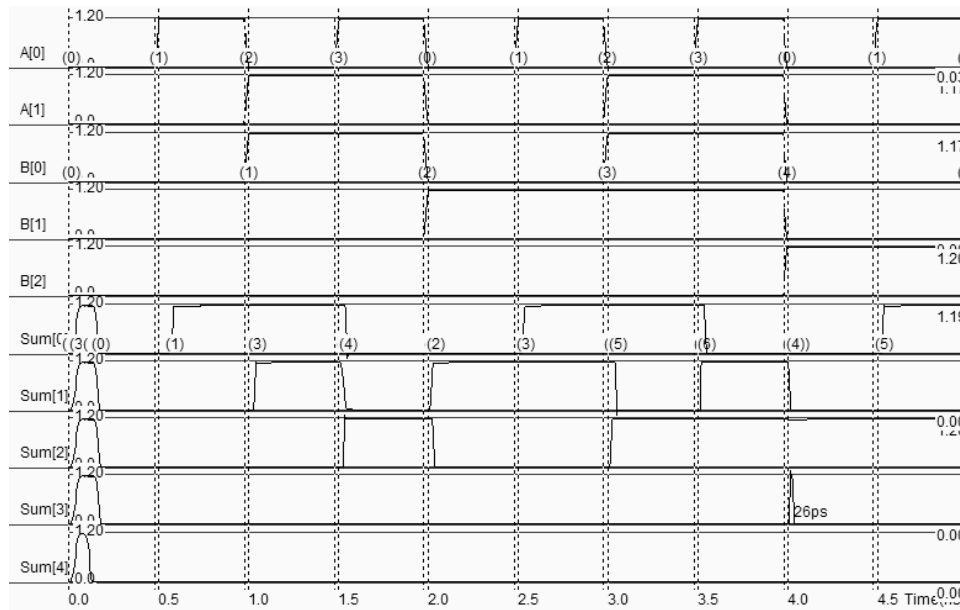


Fig. 6-13. Simulation of the four-bit adder (ADD4.MSK).

6.8 Comparator

The truth table and the schematic diagram of the comparator are given below. The $A=B$ equality represents an XNOR gate, and $A>B$, $A<B$ are operators obtained by using inverters and AND gates.

Comparator				
A	B	A>B	A<B	A=B
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

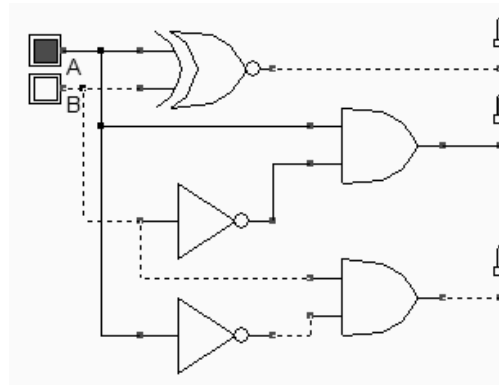


Fig. 6-14 The truth table and schematic diagram of the comparator (COMP.SCH).

Using DSCH2, the logic circuit of the comparator is designed and verified at logic level. Then the conversion into Verilog is invoked (**File → Make verilog File**). Microwind2 compiles the verilog text into layout. The simulation of the comparator is given in Figure 6-15. The XNOR gate is located at the left side of the design. The inverter and NOR gates are at the right side. After the initialization, $A=B$ rises to 1. The clocks A and B produce the combinations 00,01,10 and 11.

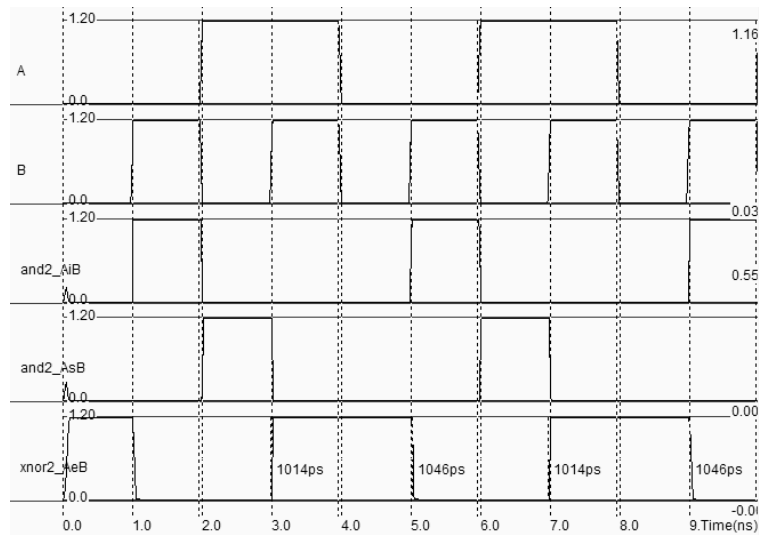


Fig. 6-15. Simulation of a comparator (COMP.MSK).

6.9 Arithmetic and Logic Unit

The purpose of this chapter is to design and simulate an 8-bit arithmetic/logic unit (ALU). The ALU is a digital function that implements basic micro-operations on the information stored in registers. In figure 6-16, the ALU receives the information from the registers *A* and *B* and performs a given operation as specified by the control function *F*. The result is produced on *S*.

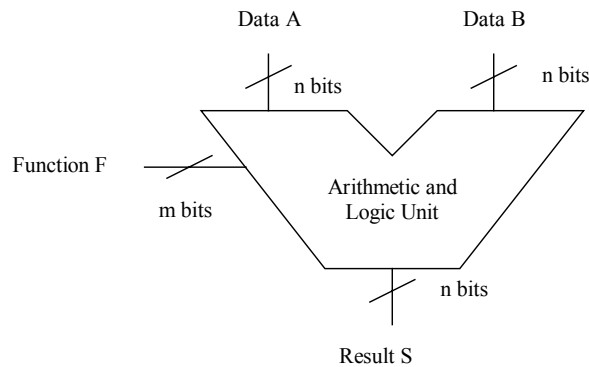


Fig. 6-16. Principles for the Arithmetic and Logic Unit

In DSCH, a simplified model of the Intel 8051 micro-controller is included. The 8051 core includes an arithmetic and logic unit to support a huge set of instructions. Most of the results are The data format is 8 bit. We consider here the following instructions, listed in table 6-3. Some instructions do not appear in this list, such as the multiplication and division.

Mnemonic	Type	Description
CLR	Clear	Clear the accumulator
CPL	Complement	Complements the accumulator, a bit or a memory contents. All the bits will be reversed.
ADD	Addition	Add the operand to the value of the accumulator, leaving the resulting value in the accumulator.
SUBB	Subtractor	Subtracts the operand to the value of the accumulator, leaving the resulting value in the accumulator.
INC	Increment	Increment the content of the accumulator, the register or the memory.
DEC	Decrement	Decrement the content of the accumulator, the register or the memory.
XRL	XOR operator	Exclusive OR operation between the accumulator and the operand, leaving the resulting value in the accumulator.
ANL	AND operator	AND operation between the accumulator and the operand, leaving the resulting value in accumulator.
ORL	OR operator	OR operation between the accumulator and the operand, leaving the resulting value in accumulator.
RR	Rotate right	Shifts the bits of the accumulator to the right. The bit 0 is loaded into bit 7.
RL	Rotate left	Shifts the bits of the accumulator to the left. The bit 7 is loaded into bit 0.

Table 6-3. Some important instructions implemented in the ALU of the 8051 micro-controller

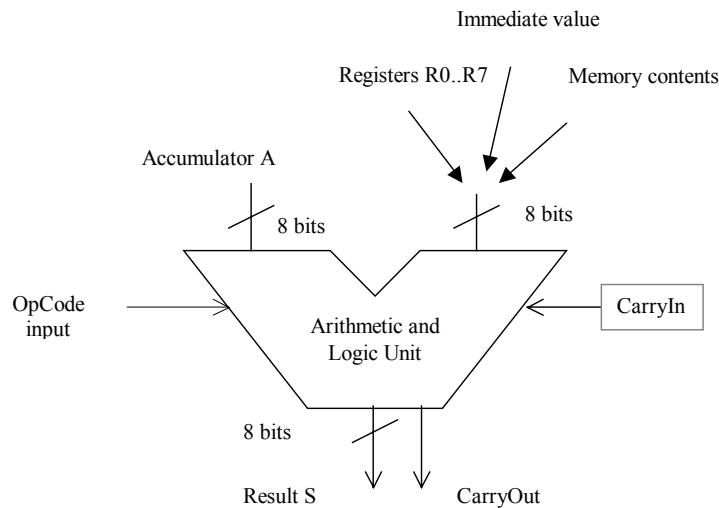


Figure 6-17. The arithmetic and logic unit of the 8051

For example:

- ADD A,R0 (Opcode 0x28) overwrites the accumulator with the result of the addition of A and the content of R0.
- SUBB A,#2 (Opcode 0x94 0x02) overwrites the accumulator with the result of the subtraction of A and the sum of the Carry and the byte 0x02.
- INC A (0x04) increments the content of the accumulator.
- DEC A (0x14) Decrements the content of the accumulator.
- ANL A,#10 (0x54) overwrites the accumulator with by the AND-gating of A and the constant 0x10.

- ORL A,R7 (0x4F) overwrites the accumulator with by the OR-gating of A and the content of R7.
- XRL A, R1 (0x69) overwrites the accumulator with the result of the XOR-gating of A and the content of the internal register R1.

A simplified model of the 8-bit micro-controller 8051 exists in DSCH2. You can add the corresponding symbol (8051.SYM) using the command **Insert** → **User Symbol**, as the symbol is not directly accessible through the symbol palette. The symbol consists mainly of general purpose input/output ports (*P0,P1,P2* and *P3*), a *clock* and a *reset* control signals. The basic connection consists of a clock on the *Clock* input and a button on the *Reset* input (Figure 6-18).

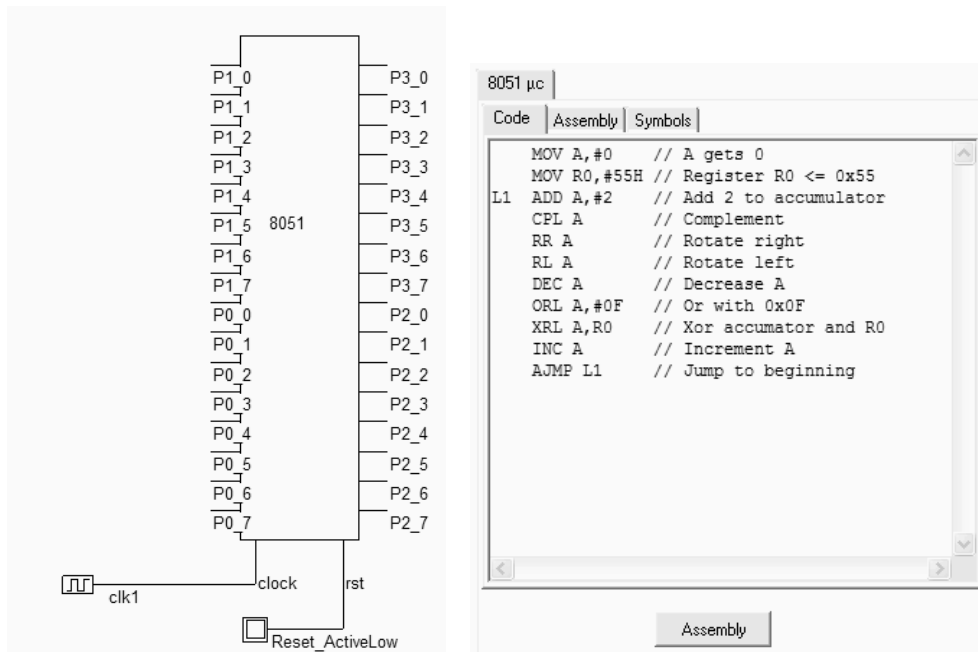


Figure 6-18. The 8051 symbol and its embedded software (8051.SCH)

After a double-click in the symbol, the embedded code appears. That code may be edited and modified (Figure 6-19). When the button **Assembly** is pressed, the assembly text is translated into executable binary format. Once the logic simulation is running, the code is executed as soon as the reset input is deactivated. The value of the program counter, the accumulator A, the current *op_code* and the registers is displayed. In the chronograms, the accumulator variations versus the time are displayed. It can be noticed that this core operates with one single clock cycle per instruction, except for some instructions such as MOV (Move data) and AJMP (Jump to a specific address).

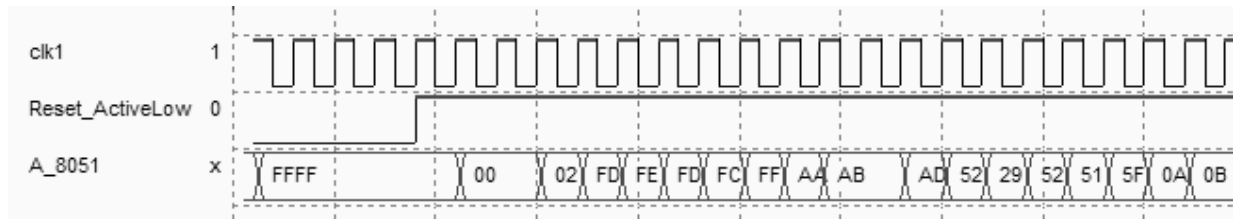


Figure 6-19. The simulation of the arithmetic and logic operation using the 8051 micro-controller (8051.SCH)

6.10 Model of the PIC 16f84

Dsch2 also includes the model of the PIC16f84 micro-controller. An example file can be found in **16f84adder.SCH**. Double click the 16f84 symbol, and click **Assembly** to convert the text lines into binary executable code.

```

; Simple program to add two numbers
;
oper1 EQU 0x0c
oper2 EQU 0x0d
result EQU 0x0e

org 0

movlw 5
movwf oper1
movlw 2
movwf oper2
movf oper1,0
addwf oper2,0
movwf result
sleep

```

Then click **OK**, run the simulation. Click the *Reset* button to activate the processor. The default code realizes the addition of two numbers (Instruction `addwf`) and stores the result in the internal registers. Modify the code to perform the AND (Instruction `andwf`), OR (Instruction `iorwf`), XOR (Instruction `xorwf`) and SUB (Instruction `subwf`) operations.

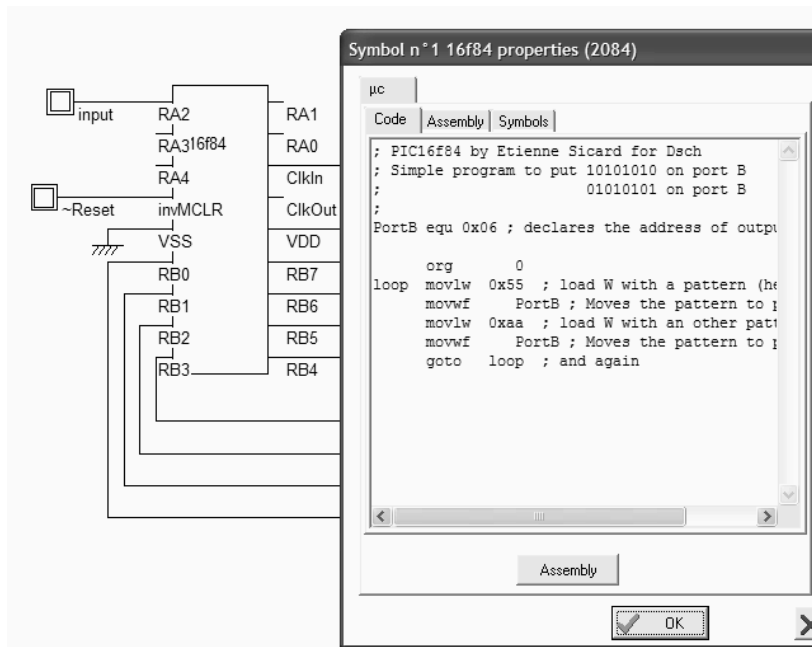


Figure 6-20. Simulation of the PIC 16f84 (16f84.SCH)

7 Latches

This chapter details the structure and behavior of latch circuits. The RS Latch, the D Latch, the edge-sensitive register and the counter are presented.

7.1 Basic Latch

The basis for storing an elementary binary value is called a latch. The simplest CMOS circuit is made from 2 inverters.

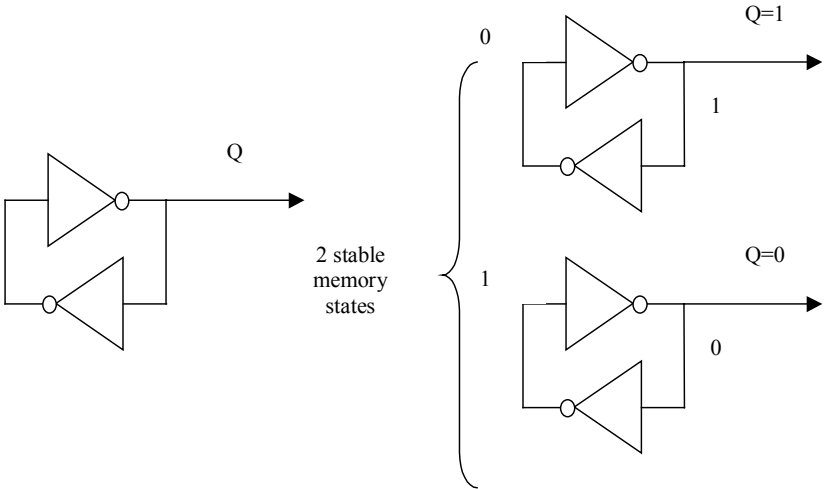


Figure 7-1: Elementary memory cell based on an inverter loop

7.2 RS Latch

The RS Latch, also called Set-Reset Flip Flop (SR FF), transforms a pulse into a continuous state. The RS latch can be made up of two interconnected NAND gates, inspired from the two chained inverters of figure 7-2. In that case, the *Reset* and *Set* inputs are active low. The memory state corresponds to *Reset=Set=1*. The combination *Reset=Set=0* should not be used, as it means that *Q* should be *Reset* and *Set* at the same time. Furthermore, the simultaneous change from *Reset=Set=0* to *Reset=Set=1* provokes what is called the metastable state, that corresponds to a parasitic ring effect that may jeopardize the behavior of the whole circuit.

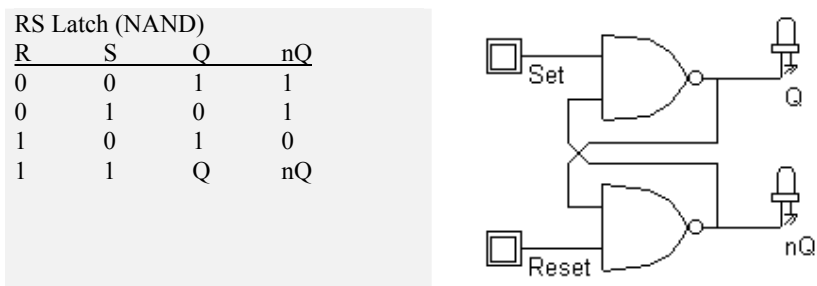


Fig. 7-2. The truth table and schematic diagram of a RS latch made (RSNor.SCH)

FULL CUSTOM LAYOUT. You may create the layout of RS latch manually. The two NAND gates may share the VDD and VSS supply achieving continuous diffusions.

LAYOUT COMPILING. Use DSCH2 to create the schematic diagram of the RS latch. Verify the circuit with buttons and lamps. Save the design under the name `RS.sch` using the command **File** → **Save As**. Generate the Verilog text by using the command **File** → **Make Verilog File**. In Microwind2, click on the command **Compile** → **Compile Verilog File**. Select the text file `RS.txt`. Click on **Compile**. When the compiling is complete, the resulting layout appears as shown below. The NOR implementation of the RS gate is completed.

```

module RSNor( Reset, Set, Q, nQ );
  input Reset, Set;
  output Q, nQ;
  nor nor1( Q, nQ, Reset );
  nor nor2( nQ, Set, Q );
endmodule

```

With the *Reset* and *Set* signals behaving like clocks, the memory effect is not easy to illustrate. A much better approach consists in declaring pulse signals with an active pulse on *Reset* followed by an active pulse on *Set*. Consequently, you must change the clock property into a pulse property. For NOR implementation, the pulse is positive.

1. Select the **Pulse** icon. Click on the node *Reset*.
2. Click the brush to clear the existing pulse properties of the pulse.
3. Enter the desired sequence, for example 01000, and click **Insert**. A piece-wise-linear sequence is generated in the table, describing the 01000 waveform in an analog way.

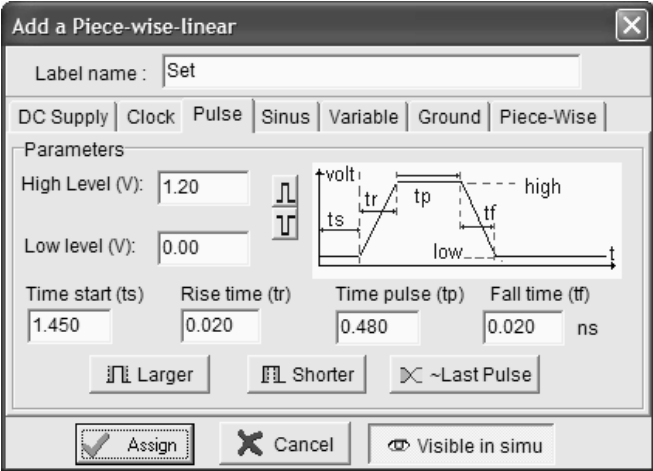


Fig. 7-3. The pulse property used to control the Reset of the latch (RsNor.MSK)

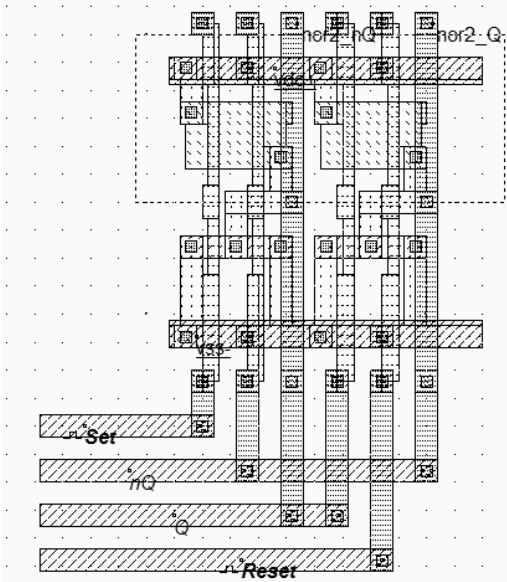


Fig. 7-4. Layout of the RS latch made (RSNor.MSK)

4. Repeat the same procedure to change the clock into a pulse for node *Set*. This time the sequence must be 000100 to delay the pulse.
5. Click on **Simulate** → **Start Simulation**. The timing diagrams of figure 7-5 appear. Click on **Close** to return to the editor.

In the simulation of Figure 7-4, a positive pulse on *Set* turns *Q* to a stable high state. Notice that when *Set* goes to 0, *Q* remains at 1, which is called the ‘memory’ state. When a positive pulse occurs on *Reset*, *Q* goes low, *nQ* goes high. In this type of simulation, the combination *Reset=Set=1* is not present.

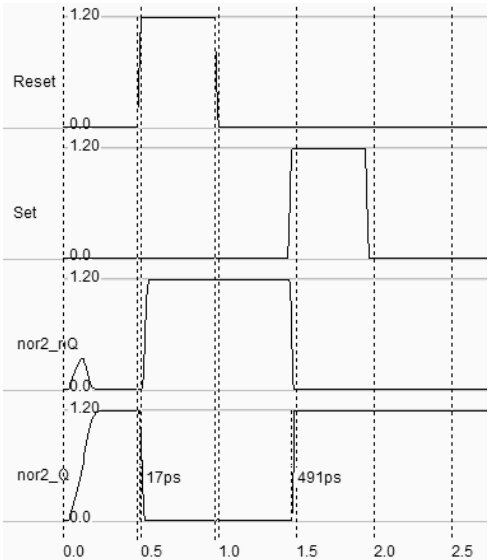


Fig. 7-5. Simulation of the RSNOR latch (RSNor.MSK)

7.3 D Latch

The truth table and schematic diagram of the static D latch, also called Static D-Flip-Flop, are shown in Figure 7-6. The data input *D* is transferred to the output if the clock input is at level 1. When the clock returns to level 0, the latch keeps its last value.

D Latch (NOR)			
D	Clock	Q	nQ
0	0	Q	nQ
0	1	0	1
1	0	Q	nQ
1	1	1	0

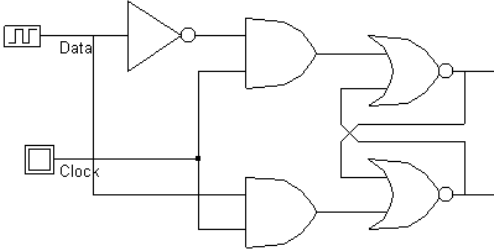


Fig. 7-6. The truth table and schematic diagram of a D Latch (File DLATCH.SCH).

MANUAL DESIGN. Note that the NOR2-AND combination can be implemented in a complex-gate style. You may find useful to invoke the one line compiler to create successively one inverter $nd = \sim d$, and two complex gates which include the AND/NOR cells using the syntax $Q = \sim (nQ | (nd \& h))$ and $nQ = \sim (Q | (d \& h))$. Build the interconnections and run the Design Rule Checker.

Assign a clock to *Clk* and a clock to *Data*. An example of such an implementation can be found in the file *DLatchCompile.MSK*. Its layout and corresponding simulation are illustrated in figure 7-7.

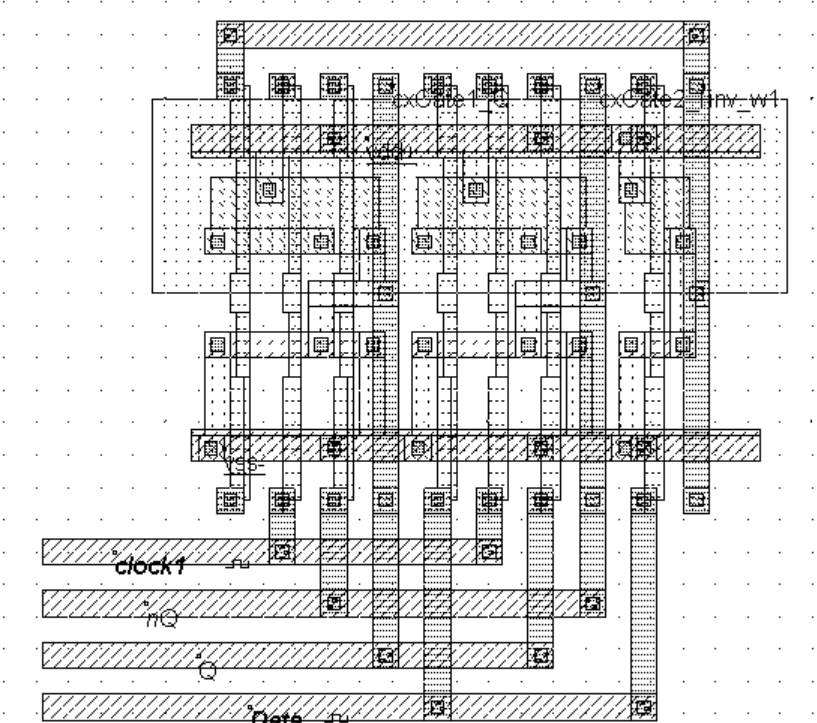


Fig. 7-8 Implementation of the D Latch (File *DlatchCompile.MSK*)

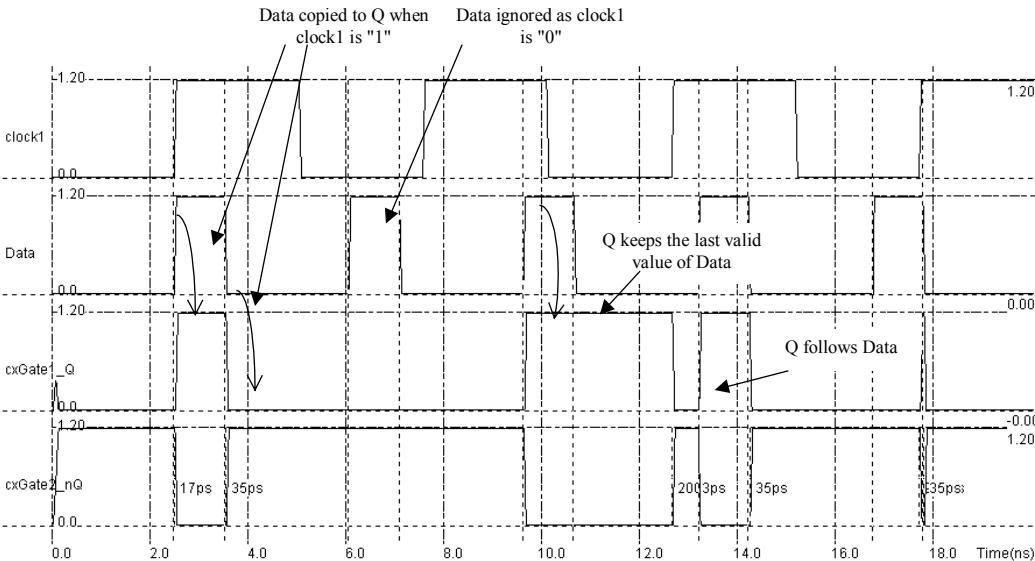


Figure 7-9: The compiled D latch at work (*DlatchCompile.MSK*).

The simulation is reported in figure 7-9. The default clocks assigned during compilation have been modified. The parameters of the clock *Data* have been changed to avoid the perfect

synchronization with $clock1$, in order to watch in one single simulation several situations. When $clock1$ is asserted, the logic information contained in $Data$ is transferred to Q , its inverted value to nQ . When $clock1$ falls down to 0, Q and nQ keep in memory state.

7.4 Edge Triggered Latch

The most common example of an edge-triggered flip flop is the JK latch. Anyhow, the JK is rarely used, a more simple version that features the same function with one single input D is preferred. This simple type of edge-triggered latch is one of the most widely used cells in microelectronics circuit design. The cell structure comprises two master-slave basic memory stages.

The most compact implementation of the edge-triggered latch is reported below. The schematic diagram is based on inverters and pass-transistors. On the left side, the two chained inverter are in memory state when the pMOS loop transistor $P1$ is on, that is when $Clk=0$. The two-chained inverters on the right side act in an opposite way. The reset function is obtained by a direct ground connection of the master and slave memories, using nMOS devices.

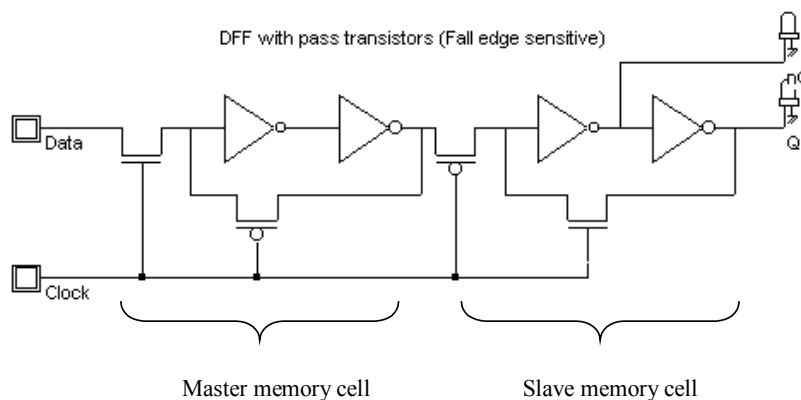


Figure 7-10 : The edge-triggered latch and its logic simulation (Dreg.MSK)

Notice that the logic model of the MOS device is not working the same way as for the real MOS switch. In the case of the logic implementation, the logic signal flows only from the source to the drain. This is not the case of the real switch where the signal can flow both ways.

In figure 7-11, $clock$ is high, the master latch is updated to a new value of the input D . The slave latch produces to the output Q the previous value of D . When $clock$ goes down, the master latch turns to memory state. The slave circuit is updated. The change of the clock from 1 to 0 is the active edge of the clock. This type of latch is a negative edge flip flop.

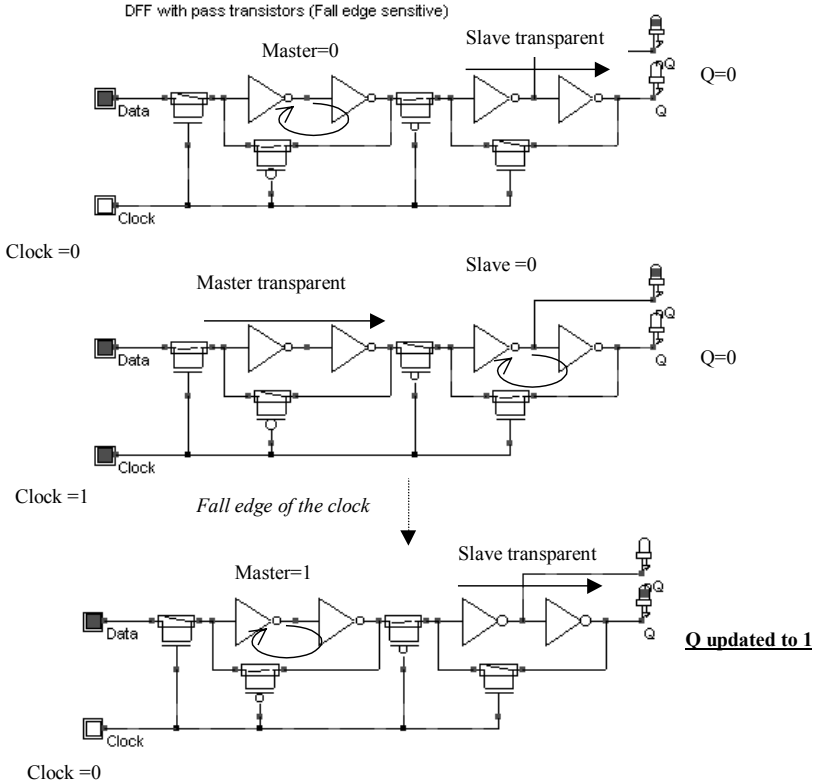


Figure 7-11 The edge-triggered latch and its logic simulation (Dreg.MSK)

Use the Verilog compiler to generate the edge-triggered latch, using the following text (dreg.txt), or by creating a schematic diagram including the “D” register symbol, in the symbol palette of DSCH2. As can be seen, the register is built up from one single call to the primitive dreg. For simulation:

- *Reset* is active on a level 1. *Reset* is activated twice, at the beginning and later, using a piece-wise linear description included in the pulse property.
- *Clk* is a clock with 10ns at 0 and 10ns at 1.
- *D* is the data chosen here not synchronized with *Clk*, in order to observe various behaviors of the register.

To compile the DREG file, use the command **Compile→Compile Verilog Text**. The corresponding layout is reported below. The piece-wise-linear data is transferred to the text “rst” automatically.

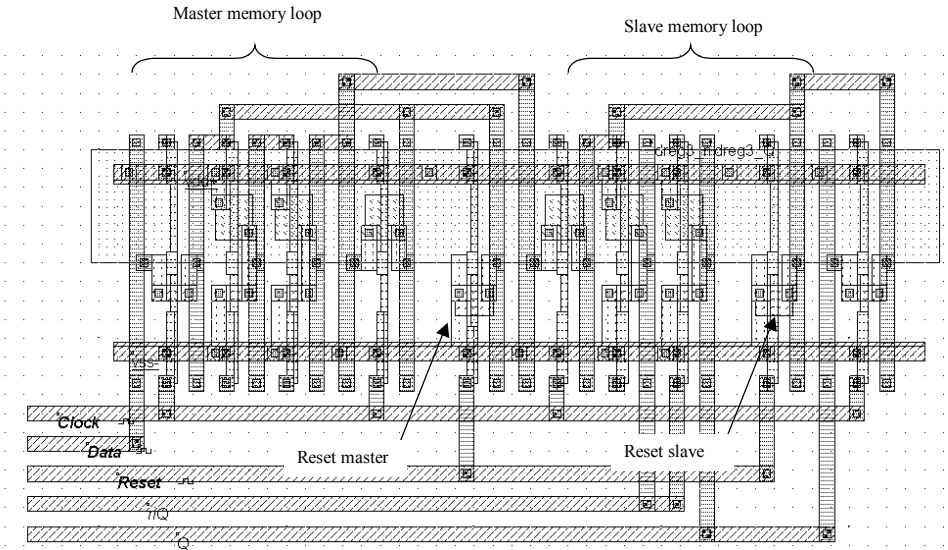


Fig. 7-12: Compiled version of the Edge-triggered D Flip Flop (DregCompile.MSK)

For testing the Dreg, the *Reset* signal is activated twice, at the beginning and later, using a piece-wise linear property. The *Clock* signal has a 2ns period. *D* is the data chosen here not synchronized with *Clock*, in order to observe various behaviors of the register.

The simulation of the edge-triggered D-register is reported in figure 7-13. The signals *Q* and *nQ* always act in opposite. When *Reset* is asserted, the output *Q* is 0, *nQ* is 1. When *Reset* is not active, *Q* takes the value of *D* at a fall edge of the clock. For all other cases, *Q* and *nQ* remain in memory state. The latch is thus sensitive to the fall edge of the clock.

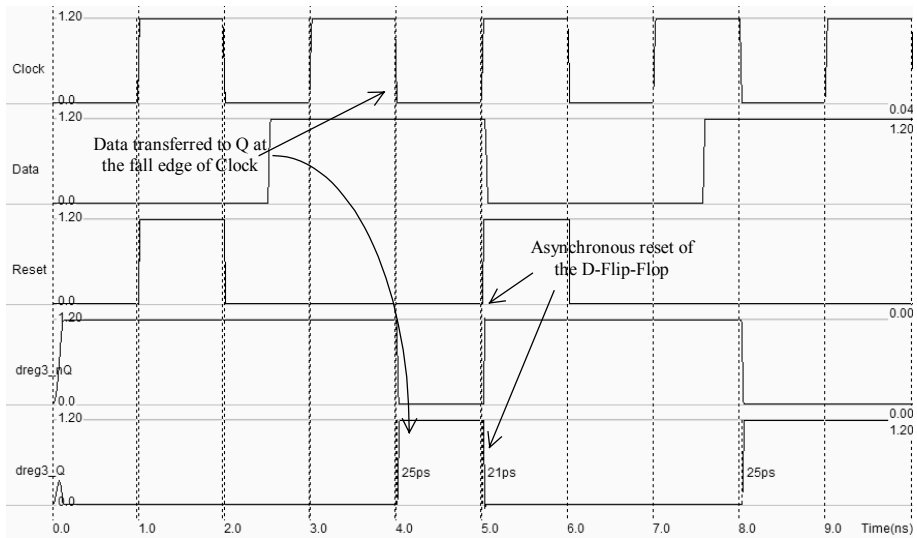


Fig. 7-13 Simulation of the DREG cell (DregCompile.MSK)

7.5 Counter

The one-bit counter is able to produce a signal featuring half the frequency of a clock. The most simple implementation consists of a D flip-flop where the output nQ is connected to D , as shown in figure 7-14. In the logic simulation, the clock changes the state of $ClockDiv2$ at each fall edge. The $Reset$ signal is active high, and stuck the output to 0.

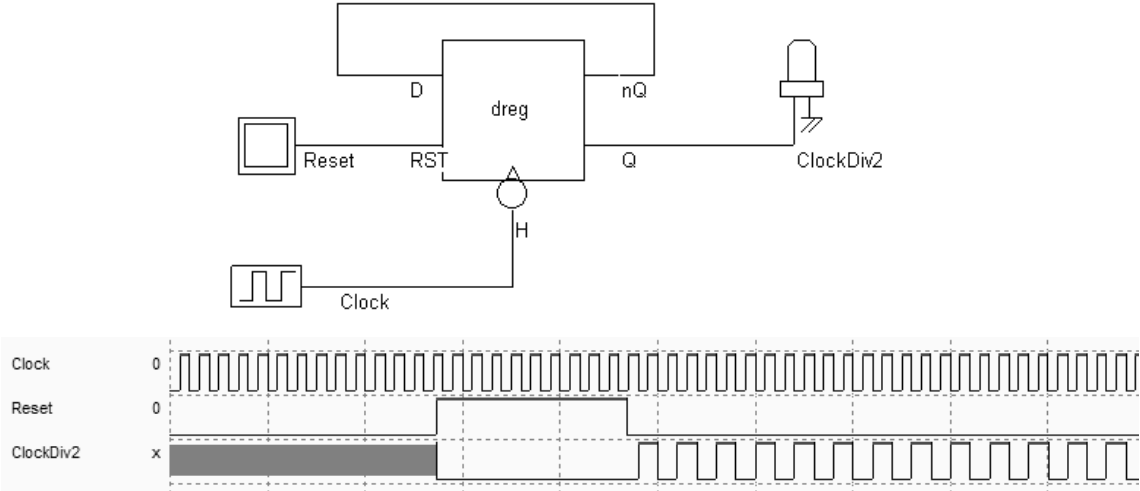


Fig. 7-14. Schematic diagram and simulation of the counter (ClockDiv2.SCH).

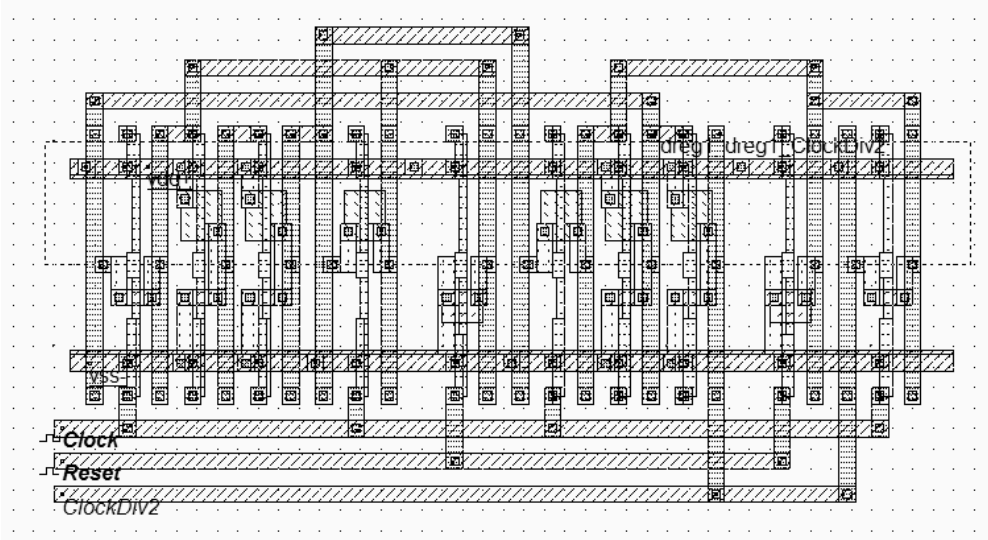


Fig 7-15. Layout of the divider-by-two (ClockDiv2.MSK)

8 Memory

Circuits

8.1 The world of Memory

Semiconductor memories are vital components in modern integrated circuits. Stand-alone memories represent roughly 30% of the global integrated circuit market. Within system-on-chip, memory circuits usually represent more than 75% of the total number of transistors.

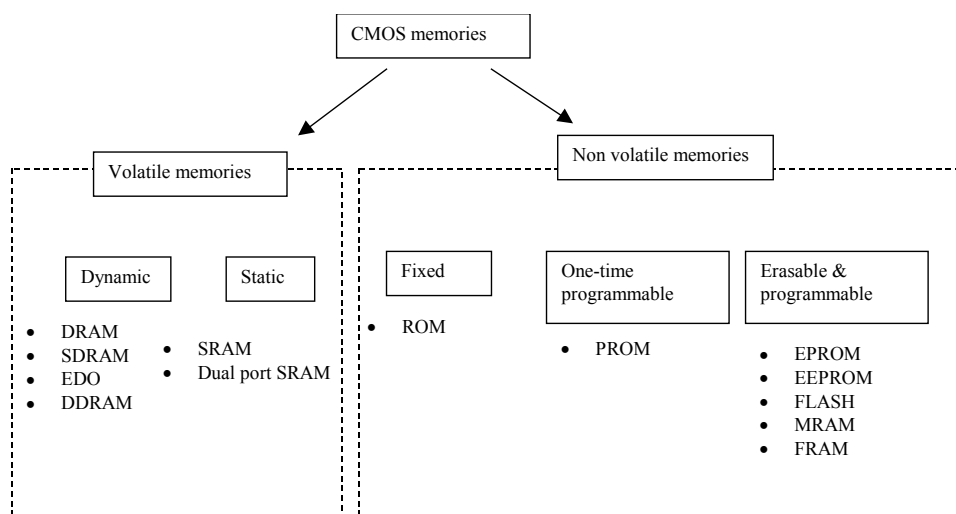


Figure 8-1 Major classes of CMOS compatible memories

Two main families of devices exist: volatile and non-volatile memories.

- In volatile circuits (Figure 8-1 left), the data is stored as long as the power is applied. The dynamic random access memory (DRAM) is the most common memory.
- Non-volatile memories are capable of storing the information even if the power is turned off (Figure 8-1 right). The read-only memory (ROM) is the simplest type of non-volatile memory. One-time programmable memories (PROM) are a second important family, but the most popular non volatile memories are erasable and programmable devices: the old electrically programmable ROM (EPROM), the more recent Electrically Erasable PROM (EEPROM, FLASH), and the new magneto resistive RAM (MRAM) and ferroelectric RAM (FRAM) memories.

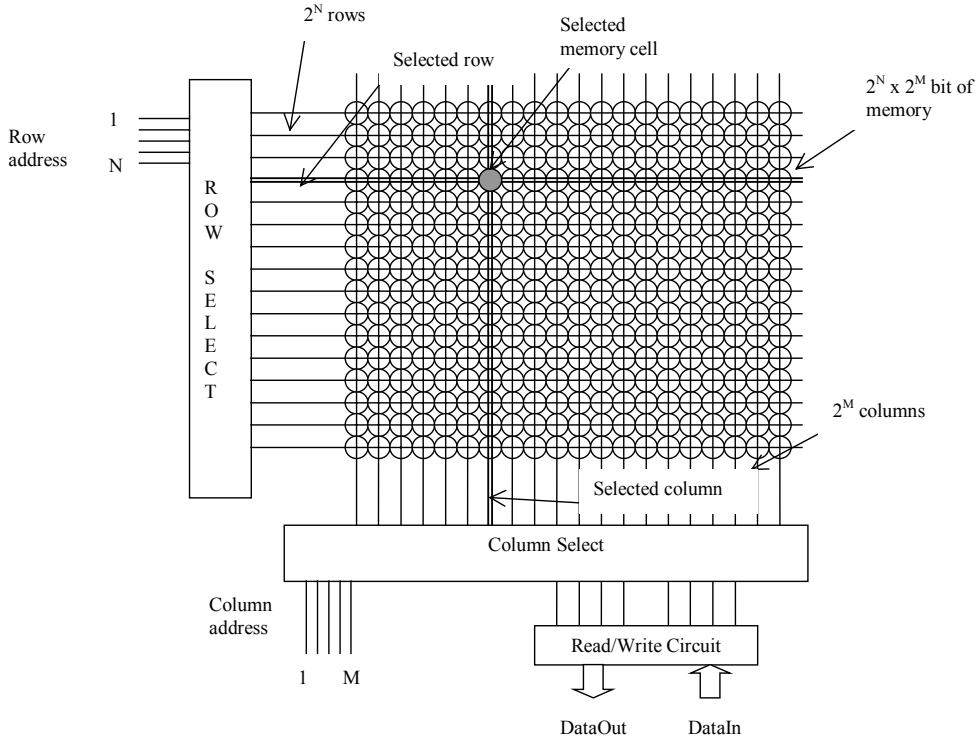


Figure 8-2 Typical memory organization

Figure 8-2 shows a typical memory organization layout. It consists of a memory array, a row decoder, a column decoder and a read/write circuit. The row decoder selects one row from 2^N , thanks to a N-bit row selection address. The column decoder selects one row from 2^M , thanks to a M-bit column selection address. The memory array is based on 2^N rows and 2^M columns of a repeated pattern, the basic memory cell. A typical value for N and M is 10, leading to 1024 rows and 1024 columns, which corresponds to 1048576 elementary memory cells (1Mega-bit).

8.2 RAM Memory

The basic cell for static memory design is based on 6 transistors, with two pass gates instead of one. The corresponding schematic diagram is given in Figure 8-3. The circuit consists again of the 2 cross-coupled inverters, but uses two pass transistors instead of one. The cell has been designed to be duplicated in X and Y in order to create a large array of cells. Usual sizes for Megabit SRAM memories are 256 column x 256 rows or higher. A modest arrangement of 4x4 RAM cells is proposed in figure 8-4. The selection lines *WL* concern all the cells of one row. The bit lines *BL* and $\sim BL$ concern all the cells of one column.

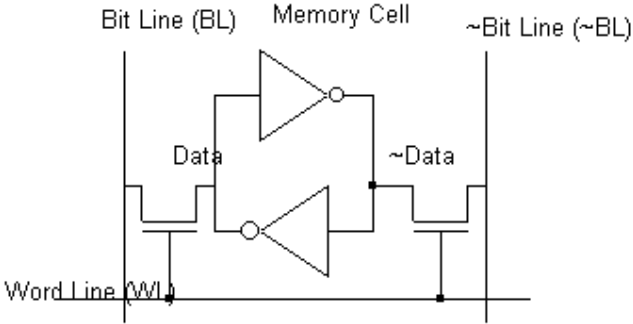


Figure 8-3: The layout of the 6 transistor static memory cell (RAM6T.SCH)

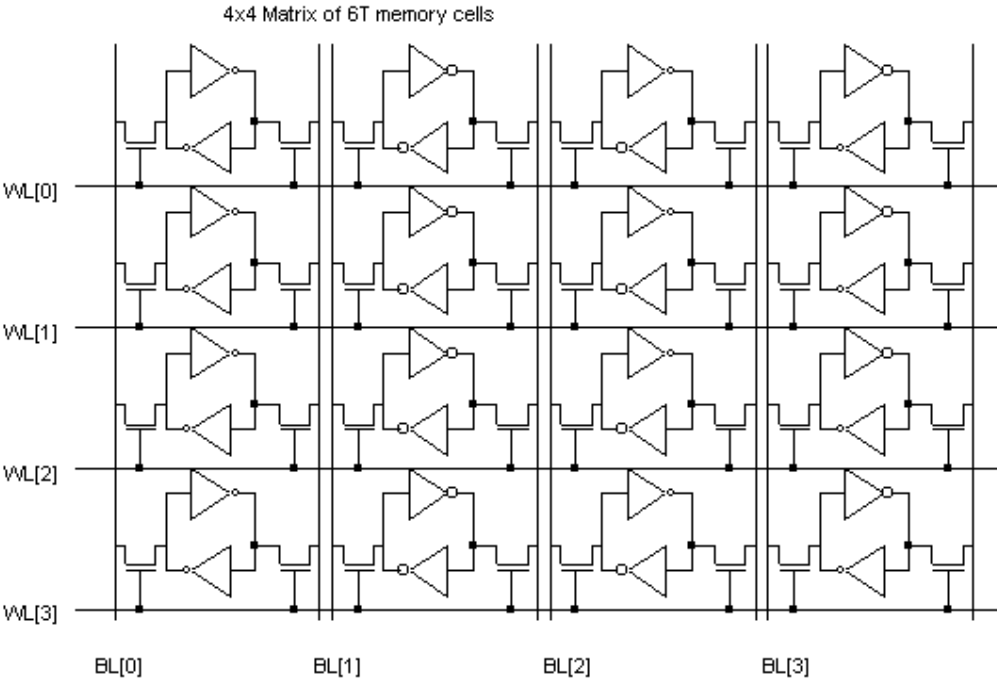


Fig. 8-4 An array of 6T memory cells, with 4 rows and 4 columns (RAM6T.SCH)

The RAM layout is given in Figure 8-5. The BL and ~BL signals are made with metal2 and cross the cell from top to bottom. The supply lines are horizontal, made with metal3. This allows easy matrix-style duplication of the RAM cell.

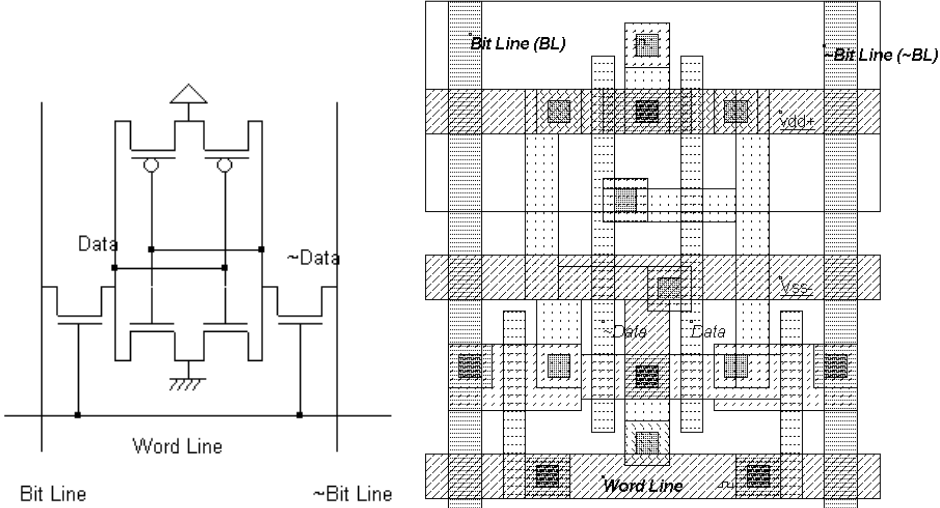


Fig. 8-5. The layout of the static RAM cell (RAM6T.MSK).

WRITE CYCLE. Values 1 or 0 must be placed on *Bit Line*, and the data inverted value on *~Bit Line*. Then the selection *Word Line* goes to 1. The two-inverter latch takes the *Bit Line* value. When the selection *Word Line* returns to 0, the RAM is in a memory state.

READ CYCLE. The selection signal *Word Line* must be asserted, but no information should be imposed on the bit lines. In that case, the stored data value propagates to *Bit Line*, and its inverted value *~Data* propagates to *~Bit Line*.

SIMULATION. The simulation parameters correspond to the read and write cycle in the RAM. The proposed simulation steps consist in writing a 0, a 1, and then reading the 1. In a second phase, we write a 1, a 0, and read the 0. The *Bit Line* and *~Bit Line* signals are controlled by pulses (Figure 8-6). The floating state is obtained by inserting the letter "x" instead of 1 or 0 in the description of the signal.

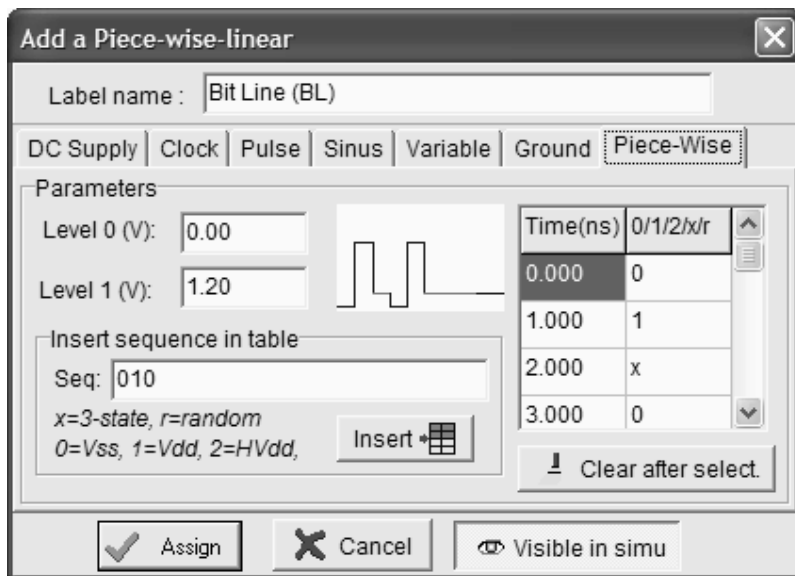


Fig. 8-6. The bit Line pulse used the "x" floating state to enable the reading of the memory cell (RamStatic6T.MSK)

The simulation of the RAM cell is proposed in figure 8-9. At time 0.0, *Data* reaches an unpredictable value of 1, after an unstable period. Meanwhile, \sim *Data* reaches 0. At time 0.5ns, the memory cell is selected by a 1 on *Word Line*. As the *Bit Line* information is 0, the memory cell information *Data* goes down to 0. At time 1.5ns, the memory cell is selected again. As the *Bit Line* information is now 1, the memory cell information *Data* goes to 1. During the read cycle, in which *Bit Line* and \sim *Bit Line* signals are floating, the memory sets these wires respectively to 1 and 0, corresponding to the stored values.

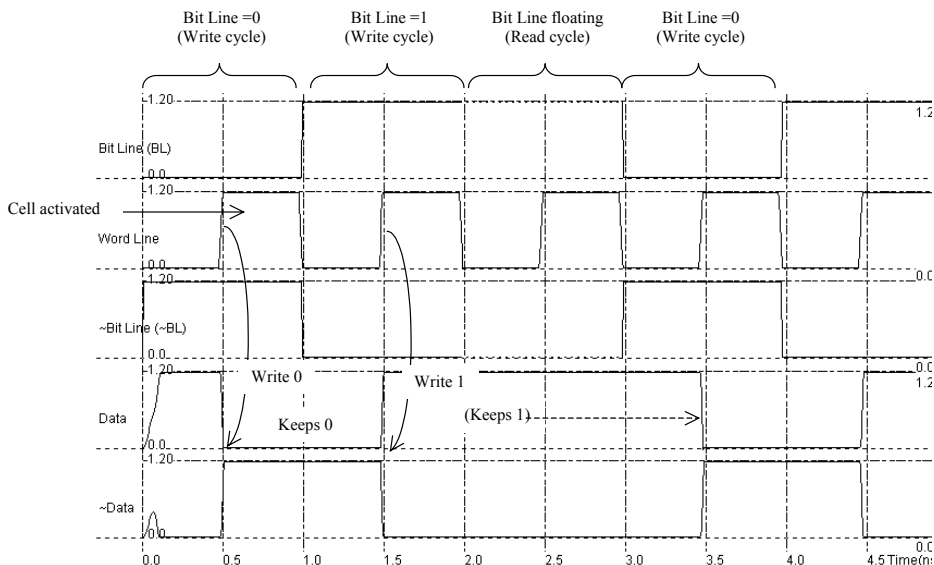


Fig. 8-7. Write cycle for the static RAM cell (RamStatic6T.MSK).

8.3 RAM Array

You can duplicate the RAM cell into a 4x4 bit array using the command **Edit → Duplicate XY**. Select the whole RAM cell and a new window appears. Enter the value « 4 » for X and « 4 » for Y into the menu. Click on « **Generate** ».

A very interesting approach to obtain a more compact memory cell consists in sharing all possible contacts: the supply contact, the ground contact and the bit line contacts. The consequence is that the effective cell size can be significantly reduced (Figure 8-8).

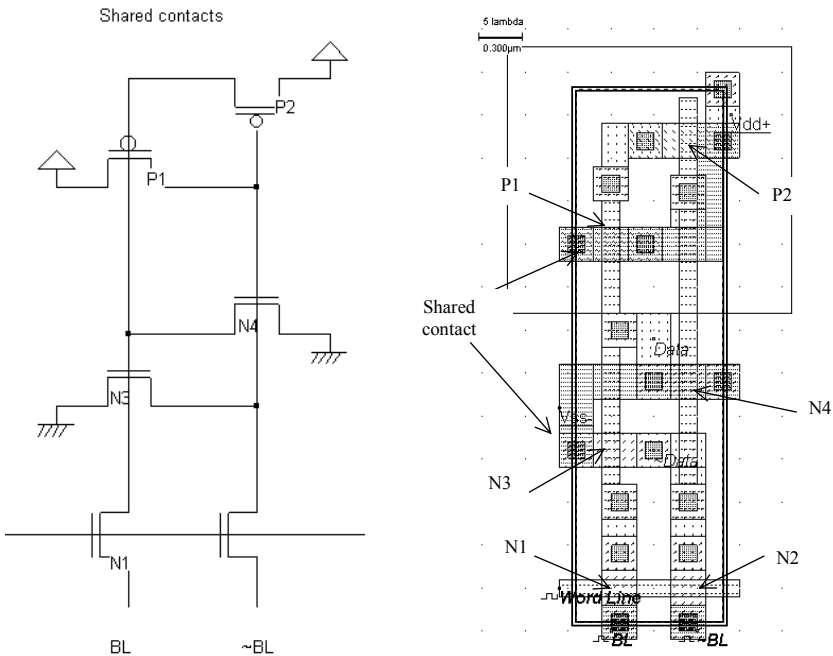


Figure 8-8. Sharing all possible contacts lead to a very compact cell design (Ram6Tcompact.MSK)

The layout is functionally identical to the previous layout. The only difference is the placement of MOS devices and contacts. We duplicate the RAM cell into a 64 bit array. The multiplication cannot be done directly by the command **Duplicate XY**, as we need to flip one cell horizontally to share lateral contacts, and flip the resulting block vertically to share vertical contacts (Figure 8-9).

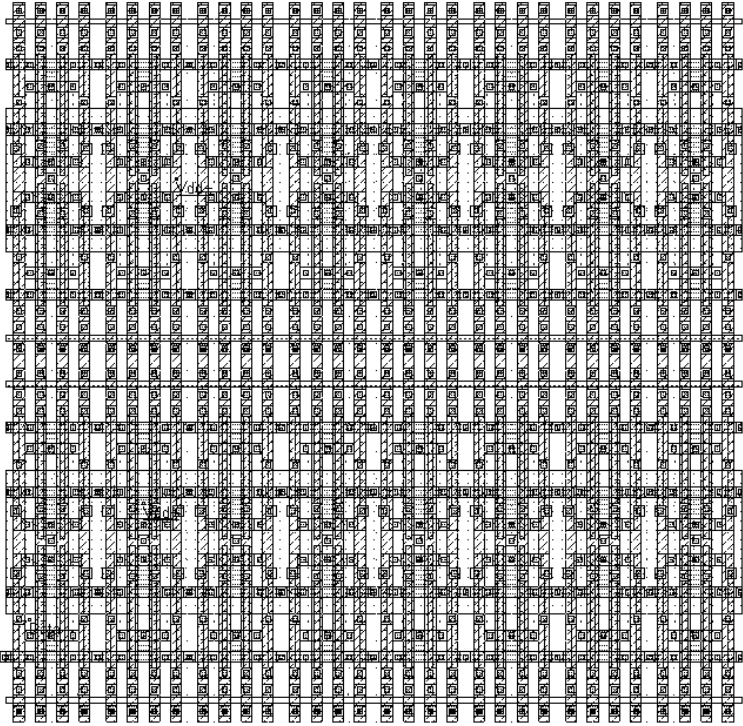


Figure 8-9. Compact 16x4 array of memory cells with shared contacts (Ram16x4Compact.MSK)

8.4 Row Selection Circuit

The row selection circuit decodes the row address and activates one single row. This row is shared by all word line signals of the row. The row selection circuit is based on a multiplexor circuit. One line is asserted while all the other lines are at zero.

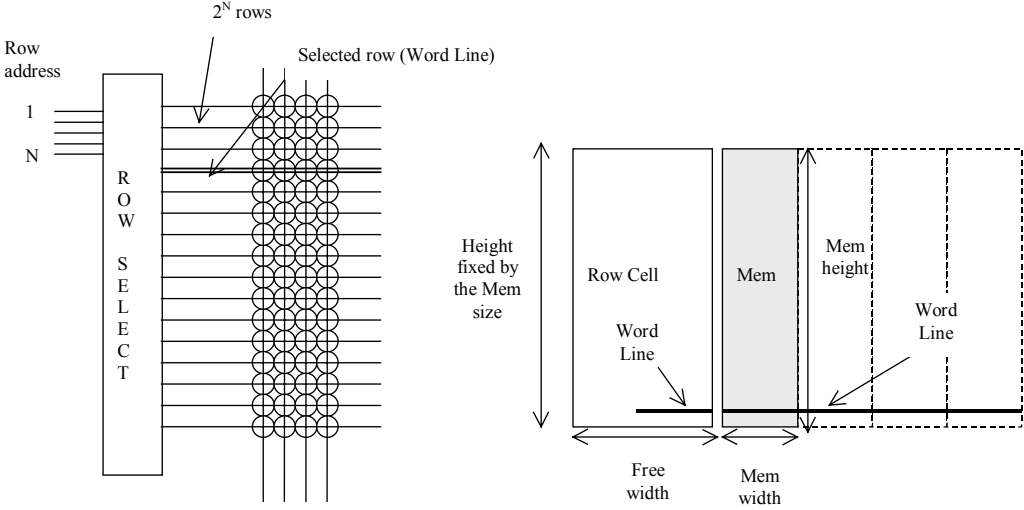


Fig. 8-10 The row selection circuit

In the row selection circuit for the 16x4 array, we simply need to decode a two-bit address. Using AND gates is one simple solution. In figure 8-11, we present the schematic diagram of 2-to-4 and 3-to-8 decoders. In the case of a very large number of address lines, the decoder is split into sub-decoders, which handle a reduced number of address lines.

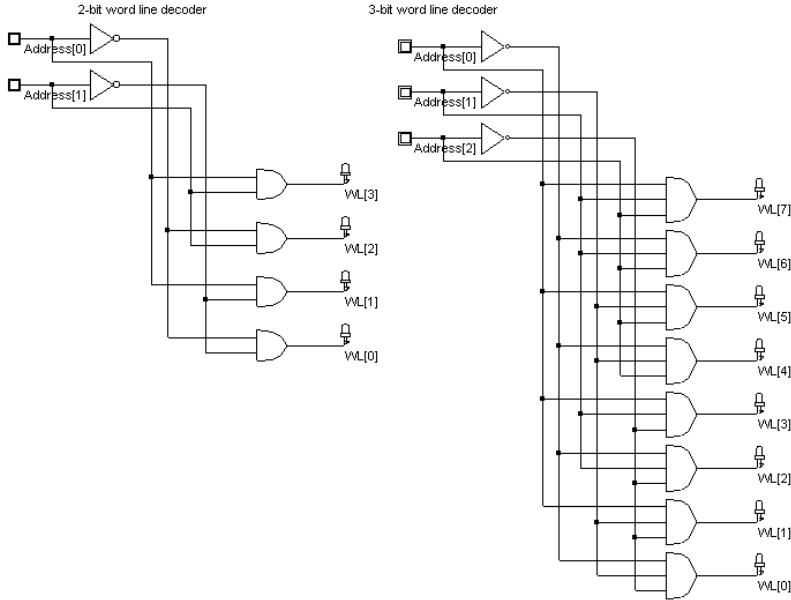


Fig. 8-11. The row selection circuit in 2 bit and 3 bit configuration (RamWordLine.SCH)

8.5 Column Selection Circuit

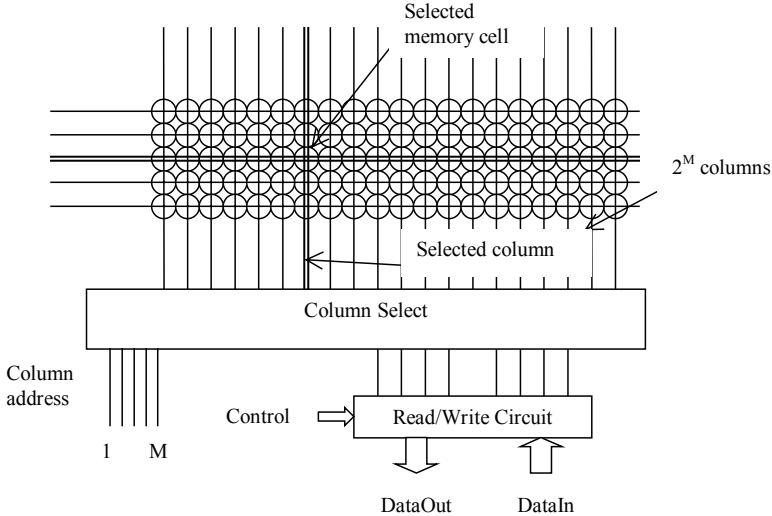


Figure 8-12. The column selection circuit principles

The column decoder selects a particular column in the memory array to read the contents of the selected memory cell (Figure 8-12) or to modify its contents. The column selector is based on the same principles as those of the row decoder. The major modification is that the data flows both ways, that is either from the memory cell to the *DataOut* signal (Read cycle), or from the *DataIn* signal to the cell (Write cycle).

Figure 8-13 proposes an architecture based on n-channel MOS pass transistors. We consider here 4 columns of memory cells, which requires 2 address signals *Address_Col[0]* and *Address_Col[1]*. The n-channel MOS device is used as a switch controlled by the column selection. When the nMOS is on and *Write* is asserted, (Figure 8-13) the *DataIn* is amplified by the buffer, flows from the bottom to the top and reaches the memory through *BL* and $\sim BL$. If *Write* is off, the 3-state inverter is in high impedance, which allows the information to be read on *DataOut*.

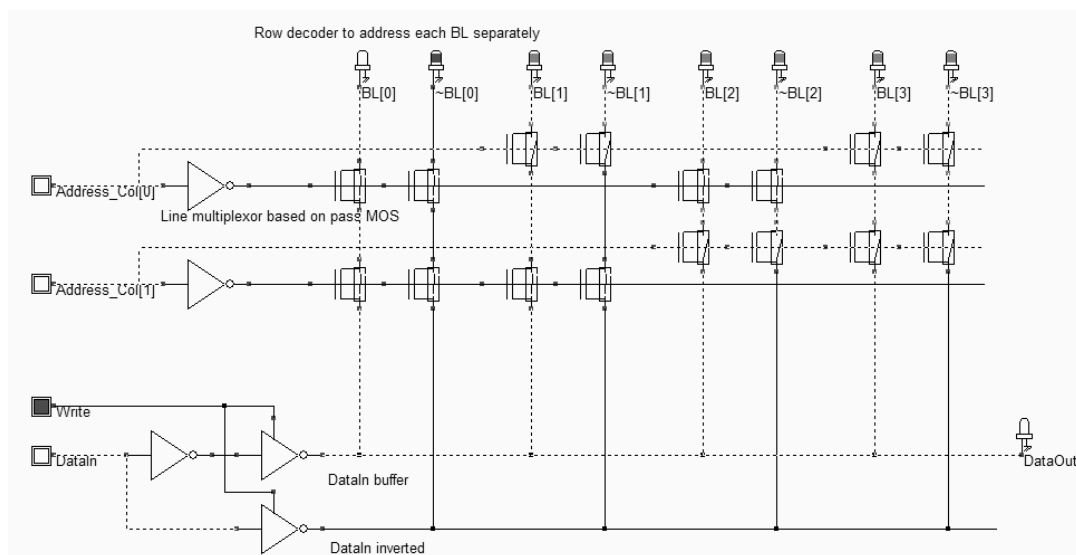


Figure 8-13. Row selection and Read/Write circuit (RamColumn.SCH)

8.6 A Complete 64 bit SRAM

The 64 bit SRAM memory interface is shown in figure 8-14. The 64 bits of memory are organized in words of 4 bits, meaning that *DataIn* and *DataOut* have a 4 bit width. Each data *D0..D15* occupies 4 contiguous memory cells in the array. Four address lines are necessary to decode one address among 16. The memory structure shown in figure 10-44 requires two address lines *A0* and *A1* for the word lines *WL[0]..WL[3]* and two address lines *A2* and *A3* for the bit line selection. The final layout of the 64 bit static RAM is proposed in Figure 8-15.

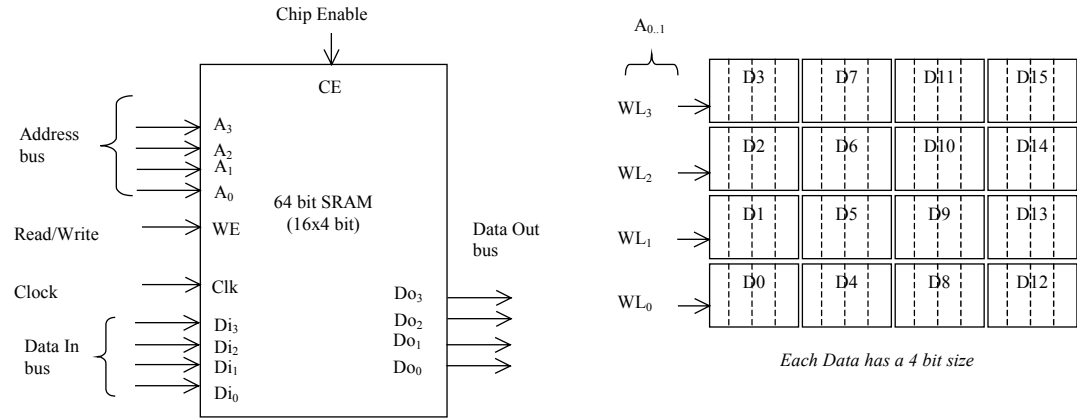


Figure 8-14. The architecture of the 64 bit RAM (RAM64.MSK)

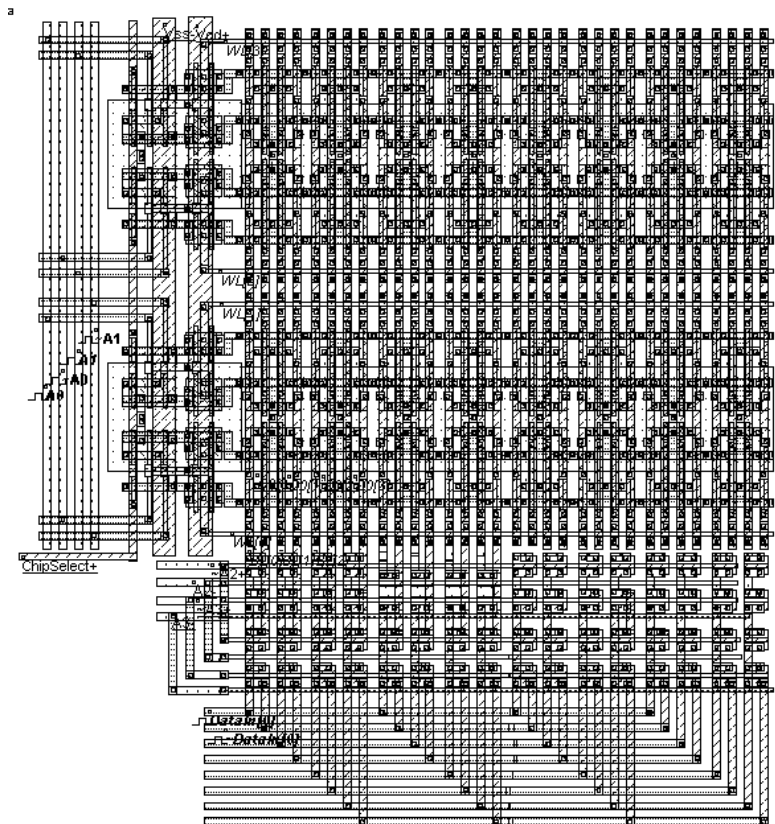


Figure 8-15. The complete RAM layout (RAM64.MSK)

8.7 Dynamic RAM Memory

The dynamic RAM memory has only one transistor, in order to improve the memory matrix density by almost one order of magnitude. The storage element is no longer the stable inverter

loop, as for the static RAM, but only a capacitor C_s , also called the storage capacitor. The DRAM cell architecture is shown in figure 8-16.

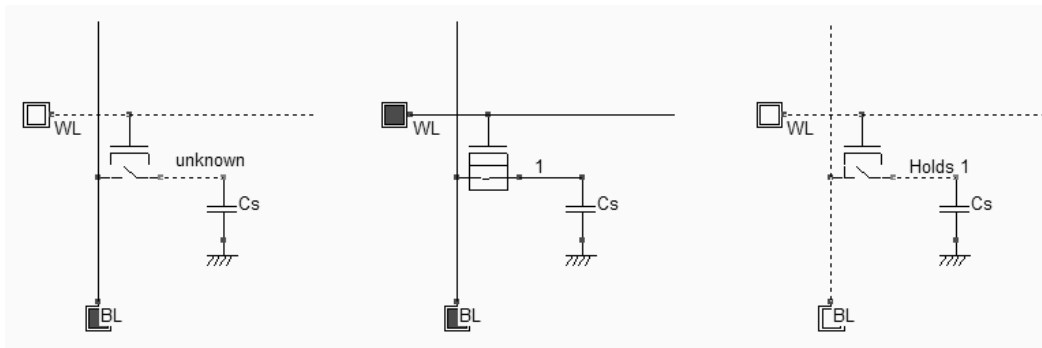


Figure 8-16: Simulation of the Write cycle for the 1 transistor dynamic RAM cell (RAM1T.SCH)

The write and hold operation for a "1" is shown in figure 8-17. The data is set on the bit line, the word line is then activated and C_s is charged. As the pass transistor is n-type, the analog value reaches $V_{DD}-V_t$. When WL is inactive, the storage capacitor C_s holds the "1".

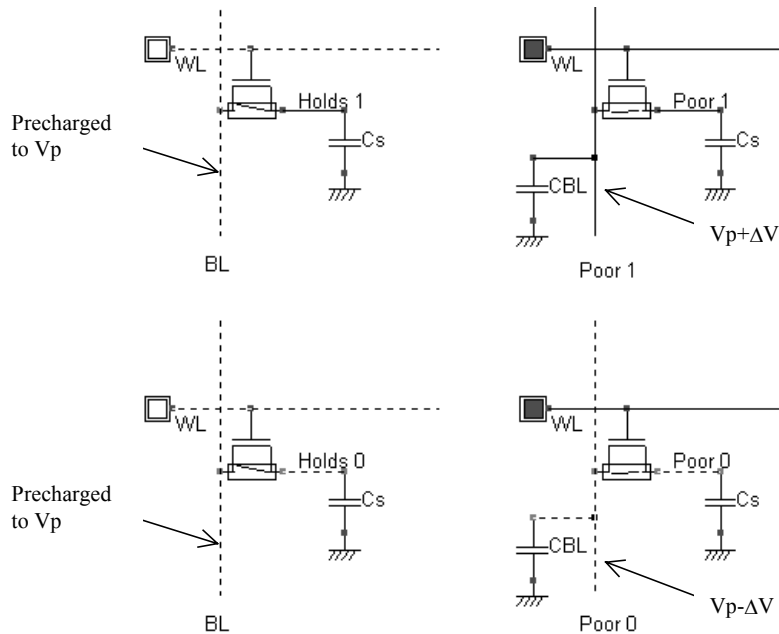


Figure 8-17: Simulation of the Read cycle for the 1 transistor dynamic RAM cell (RAM1T.SCH)

The reading cycle is destructive for the stored information. Suppose that C_s holds a 1. The bit line is precharged to a voltage V_p (Usually around $V_{DD}/2$). When the word line is active, a communication is established between the bit line, loaded by capacitor C_{BL} , and the memory, loaded by capacitor C_s . The charges are shared between these nodes, and the result is a small increase of the voltage V_p by ΔV , thanks to the injection of some charges from the memory.

Commercial dynamic RAM memories use storage capacitors with a value between 10fF and 50fF. This is done by creating a specific capacitor for the storage node appearing in figure 8-18 left thanks to the following technological advances: the use of specific metal layers to create the lower plate and external walls of the RAM capacitor, an enlarged height between the substrate surface and metal1, and the use of high permittivity dielectric oxide. The silicon dioxide SiO₂ has a relative permittivity ϵ_r of 3.9. Other oxides, compatible with the CMOS process have a higher permittivity (Higher "K") : Si₃N₄ with ϵ_r equal to 7, and Ta₂O₅ with ϵ_r equal to 23.

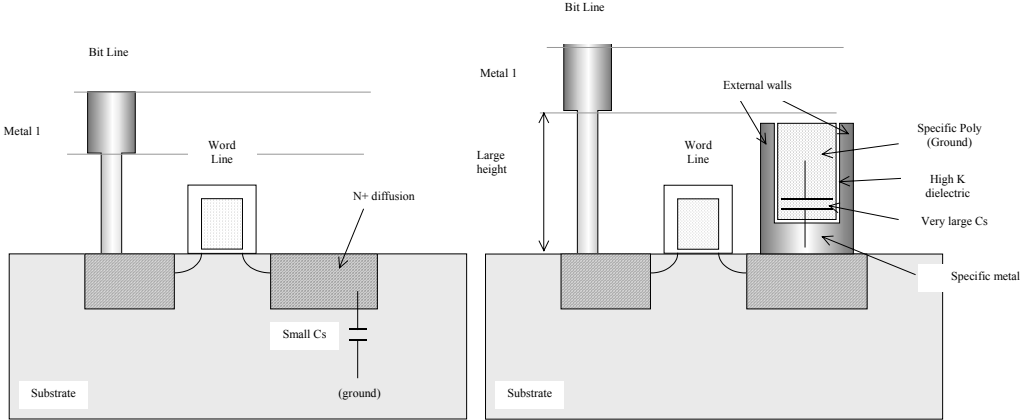


Figure 8-18: Increasing the storage capacitance (Left: junction capacitor, right, embedded capacitor)

The cross-section of the DRAM capacitor is given in figure 8-19. The bit line is routed in metal2, and is connected to the cell through a metal1 and diffusion contact. The word line is the polysilicon gate. On the right side, the storage capacitor is a sandwich of conductor material connected to the diffusion, a thin oxide (SiO₂ in this case) and a second conductor that fills the capacitor and is connected to ground by a contact to the first level of metal. The capacitance is around 20fF in this design. Higher capacitance values may be obtained using larger option layer areas, at the price of a lower cell density.

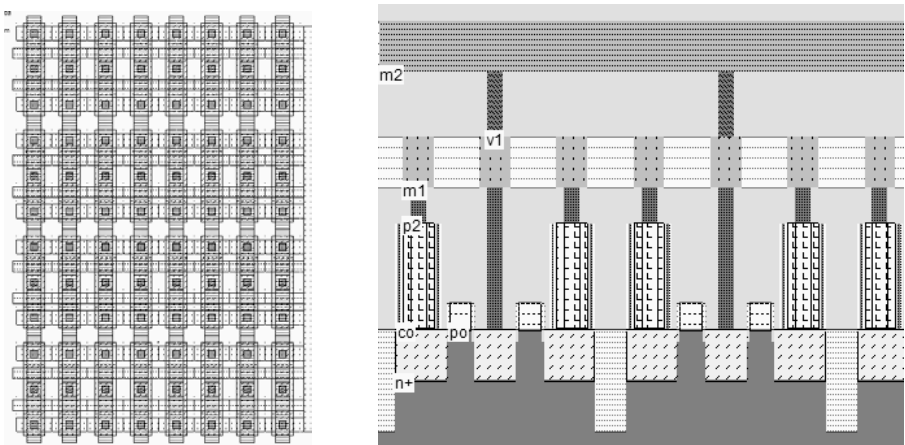


Figure 8-19: The stacked capacitor cell compared to the diffusion capacitor cell (DramEdram.MSK)

8.8 EEPROM

The basic element of an EEPROM (Electrically Erasable PROM) memory is the floating-gate transistor. The concept was introduced several years ago for the EPROM (Erasable PROM). It is based on the possibility of trapping electrons in an isolated polysilicon layer placed between the channel and the controlled gate. The charges have a direct impact on the threshold voltage of a double-gate device. When there is no charge in the floating gate (Figure 8-20, upper part), the threshold voltage is low, meaning that a significant current may flow between the source and the drain, if a high voltage is applied on the gate. However, the channel is small as compared to a regular MOS, and the Ion current is 3 to 5 times lower, for the same channel size.

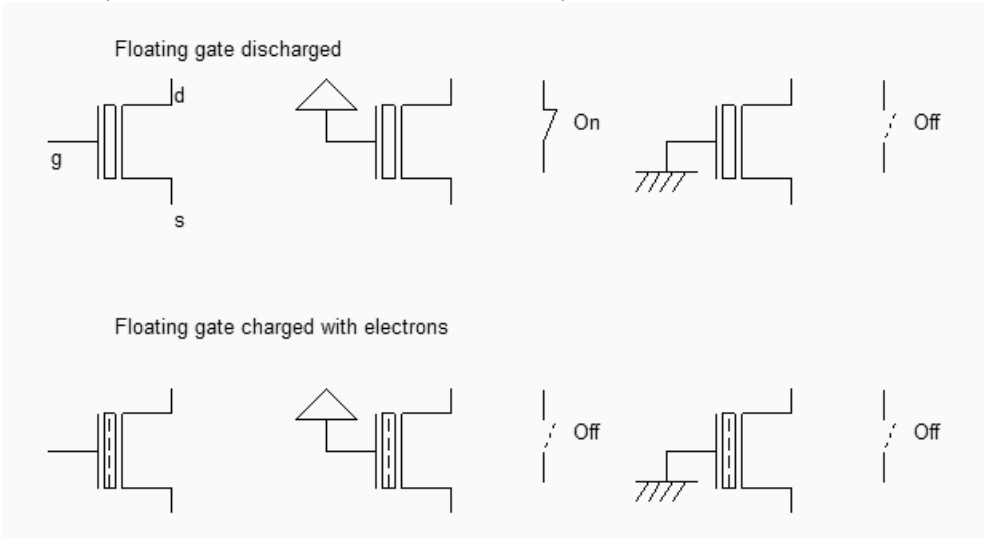


Fig. 8-20: The two states of the double gate MOS (EepromExplain.SCH)

When charges are trapped in the floating polysilicon layer (Figure 8-20, lower part), the threshold voltage is high, almost no current flows through the device, independently of the gate value. As a matter of fact, the electrons trapped in the floating gate prevent the creation of the channel by repelling channel electrons. Data retention is a key feature of EEPROM, as it must be guaranteed for a wide range of temperatures and operating conditions. Optimum electrical properties of the ultra thin gate oxide and inter-gate oxide are critical for data retention. The typical data retention of an EEPROM is 10 years.

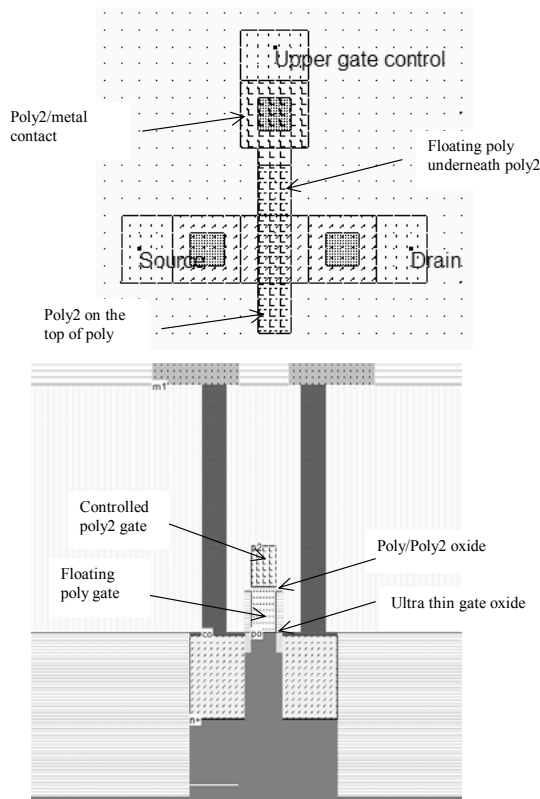


Fig. 8-21: The double gate MOS generated by Microwind (Eeprom.MSK)

The double gate MOS layout is shown in figure 8-21. The structure is very similar to the n-channel MOS device, except for the supplementary *poly2* layer on top of the polysilicon. The lower polysilicon is unconnected, resulting in a floating node. Only the *poly2* upper gate is connected to a metal layer through a *poly2/metal* contact situated at the top. The cross-section of figure 8-21 reveals the stacked *poly/poly2* structure, with a thin oxide in between.

8.8.1 Double-gate MOS Charge

The programming of a double-poly transistor involves the transfer of electrons from the source to the floating gate through the thin oxide (Figure 8-22). Notice the high drain voltage (3V) which is necessary to transfer enough temperature to some electrons to become "hot" electrons, and the very high gate control to attract some of these hot electrons to the floating poly through the ultra thin gate oxide. The very high voltage varies from 7V to 12V, depending on the technology. Notice the "++" symbols attached to the upper gate and drain regions which indicate that a voltage higher than the nominal supply is used.

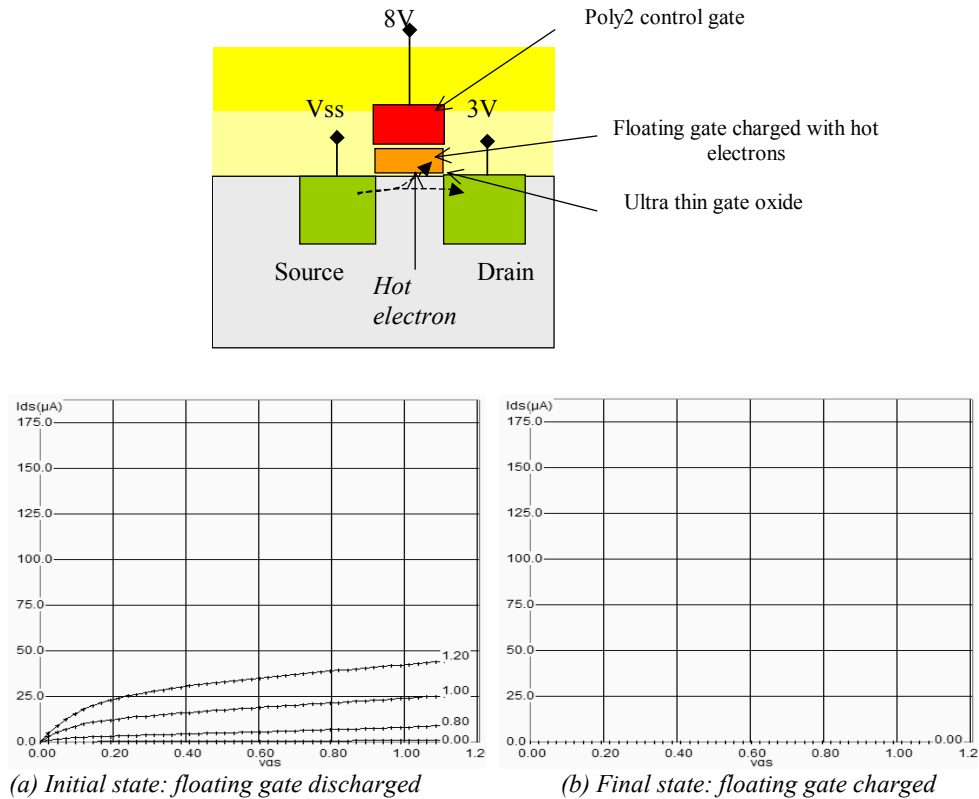


Fig. 8-22: Double-gate MOS characteristics without (a) and with charges (EepromCharge.MSK)

At initialization (Figure 8-22-a) no charge exists in the floating gate, resulting in a possibility of current when the poly2 gate voltage is high. However, the device is much less efficient than the standard n-channel MOS due to an indirect control of the channel. The maximum current is small but significant. The programming operation is performed using a very high gate voltage on poly2, here 8V. The mechanism for electron transfer from the grounded source to the floating polysilicon gate, called tunneling, is a slow process. In Microwind, around 1000ns are required. With a sufficiently positive voltage on the poly2 gate, the voltage difference between poly and source is high enough to enable electrons to pass through the thin oxide. The electrons trapped on the floating gate increase the threshold voltage of the device, thus rapidly decreasing the channel current. When the gate is completely charged, no more current appears in the I_d/V_d characteristics (Figure 8-22-b).

8.8.2 Double-gate MOS Discharge

The floating gate may be discharged by ultra violet light exposure or by electrical erasure. The U.V. technique is a heritage of the EPROM which requires a specific package with a window to expose the memory bank to the specific light. The process is very slow (Around 20mn). After

the U.V exposure, the threshold voltage of the double gate MOS returns to its low value, which enables the current to flow again. In Microwind, the command **Simulate** → **U.V exposure to discharge floating gates** simulates the exposure of all double gate MOS to an ultra violet light source. Alternatively, the charge can be accessed individually using the command **Simulate**→**Mos characteristics**. Changing the *Charge* cursor position modifies dynamically the MOS characteristics.

For the electrical erase operation, the poly2 gate is grounded and a high voltage (Around 8V) is applied to the source. Electrons are pulled off the floating gate thanks to the high electrical field between the source and the floating gate. This charge transfer is called Fowler-Nordheim electron tunneling (Figure 8-23).

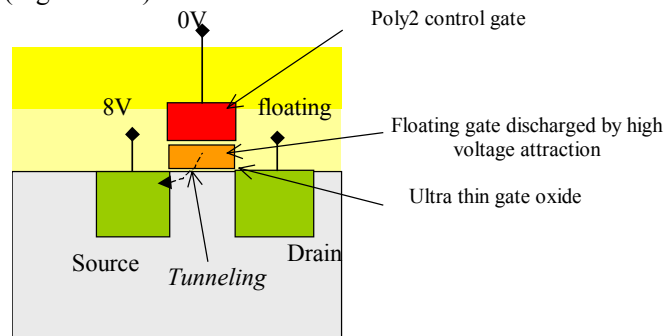


Fig. 8-23. Discharging the double gate MOS device (*EepromDischarge.MSK*)

The basic structure for reading the EEPROM information is described in the schematic diagram of figure 8-24. After a precharge to VDD, and once *WL* is asserted, the bit line may either drop to VSS if the floating gate is empty of charges, or keep in a high voltage if the gate is charged, which disables the path between *BL* and the ground through the EEPROM device. In the case of figure 8-24 left, the floating gate has no charge, so *BL* is tied to ground after the precharge, meaning that *DataOut* is 1.

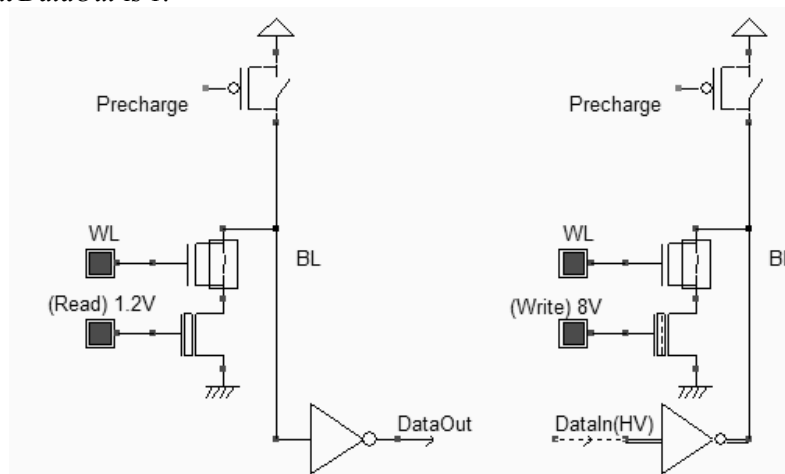


Fig. 8-24 Reading and writing in the EEPROM (*Eeprom.MSK*)

The write operation consists in applying a very high voltage on the gate (8V), and to inject a high or low state on *BL*. A zero on *DataIn* is equivalent to a high voltage on *BL*, which provokes the hot electron effect and charges the floating gate. In contrast, a one on *DataIn* keeps *BL* low, and no current flows on the EEPROM channel. In that case, the floating gate remains discharged.

8.9 Flash Memories

Flash memories are a variation of EEPROM memories. Flash arrays can be programmed electrically bit-by-bit but can only be erased by blocks. Flash memories are based on a single double poly MOS device, without any selection transistor (Figure 8-26). The immediate consequence is a more simple design, which leads to a more compact memory array and more dense structures. Flash memories are commonly used in micro-controllers for the storage of application code, which gives the advantage of non volatile memories and the possibility of reconfiguring and updating the code many times.

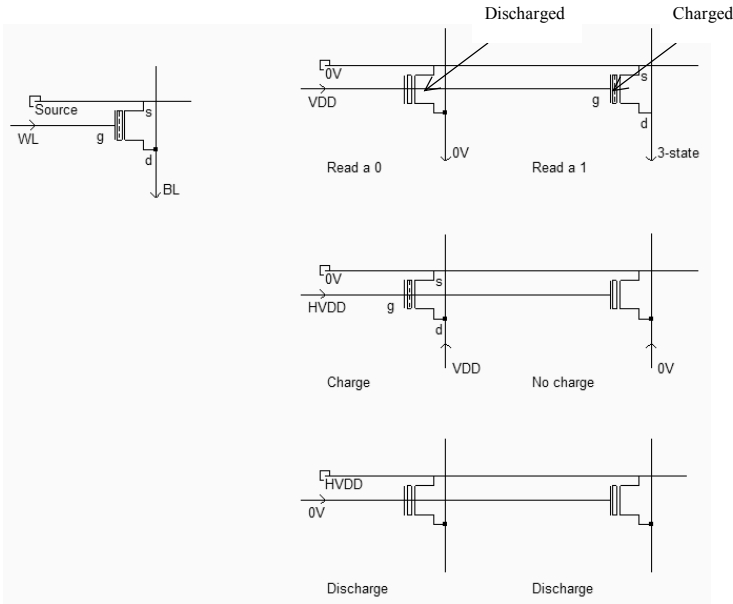


Fig. 8-26. The flash memory point and the principles for charge/discharge (FlashMemory.SCH)

The Flash memory point usually has a "T-shape", due to an increased size of the source for optimum tunneling effect [Sharma]. The horizontal polysilicon2 is the bit line, the vertical metal2 is the word line which links all drain regions together. The horizontal metal line links all sources together. It is a common practice to violate usual design rules, in order to achieve a more compact layout. In the case of figure 8-27, the poly extension is reduced from 3 lambda to 2 lambda.

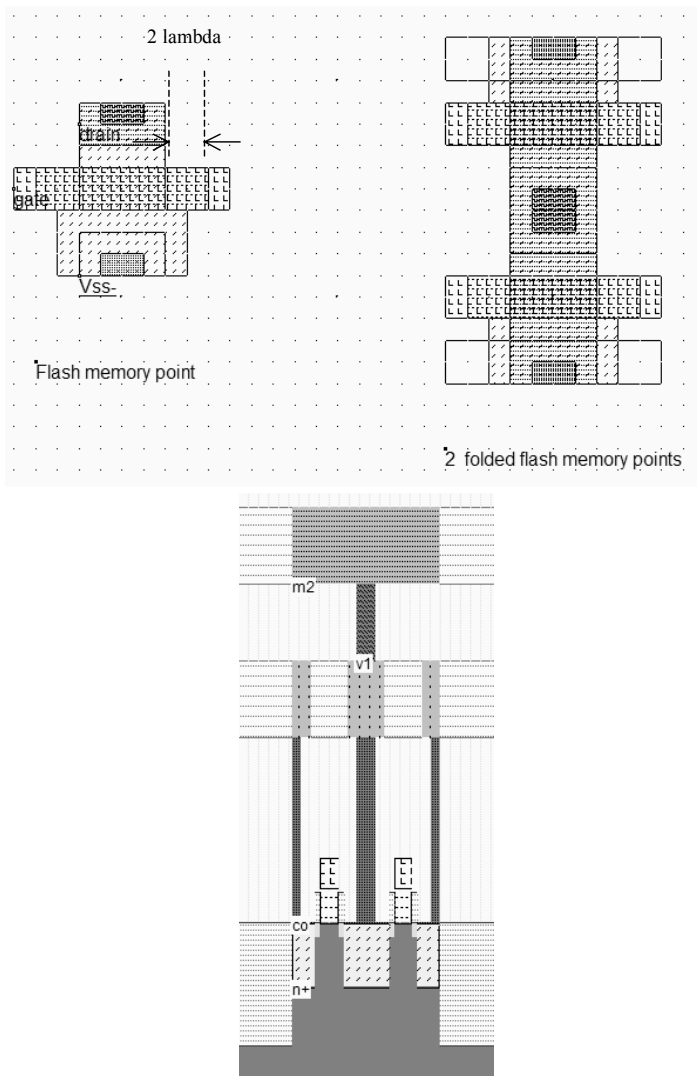


Fig. 8-27. The flash memory point and the associated cross-section (Flash8x8.MSK)

8.10 Ferroelectric RAM memories

Ferroelectric RAM memories are the most advanced of the Flash memory challengers [Geppert]. The FRAM is exactly like the DRAM except that the FRAM memory point is based on a two-state ferroelectric insulator, while the DRAM relies on a silicon dioxide capacitor. Mega-bit FRAM are already available as stand alone products. However, FRAM embedded memories have been made compatible since the 90nm CMOS technology. The Microwind software should first be configured in 90nm to access the FRAM properties using the command **File → Select Foundry**.

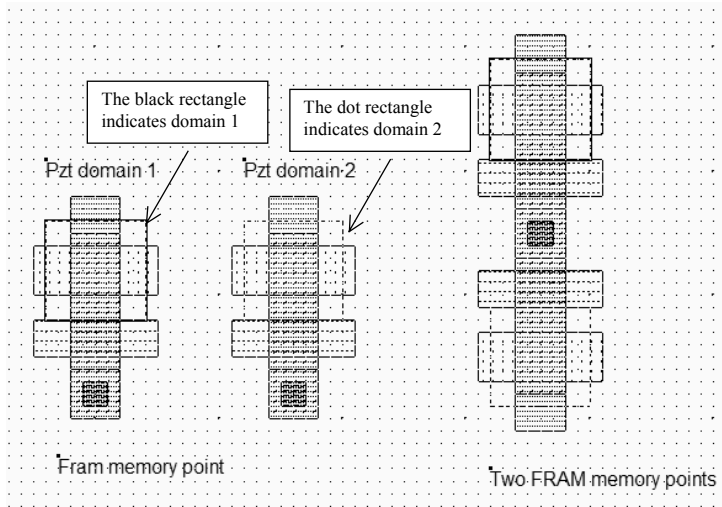


Fig. 8-28. The two domains of the FRAM memory (FramCell.MSK)

The 2D cross-section (Figure 8-29) shows the ferroelectric crystalline material made from a compound of lead, zirconium and titanium (PZT). The chemical formulation of PZT is $PbZr_{1-x}Ti_xO_3$. Adjusting the proportion of zirconium and titanium changes the electrical properties of the material.

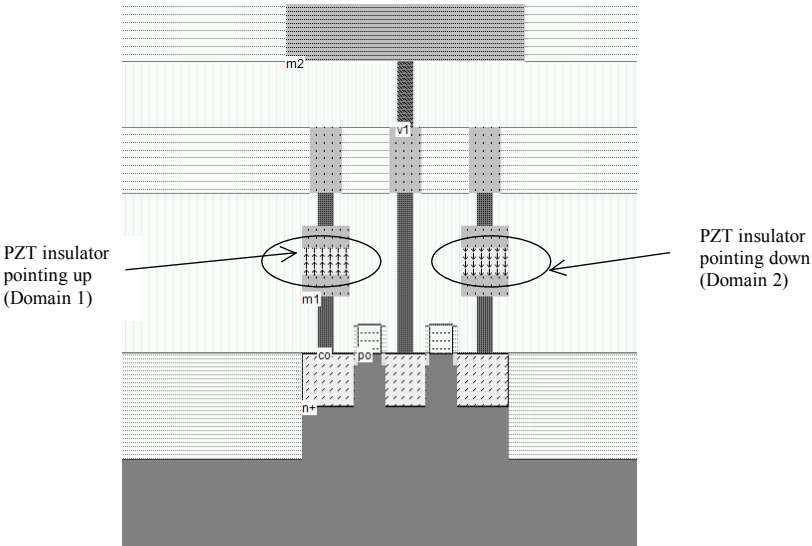


Fig. 8-29. The two domains of the FRAM memory (FramCell.MSK)

The $PbTiO_3$ molecular structure is given in figure 10-90. It is equivalent to a cube, where each of the eight corners is an atom of lead (Pb). In the center of the cube is a titanium atom, which is a class IVb element, with oxygen atoms at its ends, shared with neighbors. The two stable states of the molecule are shown in the figure 8-30. The titanium atom may be moved inside the cell by applying an electrical field. The remarkable properties of this insulator material are: the

stable state of the titanium atom even without any electrical field, the low electrical field required to move the atom, and its very high dielectric constant (Around 100).

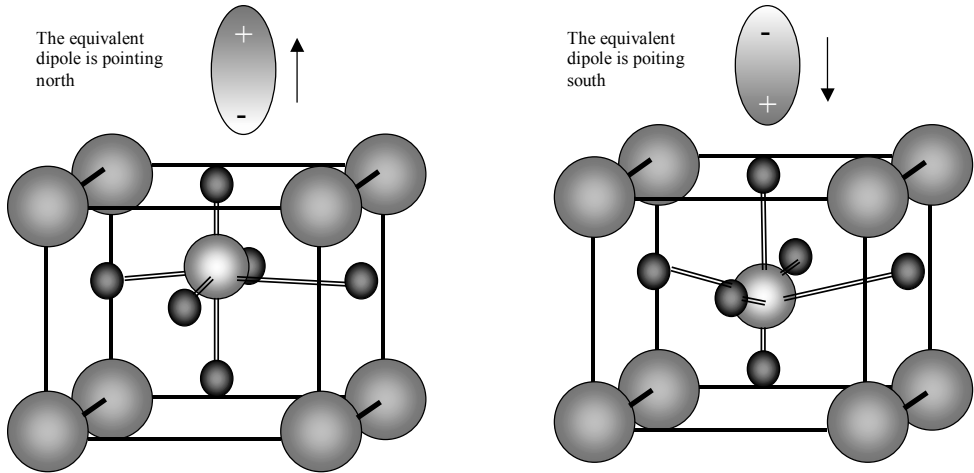


Fig. 8-30. The two domains of the structure which change the orientation of the equivalent dipole

The PZT capacitor behavior is usually represented by an hysteresis curve shown in figure 8-31. In the X Axis, the electrical field applied to the electrodes is displayed. The Y axis represents the dipole orientation for each molecule. It can be seen that if a minimum field is applied on the capacitor, the polarization changes. An inverted electrical field is required to change the state of the material.

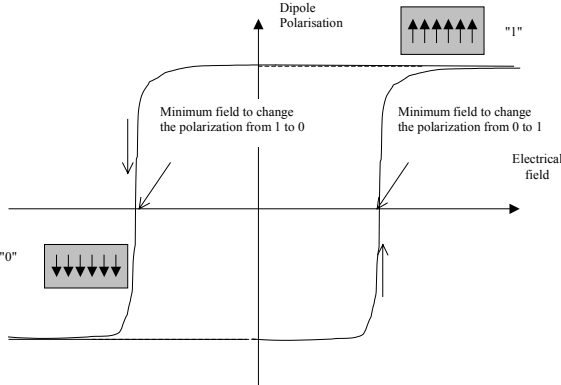


Fig. 8-31. The hysteresis curve of the PZT insulator

Consequently, the write cycle simply consists, for a 1, in applying a large positive step which orients the dipoles north, and for a zero in applying a negative voltage step, which orients the dipoles south (Figure 8-32).

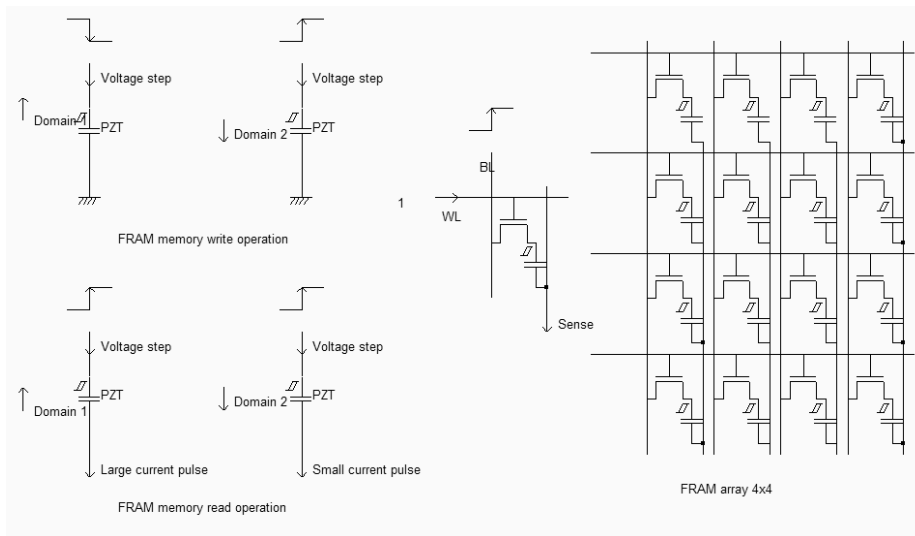


Fig. 8-32 The FRAM circuit principles and architecture (Fram4x4.SCH)

To read the domain information, an electrical field is applied to the PZT capacitor, through a voltage pulse. If the electric field is oriented in the opposite direction of the elementary dipole and is strong enough, the inner atom orientation is changed, which creates a significant current which is amplified and considered as a 1. If the electric field is oriented in the same direction as the elementary dipole, only a small current pulse is observed which is considered as a 0. Reading the logical information is equivalent to observing the current peak and deciding whether the current peak is small (0), or large (1). Notice that the read operation destroys the data stored in the PZT material, as for the DRAM. Just after the memory information is read, the logic information must be written back to the memory cell.

8.11 Memory Interface

All inputs and outputs of the RAM are synchronized to the rise edge of the clock, and more than one word can be read or written in sequence. The typical chronograms of a synchronous RAM are shown in figure 8-33. The active edge of the clock is usually the rise edge. One read cycle includes 3 active clock edges in the example shown in figure 8-33. The row address selection is active at the first rise edge, followed by the column address selection. The data is valid at the third fall edge of the system clock.

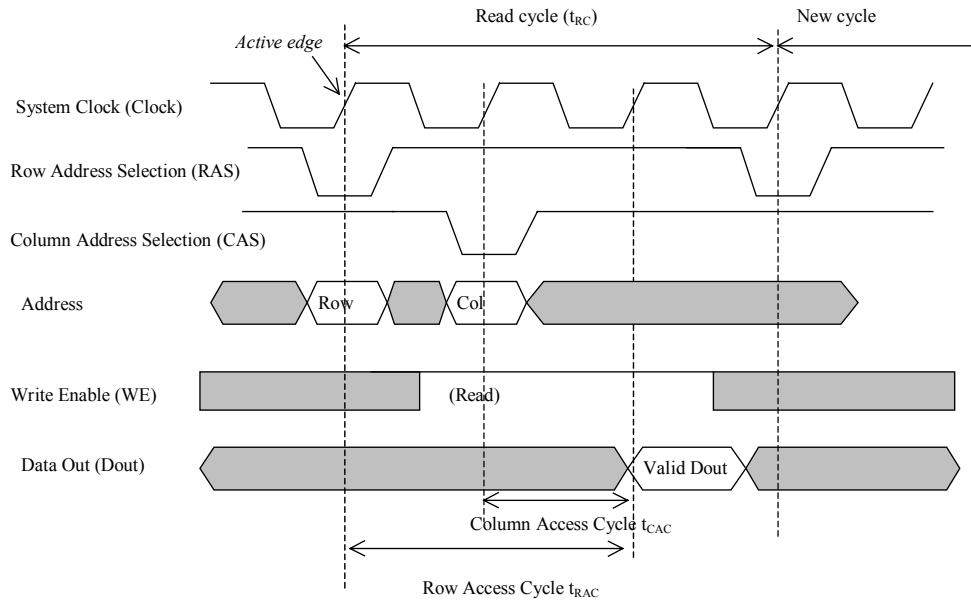


Figure 8-33: Synchronous RAM timing diagram

Double data Rate memories involve both the rise and fall edge of the clock [Sharma]. Furthermore, a series of data from adjacent memories may be sent on the data bus. Two contiguous data are sent, one on the rise edge of the clock, the other on the fall edge of the clock. This technique is called "burst-of-two". An example of double data rate and burst-of-two data in/out is proposed in figure 8-34. Notice that *Data In* and *Data Out* work almost independently.

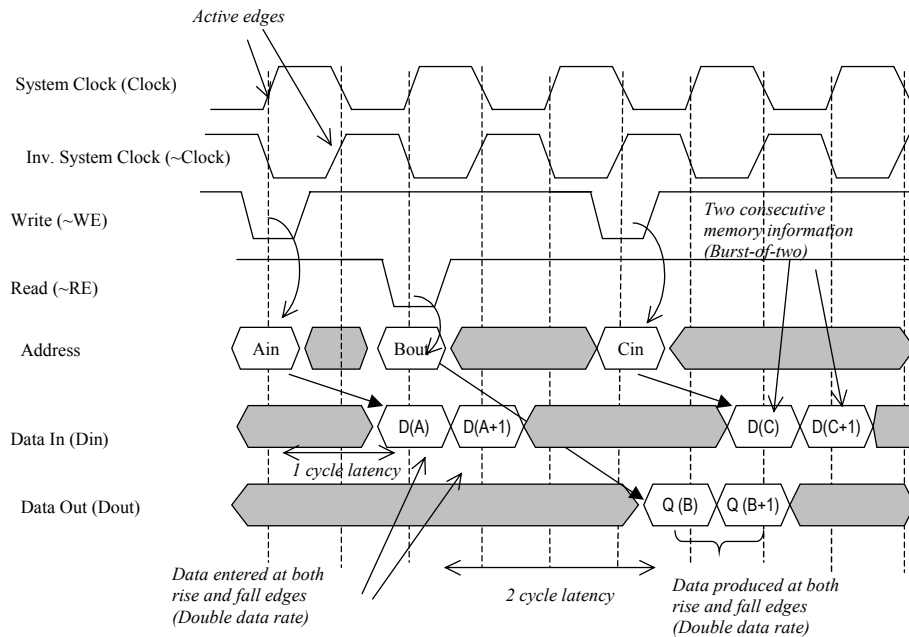


Figure 8-34: Double data rate diagram

8.12 EXERCISES

- Compare the leakage current on a DRAM cell for the following technologies : 0.35 μm , 0.12 μm and 90nm.
- Given a 4x4 EEPROM memory array, create the chronograms to write the words 0001, 0010, 0100 and 1000, and then to read these values.
- Modify the ROM array to write the word "Welcome".

9 Analog Cells

This chapter deals with analog basic cells, from the simple resistor and capacitor to the operational amplifier.

9.1 Resistor

An area-efficient resistor available in CMOS process consists of a strip of polysilicon [Hasting]. The resistance between *s1* and *s2* is usually counted in a very convenient unit called "ohm per square", noted Ω/\square . The default value polysilicon resistance per square is 10Ω , which is quite small, but rises to 200Ω if the salicide material is removed (Figure 9-1).

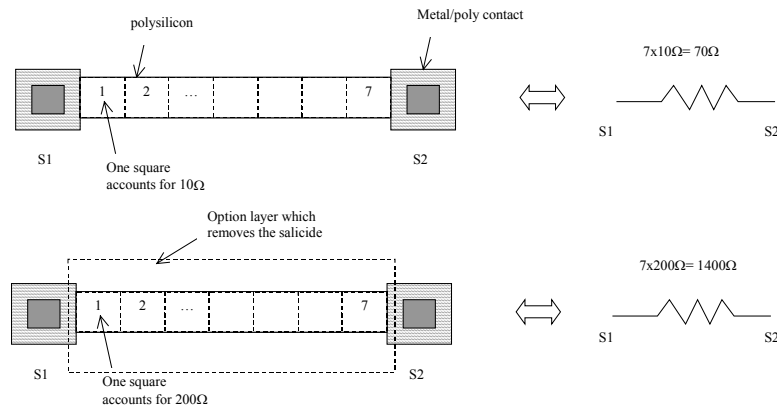


Figure 9-1 : The polysilicon resistance with unsalicide option

In the cross-section shown in figure 9-2, the salicide material deposited on the upper interface between the polysilicon layer and the oxide creates a metal path for current that reduces the resistance dramatically. Notice the shallow trench isolation and surrounding oxide that isolate the resistor from the substrate and other conductors, enabling very high voltage biasing (up to 100V). However, the oxide is a poor thermal conductor which limits the power dissipation of the polysilicon resistor.

The salicide is part of the default process, and is present at the surface of all polysilicon areas. However, it can be removed thanks to an option layer programmed by a double click in the option layer box, and a tick at "Remove Salicide". In the example shown in figure 9-3, the default resistance is 76Ω , and the unsalicide resistance rises to 760Ω .

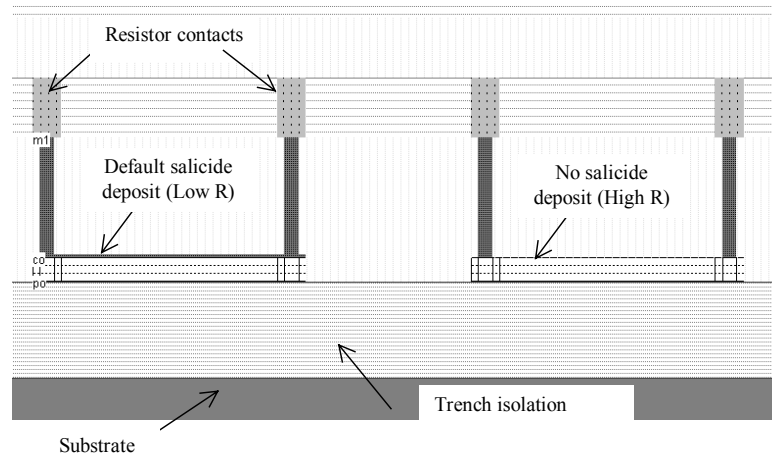


Figure 9-2 : Removing the salicide material to increase the sheet resistance (ResPoly.MSK)

This block contains two diagrams and a screenshot. The top diagram, titled 'Default poly (low resistance per square due to salicide)', shows a resistor with $R(\text{poly})=33$. The bottom diagram, titled 'Unsalicide poly (high resistance per square)', shows a resistor with $R(\text{poly})=330$. To the right is a screenshot of the 'Navigator' window, showing the 'Options' tab. Under 'Mos options', the 'Remove salicide to increase resistance' checkbox is checked and circled. An arrow points from this checkbox to the text 'Option layer used to remove the salicidation'.

Figure 9-3 : Removing the salicide material thanks to an option layer

Other resistors consist of N+ or P+ diffusions. An interesting feature of diffusion resistor is the ability to combine a significant resistance value and a diode effect. The diffusion resistor is used in input/output protection devices.

The resistor value varies because of lithography and process variations. In the case of the poly resistance, the width, height and doping may vary (Figure 9-4 left). Polysilicon resistors are rarely designed with the minimum 2λ width, but rather 4 or 6λ , so that the impact of the width variations is smaller. But the equivalent resistance is smaller, meaning less silicon efficiency. A variation ΔW of 0.2λ on both edges results in a 20% variation of the resistance on a 2λ width resistor, but only a 10% variation for a larger resistor designed with a width of 4λ .

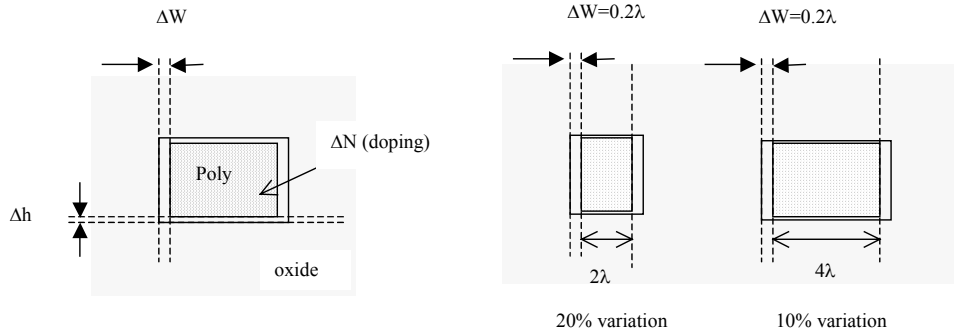


Figure 9-4 : Resistance variations with the process

There exist efficient techniques to reduce the resistance variations within the same chip. In figure 9-5, the resistor design on the left upper part is not regular, uses various polysilicon widths, and sometimes uses too narrow conductors. Although the design rules are not violated, the process variations will enlarge the spread of resistance values. To minimize the effects of process variations, resistors should:

- Always be laid out with an identical width
- Use at least twice the minimum design rules
- Use the same orientation
- Use dummy resistance. These boxes of poly have no active role. Their role is to have a regular width variation on the active part.

Dog-bone resistors may not pack as densely as serpentine resistors as two metal layers are used and induce supplementary design rules [Baker][Hastings] but are said to be less sensitive to process variations.

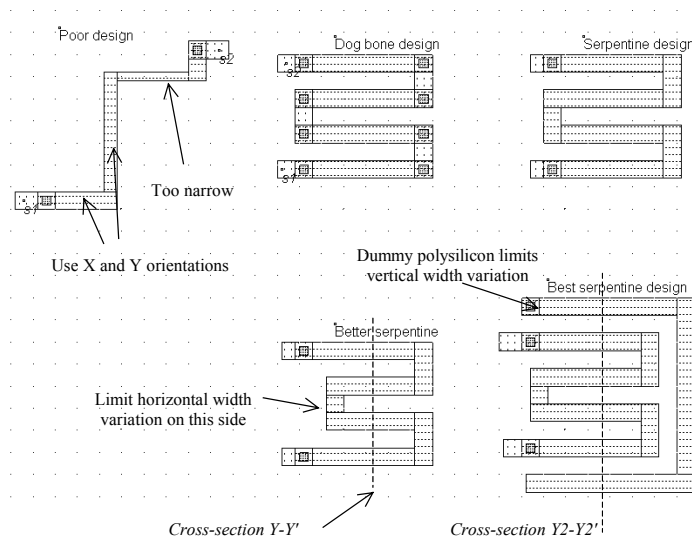


Figure 9-5 : Resistance design (ResPoly.MSK)

9.2 Capacitor

Capacitors are used in analog design to build filters, compensation, decoupling, etc.. Ideally, the value of the capacitor should not depend on the bias conditions, so that the filtering effect would be situated at constant frequencies.

9.2.1 Diode Capacitor

Diodes in reverse mode exhibit a capacitor behavior, however, the capacitance value is strongly dependent on the bias conditions [Hastings]. A simple N+ diffusion on a P-substrate is a NP diode, which may be considered as a capacitor as long as the N+ region is polarized at a voltage higher than the P-substrate voltage which is usually the case as the substrate is grounded (0V). In 0.12μm, the capacitance is around 300aF/μm2 (1 atto-Farad is equal to 10⁻¹⁸ Farad).

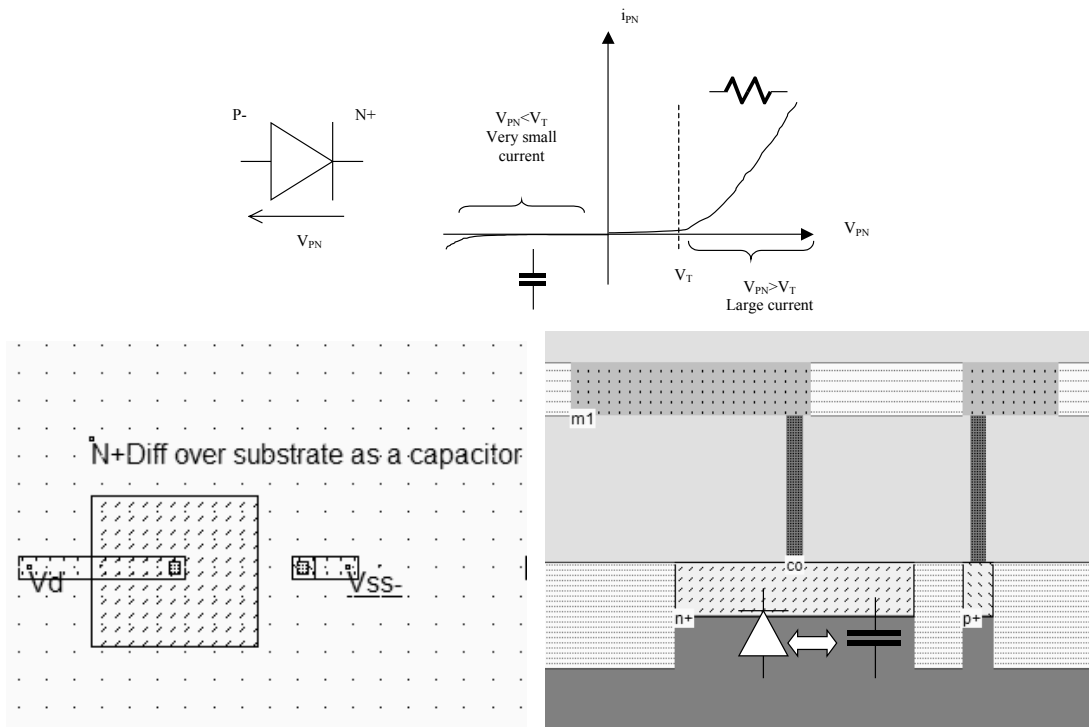


Figure 9-6: The diffusion over substrate as a non-linear capacitor (Capa.MSK)

The typical variation of the capacitance with the diffusion voltage V_N is given in figure 9-7. The capacitance per μm2 provided in the electrical rules is a rude approximation of the capacitance variation. A large voltage difference between V_N and the substrate result in a thick zone with empty charges, which corresponds to a thick insulator, and consequently to a small capacitance. When V_N is lowered, the zone with empty charges is reduced, and the capacitance increases. If V_N goes lower than the substrate voltage, the diode starts to conduct.

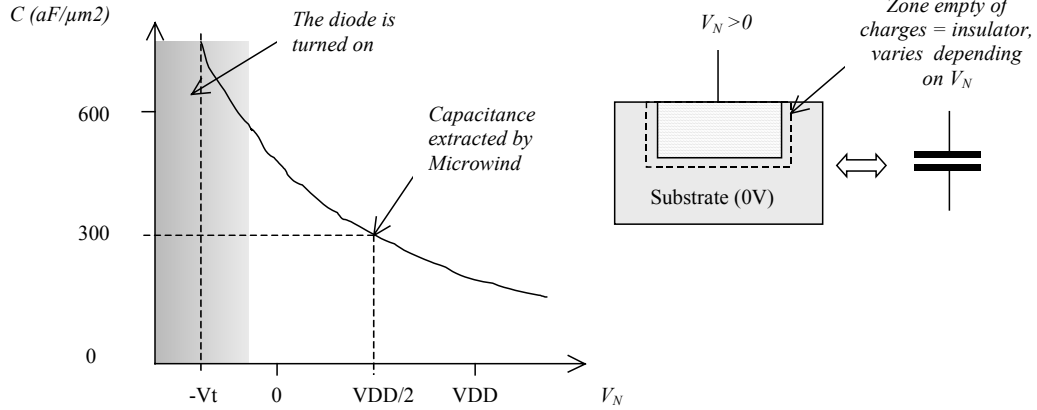


Figure 9-7: The diffusion capacitance varies with the polarization voltage

9.3 Mos Capacitor

The MOS transistor is often the simplest choice to build a capacitor. In 0.12μm, the gate oxide has an equivalent thickness of 2nm (20 angstrom, also written 20Å), which leads to a capacitance evaluated by formulation 9-1.

$$C_{thinox} = \frac{\epsilon_0 \epsilon_r}{e} = \frac{8,85e^{-12} \times 3.9}{2.0 \times 10^{-9}} = 17e^{-3} F/m^2 = 17e^{-15} F/\mu m^2 = 17fF/\mu m^2 \quad (\text{Eq. 9-1})$$

where

e= gate oxide thickness (m)

ε₀=vacuum permittivity (F/m)

ε_r=relative permittivity (no unit)

The design of a gate capacitor using a large MOS device is shown in figure 9-8, with a capacitance of around 300fF. In analog design, the gate capacitor is often surrounded by a guard ring. It is usually very difficult to integrate capacitors of more than a few hundred pico-farads. If nano-farad capacitors are required, these components are too large to be integrated on-chip, and must be placed off-chip.

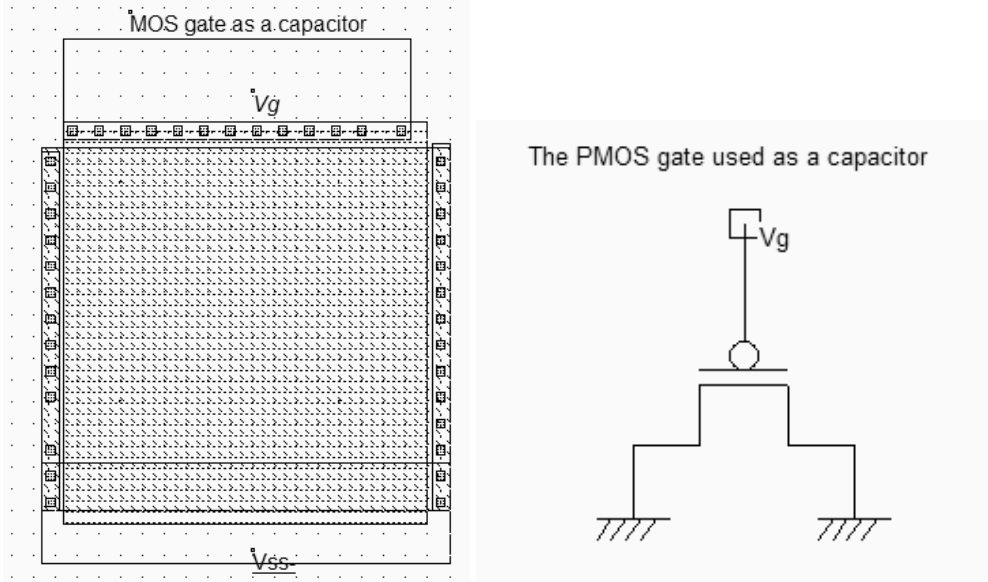


Figure 9-8: Generating an efficient capacitor based on a MOS device with very large length and width (CapaPoly.MSK)

9.4 Poly-Poly2 Capacitor

Most deep-submicron CMOS processes incorporate a second polysilicon layer (poly2) to build floating gate devices for EEPROM. An oxide thickness around 20nm is placed between the poly and poly2 materials, which induces a plate capacitor around $1,7\text{fF}/\mu\text{m}^2$. In Microwind, the command "Edit → Generate → Capacitor" gives access to a specific menu for generating capacitor (Figure 9-9). The parameter in the design rules used to configure the poly-poly2 capacitor is CP2PO.

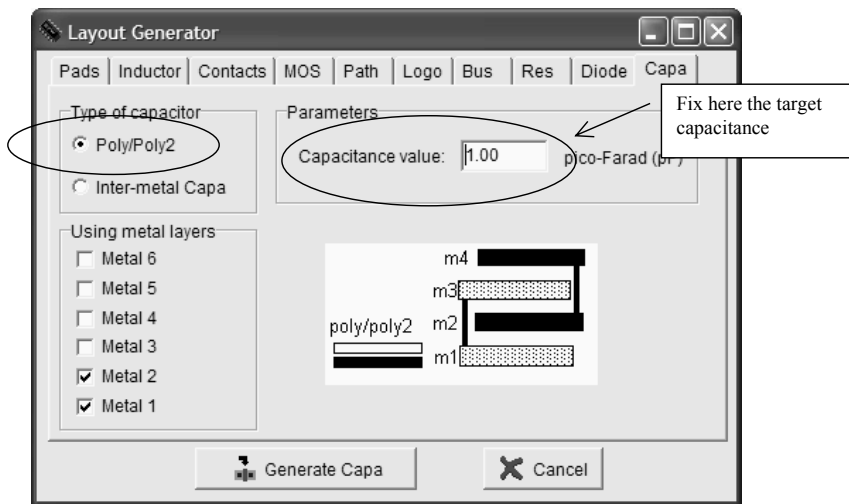


Figure 9-9: The generator menu handles the design of poly/poly2 capacitor and inter-metal capacitors

The poly/poly2 capacitor simply consists of a sheet of polysilicon and a sheet of poly2, separated by a specific dielectric oxide which is 20nm in the case of the default CMOS 0.12µm process.

9.5 Inter-Metal Capacitor

The multiplication of metal layers create lateral and vertical capacitance effects of rising importance. Although the inter-metal oxide thickness is 10 to 50 times larger than the ultra-thin gate oxide thickness, the spared silicon area in upper metal layers may be used for small size capacitance, which might be attractive for compensation or local decoupling. Depending on the desired capacitor value, Microwind computes the size of the square structure, made of metal plates, that reaches the capacitance value. In figure 9-10, a sandwich of metal1, metal2, metal3 and metal 4 is selected, for a target value of 100fF.

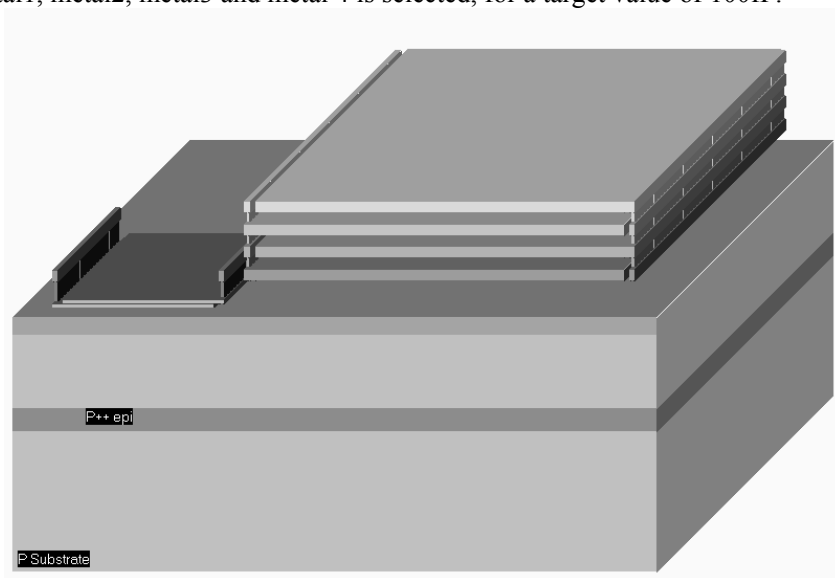


Figure 9-10: Generating an inter-metal capacitor (CapaPolyMetalComp.MSK)

9.6 Diode-connected MOS

The schematic diagram of the diode-connected MOS is proposed in figure 9-11. This circuit features a high resistance within a small silicon area. The key idea is to build a permanent connection between the drain and the gate. Most of the time, the source is connected to ground in the case of n-channel MOS, and to VDD in the case of p-channel MOS.

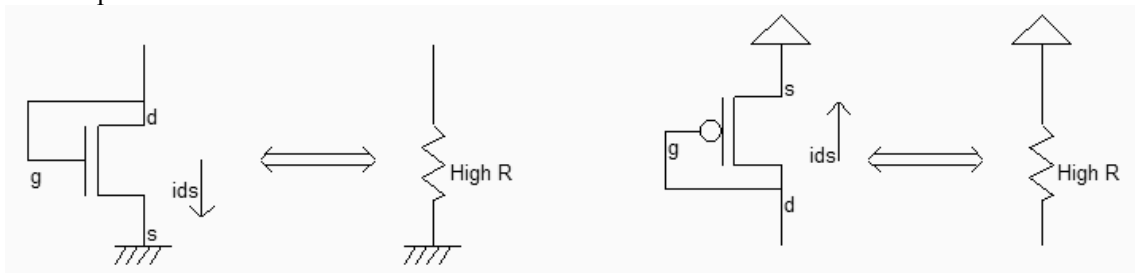


Figure 9-10 : Schematic diagram of the MOS connected as a diode (MosRes.SCH)

To create the diode-connected MOS, the easiest way is to use the MOS generator. Enter a large length and a small width, for example $W=0.24\mu\text{m}$ and $L=2.4\mu\text{m}$. This sizing corresponds to a long channel, featuring a very high equivalent resistance. Add a poly/metal contact and connect the gate to one diffusion. Add a clock on that node. Add a VSS property to the other diffusion. The layout result is shown in figure 9-11.

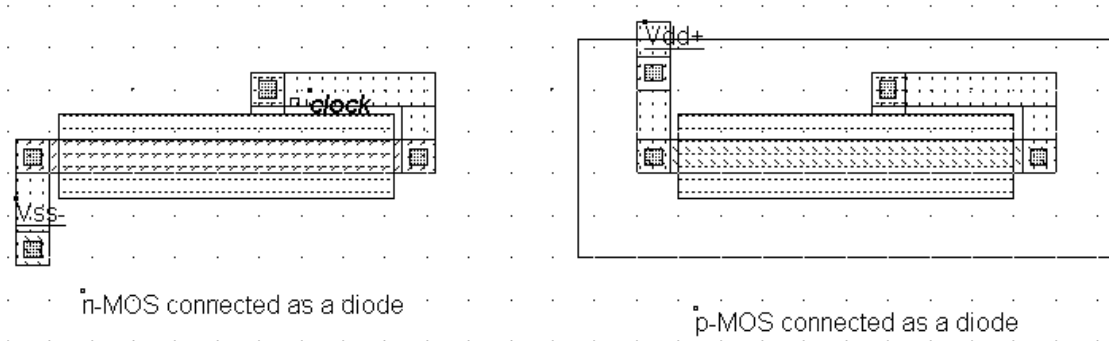


Figure 9-11 : Schematic diagram of the MOS connected as a diode (ResMos.MSK)

Now, click **Simulation on Layout**. In a small window, the MOS characteristics are drawn, with the functional point drawn as a color dot (Figure 9-12). It can be seen that the I/V characteristics correspond to a diode. The resistance is the invert value of the slope in the I_d/V_d characteristics. For V_{ds} larger than 0.6V, the resistance is almost constant. As the current I_{ds} increases of $10\mu\text{A}$ in 0.4V, the resistance can be estimated around $40\text{K}\Omega$. A more precise evaluation is performed by Microwind if you draw the slope manually. At the bottom of the screen, the equivalent resistance appears, together with the voltage and current.

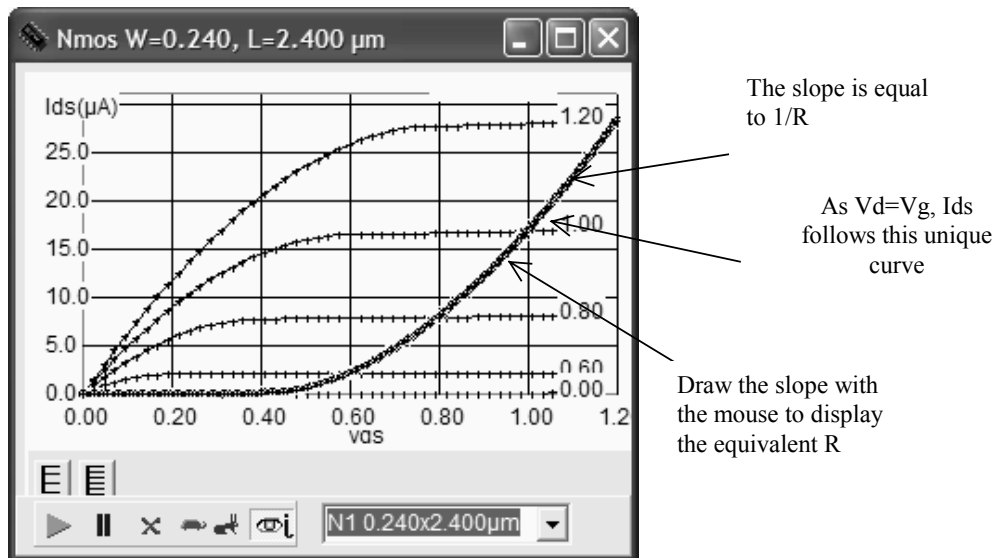


Figure 9-12 : Using the Simulation on Layout to follow the characteristics of the diode-connected MOS (ResMos.MSK)

In summary, the MOS connected as a diode is a capacitance for $V_{gs} < V_t$, a high resistance when V_{gs} is higher than the threshold voltage V_t . The resistance obtained using such a circuit can easily reach 100K Ω in a very small silicon area.

9.7 Voltage Reference

The voltage reference is usually derived from a voltage divider made from resistance. The output voltage V_{ref} is defined by equation 9-1.

$$V_{ref} = \frac{R_N}{R_N + R_P} V_{DD} \quad (\text{Eq. 9-1})$$

with

V_{DD} =power supply voltage (1.2V in 0.12 μm)

R_N =equivalent resistance of the n-channel MOS (ohm)

R_P =equivalent resistance of the p-channel MOS (ohm)

The value of the resistance must be high enough to keep the short -circuit current low, to avoid wasted power consumption. A key idea is to use MOS devices rather than polysilicon or diffusion resistance to keep the silicon area very small. Notice that two n-MOS or two p-MOS properly connected feature the same function. P-MOS devices offer higher resistance due to lower mobility, compared to n-channel MOS. Four voltage reference designs are shown in figure 9-13. The most common design uses one p-channel MOS and one n-channel MOS connected as diodes.

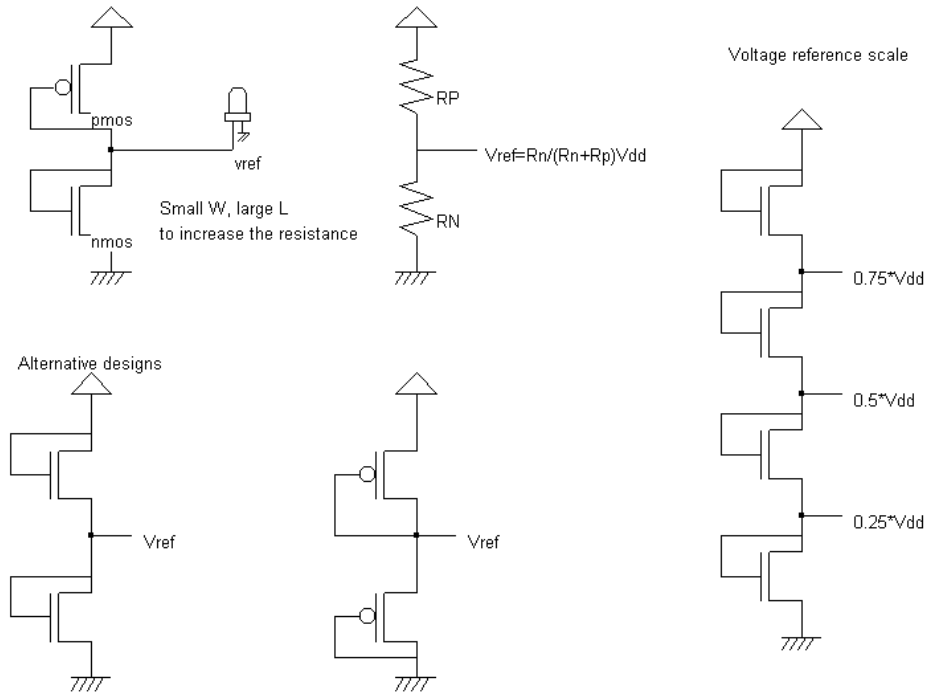


Figure 9-13 : Voltage reference using PMOS and NMOS devices as large resistance

The alternative solutions consist in using two n-channel MOS devices only (Left lower part of the figure), or their opposite built from p-channel devices only. Not only one reference voltage may be created, but also three, as shown in the right part of the figure, which use four n-channel MOS devices connected as diodes.

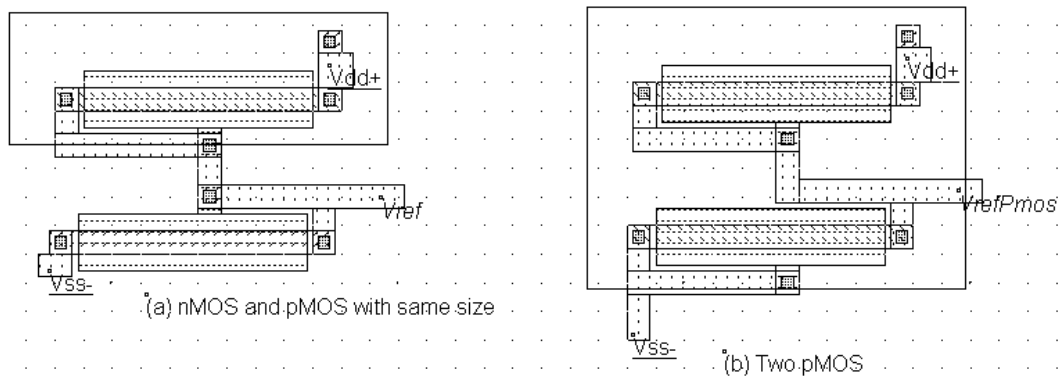


Figure 9-14 : Voltage reference circuits (a) with one nMOS and one pMOS (b) with two pMOS ($V_{ref.MSK}$)

In the layout of figure 9-14, the pMOS and nMOS have the same size. Due to lower pMOS mobility, the resulting V_{ref} is a little lower than $VDD/2$. Using BSIM4 instead of model 3, we see that the voltage reference obtained with two identical pMOS devices is not $VDD/2$ either, as shown in the simulation of figure 9-15. This is due to the non-symmetrical polarization of the pMOS regarding the substrate voltage V_{bs} , which has a significant impact on the current. Consequently, a good $VDD/2$ voltage reference requires a precise adjustment of MOS sizing, a good confidence in the accuracy of the model, and several iterations of design/simulation until the target reference voltage is reached.

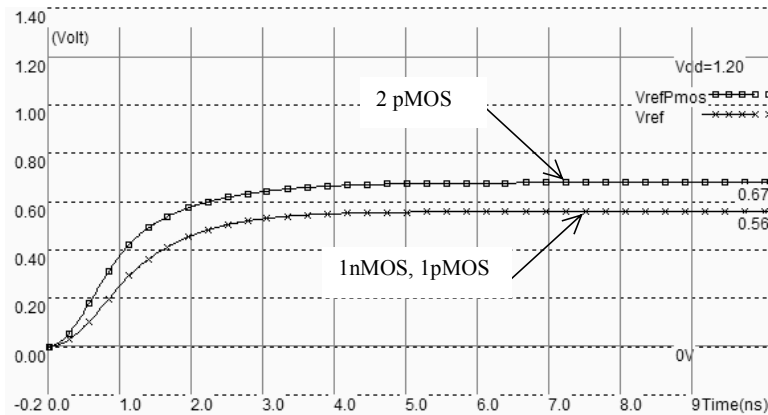


Figure 9-15 : Simulation of the two voltage reference circuits (*Vref.MSK*) using BSIM4 model

9.8 Current mirror

The current mirror is one of the most useful basic blocs in analog design. It is primarily used to copy currents. When a current flows through a MOS device $N1$, an almost identical current flows through the device $N2$, as soon as $N1$ and $N2$ are connected as current mirrors. In its most simple configuration, the current mirror consists of two MOS devices connected as shown in figure 9-16. A current $I1$ flowing through the device *Master* is copied onto the MOS device *Slave*. If the size of *Master* and *Slave* are identical, in most operating conditions, the currents $I2$ and $I1$ are identical. The remarkable phenomenon is that the current is almost independent from the load represented in figure 9-16 by a resistor R_{load} .

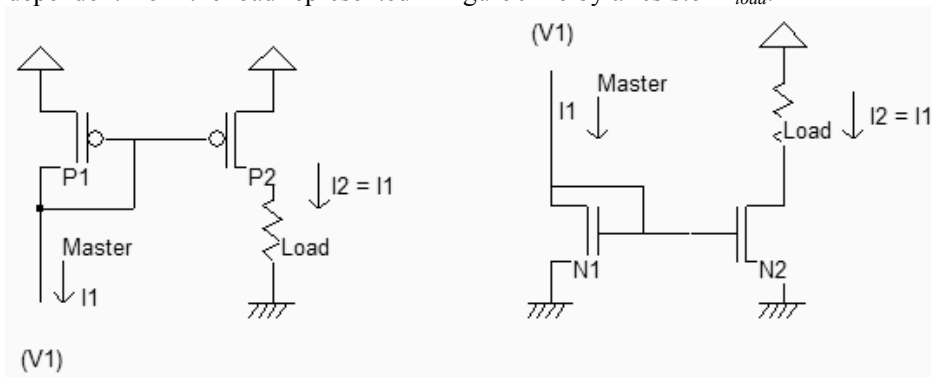


Figure 9-16: Current mirror principles in nMOS and pMOS version (*CurrentMirror.SCH*)

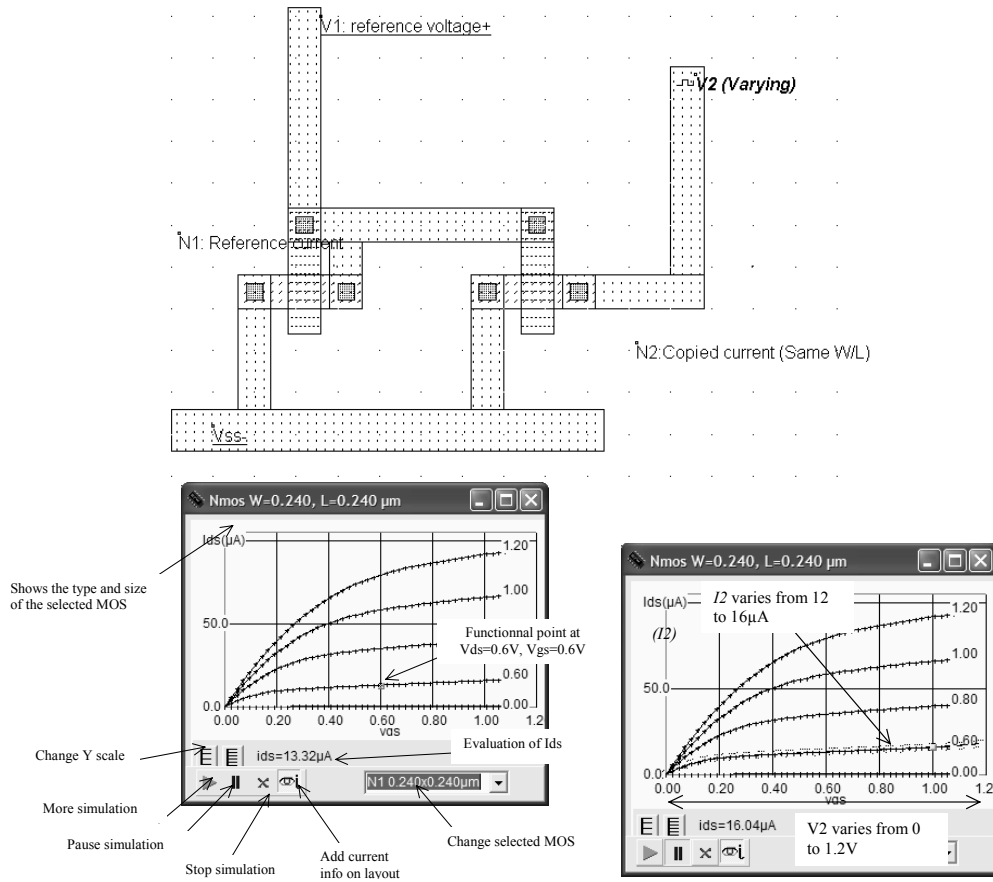


Figure 9-17: Layout of an n-channel current mirror with identical size and simulation of the mos devices (CurrentMirror.MSK)

The illustration of the current mirror behavior is shown in the case of two identical N-channel MOS (Figure 9-17). The current of the master $N1$ is fixed by $V1$, which is around 0.6V in this case. We use the simulation on layout to observe the current flowing in $N1$ and $N2$. Concerning $N1$, the gate and drain voltage is fixed to 0.6V, which corresponds to a constant current of around $13\mu\text{A}$, as shown in figure 11-42. The voltage $V2$ varies thanks to a clock. We observe that the current $I2$ is almost equal to $13\mu\text{A}$, independently of $V2$, except when $V2$ is lower than 0.2V. More precisely, the variation of $I2$ is between 12 and $16\mu\text{A}$ when $V2$ varies from 0.2 to 1.2V.

Mos Matching

A set of design techniques can improve the current mirror behavior, which are described hereafter.

- All MOS devices should have the same orientation. During fabrication, the chemical process has proven to be slightly different depending on the orientation, resulting in variations of effective channel length. This mismatch alters the current duplication if one nMOS are implemented horizontally, the other vertically.
- Long channel MOS devices are preferred. In such devices, the channel length modulation is small, and consequently I_{ds} is almost independent of V_{ds} .

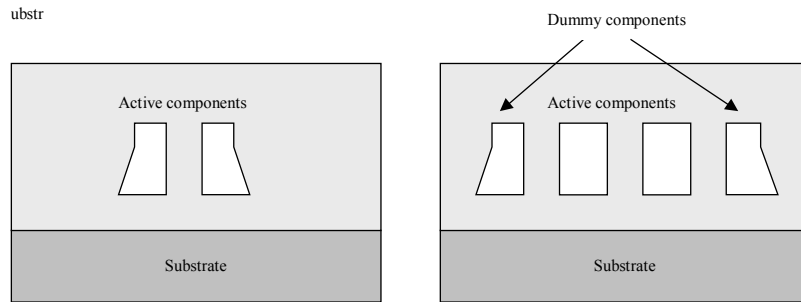


Figure 9-18: 2D aspect of the circuit without and with dummy components

- Dummy gates should be added on both sides of the current mirror. Although some silicon area is lost, due to the addition of inactive components, the patterning of active gates leads to very regular structures, ensuring a high quality matching (See figure 9-18).
- MOS devices should be in parallel. If possible, portions of the two MOS devices should be interleaved, to reduce the impact of an always-possible gradient of resistance, or capacitance with the location within the substrate (Figure 9-19).

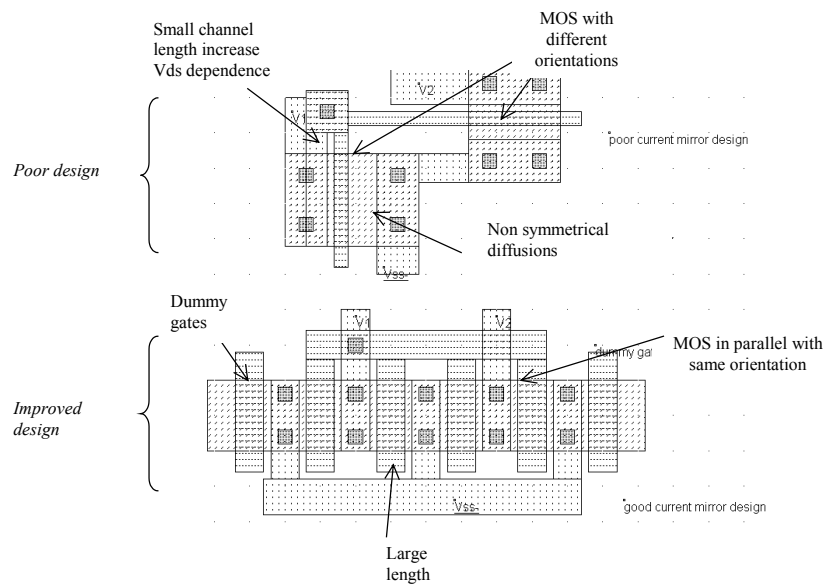


Figure 9-19: Design of high performance current mirrors (MirrorMatch.MSK)

9.9 Amplifier

The goal of the amplifier is to multiply by a significant factor the amplitude of a sinusoidal voltage input V_{in} , and deliver the amplified sinusoidal output V_{out} on a load. The single stage amplifier may consist of a MOS device (we choose here a n-channel MOS) and a load. The load can be a resistance or an inductance. In the circuit, we use a resistance made with a p-channel MOS device with gate and drain connected (Figure 9-20). The pMOS which replaces the passive load is called an active resistance.

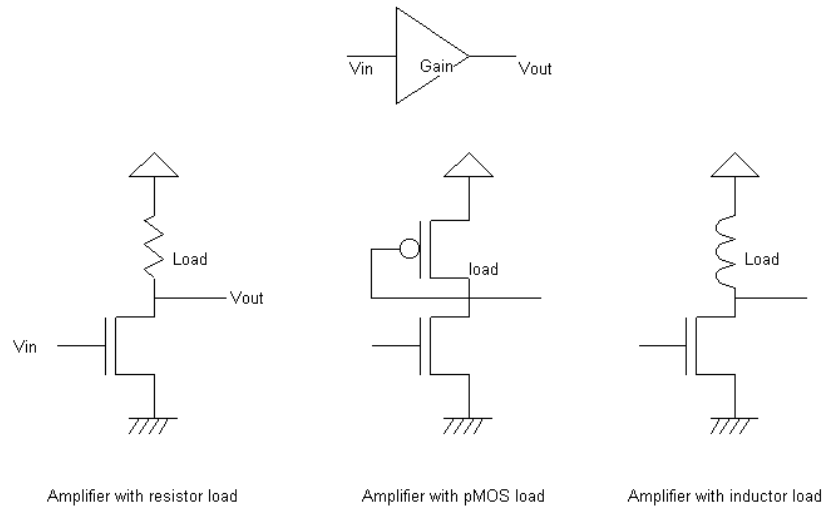


Figure 9-20: Single stage amplifier design with MOS devices (AmpliSingle.SCH)

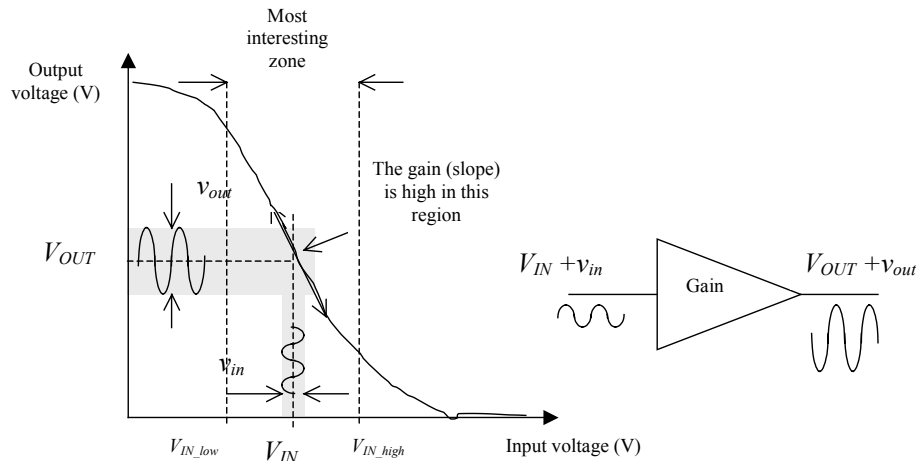


Figure 9-21: The amplifier has a high gain in a certain input range, where a small input signal v_{in} is amplified to a large signal v_{out} .

The single stage amplifier characteristics between V_{in} and V_{out} have a general shape shown in figure 9-21. The most interesting zone corresponds to the input voltage range where the transfer function has a linear shape, that is between V_{IN_low} and V_{IN_high} . Outside this voltage range, the behavior of the circuit does not correspond anymore to an amplifier. If we add a small sinusoidal input v_{in} to V_{IN} , a small variation of current i_{ds} is added to the static current I_{DS} , which induces a variation v_{out} of the output voltage V_{OUT} . The link between the variation of current i_{ds} and the variation of voltage v_{in} can be approximated by equation 9-2.

$$i_{ds} = g_m v_{gs} \tag{Equ. 9-2}$$

In figure 9-22, a nMOS device with large width and minimum length is connected to a high resistance pMOS load. A 50mV sinusoidal input (v_{in}) is superimposed to the static offset 0.6V (V_{IN}). What we expect is a 500mV sinusoidal wave (v_{out}) with a certain DC offset (V_{OUT}).

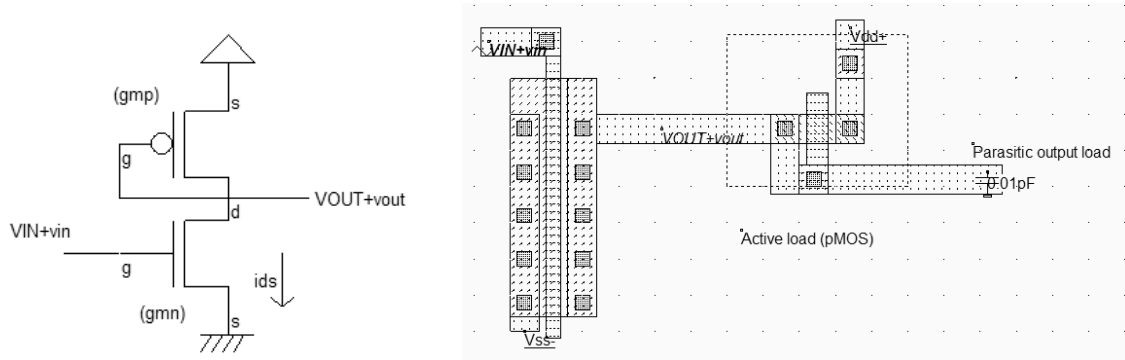


Figure 9-22: Single stage amplifier layout with a pMOS as a load resistor (AmpliSingle.MSK)

What we need now is to find the characteristics V_{out}/V_{in} in order to tune the offset voltage V_{IN} . In the simulation window, click **Voltage vs voltage** and **More**, to compute the static response of the amplifier (Figure 9-23). The range of voltage input that exhibits a correct gain appears clearly. For V_{DS} higher than 0.25V and lower than 0.4V, the output gain is around 3. Therefore, an optimum offset value is 0.35V. Change the parameter **Offset** of the input sinusoidal wave to place the input voltage in the correct polarization.

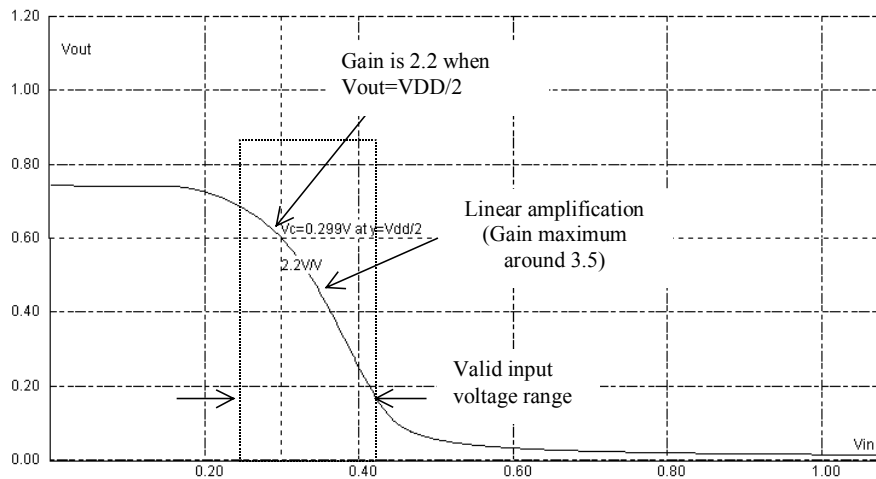


Figure 9-23: Single stage amplifier static response showing the valid input voltage range.

We change the sinusoidal input offset and start again the simulation. A gain of 3.5 is observed when the offset V_{IN} is 0.35V. In figure 9-24, the input amplitude is 100mV peak to peak, the output amplitude is 350mV peak-to-peak. These pieces of information appear in the information bar of the main window.

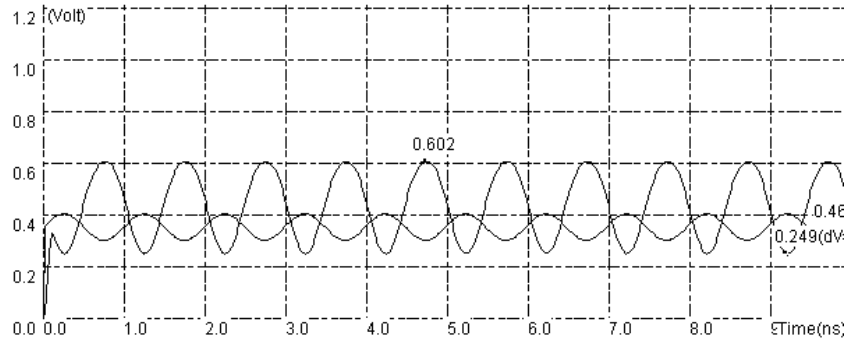


Figure 9-24: Single Stage amplifier with correct polarization $V_{IN}=0.35V$

9.10 Simple Differential Amplifier

The goal of the differential amplifier is to compare two analog signals, and to amplify their difference. The differential amplifier formulation is reported below (Equation 9-3). Usually, the gain K is high, ranging from 10 to 1000. The consequence is that the differential amplifier output saturates very rapidly, because of the supply voltage limits.

$$V_{out} = K(V_p - V_m) \quad (\text{Equ. 9-3})$$

The most simple MOS implementation for the differential amplifier is given in figure 9-25. We suppose that both V_p and V_m have an identical value V_{in} . Consequently, the two branches have an identical current I_1 so that no current flows to charge or discharge the output capacitor, which is connected to the output V_{out} (Left figure). Now, if the gate voltage of the N_1 device is increased to $V_{in}+dV$, the current through the left branch is increased to I_1' , greater than I_1 . The current mirror copies this I_1' current on the right branch, so that I_1' also flows through P_2 . As the N_2 gate voltage remains at V_{in} , the over-current $I_1'-I_1$ is evacuated to the output stage and charges the capacitor. The output voltage V_{out} rises.

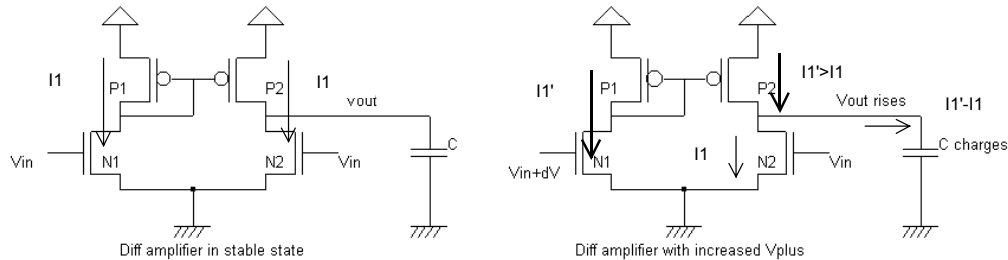


Figure 9-25: The differential amplifier at work (AmpliDiff.SCH)

The schematic diagram of a basic differential amplifier is proposed in figure 9-26. An nMOS device has been inserted between the differential pair and the ground to improve the gain. The gate voltage V_{bias}

controls the amount of current that can flow on the two branches. This pass transistor permits the differential pair to operate at lower V_{ds} , which means better analog performances and less saturation effects.

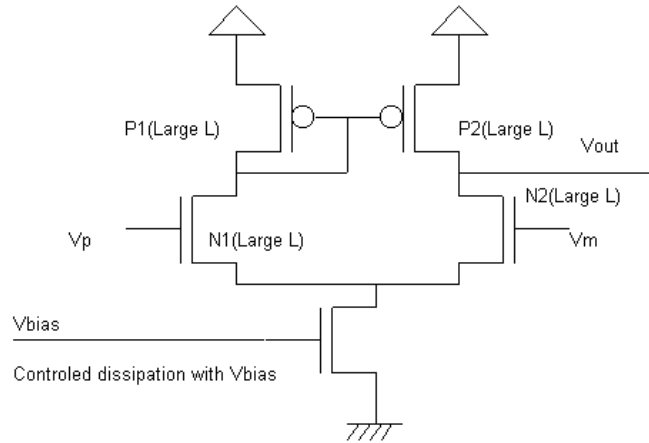


Figure 9-26: An improved differential amplifier (AmpliDiff.SCH)

The best way to measure the input range is to connect the differential amplifier as a follower, that is V_{out} connect to V_m . The V_m property is simply removed, and a contact poly/metal is added at the appropriate place to build the bridge between V_{out} and V_m . A slow ramp is applied on the input V_{in} and the result is observed on the output. We use again the « Voltage vs. Voltage » to draw the static characteristics of the follower. The BSIM4 model is forced for simulation by a label "BSIM4" on the layout.

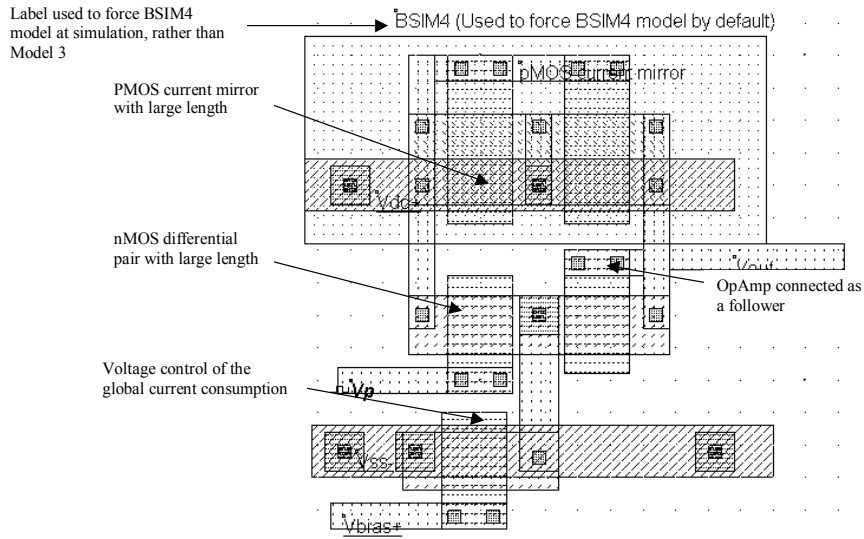


Figure 9-27: The layout corresponding to the improved differential amplifier (AmpliDiffLargeLength.SCH)

One convenient way to stimulating the follower response is to assign V_p a clock with a very slow rise and fall time. As can be seen from the resulting simulation reported in figure 9-28, a low V_{bias} features a larger voltage range, specifically at high voltage values. The follower works properly starting 0.4V, independently of the V_{bias} value. A high V_{bias} leads to a slightly faster response, but reduces the input range and consumes more power as the associated nMOS transistor drives an important current. The voltage V_{bias} is

often fixed to a value a little higher than the threshold voltage V_{tn} . This corresponds to a good compromise between switching speed and input range.

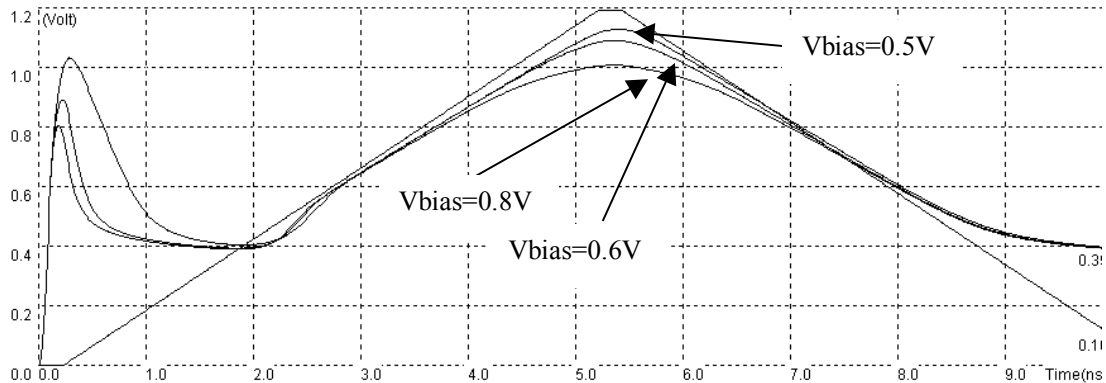


Figure 9-28: Effect of V_{bias} on the differential amplifier performance (*AmpliDiffVbias.MSK*)

9.11 Push-Pull Amplifier

The push-pull amplifier is built using a voltage comparator and a power output stage. Its schematic diagram is reported in figure 9-29, with some details about the important voltage nodes. The difference between V_p and V_m is amplified and produces a result, codified V_{out} . Transistors N_b and P_b are connected as diodes in series to create an appropriate voltage reference V_{bias} , fixed between the nMOS threshold voltage V_{tn} and half of VDD. The differential pair consists of transistors $N1$ and $N2$. This time, two stages of current mirrors are used: $P1$, $P2$ and $P3$, $P0$.

Node	Description	Typical value
V+	Positive analog input	Close to VDD/2
V-	Negative analog input	Close to VDD/2
Vbias	Bias voltage	A little higher than VTN
Vout	Analog output	0..VDD

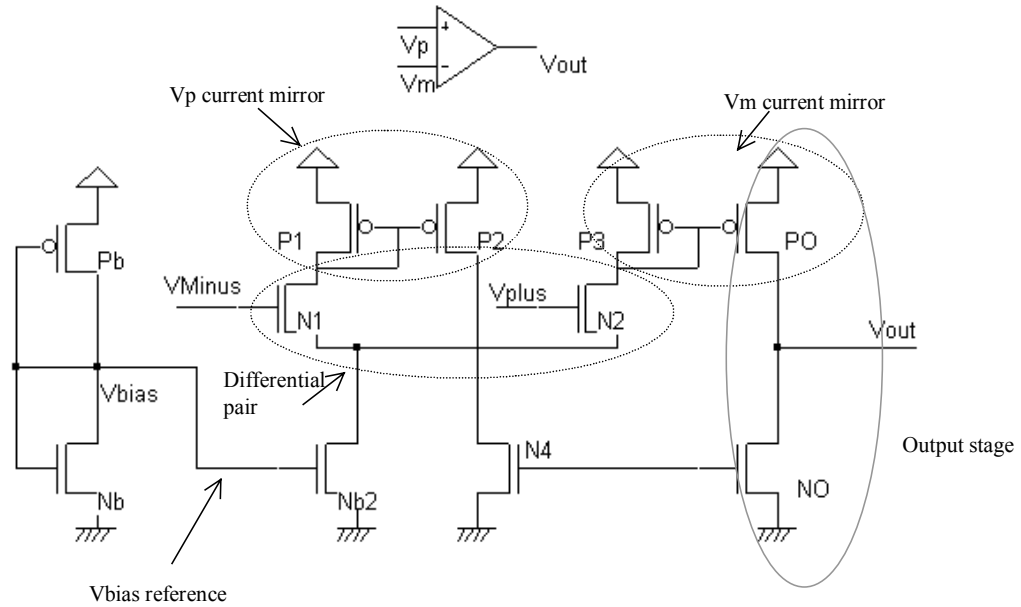


Figure 9-29: Push-pull operational amplifier (AmpliPushPull.SCH)

The output stage consists of transistors PO and NO. These transistors are designed with large widths in order to lower the output resistance. Such a design is justified when a high current drive is required: high output capacitor, antenna dipole for radio-frequency emission, or more generally a low impedance output. The ability to design the output stage according to the charge is a key advantage of this structure compared to the simple differential pair presented earlier. The implementation shown in figure 9-30 uses NO and PO output stage devices with a current drive around five times larger than the other devices. In practice, the ratio may rise up to 10-20.

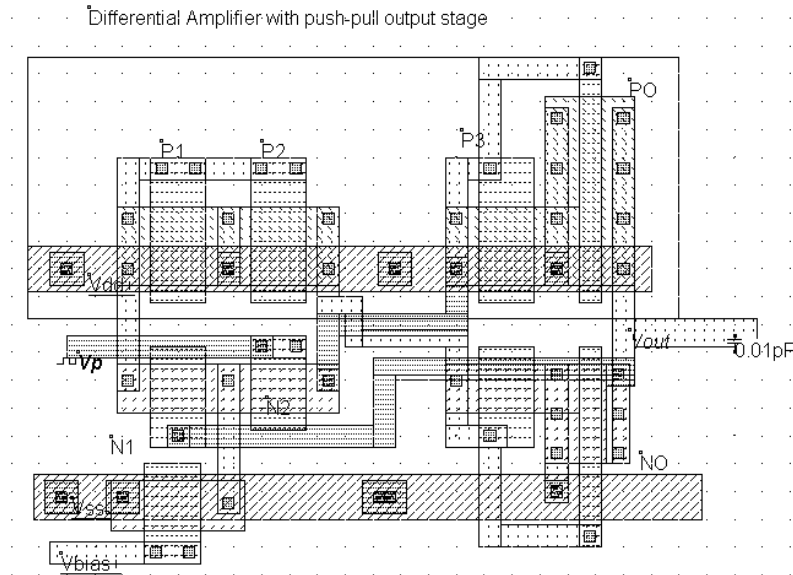


Figure 9-30: Push-pull operational amplifier (AmpliDiffPushPull.MSK)

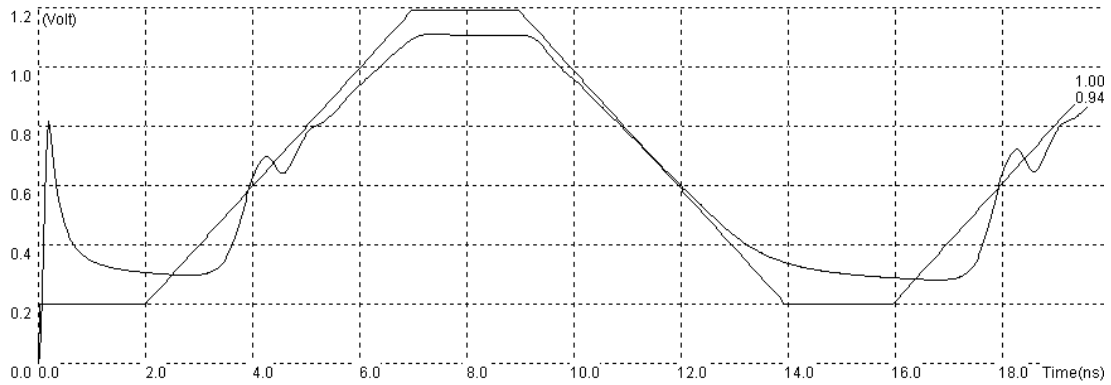


Figure 9-31: Simulation of the push-pull operational amplifier (AmpliDiffPushPull.MSK)

The transient simulation (Figure 9-31) shows an interesting phenomenon called ringing. The oscillation appearing at time 4.0ns is typical for a feedback circuit with a large loop delay and a very powerful output stage. Although an extra 10fF have been added to load the output artificially, its voltage is strongly driven by the powerful devices PO and NO. The oscillation can be dangerous as it introduces instability. It also signifies that the output stage is too strong compared to its charge.

9.12 Exercises

- The cascode current mirror [Gregorian] has several advantages over the simple current mirror. The output impedance is higher, and the current mirroring capabilities are better in terms of accuracy. The schematic diagram of the cascode current mirror is given in figure 11-87. The disadvantage of the structure is the minimum level of the output voltage which is reduced as compared to the regular current mirror. Design a nMOS or pMOS cascode mirror and compare it to the standard structure.

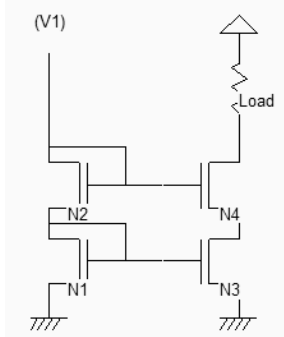


Figure 9-32 : The cascode current mirror

- Let us consider the amplifier of figure 9-33 [Gregorian]. What is the typical value for V_o ? What is the impact of a change in the voltage V_o ? If the width of P2 W_{P2} is $10 \times W_{P1}$, and P3 is $50 \times W_{P1}$, what is the value of the current flowing through P2 and P3, compared to I_1 crossing P1? Where are the V- and V+ inputs located? What is the role of P4 and P5?
- What is the equivalent function for N1 and N2? Locate the two stage output driver. Design the operational amplifier and extract the gain.

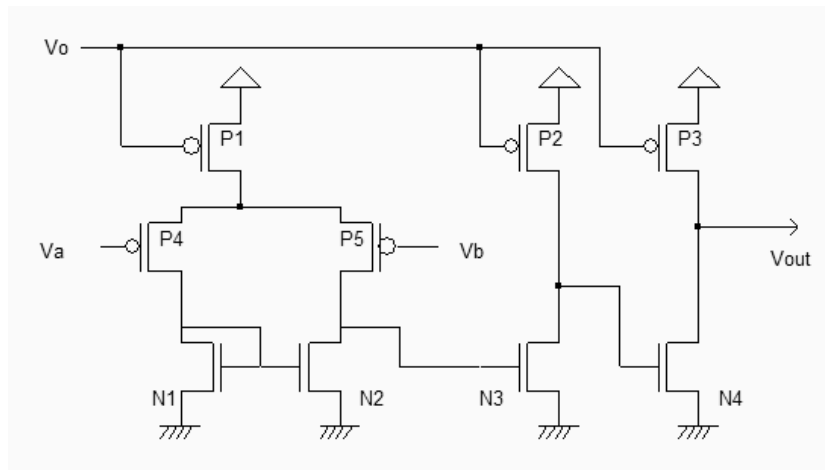


Figure 9-33 : An operational amplifier

10 Radio Frequency Circuits

10.1 On-Chip Inductors

Inductors are commonly used for filtering, amplifying, or for creating resonant circuits used in radio-frequency applications. The inductance symbol in DSCH and Microwind is as follows (Figure 10-1).

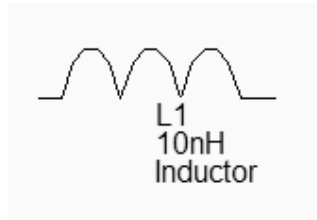


Figure 10-1. The inductance symbol

The inductance formula used in Microwind (Equation 10-1) is one of the most widely known approximation, proposed by [Wheeler], which is said to be still accurate for the evaluation of the on-chip inductor. With 5 turns, a conductor width of 20 μm , a spacing of 5 μm and a hollow of 100 μm , we get $L=11.6\text{nH}$.

$$L = 37.5\mu_0 \cdot \frac{n^2 \cdot a^2}{(22 \cdot r - 14 \cdot a)} \quad (\text{Equ. 10-1})$$

with

$$r = n \cdot (w + s)$$

$$\mu_0 = 4\pi \cdot 10^{-7}$$

n =number of turns

w = conductor width (m)

s =conductor spacing (m)

r =radius of the the coil (m)

a =square spiral's mean radius (m)

The quality factor Q is a very important metric to quantify the resonance effect. A high quality factor Q means low parasitic effects compared to the inductance. The formulation of the quality factor is not as easy as it could appear. An extensive discussion about the formulation of Q depending on the coil model is given in [Lee]. We consider the coil as a serial inductor $L1$, a parasitic serial resistor $R1$, and two parasitic capacitors $C1$ and $C2$ to the ground, as shown in figure 10-2. Consequently, the Q factor is approximately given by equation 10-2.

$$Q = \frac{\sqrt{\frac{L1}{C1 + C2}}}{R1} \quad (\text{Equ. 10-2})$$

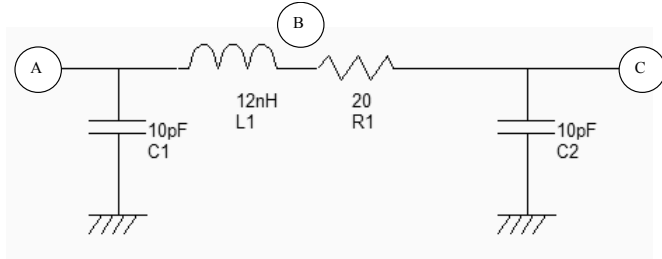


Figure 10-2. The equivalent model of the 12nH default coil and the approximation of the quality factor Q

10.2 Inductor Design in Microwind

We investigate here the design of a rectangular on-chip inductor, the layout options and the consequences on the inductor quality factor. The inductor can be generated automatically by Microwind using the command **Edit → Generate → Inductor** . The inductance value appears at the bottom of the window, as well as the parasitic resistance and the resulting quality factor Q.

Using the default parameters, the coil inductance approaches 12nH, with a quality factor of 1.15. The corresponding layout is shown in figure 10-3. Notice the virtual inductance (L1) and resistance (R1) symbols placed in the layout. The serial inductor is placed between A and B and a serial resistance between B and C. If these symbols were omitted, the whole inductor would be considered as a single electrical node.

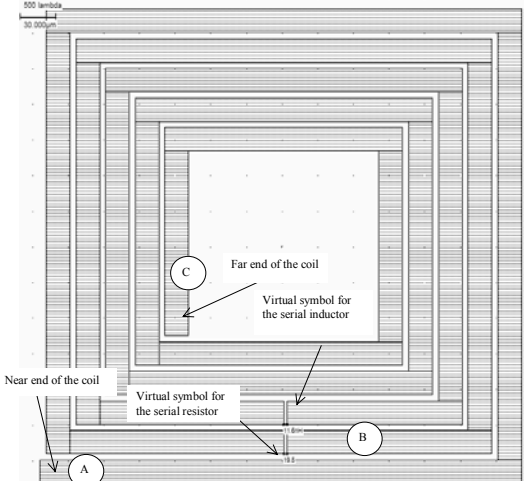


Figure 10-3. The inductor generated by default (inductor12nH.MSK)

A high quality factor Q is attractive because it permits high voltage gain, and high selectivity in frequency domain. A significant improvement in Q consists in using metal layers in parallel (Figure 10-4). The selection of metal2, metal3, up to metal6 reduces the parasitic resistance of RI by a significant factor. The result is a quality factor near 13, for a 3nH inductor.

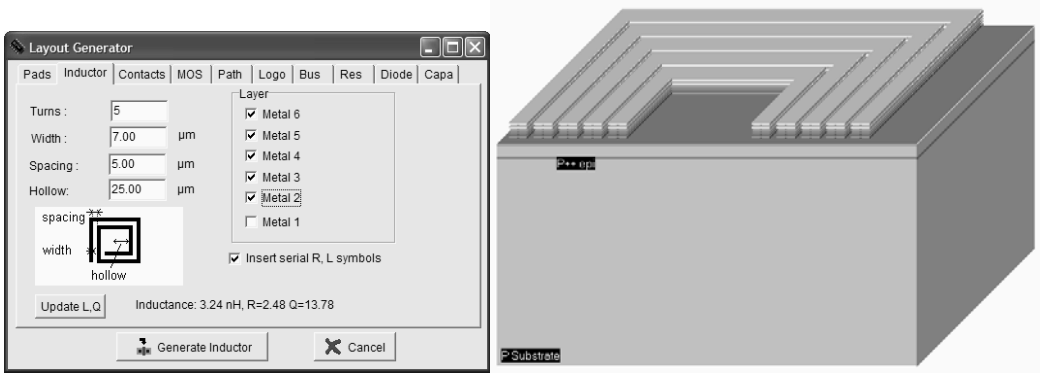


Figure 10-4: A 3D view of a high Q inductor using metal layers in parallel (Inductor3nHighQ.MSK)

Resonance

The coil can be considered as a RLC resonant circuit. At very low frequencies, the inductor is a short circuit, and the capacitor is an open circuit (Figure 10-5 left). This means that the voltage at node C is almost equal to A, if no load is connected to node C, as almost no current flows through RI. At very high frequencies, the inductor is an open circuit, the capacitor a short circuit (Figure 10-5 right). Consequently, the link between C and A tends towards an open circuit.

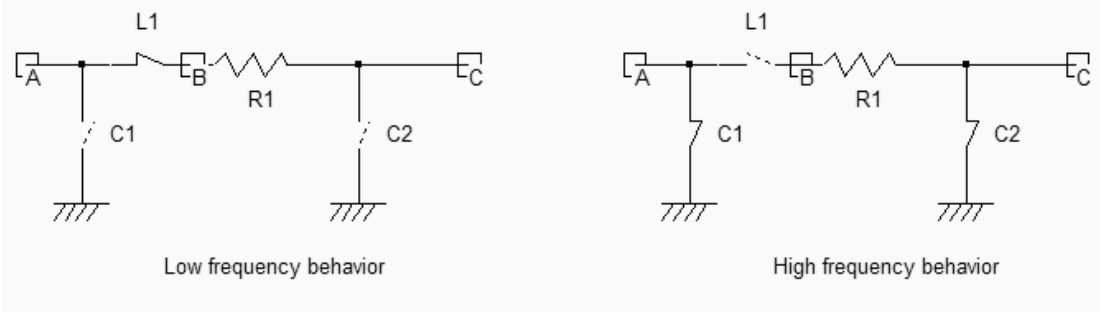


Figure 10-5. The behavior of a RLC circuit at low and high frequencies (Inductor.SCH)

At a very specific frequency the LC circuit features a resonance effect. The theoretical formulation of this frequency is given by equation 10-4.

$$f_r = \frac{1}{2\pi\sqrt{L1(C1 + C2)}} \quad (\text{Equ. 10-4})$$

We may see the resonance effect of the coil and an illustration of the quality factor using the following procedure. The node *A* is controlled by a sinusoidal waveform with increased frequency (Also called “chirp” signal). We specify a very small amplitude (0.1V), and a zero offset.

The resonance can be observed when the voltage at nodes *B* and *C* is higher than the input voltage *A*. The ratio between *B* and *A* is equal to the quality factor *Q*.

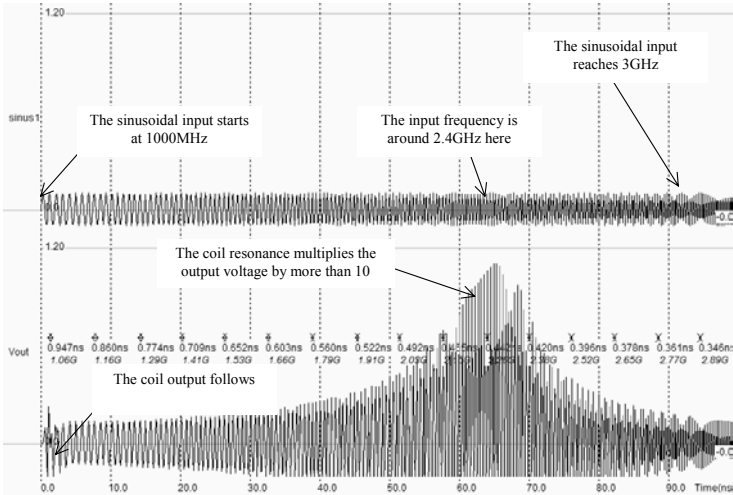


Figure 10-6. The behavior of a RLC circuit near resonance (Inductor3nHighQ.MSK)

10.3 Power Amplifier

The power amplifier is part of the radio-frequency transmitter, and is used to amplify the signal being transmitted to an antenna so that it can be received at the desired distance. Most CMOS power amplifiers are based on a single MOS device, loaded with a “Radio-Frequency Choke” inductor L_{RFC} , as shown in figure 10-7.

The inductor serves as a load for the MOS device (At a given frequency f , the inductor is equivalent to a resistance $L \cdot 2\pi \cdot f$), with two significant advantages as compared to the resistor: the inductor do not consume DC power, and the combination of the inductor and the load capacitor C_L creates a resonance. The power is delivered to the load RL , which is often fixed to 50Ohm. This load is for example the antenna monopole, which can be assimilated to a radiation resistance, as described in the previous section. The resonance effect is obtained between L_{RFC} and C_L .

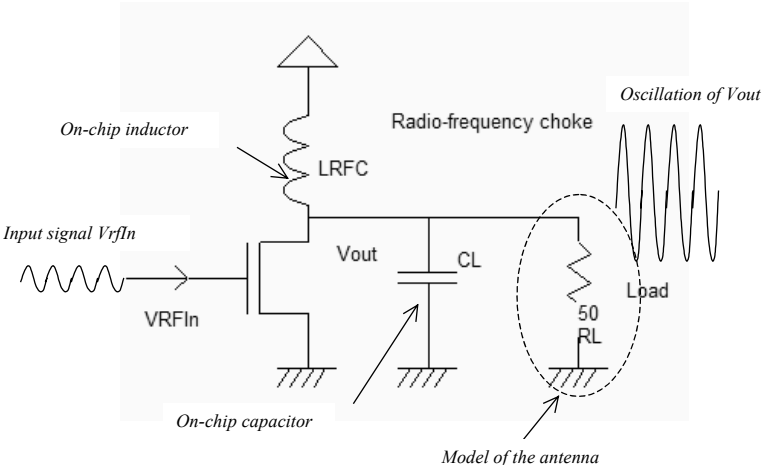


Figure 10-7: The basic diagram of a power amplifier (PowerAmp.SCH)

An example of powerful MOS device is shown in figure 10-8. The maximum current is close to 40mA. A convenient way to generate the polarization ring consists in using the Path generator command, and selection the option **Metal and p-diffusion**. Then draw the location for the polarization contacts in order to complete the ring.

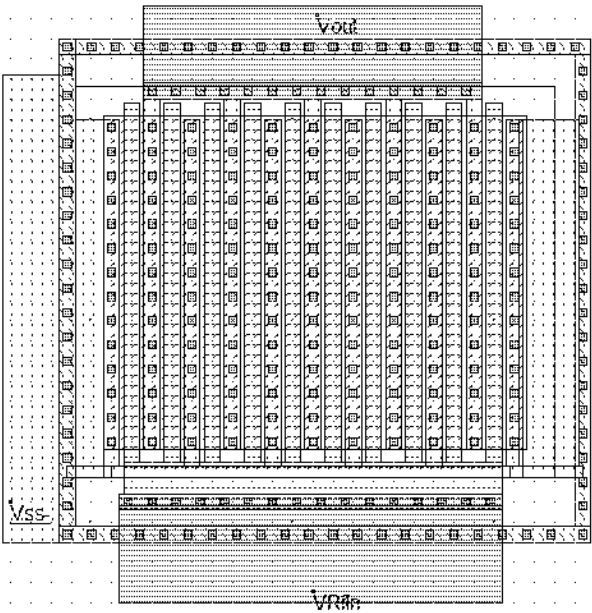


Figure 10-8: The layout of the power MOS also includes a polarization ring, and the contacts to metal2 connections to VRF_in and VOut (PowerAmplifier.MSK)

The distinction between class A,B,AB, etc.. amplifiers is mainly given with the polarization of the input signal. A Class A amplifier is polarized in such a way that the transistor is always conducting. The MOS device operates almost linearly. An example of power amplifier polarized in class A is shown in figure 10-9. The power MOS is designed very big to improve the power efficiency.

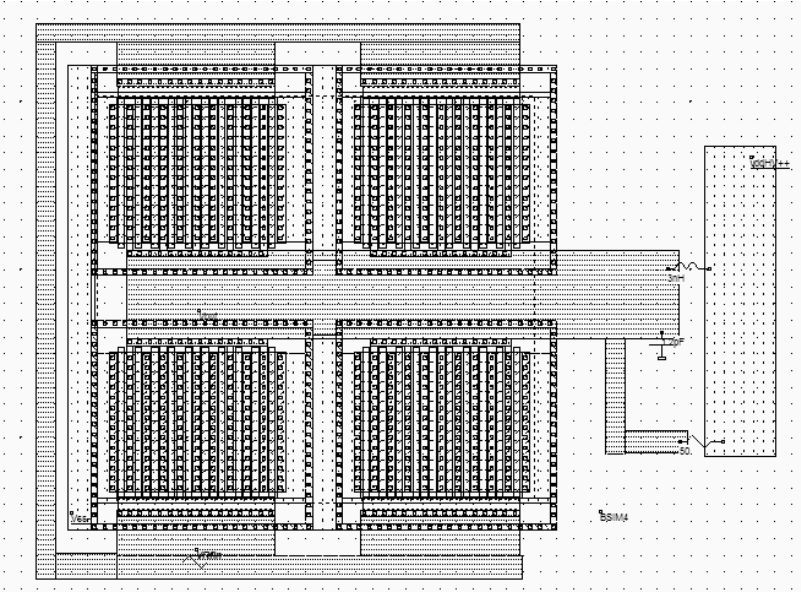


Figure 10-9. The class A amplifier design with a very large MOS device (PowerAmplifierClassA.MSK)

The sinusoidal input offset is 1.3V, the amplitude is 0.4V. The power MOS functional point trajectory is plotted in figure 10-10, and is obtained using the command **Simulate on Layout**. We see the evolution of the functional point with the voltage parameters: as V_{gs} varies from 0.9V to 1.7V, I_{ds} fluctuates between 20mA to 70mA. The MOS device is always conducting, which corresponds to class A amplifiers.

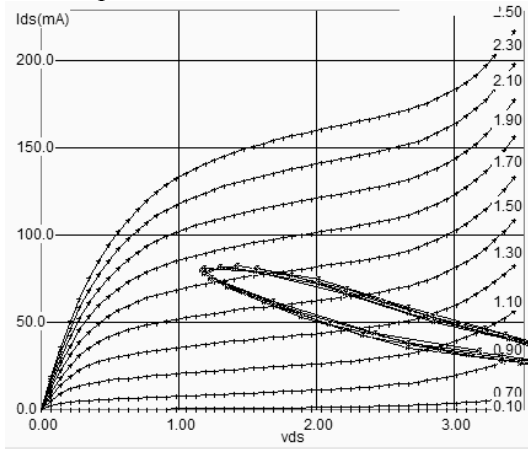


Figure 10-10. The class A amplifier has a sinusoidal input (PowerAmplifierClassA.MSK)

The main drawback of Class A amplifiers is the high bias current, leading to a poor efficiency. In other words, most of the power delivered by the supply is dissipated inefficiently. The power efficiency is around 11% in this layout. The main advantage is the amplifier linearity, which is illustrated by a quasi-sinusoidal output V_{out} , as seen in figure 10-11.

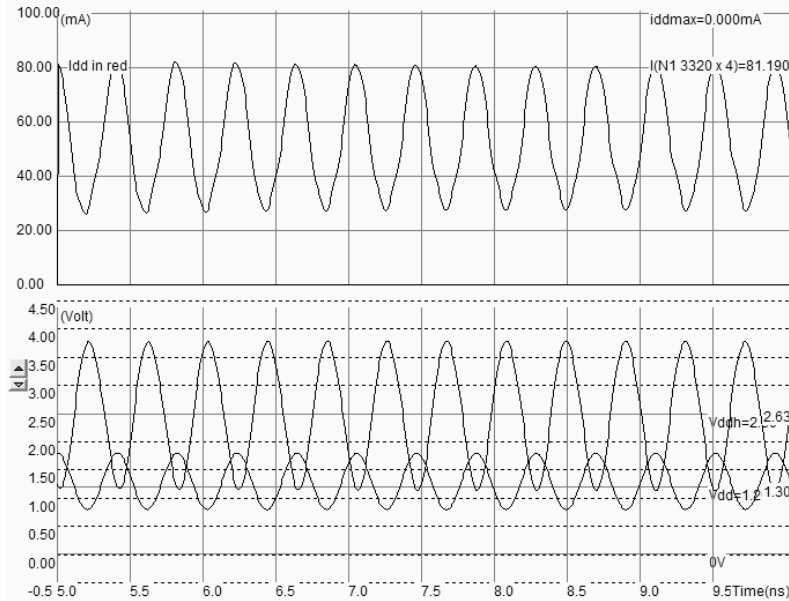


Figure 10-11. The Class A Amplifier simulation (PowerAmplifierClassA.MSK)

10.4 Oscillator

The role of oscillators is to create a periodic logic or analog signal with a stable and predictable frequency. Oscillators are required to generate the carrying signals for radio frequency transmission, but also for the main clocks of processors. The ring oscillator is a very simple oscillator circuit, based on the switching delay existing between the input and output of an inverter. If we connect an odd chain of inverters, we obtain a natural oscillation, with a period which corresponds roughly to the number of elementary delays per gate. The fastest oscillation is obtained with 3 inverters (One single inverter connected to itself does not oscillate). The usual implementation consists in a series of five up to one hundred chained inverters (Figure 10-12).

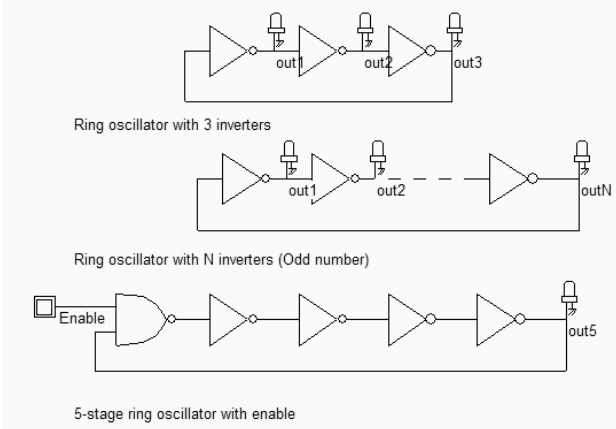


Figure 10-12: A ring oscillator is based on an odd number of inverters (Inv3.SCH)

The main problem of this type of oscillators is the very strong dependence of the output frequency on virtually all process parameters and operating conditions. As an example, the power supply voltage VDD has a very significant importance on the oscillating frequency. This dependency can be analyzed using the **parametric analysis** in the **Analysis** menu. Several simulations are performed with VDD varying from 0.8V to 1.4V with a 50mV step. In figure 10-13, we clearly observe a very important increase of the output frequency with VDD (Almost a factor of 2 between the lower and upper bounds). This means that any supply fluctuation has a significant impact on the oscillator frequency.

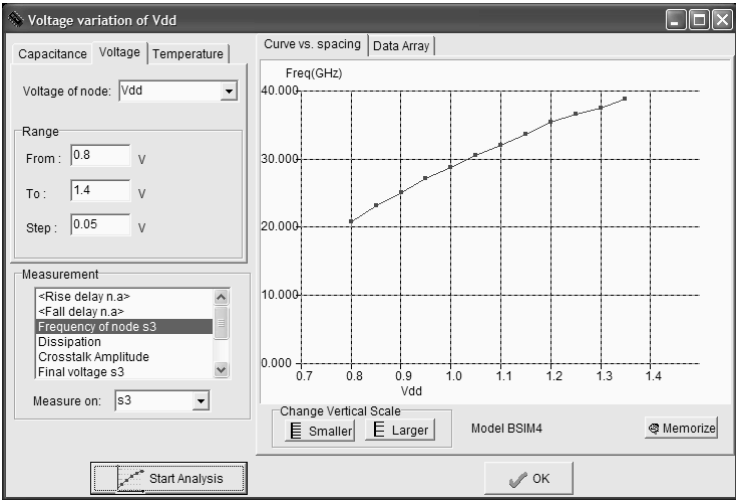


Figure 10-13: The oscillator frequency variation with the power supply (Inv3.MSK)

The LC oscillator proposed below is not based on the logic delay, as with the ring oscillator, but on the resonant effect of a passive inductor and capacitor circuit. In the schematic diagram of figure 10-14, the inductor L1 resonates with the capacitor C1 connected to S1 combined with C2 connected to S2.

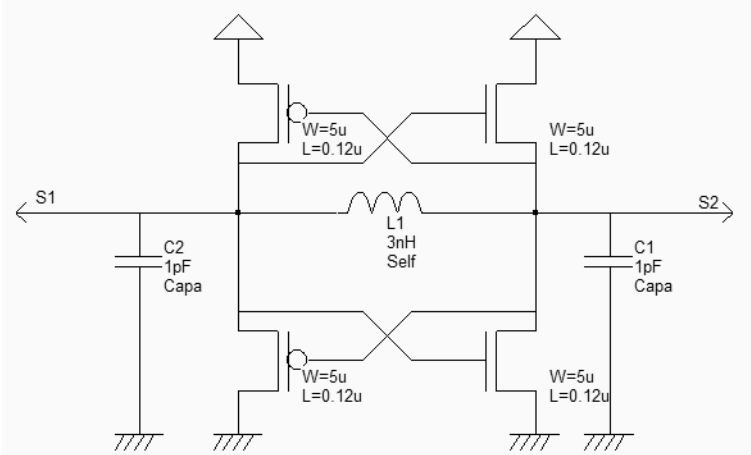


Figure 10-14: A differential oscillator using an inductor and companion capacitor (OscillatorDiff.SCH)

The layout implementation is performed using a 3nH virtual inductor and two 1pF capacitor. The large width of active devices to ensure a sufficient current to charge and discharge the huge capacitance of the output node at the desired frequency. Using virtual capacitors instead of on-chip physical coils is recommended during the development phase. It allows an easy tuning of the inductor and capacitor elements in order to achieve the correct behavior. Once the circuit has been validated, the L and C symbols can be replaced by physical components. The time-domain simulation (Figure 10-15) shows a warm-up period around 1ns where the DC supply rises to its nominal value, and where the oscillator effect reaches a permanent state after some nano-seconds.

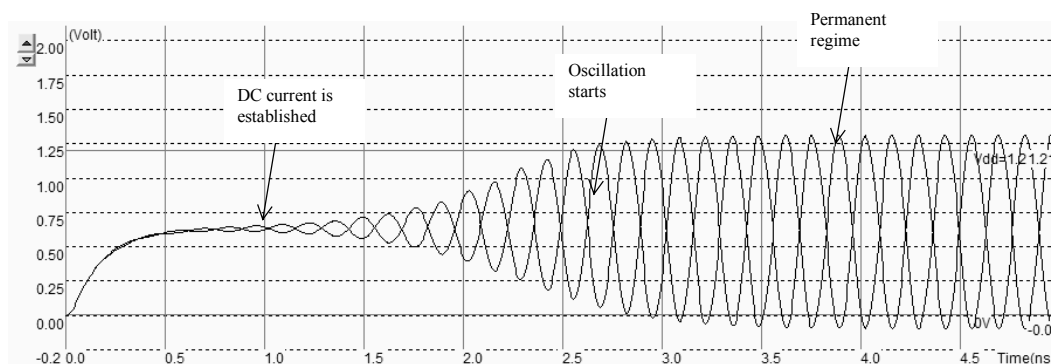


Figure 10-15: Simulation of the differential oscillator (OscillatorDiff.MSK)

The Fourier transform of the output $s1$ reveals a main sinusoidal contribution at $f_0=3.725\text{GHz}$ as expected, and some harmonics at $2xf_0$ and $3xf_0$ (Figure 10-16). The remarkable property of this circuit is its ability to remain in a stable frequency even if we change the supply voltage or the temperature, which features a significant improvement as compared to the ring oscillator. Furthermore, the variations of the MOS model parameters have almost no effect on the frequency.

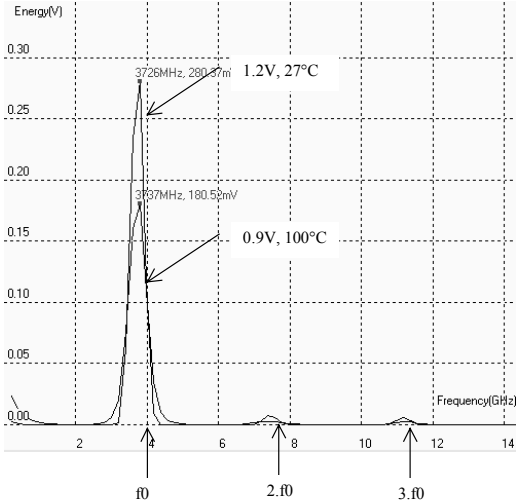


Figure 10-16: The frequency spectrum of the oscillator (OscillatorDiff.MSK)

10.5 Voltage Controlled Oscillator

The voltage controlled oscillator (VCO) generates a clock with a controllable frequency. The VCO is commonly used for clock generation in phase lock loop circuits, as described later in this chapter. The clock may vary typically by +/-50% of its central frequency. A current-starved voltage controlled oscillator is shown in figure 10-17 [Weste] . The current-starved inverter chain uses a voltage control $V_{control}$ to modify the current that flows in the $N1,P1$ branch. The current through $N1$ is mirrored by $N2,N3$ and $N4$. The same current flows in $P1$. The current through $P1$ is mirrored by $P2, P2$, and $P4$. Consequently, the change in $V_{control}$ induces a global change in the inverter currents, and acts directly on the delay. Usually more than 3 inverters are in the loop. A higher odd number of stages is commonly implemented, depending on the target oscillating frequency and consumption constraints.

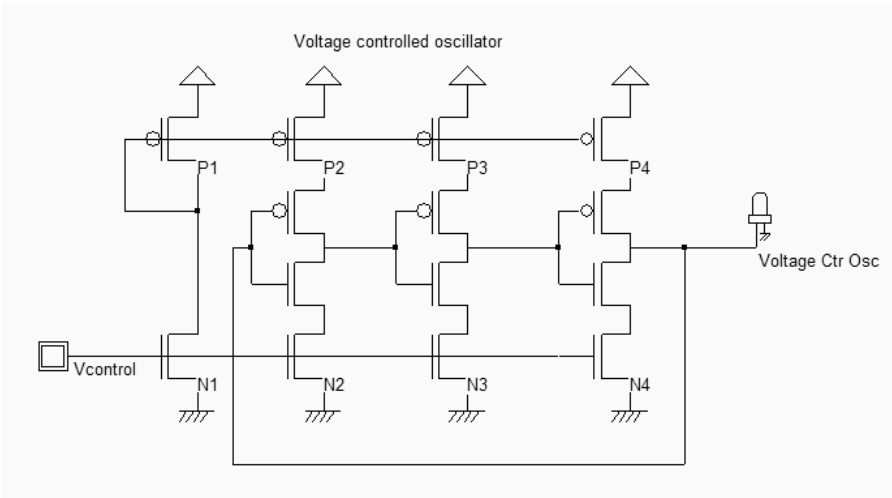


Figure 10-17: Schematic diagram of a voltage controlled oscillator (VCOMos.SCH)

The implementation of the current-starved VCO for a 5-inverter chain is given in figure 10-18. The current mirror is situated on the left. Five inverters have been designed to create the basic ring oscillator. Then a buffer inverter is situated on the right side of the layout.

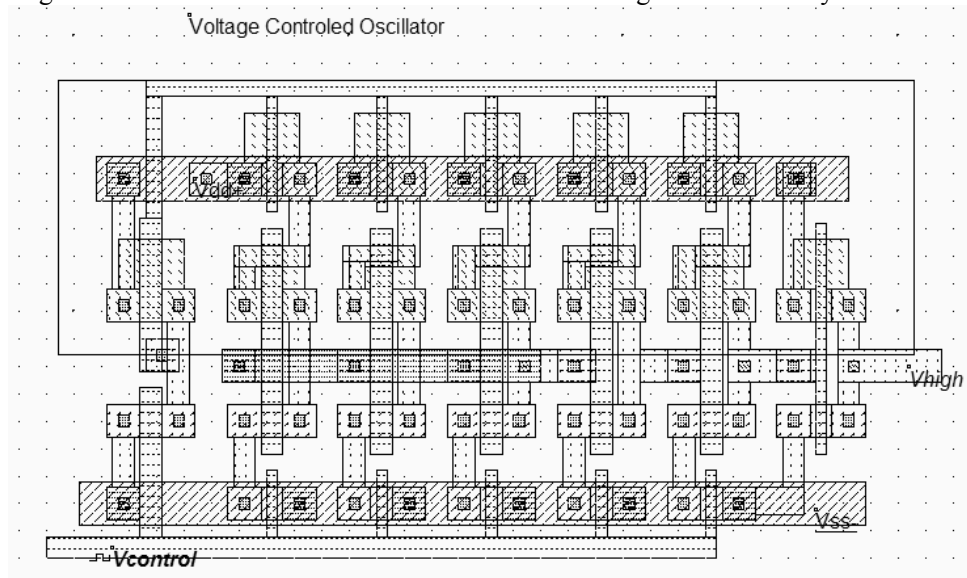


Figure 10-18. A VCO implementation using 5 chained inverters (VCO.MSK)

There exist a convenient simulation mode to display the frequency variations versus time together with the voltage variations (**Frequency vs. Time** in the simulation menu). The frequency is evaluated on the selected node, which is the output node V_{high} in this case. No oscillation is observed for an input voltage $V_{control}$ lower than 0.5V. Then the VCO starts to oscillate, but the frequency variation is clearly not linear. The maximum frequency is obtained for the highest value of $V_{control}$, around 8.4GHz. By increasing the number of inverters and altering the size of the MOS current sources, we may modify the oscillating frequency easily.

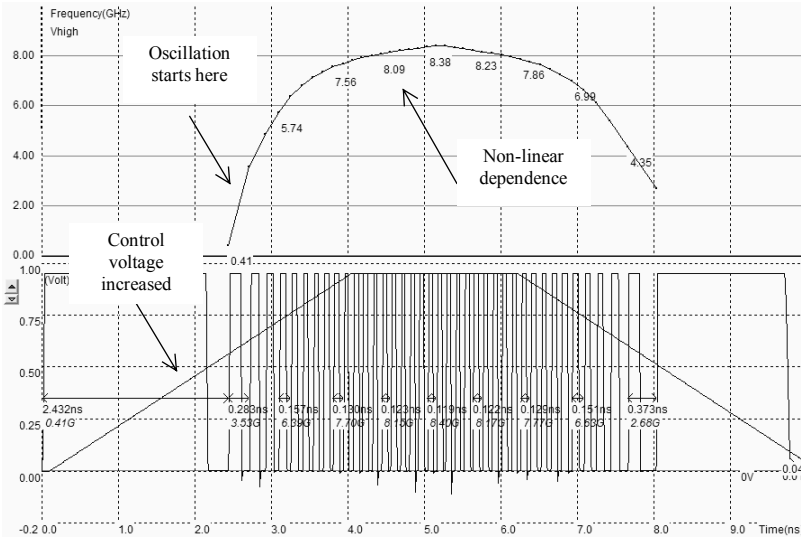


Figure 10-19. The frequency variations versus the control voltage show a non-linear dependence (VCO.MSK)

10.6 Phase-lock-loop

The phase-lock-loop (PLL) is commonly used in microprocessors to generate a clock at high frequency (F_{out}=2GHz for example) from an external clock at low frequency (F_{ref} = 100MHz for example). Clock signals in the range of 1GHz are very uneasy to import from outside the integrated circuit because of low pass effect of the printed circuit board tracks and package leads. The PLL is also used as a clock recovery circuit to generate a clock signal from a series of bits transmitted in serial without synchronization clock (Figure 10-20). The PLL may also be found in frequency demodulation circuits, to transform a frequency varying waveform into a voltage.

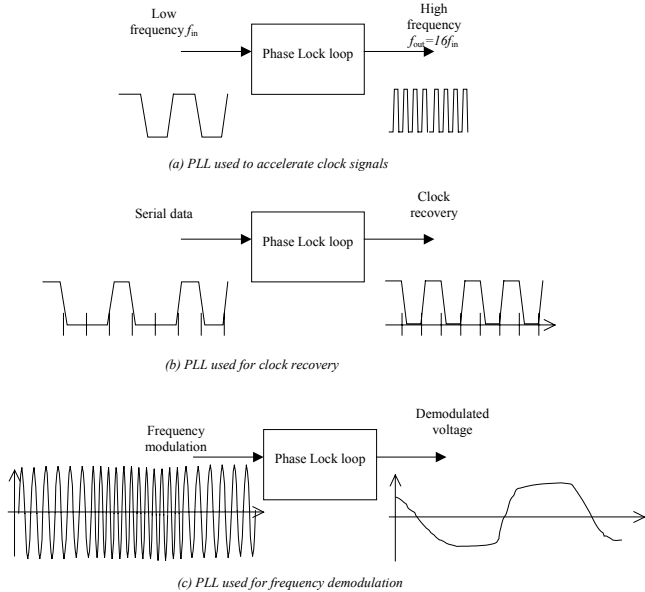


Figure 10-20. Principles of phase lock loops

The PLL uses a high frequency oscillator with varying speed, a counter, a phase detector and a filter (figure 10-21). The PLL includes a feedback loop which lines up the output clock $ClkOut$ with the input clock $ClkIn$ through a phase locking stabilization process. When locked, the high input frequency f_{out} is exactly $Nx f_{in}$. A variation of the input frequency f_{in} is transformed by the phase detector into a pulse signal which is converted in turn into variation of the analog signal Vc . This signal changes the VCO frequency which is divided by the counter and changes $clkDiv$ according to f_{in} .

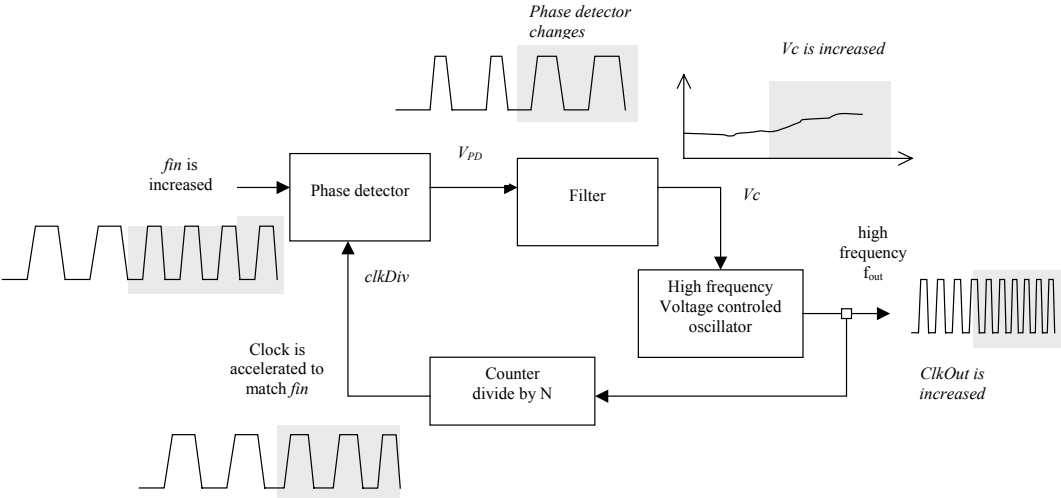


Figure 10-21. Principles of phase lock loops

10.6.1 Phase Detector

The most simple phase detector is the XOR gate. The XOR gate output produces a regular square oscillation V_{PD} when the input $clkIn$ and the signal $divIn$ have one quarter of period shift (or 90° or $\pi/2$). For other angles, the output is no more regular. In figure 10-22, two clocks with slightly different periods are used in Dsch2 to illustrate the phase detection.

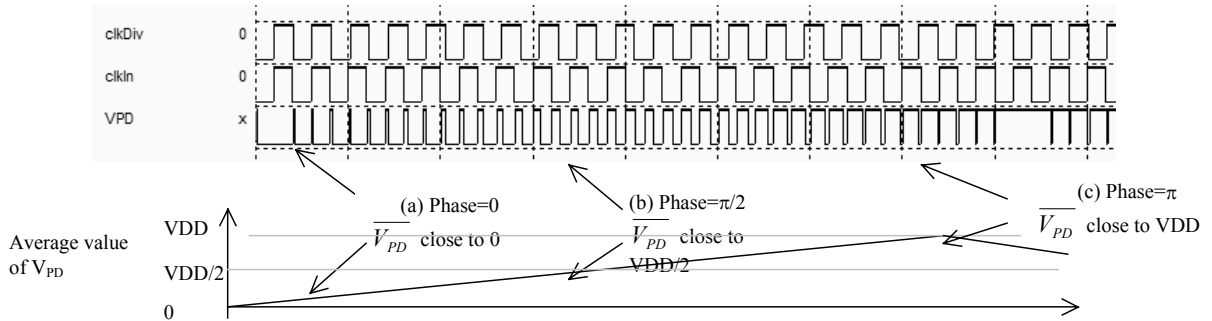


Figure 10-22. The XOR phase detector at work (PhaseDetectXor.SCH)

At initialization, (Figure 12-52) the average value of the XOR output $\overline{V_{PD}}$ is close to 0. When the phase between $clkDiv$ and $clkIn$ is around $\pi/2$, $\overline{V_{PD}}$ is $VDD/2$. Then it increases up to VDD . The gain of the phase detector is the ratio between $\overline{V_{PD}}$ and $\Delta\phi$ (figure 10-23). When the phase difference is larger than π , the slope sign is negative until 2π . When locked, the phase difference should be close to $\pi/2$.

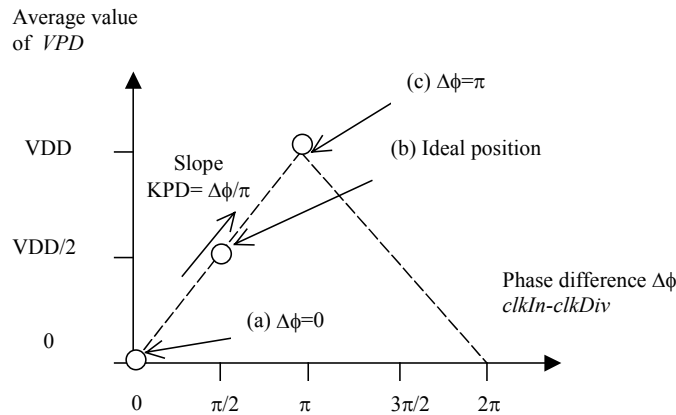


Figure 10-23. The XOR phase detector at work (PhaseDetectXor.SCH)

10.6.2 Filter

The filter is used to transform the instantaneous phase difference V_{PD} into an analog voltage V_c which is equivalent to the average voltage $\overline{V_{PD}}$. The rapid variations of the phase detector output are converted into a slow varying signal V_c which will later control the voltage controlled oscillator. Without filtering, the VCO control would have too rapid changes which would lead to instability. The filter may simply be a large capacitor C , charged and discharged through the R_{on} resistance of the switch. The $R_{on} \cdot C$ delay creates a low-pass filter. Figure 10-24 shows a XOR gate with the output charged with a large poly/poly2 capacitor and a serial resistance to create the desired analog voltage control V_c .

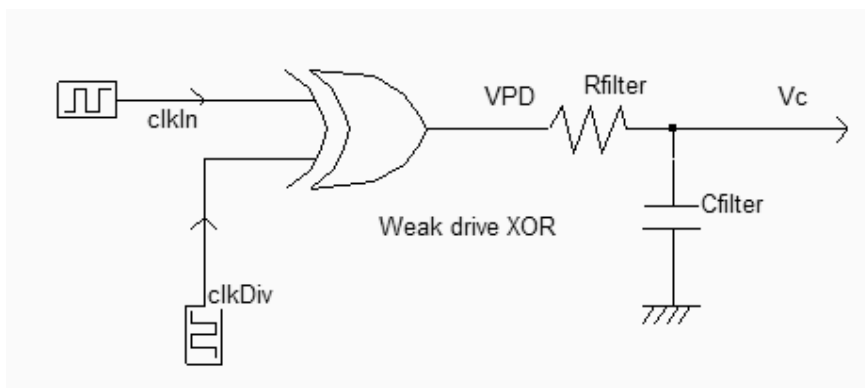


Figure 10-24. Large load capacitance and weak XOR output stage to act as a filter (phaseDetectAndFilter.SCH)

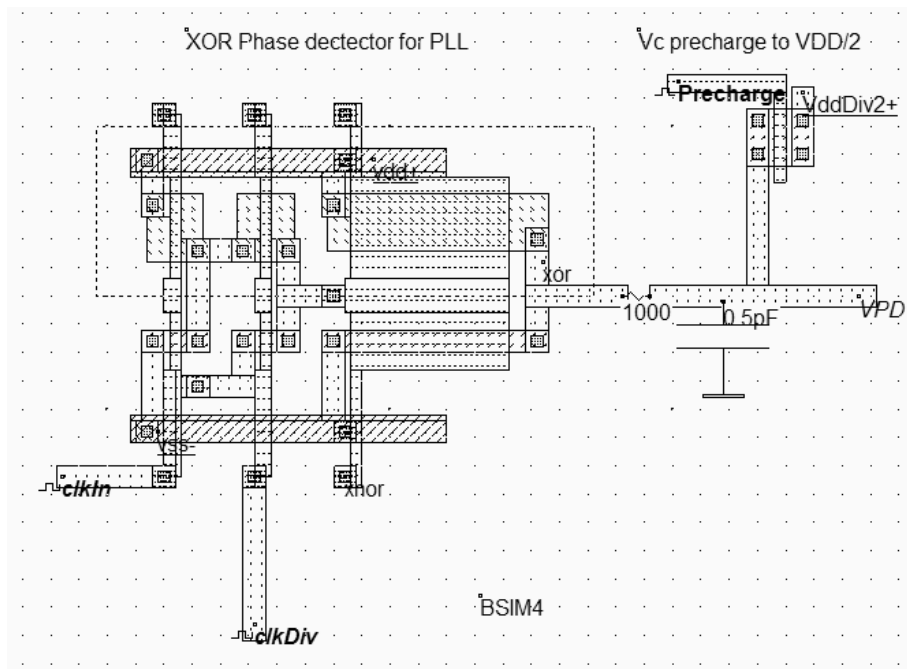


Fig. 10-25. The phase detector and the filter (*phaseDetect.MSK*)

In the figure above, the filtered version of the XOR gate output VPD is shown. The signal VPD should be around $VDD/2$ when the phase difference is $\pi/2$ or $-\pi/2$. The duty cycle of the phase detector output should be as close as possible to 50%, so that Vc is very close to $VDD/2$ when the inputs are in phase. If this is not the case, the PLL would have problems locking or would not produce a stable output clock.

10.6.3 Voltage controlled oscillator for PLL

Important characteristics of the PLL can be listed:

- The oscillating frequency should be restricted to the required bandwidth. For example, in European mobile phone applications, the VCO frequency should be varying between $f_{low}=1700$ and $f_{high}=1800$ MHz (Figure 10-26)
- Due to process variations, the VCO frequency range should be extended to f_{min} , f_{max} , typically 10% higher and lower than the request range
- When the control voltage Vc is equal to $VDD/2$, the VCO clock should be centered in the middle of the desired frequency range.
- The duty cycle of the VCO clock output should be as close as possible to 50% [Baker]. If this is not the case, the PLL would have problems locking or would not produce a stable output clock.

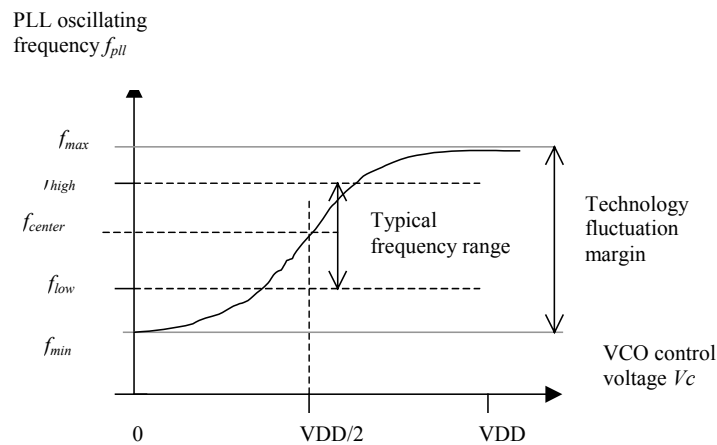


Figure 10-26: Requirements for the VCO used in the PLL

10.6.4 Complete Phase Lock Loop

The implementation of the PLL shown in figure 10-27 includes a resistor R_{filter} (1000Ohm) and R_{vdd2} (5000 Ohm). The capacitor C_{filter} is a virtual component fixed to 0.3pF. These resistance and capacitance are easy to integrate on-chip.

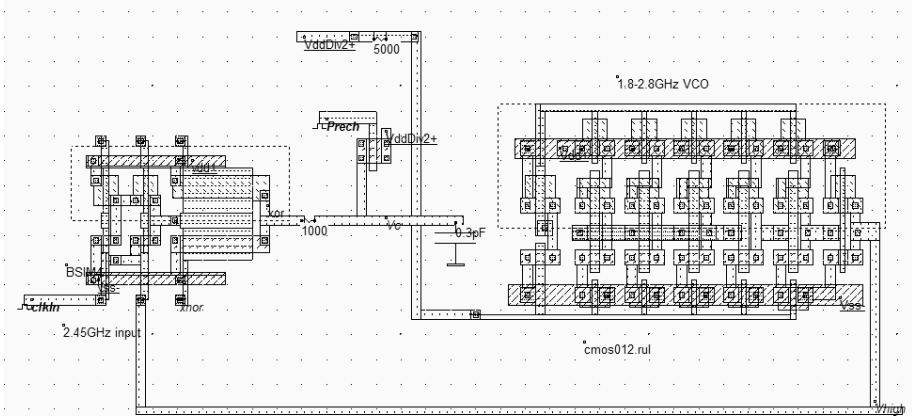


Figure 10-27: Connecting the current-starved VCO to the phase detector (VCOpll.SCH)

The input frequency is fixed to 2.44GHz. During the initialization phase (Simulation of figure 10-28), the precharge is active, which pushes rapidly the voltage of V_c around $V_{DD}/2$. The VCO oscillation is started and the phase detector starts operating erratically. The output $Xnor$ is an interesting indication of what happens inside the phase detector. We see that the phase difference is very important during the first 10 nanoseconds. Then, the VCO output starts to converge to the reference clock. In terms of voltage control, V_c tends to oscillate and then converge to a stable state where the PLL is locked and stable. The output is equal to the input, and the phase difference is equal to one fourth of the period ($\pi/2$) according to the phase detector principles.

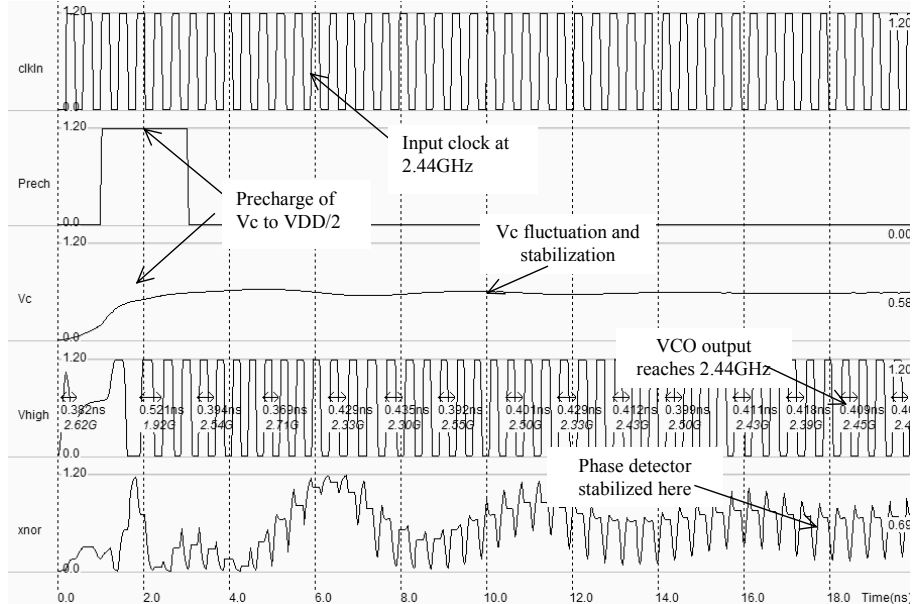


Figure 10-28: Simulation of the PLL showing the locking time (VCOPLL.SCH)

10.7 Gilbert Mixer

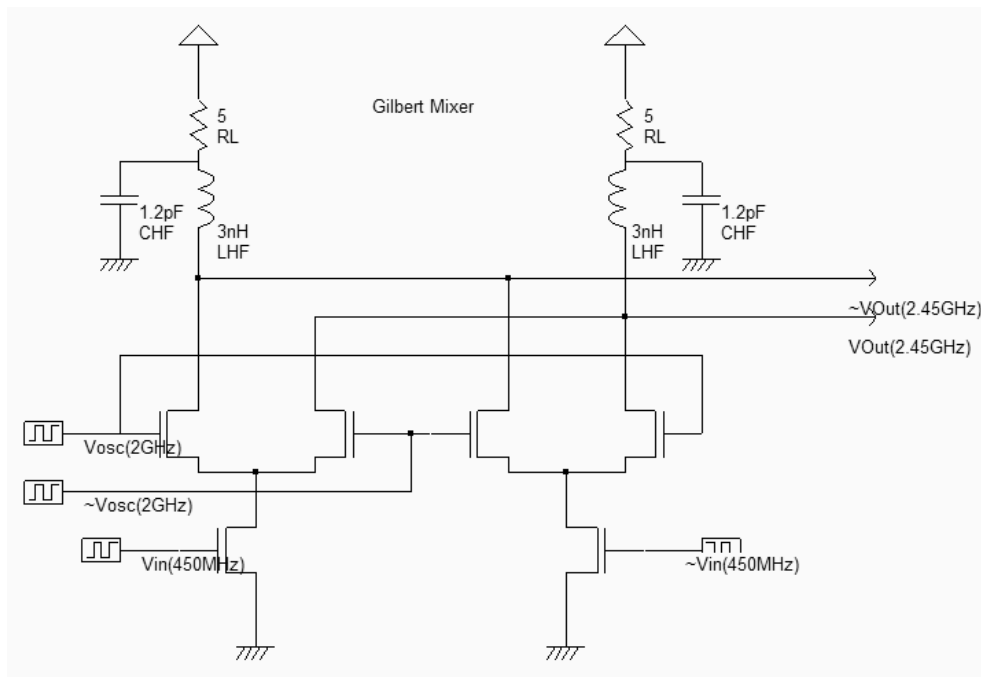


Figure 10-29: The Gilbert mixer (MixerGilbert.SCH)

The Gilbert mixer is used to shift the frequency of an input signal V_{in} to a high frequency. In the implementation shown in figure 10-29, the result is an output frequency at $V_{osc} + V_{in}$. The process for rising the frequency of the signal is called up-conversion. The Gilbert cell consists of only six transistors, and performs a high quality multiplication of the sinusoidal waves [Lee]. The schematic diagram shown in figure 10-30 uses the tuned inductor as loads, so that V_{out} and $\sim V_{out}$ oscillate around the supply VDD.

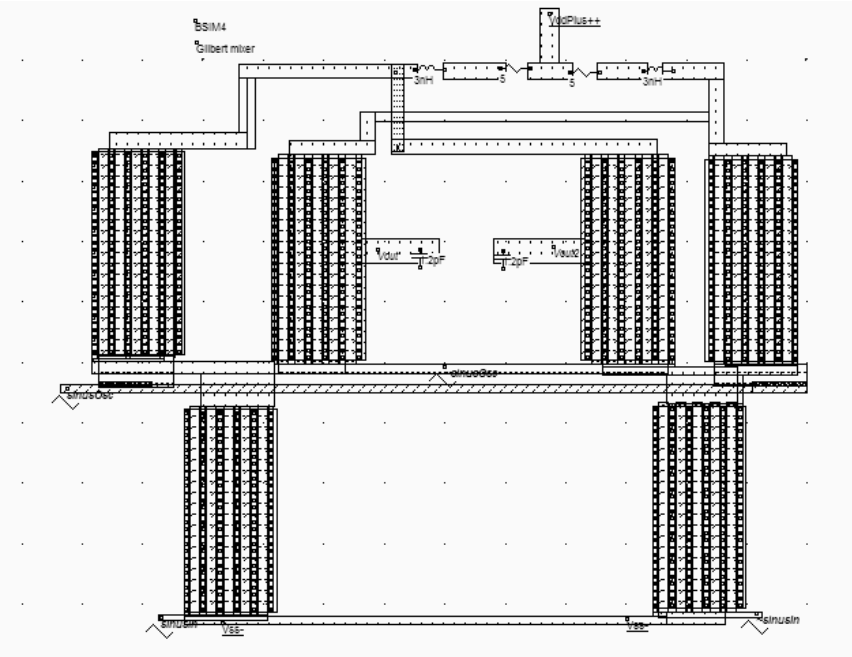


Figure 10-30: The Gilbert mixer implementation with virtual R,L and C (MixerGilbert.MSK)

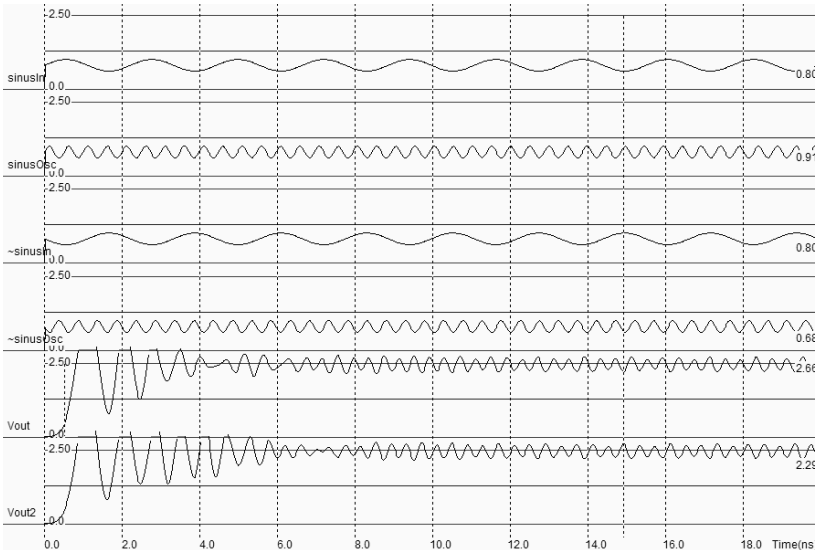


Figure 10-31: Time-domain simulation of the Gilbert mixer (MixerGilbert.MSK)

The implementation shown in figure 10-31 makes again an extensive use of virtual R,L and C elements. The 3nH inductor is in series with a parasitic 5 ohm resistance, on both branches. The time domain simulation reveals a transient period from 0.0 to 8ns during which the inductor and capacitor warm-up. This initialization period is not of key interest. The most interesting part starts from 8ns, when the output V_{out} and V_{out2} are stabilized and oscillate in opposite phase around 2.5V.

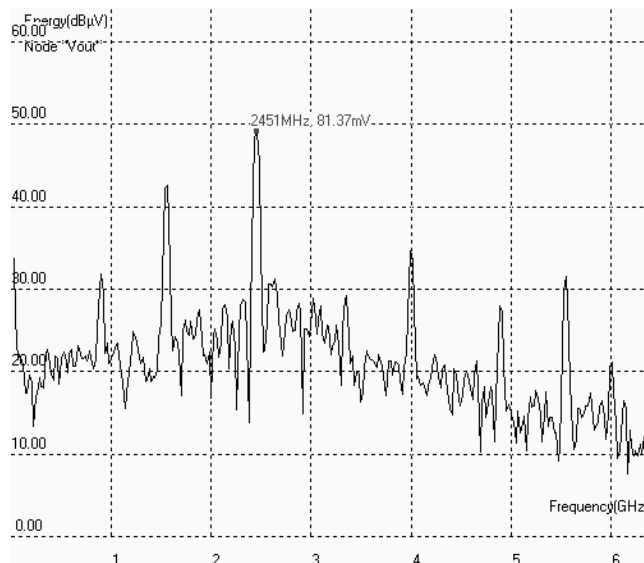


Figure 10-32: Fourier transform of the Gilbert mixer output (MixerGilbert.MSK)

The Fourier transforms of nodes V_{out} and V_{out2} are almost identical. We present the Fourier transform in logarithm scale to reveal the small harmonic contributions. As expected, the 2GHz f_{osc} signal and 450MHz f_{in} signals have disappeared, thanks to the cancellation of contributions. The two major contributors are $f_{osc}+f_{in}$ and $f_{osc}-f_{in}$.

10.8 Exercises

- Design a 10mW power amplifier operating near 1.9GHz (UMTS frequency range). Add a second power MOS device to be able to tune the output power from 10mW to 30mW, using logic controls.
- Optimize the power amplifier shown in this chapter for a maximum power efficiency delivered to a 30Ω load, as the radiating resistance is often closer to that value than the standard 50Ω.
- Design a LC oscillator targeted to 5.4GHz, corresponding to the frequency used in wireless area network protocols such as IEEE 802.11a.
- Redesign the high-performance VCO to oscillate around 5.4GHz, with a span of 0.5GHz (Corresponds to IEEE 802.11a).

11 Converters

In this chapter, we shall discuss the basic principles of data converters and give an overview of their architecture. The data converter design and implementation is also discussed, with emphasis on basic building blocks, the use of comparators, voltage reference, sampling structures, etc... We also discuss the implementation of temperature and light sensors, compatible with the CMOS standard process.

11.1 Introduction

The analog to digital converters (ADC) and digital to analog converters (DAC) are the main links between the analog signals and the digital world of signal processing. The ADC and DAC viewed as black boxes are shown in figure 11-1. On the right side, the ADC takes an analog input signal V_{in} and converts it to a digital output signal A . The digital signal A is a binary coded representation of the analog signal using N bits: $A_{N-1} \dots A_0$. The maximum number of codes for N bits is 2^N . The digital signal is usually treated by a microprocessor unit (MPU) or by a specific digital signal processor (DSP) before being restituted as an output B . Then, the DAC, which has the opposite function compared to the ADC, converts the digital signal to the final analog output signal V_{out} .

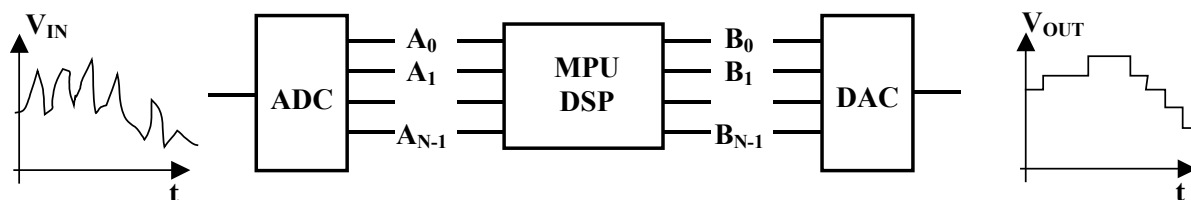


Figure 11-1. Basic principle of N bits analog to digital and digital to analog converters.

11.2 Digital-Analog Converters architectures

Digital-to-analog converters (DAC) traduce a digital number B into an analog signal V_{OUT}^* . The output of the DAC is not as smooth as we could wish, due to a finite number of available analog

levels. A low pass filter eliminates the higher order harmonics caused by the conversion on the signal V_{OUT}^* , and returns an analog signal V_{OUT} .

11.2.1 Resistor string converter

The most basic DAC is based on a resistance ladder. This type of DAC consists of a simple resistor string of 2^N identical resistors, and a binary switch array whose inputs are a binary word. The analog output is the voltage division of the resistors flowing via pass switches. In the example of figure 11-2, the resistance ladder includes 8 identical resistors, which generate 8 reference voltage equally distributed between the ground voltage and V_{dac} .

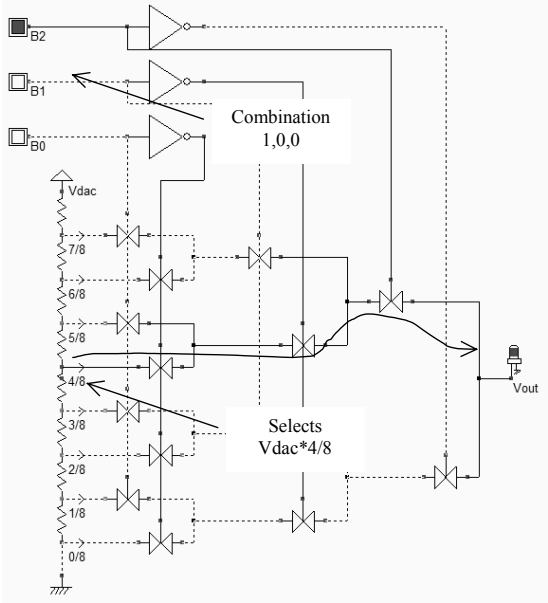


Figure 11-2 Schematic diagram of the digital-analog converter (DAC3bit.SCH)

The digital-analog converter uses the three-bit input (B2,B1,B0) to control the transmission gate network which selects one of the voltage references (A portion of V_{dac}) which is then transferred to the output V_{out} .

B2	B1	B0	Vout*	Analog output Vout* (V) with Vdac=1.2V
0	0	0	0/8 Vdac	0.0
0	0	1	1/8 Vdac	0.15
0	1	0	2/8 Vdac	0.3
0	1	1	3/8 Vdac	0.45
1	0	0	4/8 Vdac	0.6
1	0	1	5/8 Vdac	0.75
1	1	0	6/8 Vdac	0.9
1	1	1	7/8 Vdac	1.05

Figure 11-3 The specifications of a 3-bit digital-to-analog converter

A long path of polysilicon between VDD and VSS may give intermediate voltage references required for the DAC circuit. Unfortunately, the polysilicon has a low resistance due to a surface deposit of metal, called salicidation. In order to increase the sheet resistance value (Around 4 Ω per square), the polysilicon resistor must be surrounded by the specific "Option" layer that may be found in the upper part of the palette of layers. The salicide is removed, and the sheet resistance is increased to 40 Ω per square. We activate the property "**Remove salicide to increase resistance**" of the option layer (Figure 11-4). Consequently, the resistor value is multiplied by 10 and can be used to design an area-efficient resistor network.

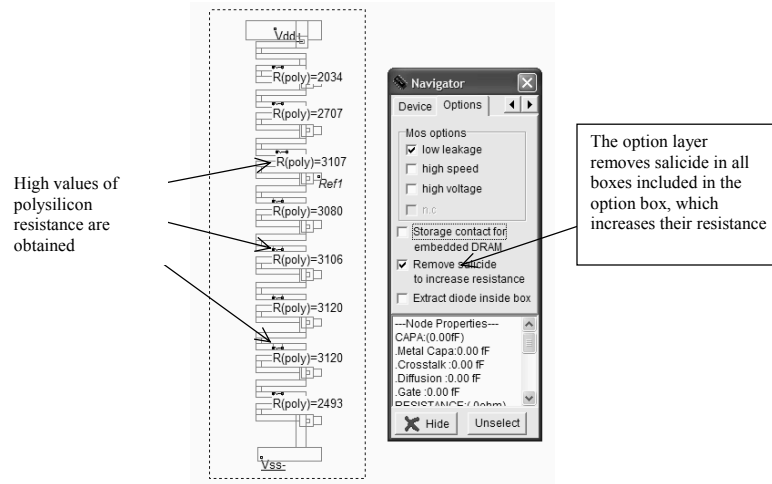


Figure 11-4. The sheet resistance is increased by removing the salicide deposit, thanks to an option layer (ADC.MSK)

To account for the serial resistance distributed along the polysilicon path, a virtual resistance symbol must be added, which will force Microwind to split the ladder into separate electrical nodes, and to extract the corresponding polysilicon resistance. The virtual resistor may be found in the upper part of the palette. Once inserted, the menu of figure 11-5 appears. It is recommended in this case to select the option **Poly resistance**. At extraction, Microwind will evaluate the equivalent resistance on both sides of the virtual symbol and update the resistance automatically according to the design and technological options.

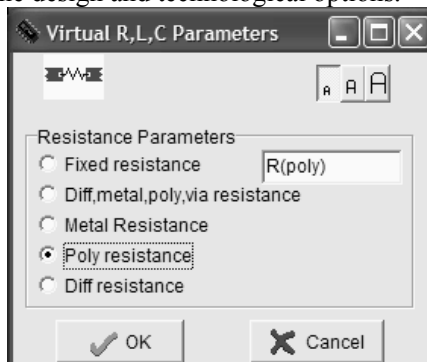


Figure 11-5: Adding a virtual resistor symbol to extract the polysilicon resistance

The resistance symbol is inserted in the layout to indicate to the simulator that an equivalent resistance must be taken into account for the next analog simulation. The layout of the 3-bit digital-to-analog converter is shown in Figure 11-6. The three inverter circuits generate the signals $\sim B2, \sim B1$ and $\sim B0$ from signals $B2, B1$ and $B0$. The transmission gates use minimum MOS device size. The total resistance approaches 24Kohm, which means a stand-by current near $50\mu\text{A}$ on a 1.2V supply power. Lower DC currents may be obtained by increasing the length of the polysilicon path.

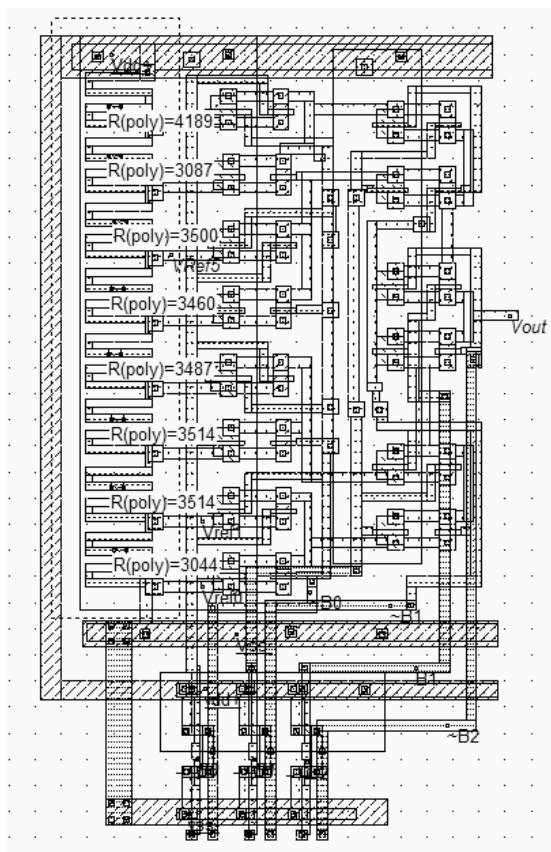


Figure 11-6. Layout of the digital-analog converter (DAC.MSK).

The simulation of the R ladder DAC (Figure 11-7) shows a regular increase of the output voltage V_{out} with the input combinations, from 000 (0V) to 111 (1.2V). Each input change provokes a capacitance network charge and discharge. Notice the fluctuation of the reference voltage V_{ref5} (One of the 8 reference voltages) too. This is due to the weak link to VDD and VSS through a highly resistive path.

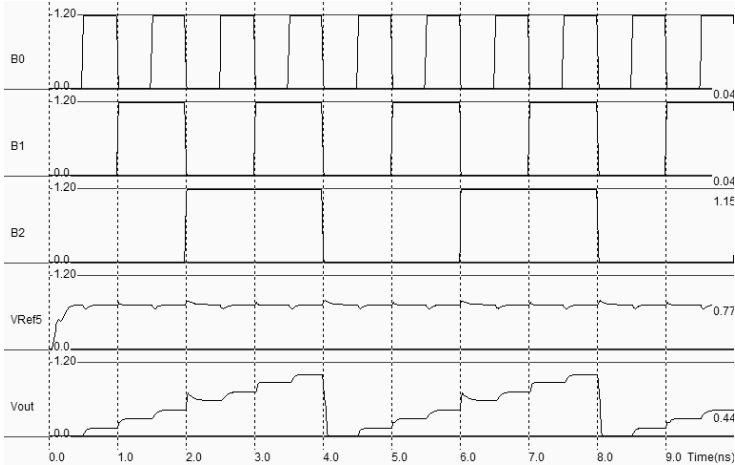


Figure 11-7. Simulation of the digital-analog converter (DAC.MSK).

The analog level V_{out} increases regularly with increasing digit input B . The converter is monotonic. However, it must be noticed that for a very short period of time, near $t=2.0ns$, the internal node discharge leads to a voltage overshoot close to one voltage step ΔV . Also notice that, according to the schematic diagram of figure 13-4, the output is connected to the N switches *On* and N switches *Off*.

11.2.2 R-2R ladder converter

It is not easy to construct a resistor-based DAC with a high resolution, due to the resistance spread and to the needs for 2^N serial resistors. A more compact choice is the R-2R ladder [Gustavsson]. Its configuration consists of a network of resistors alternating between R and $2R$. For a N bits DAC, only N cells based on 2 resistors R and $2R$ in series are required. The 4-bit and 8-bit implementation of this circuit are reported in figure 11-8. At the right end of the network is the output voltage V_{out} .

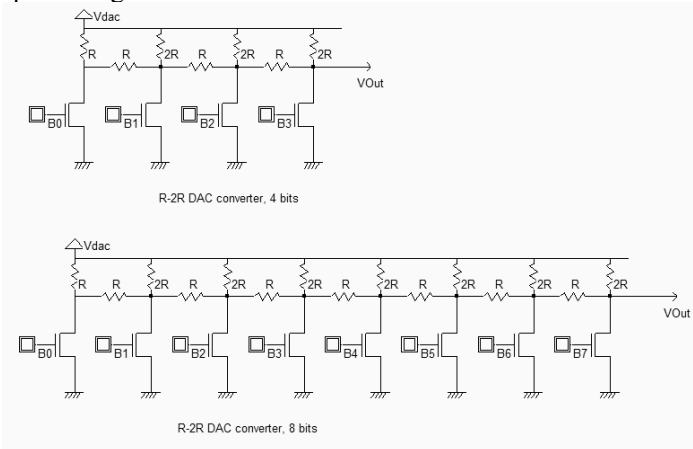


Figure 11-8. 4-bit and 8-bit DAC converter using the R-2R ladder (DACR2R.SCH)

11.3 Sample and Hold circuits

Sample and Hold (S/H) circuits are critical in converting analog signals into digital signals. The sample-and-hold main function is to capture the signal value at a given instant and hold it until the ADC has processed the information (Figure 11-9). The operation is repeated in time with a regular sampling period. Several parasitic effects may be observed : when the switch is turned off, a parasitic offset may appear due to capacitance couplings which modifies the voltage V_{in}^* . Also, after some nanoseconds, the stored voltage may be altered by parasitic discharge, appearing as an unpredictable droop.

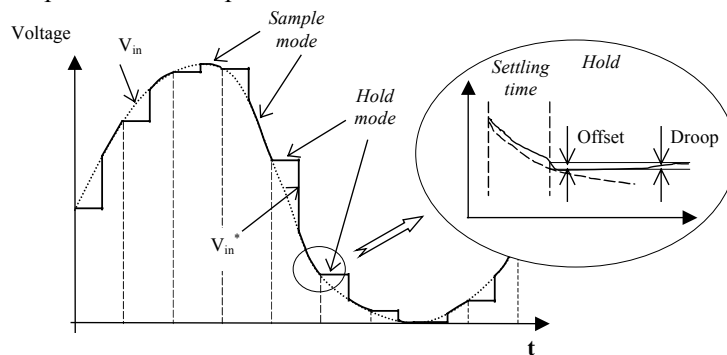


Figure 11-9. Sampling of an analog voltage (sample and hold modes)

The transmission gate can be used as a sample and hold circuit. The schematic diagram of the sample/hold circuit is proposed in figure 11-10. It corresponds to the classical transmission gate. The only important supplement is the storage capacitor, called C_{store} , appearing at the output V_{in}^* , the sampled version of V_{in} . The capacitor retains the voltage information during the conversion phase. By default, a parasitic capacitance always exists due to diffusion areas of the p-channel MOS and n-channel MOS devices. However, C_{store} includes a supplementary capacitor connected to the node V_{in}^* , with a capacitance value sufficiently high to counterbalance the effects of leakage currents.

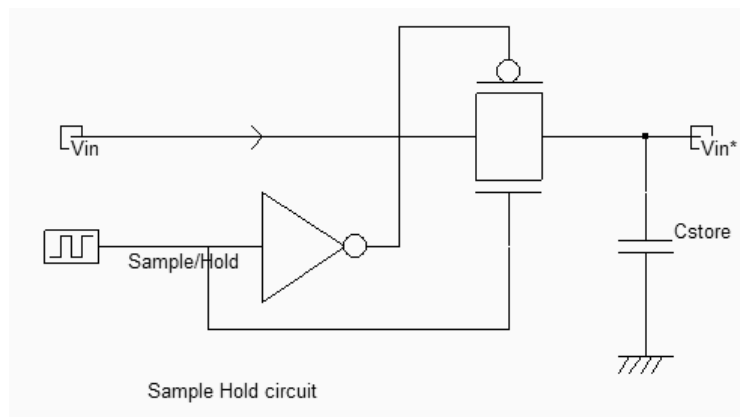


Figure 11-10. Schematic diagram of the Sample-Hold circuit (SampleHold.SCH)

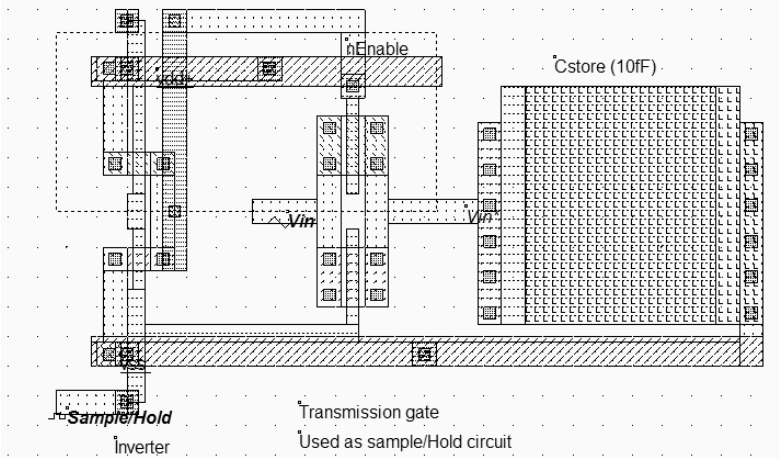


Figure 11-11. The transmission gate used to sample analog signals (SampleHold.MSK)

The layout of the transmission gate is reported in figure 11-11. The *sample/hold* command is situated on the left, and controls the transmission gate. The inverter is required for the pMOS device. The *Vin** signal is connected to a 10fF capacitor made of poly/Poly2. The effect of sample and hold is illustrated in figure 11-12. The voltage curves have been superimposed by using the simulation mode **Current and Voltage vs. Time**. When sampling, the transmission gate is turned on so that the sampled data *Vin** reaches the value of the sinusoidal wave *Vin*.

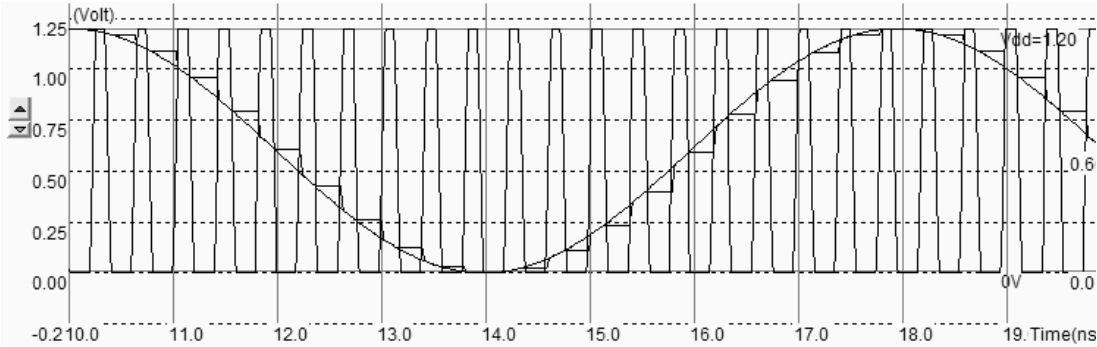


Figure 11-12. Effect of sampling (SampleHold.MSK)

11.4 Analog-Digital Converters architectures

The analog to digital converter is considered as an encoding device, where an analog sample is converted into a digital quantity with a number N of bits. ADCs can be implemented by employing a variety of architectures. We describe in the following chapters the Flash converter and successive approach converters.

11.4.1 The Flash converter Principles

The 2-bit analog-digital converter converts an analog value V_{in} into a two-bit digital value A coded on 2-bit A_1, A_0 . The flash converter uses three amplifiers which produce results C_0, C_1 and C_2 , connected to a coding logic to produce A_1 and A_0 in a very short delay (Figure 11-13). The flash converters are widely used for very high sampling rates, at the cost of very important power dissipation.

Analog Input V_{in}	C_2	C_1	C_0	A_1	A_0
$V_{in} < V_{ref0}$	0	0	0	0	0
$V_{ref0} < V_{in} < V_{ref1}$	0	0	1	0	1
$V_{ref1} < V_{in} < V_{ref2}$	0	1	1	1	0
$V_{in} > V_{ref2}$	1	1	1	1	1

Table 11-1. The specifications for a 2-bit flash ADC converter

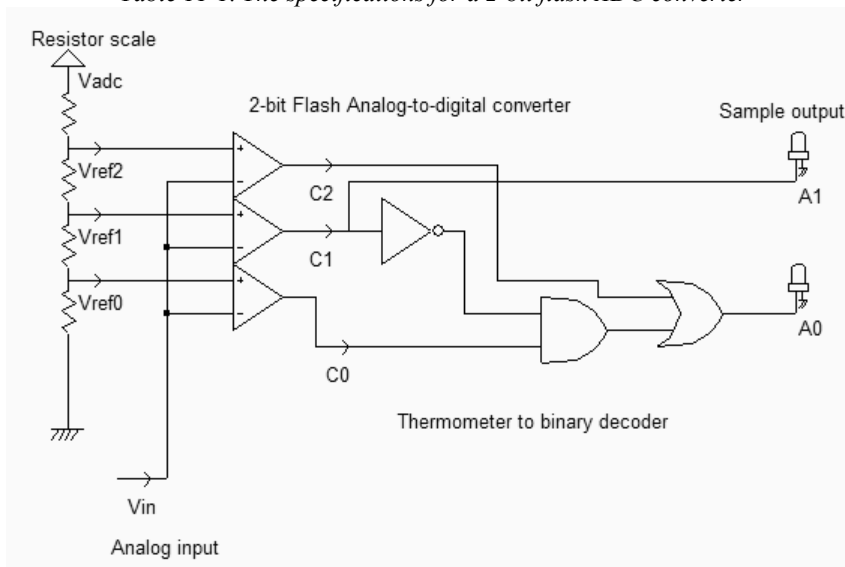


Figure 11-13. The schematic diagram of the 2-bit flash ADC converter (AdcFlash2bits.SCH)

A schematic diagram for the 2-bit flash converter is proposed in figure 11-13. The resistor scale produce reference voltages V_{ref0} , V_{ref1} and V_{ref2} . Three comparator circuits compute the difference between V_{in} and the reference voltage. Their outputs C_2, C_1 and C_0 are almost logic signals as the comparators are connected in open-loop. The main problem of the comparator-based architecture is that the output A_1, A_0 is not directly available from C_2, C_1 and C_0 . The comparator outputs represent the "thermometer coding" of the input. The ones propagate from C_0 to C_2 as the input V_{in} rises, as specified in table 11-1. A conversion circuit from thermometer code to binary code is needed. In the case of a 2-bit flash converter, the circuit is quite simple (figure 11-13), and can be efficiently implemented using one inverter (A_1) and a complex gate (A_0).

The resistor ladder generates intermediate voltage references used by the voltage comparators located in the middle of the layout. An unsalicide option layer multiplies the sheet resistance of the polysilicon ladder for an area-efficient implementation. The resistance symbol $R(\text{poly})$ is inserted in the layout to indicate to the simulator that an equivalent resistance must be taken into account for the analog simulation. Open-loop amplifiers are used as voltage comparators. The comparators address the decoding logic situated to the right and that provides correct A_0 and A_1 coding.

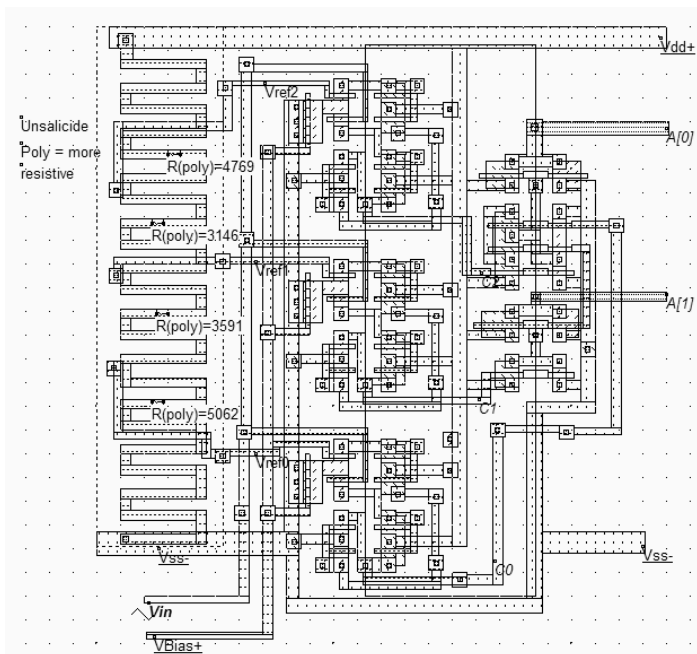


Figure 11-14. Design of the analog-digital converter (ADC.MSK).

In the simulation shown in Figure 11-15, the comparators C_0 and C_1 work well but the comparator C_0 is used in the lower limit of the voltage input range. The generation of combinations "01", "10" and "11" is produced rapidly but the generation of "00" is slow. The comparator C_0 may be modified to provide a faster response in comparison with low voltage, by changing the biasing conditions. An alternative is to reduce the input voltage range, which means that the resistance scale would be supplied by V_{dac-} larger than V_{SS} and V_{dac+} smaller than V_{DD} .

The main drawback of flash converters is the silicon area and the power consumption: every bit increase in resolution almost doubles the size of the ADC circuit and significantly increases the power consumption.

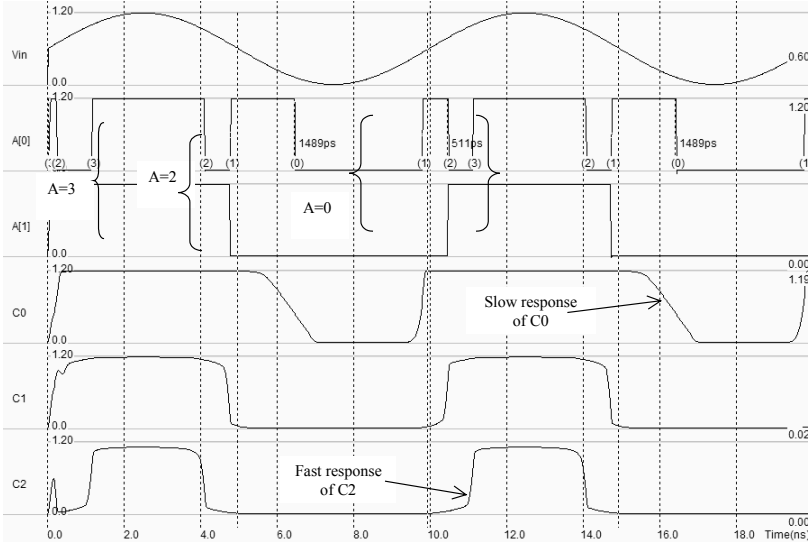


Figure 11-15. Simulation of the analog-digital converter (ADC.MSK).

11.4.2 Low speed ADC Converters

The most common low speed converter is the iterative converter. As shown in figure 11-16, it consists of a digital-to-analog converter, a counter and an analog comparator. Starting with the lowest voltage, the counter is increased until the DAC voltage V_{dac} is higher than the input voltage V_{in} .

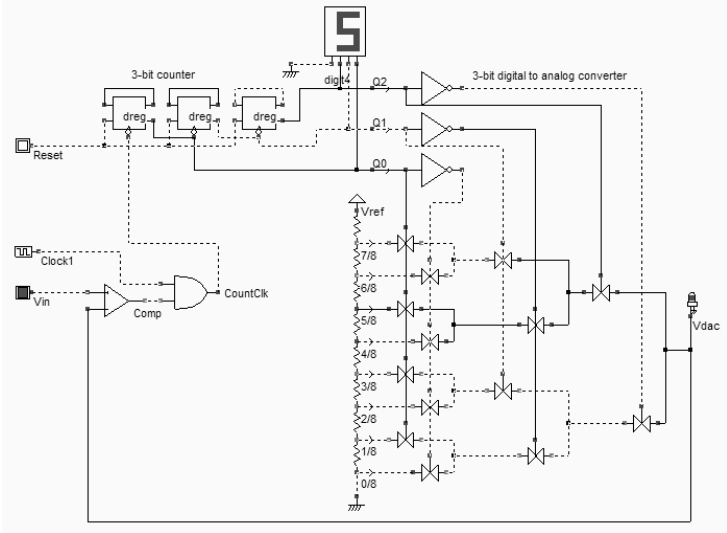


Figure 11-16. Iterative converter using a DAC (ADCIterative.SCH)

In the particular example shown in the figure, we suppose that V_{in} is a little higher than $V_{ref}/2$. The counter has reached the value 5 (101), which corresponds to the transfer of the reference

voltage $V_{ref} \times 5/8$ to V_{dac} . As V_{in} is lower than this reference, the comparator produces a 0, which stops the counter clock $CountClk$.

This converter is very simple to design but slow. Up to 2^N clock cycles are necessary to complete the conversion, where N is the resolution of the DAC and of the ADC converter. For example, with a 16 bit data converter with a 100 MHz clock frequency, the conversion rate is as low as 750 Hz.

11.5 Exercises

- Design of a 3-bit thermometer to binary coder corresponding to a 3-bit flash ADC
- Design a 3-bit flash converter using the thermometer coder and a set of 7 comparators.
- Design an iterative converter using a 4-bit counter and the 2-2R 4-bit DAC.
- Design a successive approach converter using a specific register and the R-2R 4-bit DAC.

12 Input/Output Interfacing

This chapter is dedicated to the interfacing between the integrated circuit and the external world. After a brief justification of the power supply decrease, the input/output pads used to import and export signals are dealt with. Then, the input pad protections against electrostatic discharge and voltage overstress are described. The design of output buffers is also presented, with focus on current drive.

12.1 Power Supply

The power supply of integrated circuits has continuously decreased with the progresses in process integration. Figure 12-1 shows the evolution of the supply voltage with the technology generation. A difference is made between the external supply and the internal supply. The external supply, usually 5V, 3.3V or 2.5V concerns the input/output interface. For compatibility reasons, the chip interface is kept at these high standard voltages, which eases the exchanges with other integrated circuits. The low internal supply concerns the core logic.

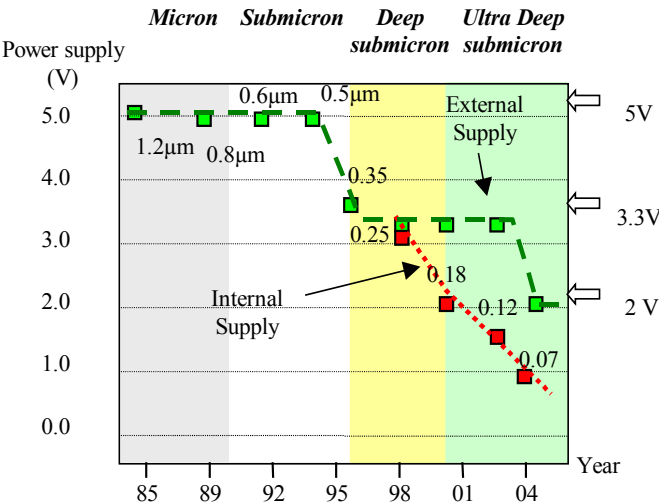


Figure 12-1: Power supply decrease with technology scale down

In 0.12µm CMOS technology, the external supply VDDH is 2.5V and the core supply VDD is 1.2V. The input/output structures work at high voltage by making extensive use of specific

MOS devices with thick oxide, while the internal devices work at low voltage with optimum performances.

12.2 The Bonding Pad

The bonding pad is the interface between the integrated circuit die and the package. The pad has a very large surface (Almost giant compared to the size of logic cells) because it is the place where the connection wire is attached to build the electrical link to the outside world. The pad is approximately $80\mu\text{m} \times 80\mu\text{m}$. The basic design rules for the pad are shown in figure 12-2. The pad-to-pad spacing, ($Rp02$), is also around $80\mu\text{m}$.

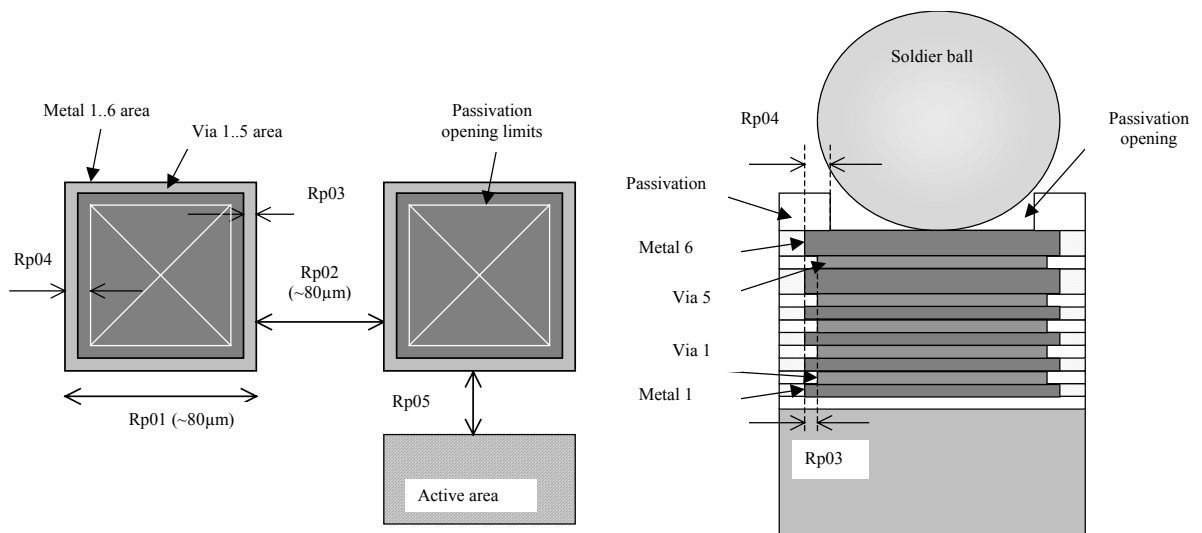


Figure 12-2: The bonding pad design rules

The cross-section shown in figure 12-2 gives an illustration of the passivation opening and associated design rule $Rp04$ on top of the metal and via stack. The thick oxide used for passivation is removed so that a bonding wire or a bonding ball can be connected by melting to the package. The pad can be generated by Microwind using the command **Edit** → **Generate** → **I/O pads**. The menu gives access to a single pad, with a default size given by the technology (around $80\mu\text{m}$ in this case), or to a complete pad ring, as detailed later.

12.3 The Pad ring

The pad ring consists of several pads on each of the four sides of the integrated circuit, to interface with the outside world. The default menu for an automatic generation of a pad ring is shown in figure 12-3. The proposed architecture is based on 5 pads on each side, meaning a total of 20 pads.

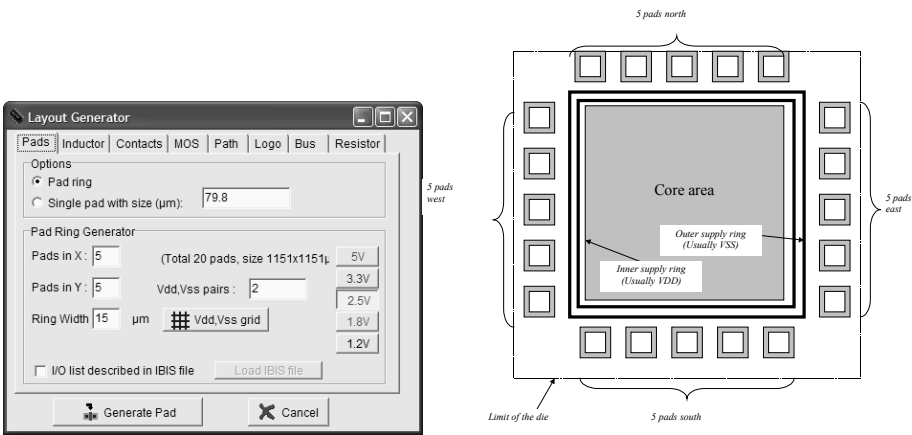


Figure 12-3: The menu for generating the pad ring and the corresponding architecture

The layout of the default pad ring generated in 0.12μm is shown in figure 12-4. Two pairs of supply VDD/VSS are automatically added to the pad structure. The first pair is fixed on the west side, the second pair on the east side. More VDD/VSS pairs may be generated. Usually one VDD/VSS pair is needed for 8-10 active input/output pads. Each I/O pad includes an over-voltage protection circuit based on two diodes which appear near the inner supply ring. These structures are justified and described later in this chapter.

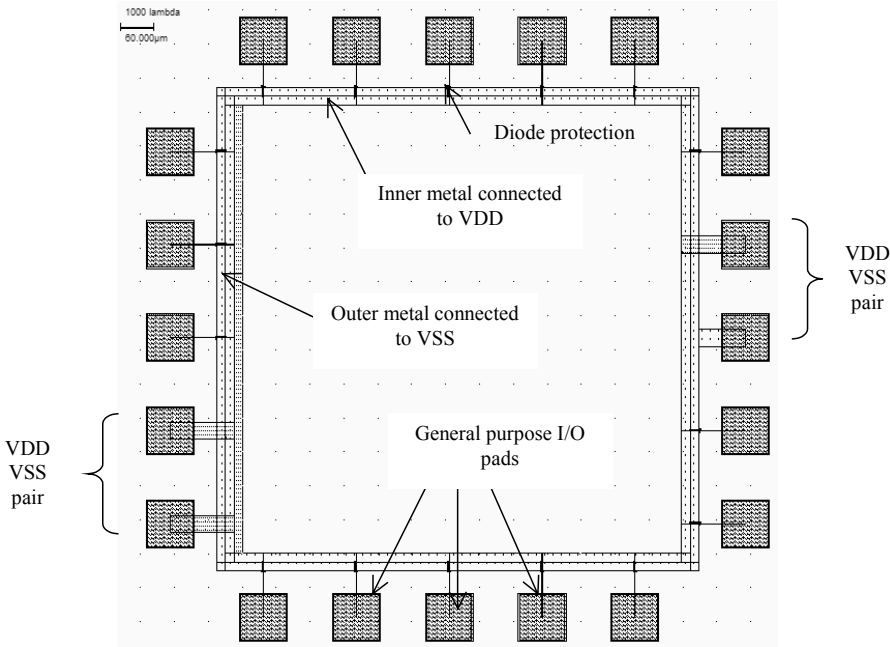


Figure 12-4: The default pad ring generated in 0.12μm, with 20 pads, including two pairs of VDD/VSS supply pins (padRing.MSK)

12.4 The supply rails

The supply voltage may be 5V, 3.3V, 2.5V, 1.8V or 1.2V. Most designs in 0.12 μm use 1.2V for the internal core supply and 2.5V for the interfacing. This is because the logic circuits of the core operate at low voltage to reduce power consumption, and the I/O structures operate at high voltage for external compatibility and higher immunity to external perturbations. Usually, an on-chip voltage regulator converts the high voltage into an internal low voltage.

In most cases, the integrated circuit uses two separate supply pads, one for the high voltage, one for the low voltage. Consequently, the integrated circuit has four-supply rings: VSS for I/Os (0.0V), VDD for I/Os (2.5V), VSS for the core (0.0V), VDD for the core (1.2V).

A metal wire cannot drive an unlimited amount of current. When the average current density is higher than 2.10^9 A/m^2 [Hastings], the grains of the polycrystalline aluminum interconnect start to migrate (The phenomenon is called electro migration) and the conductor ultimately melts. To handle very high current density, the supply metal lines must be enlarged. A typical rule of thumb is $2\text{mA}/\mu\text{m}$ width for aluminum supply lines and $5\text{mA}/\mu\text{m}$ for copper, which means that a copper interconnect is superior to aluminum in sustaining large currents.

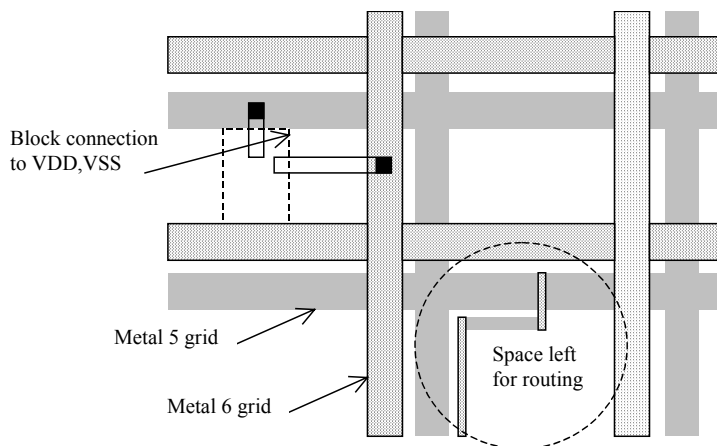


Figure 12-5: The supply rails are routed in metal5 and metal6 with a regular grid to provide power supply in all regions of the integrated circuit

A complex logic core may consume amperes of current. In that case, the supply lines must be enlarged in order to handle very large currents properly. The usually design approach consists in creating a regular grid structure, as illustrated in figure 12-5, which provides the supply current at all points of the integrated circuit. In that test circuit, the VDD supply is assigned to metal5, VSS to metal 6.

12.5 Input Structures

The input pad includes some over-voltage and under-voltage protections due to external voltage stress, electrostatic discharge (ESD) coupling with external electromagnetic sources, etc.. Such protections are required as the oxide of the gate connected to the input can easily be destroyed by over voltage. The electrostatic discharges may attain 1000 to 5000Volt.

One of the most simple ESD protections is made up of one resistance and two diodes (Fig. 12-6). The resistor helps to dissipate the parasitic energy and reduces the amplitude of the voltage overstress. One diode handles the negative voltage flowing inside the circuit (N+/P substrate diode), the other diode (P+/N well) handles the positive voltage. The combination of the serial resistor and the diode bridge represents an acceptable protection circuit against transient voltage overstress around +/-50V.

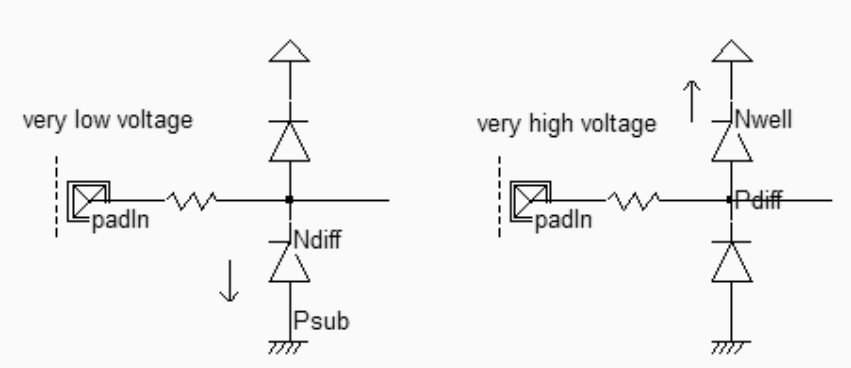


Figure 12-6: Input protection circuit (IOPadIn.SCH)

Diodes are essential parts of the ESD protection. Used since the infancy stage of microelectronics, the diodes are still widely used because of their efficiency and simplicity [Dabral]. The native diodes in CMOS technology consist of an N+ diffusion in the p-substrate and a P+ diffusion in the n-well.

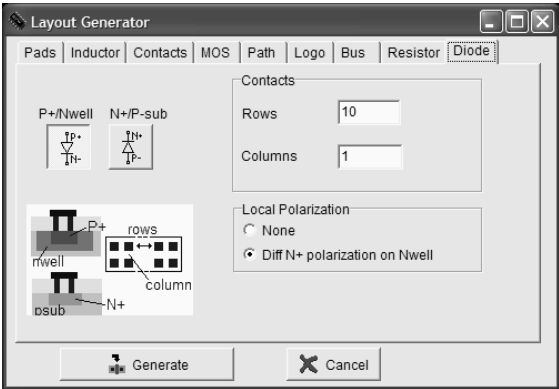


Figure 12-7: The diode generating menu in Microwind (By default a P+/well diode)

The command used to generate a protection diode in Microwind is **Edit →Generate →Diode**. Click either the P+/nwell diode or the N+/P substrate diode. By default, the diode is quite large,

and connected to the upper metal by a row of 10 contacts. The N+ diode region is surrounded by a polarization ring made of P+ diffusion, as shown in figure 12-7. The large number of rows ensures a large current capability, which is very important in the case of ESD protection devices.

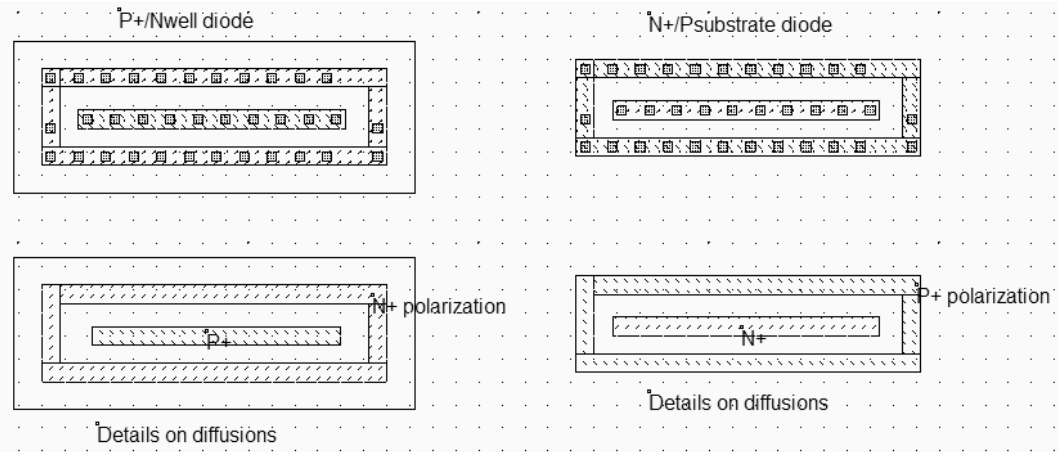


Figure 12-8: Generating a default protection diode (IODiode.MSK)

A protection circuit example is simulated in figure 12-9. It consists of a pad 50x50µm, a serial resistor around 200 ohm and two diodes. When a very high sinusoidal waveform (+/- 10V) is injected, the diodes exhibit a clamping effect both for the positive and negative overstress. The best simulation mode is **Voltage and Currents**. The voltage scale may be changed using the arrows on the left side of the lower voltage window. The internal voltage remains within the voltage range [0..VDDH] while the voltage near the pad is -10 to +10V wide. Notice that the current flowing in the diodes is around 1mA (Figure 12-9).

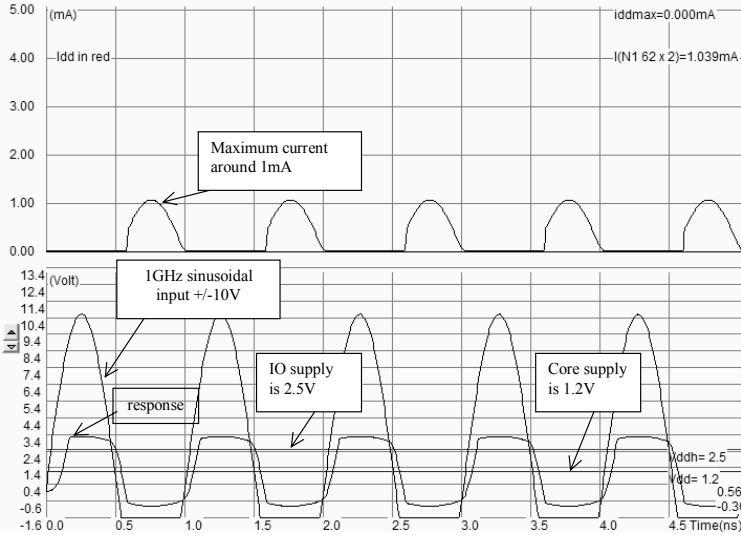


Figure 12-9: The diodes clamp the positive and negative overstress so that the internal voltage keeps close to the voltage range [0..VDDH] (IoPadIN.MSK)

12.6 High voltage MOS

The general diagram of an input structure is given in figure 12-10. A high voltage buffer is used to handle voltage overstress issued from electrostatic discharges. The logic signal is then converted into a low voltage signal to be used in the core logic. For interfacing with input/output, specific high voltage MOS are introduced. These MOS devices are called high voltage MOS. They use a double gate oxide to handle the high voltage of the I/Os. The thin oxide used for internal logic devices would be damaged by the high I/O voltage. The high voltage device symbols are drawn with a double line. The symbol *Vdd_HV* represents the I/O voltage, which is usually 2.5V in CMOS 0.12µm.

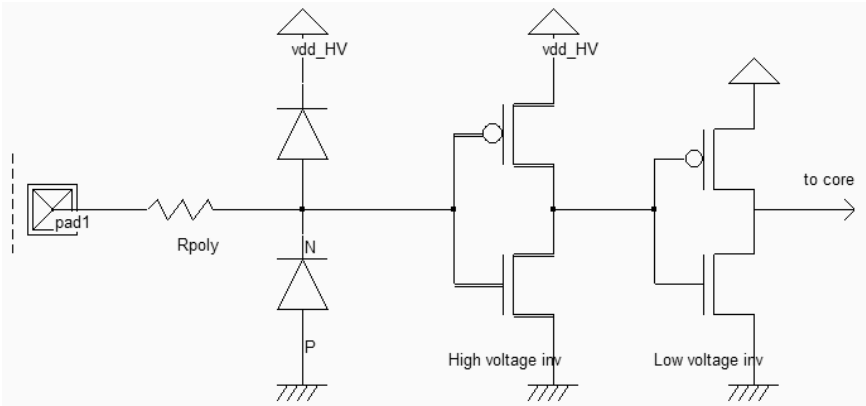


Figure 12-10: The basic principles for an input circuit, including the ESD protection and the voltage translator (IOPadIn.SCH)

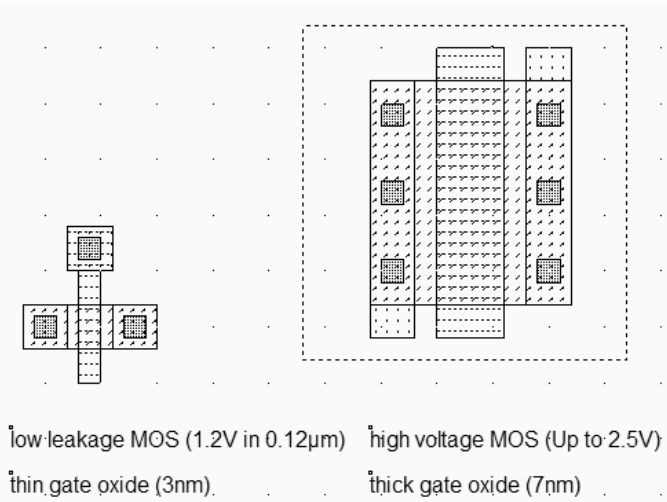


Figure 12-11: High voltage MOS device vs. normal MOS (MOSHHighVoltage.MSK)

The high voltage MOS layout differs slightly from the normal MOS, as shown in the comparative layout view of figure 12-11. The high voltage MOS uses a gate width which is much larger than that of the regular MOS. Usually, the lateral drain diffusion, which aims at limiting the hot-carrier effect at boosting the device lifetime, is removed in high voltage MOS devices. In 0.12 μm , the gate oxide of the high voltage MOS is around 5nm, while the core MOS is 2nm.

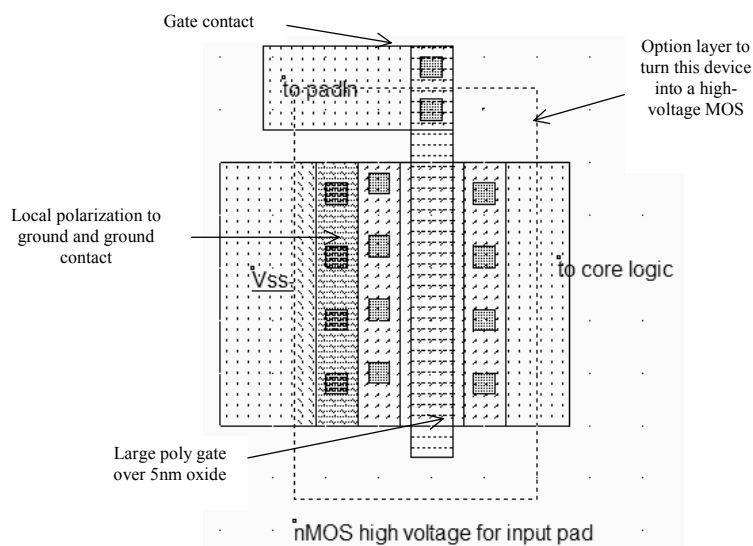


Figure 12-12: Layout of the input MOS device (IOPadMos.MSK)

The bird's view of the layout (Figure 12-12) reveals that the polysilicon gate is not the usual 2-lambda length. In the case of high voltage MOS devices, the minimum length is 4 lambda. The 2 lambda sizing is not compatible with the double gate oxide and the high voltage operation. The gate oxide is twice thicker than the low voltage MOS. The high voltage device performance corresponds approximately to a 0.25 μm MOS device. To turn a normal MOS into a high voltage MOS, the designer must add an option layer (The dot rectangle in figure 10-12). The tick in front of **High voltage MOS** assigns high voltage properties to the device: double oxide, removed LDD, different rules for minimum length, and different MOS model parameters.

12.7 Input pad with Schmitt Trigger

Using a Schmitt trigger instead of an inverter helps to transform a very noisy input signal into a clean logic signal. The Schmitt trigger circuit switches at different thresholds, in order to increase the noise margin of the input buffer. The main difference between the inverter and the Schmitt trigger appears in the simulation shown in figure 12-13.

While the inverter may transform a noisy input into several glitches at the output near the commutation point of the inverter, the Schmitt trigger produces one single commutation.

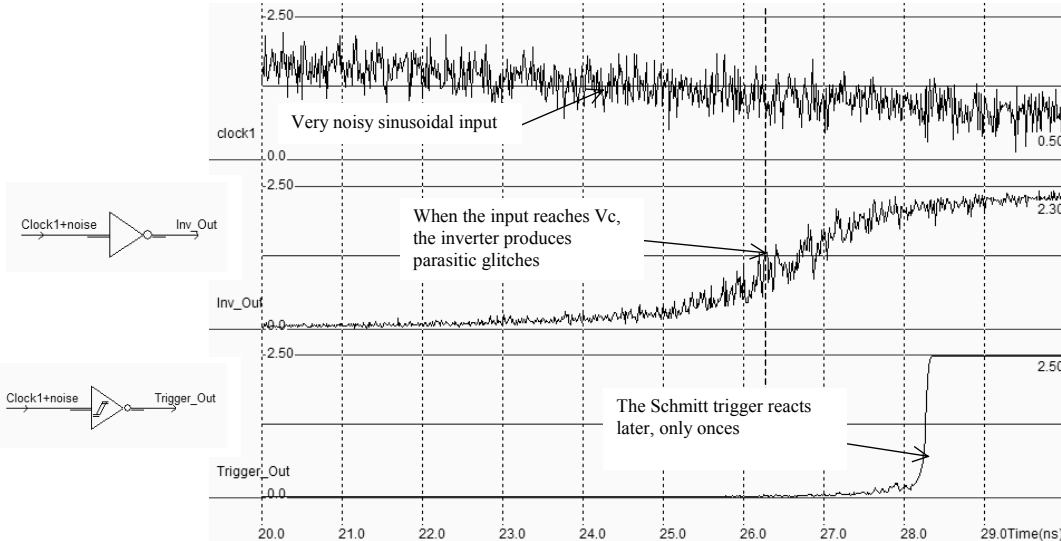


Figure 12-13: The filtering effect of the Schmitt trigger in presence of noisy input signal (TriggerCompInv.MSK)

The schematic diagram of the trigger is proposed in figure 12-14 [Bellaouar]. A brilliant idea lies beyond this circuit - it is based on a modification of the commutation point, thanks to feedback MOS devices. The pMOS feedback device adds a path to ground when *Trigger_Out* is low. Consequently, the threshold voltage is lowered to a commutation point V_{c_low} , lower than the commutation point of a regular inverter V_C . The nMOS feedback device adds a path to V_{DD_HV} when *Trigger_Out* is high. Consequently, the threshold voltage has risen to V_{c_high} , higher than the commutation point V_C .

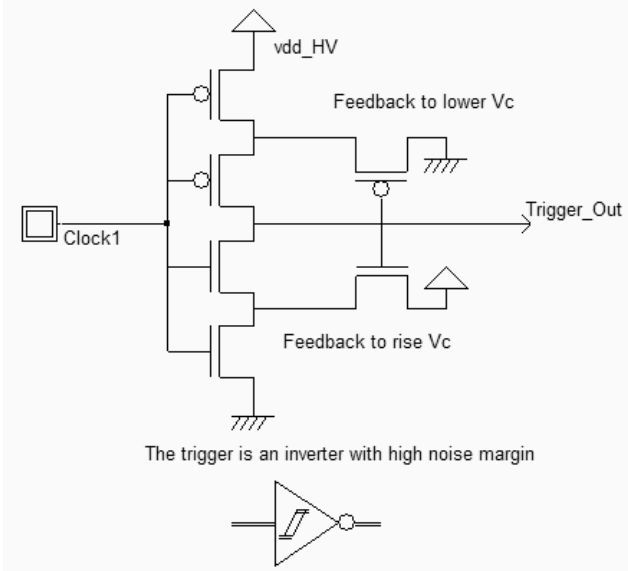


Figure 12-14. Schematic diagram of the trigger (Trigger.SCH)

The layout of the trigger is shown in figure 12-15. The feedback MOS devices are situated on the right of the trigger core. An inverter is added for comparison. The most demonstrative simulation is probably the compared static characteristics of the inverter and the trigger (Figure 12-16).

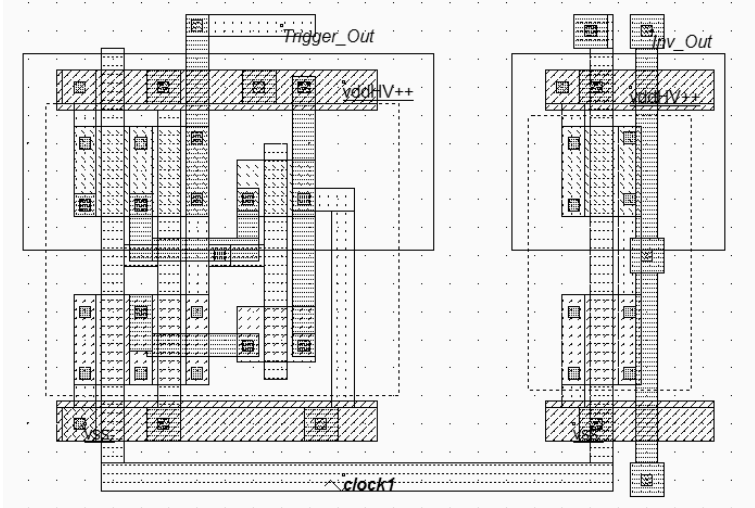


Figure 12-15. Layout of the trigger and the inverter, using high voltage MOS devices (TriggerCompInv.MSK)

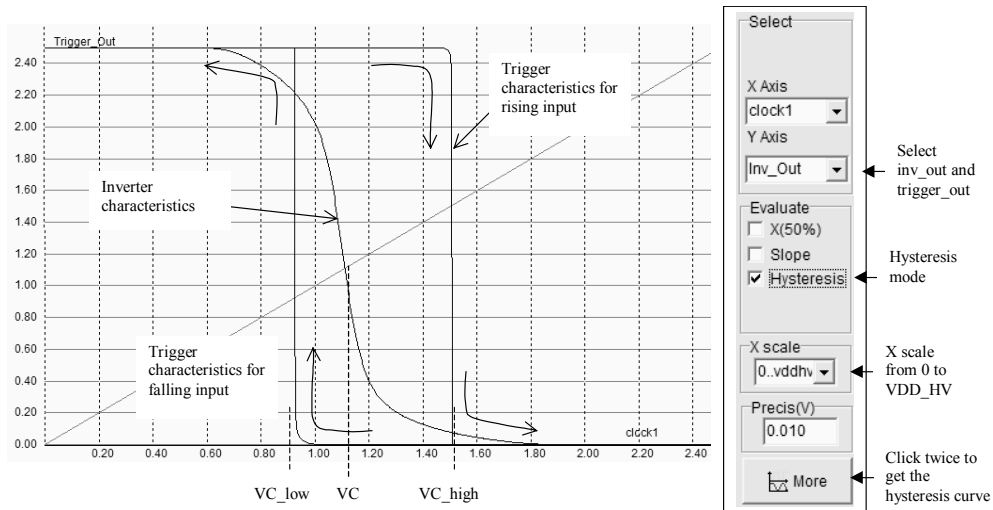


Figure 12-16. Static characteristics of the trigger compared to the inverter (Trigger.MSK)

The static simulation is available in **Voltage vs. voltage** mode. Firstly, the X scale must be adjusted to $0..VDD_{HV}$. Secondly, the hysteresis mode must be activated: at each simulation, the input signal is either decreased or increased. Finally, the trigger characteristics may be added to the inverter by changing the selected output.

12.8 Digital Output Structures

The role of the output buffer is to ensure that the signal coming out of the integrated circuit is propagated safely to the receiver which is usually the input of a second integrated circuit. The schematic diagram of the basic output buffer is given in figure 12-17. A very simple structure is used to protect the output buffer from electrostatic discharge, and more generally any over or under voltage. A Zener diode may be used, or a set of two diodes, as for the input pad.

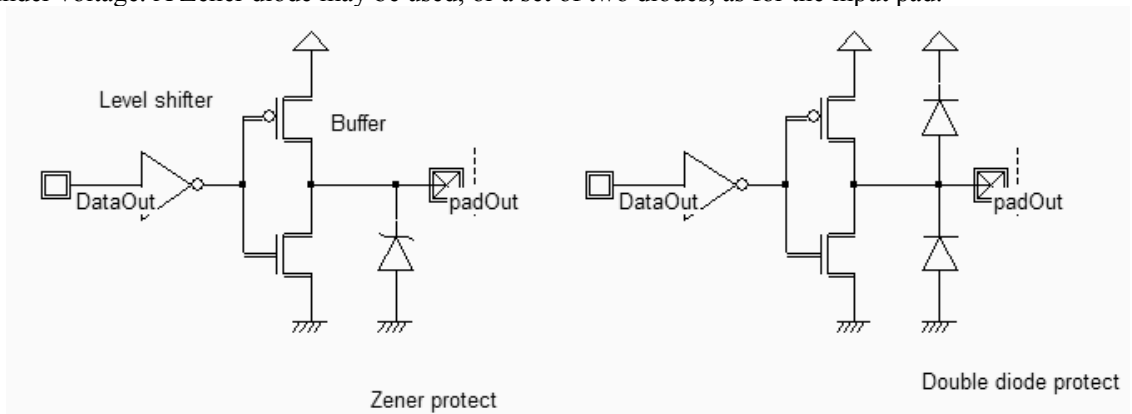


Figure 12-17: The output buffer design including the protection against electrostatic discharge (IOPadOut.SCH)

12.8.1 Level shifter

The role of the level shifter is to translate the low voltage logic signal *Data_Out* into a high voltage logic signal which controls the buffer devices. Figure 12-18 gives the schematic diagram of a level shifter circuit which has no problem of parasitic DC power dissipation. The circuit consists of a low voltage inverter, the level shifter itself and the buffer. The circuit has two power supplies: a low voltage *VDD* for the left-most inverter, and a high voltage *VddHV* for the rest of the circuit.

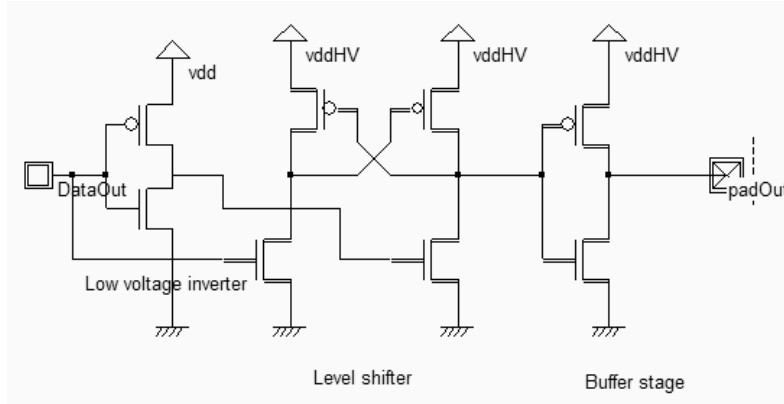


Figure 12-18: Schematic diagram of a level shifter (IOPadOut.SCH)

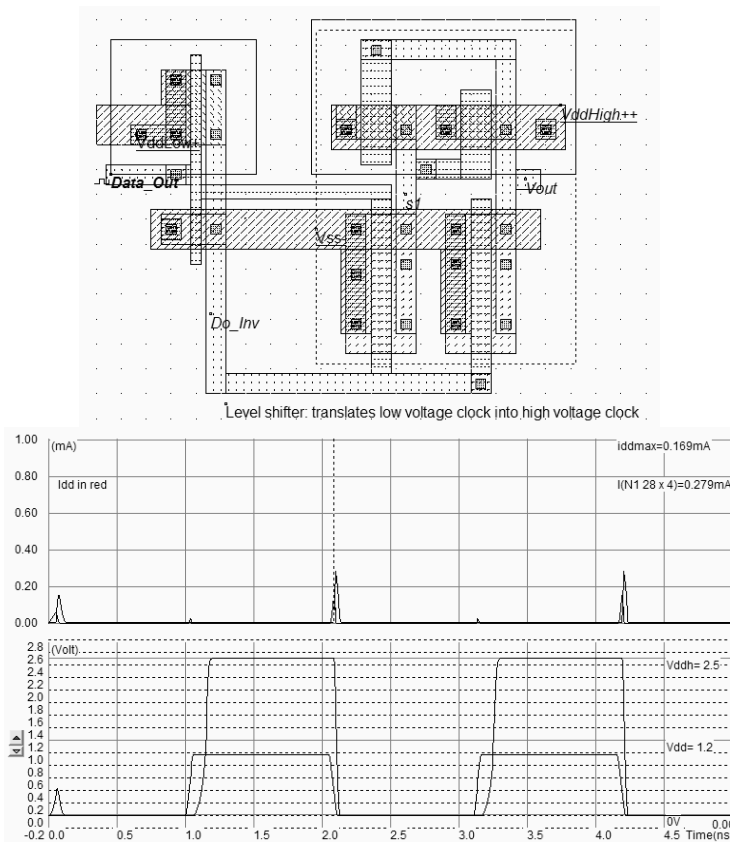


Figure 12-19: Layout and simulation of the level shifter (LevelShift.MSK)

The layout of the level shifter is shown in figure 12-19. The left part works at low voltage 1.2V, the right part works with high-voltage MOS devices, at a supply of 2.5V (*VddHigh*). The data signal *Data_Out* has a 0-1.2V voltage swing. The output *Vout* has a 0-2.5V voltage swing. This time, no DC consumption appears except during transitions of the logic signals, as shown in the simulation of figure 12-9.

12.9 CORE/PAD LIMITATION

When the active area of the chip is the main limiting factor, the pad structure may be designed in such a way that the width is large but the height is as small as possible. In that case, the oversize due to the pads is minimized. Protections are placed on both sides of the pad area. This situation is often called "Core Limited", and corresponds to the design shown in figure 12-20. In most pad libraries, the core limited structures have a minimum height, which often implies to place the protection circuits on both sides of the pad.

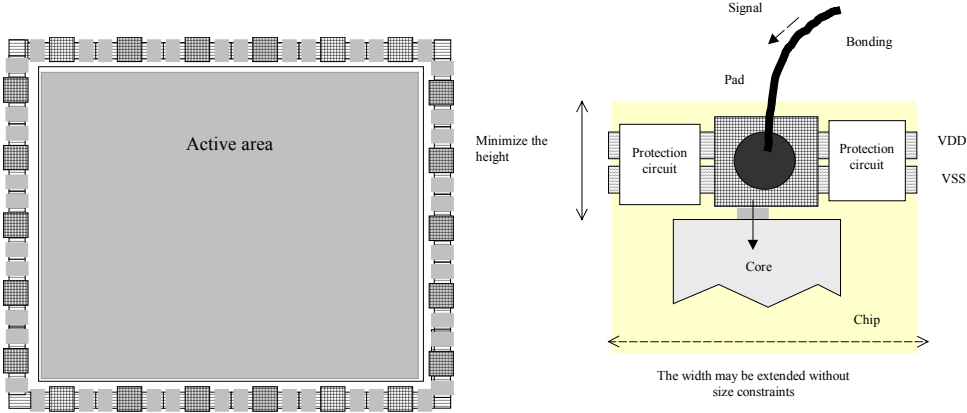


Figure 12-20 : Chip size fixed by the core

When the number of pads of the chip is the main limiting factor, the situation is called "Pad Limited", and corresponds to the design shown in figure 12-21. The pad structure may be designed in such a way that the width is small but the height is large. In that case, the oversize due to the pads is minimized. Protections are placed under the pad area.

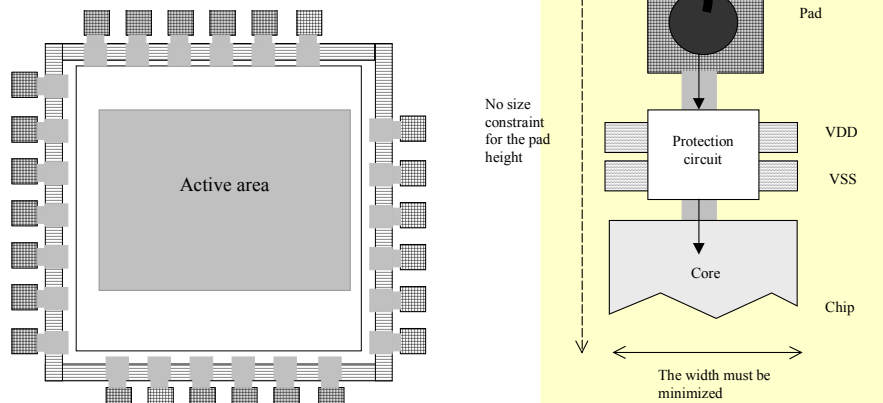


Figure 12-21. Chip size fixed by the number of pads

The spared silicon area may be avoided by using a double pair of I/O pads, as illustrated in figure 12-22. This attractive feature has been made available starting 0.25 μ m technology. An example of a test-chip using a double pad ring is reported in the figure below, which corresponds to a CMOS 0.18 μ m test-chip fabricated by ST-Microelectronics for research purposes. The pad pitch is significantly reduced thanks to the double row of bonding pads. The pad pitch for a single row is the sum of the minimum pad width $Rp01$ and of the pad distance $Rp02$. In the double ring structure, the pad pitch is divided by a factor of 2.

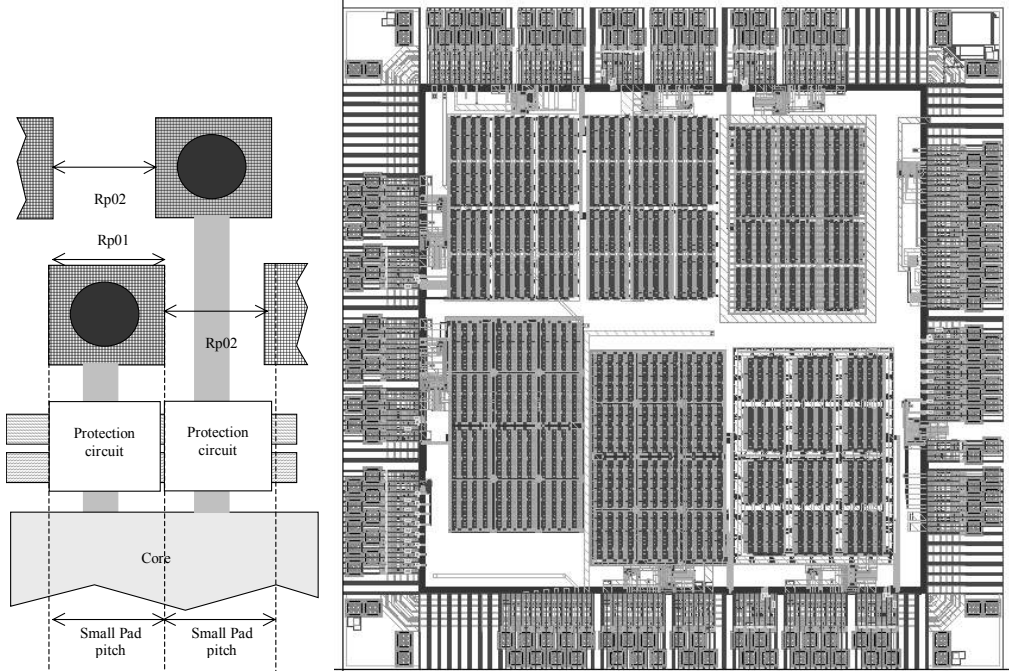


Figure 12-22. An example of a double ring test-chip in 0.18 μ m technology (Courtesy of ST-Microelectronics)

12.10 I/O Pad description using Ibis

IBIS is a standard for electronic behavioral specifications of integrated circuit input/output analog characteristics. In order to enable an industry standard method to transport electronically IBIS modeling data between semiconductor vendors, simulation vendors, and end customers, a format has been proposed by the IBIS group [Ibis]. The version 3.2 of IBIS was finalized by a wide group of industry experts representing various companies and interests. A complete backup of slides and meeting notes for the latest IBIS open forum is available on the Ibis web site.

Microwind2 uses IBIS to pilot the generation of the I/O pads, when compiling a Verilog file. Click the button **Load** in front of the check box **Fixed I/Os**, in the Verilog menu. The default IBIS file is **default.IBS**.

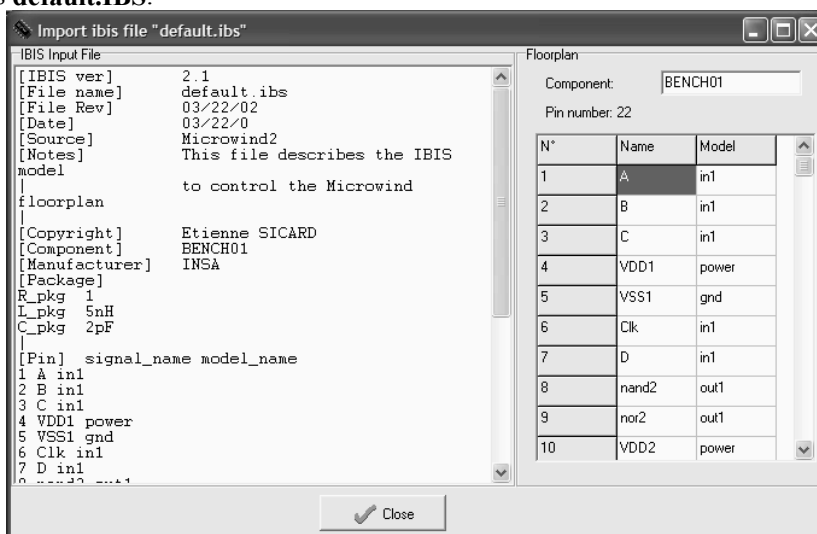


Figure 12-24. The IBIS description file loaded for controlling the pin assignment

It can be seen that IBIS is a text file, with a simple structure based on keywords. We only use a very reduced set of the available keywords, listed in table 12-2.

[IBIS Ver]	Specifies the IBIS template version. This keyword informs electronic parsers of the kinds of data types that are present in the file.
[File Rev]	Tracks the revision level of a particular .ibs file. Revision level is set at the discretion of the engineer defining the file.
[Component]	Marks the beginning of the IBIS description of the integrated circuit named after the keyword.
[Manufacturer]	Specifies the manufacturer's name of the component. Each manufacturer must use a consistent name in all .ibs files.
[Package]	Defines a range of values for the default packaging resistance, inductance, and capacitance of the component pins. Sub-Parameters are named R_pkg, L_pkg, C_pkg
[Pin]	Associates the component's I/O models to its various external pin names and signal names. Each line must contain either three or six columns. A pin line with three

	columns associates the pin's signal and model. Six columns can be used to override the default package values. In that case headers R pin, L pin, and C pin appear.
--	---

Table 12-2. The IBIS keywords understood by Microwind

At a click on **Generate Pad**, the layout is created, which corresponds to the list of pins declared in the IBIS file.

13 Design Rules

13.1 Select a Design Rule File

The software can handle various technologies. The process parameters are stored in files with the appendix '.RUL'. The default technology corresponds to a generic 6-metal 0.25µm CMOS process. The default file is CMOS012.RUL.

- To select a new foundry, click on **File → Select Foundry** and choose the appropriate technology in the list.
- To set a specific foundry as the default foundry, click **File → Properties , 'Set as Default Technology'**.

13.2 Start Microwind with a specific design Rule File

To start Microwind with a specific design rule file, click with the right button of the mouse on the Microwind icon, select the "Properties" item, then the target. The default target may be:

```
C:\microwind2\Microwind2.exe
```

The command line may include two more parameters:

- The *First* parameter is the default mask file loaded at initialization
- The *Second* parameter is the design rule file loaded at initialization

The following command executes MICROWIND2 with a default mask file « **test.MSK** » and the rule file « **cmos018.RUL** ».

```
C:\microwind2\Microwind2.exe test cmos018.rul
```

This section gives information about the design rules used by Microwind2. You will find all the design rule values common to all CMOS processes. All that rules, as well as process parameters and analog simulation parameters are detailed here.

13.3 Lambda Units

The Microwind software works is based on a lambda grid, not on a micro grid. Consequently, the same layout may be simulated in any CMOS technology. The value of lambda is half the minimum polysilicon gate length. Table A-xxx gives the correspondence between lambda and micron for all CMOS technologies available in the companion CD-ROM.

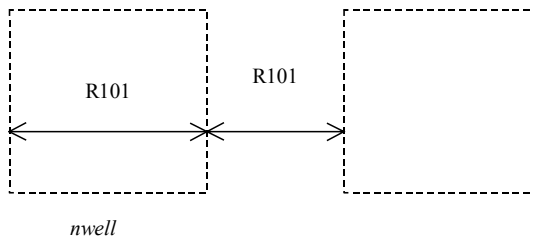
Technology file available in the CD-Rom	Minimum gate length	Value of lambda
Cmos12.rul	1.2µm	0.6µm
Cmos08.rul	0.7µm	0.35µm
Cmos06.rul	0.5µm	0.25µm
Cmos035.rul	0.4µm	0.2µm
Cmos025.rul	0.25µm	0.125µm
Cmos018.rul	0.2µm	0.1µm
Cmos012.rul	0.12µm	0.06µm
Cmos90n.rul	0.1µm	0.05µm
Cmos70n.rul	0.07µm	0.035µm
Cmos50n.rul	0.05µm	0.025µm

Table 13-1: correspondence between technology and the value of lambda in µm

The software can handle various technologies. The process parameters are stored in files with the appendix '.RUL'. The default technology corresponds to a generic 6-metal 0.12µm CMOS process. The default file is CMOS012.RUL. To select a new foundry, click on **File -> Select Foundry** and choose the appropriate technology in the list.

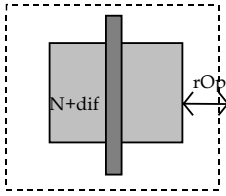
13.4 N-Well

- r101 Minimum well size 12 λ
- r102 Between wells 12 λ
- r110 Minimum well area 144 λ²



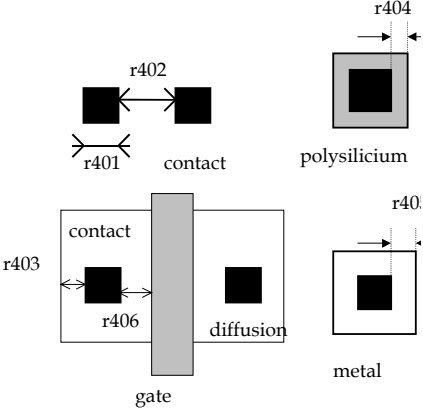
13.5 Diffusion

rOpt Border of "option" layer over diff 7λ
N+ and diff P+



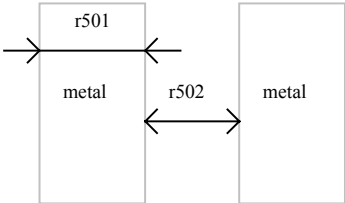
13.9 Contact

r401	Contact width	2 λ
r402	Between two contacts	5 λ
r403	Extra diffusion over contact	2 λ
r404	Extra poly over contact	2 λ
r405	Extra metal over contact	2 λ
r406	Distance between contact and poly gate	3 λ
r407	Extra poly2 over contact	2 λ



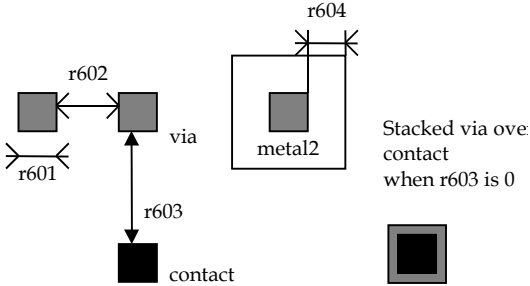
13.10 Metal 1

r501	Metal width	4 λ
r502	Between two metals	4 λ
r510	Minimum surface	16 λ ²



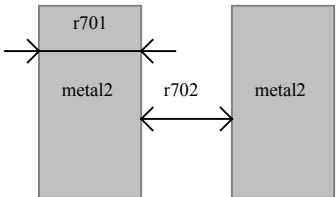
13.11 Via

r601	Via width	2 λ
r602	Between two Via	5 λ
r603	Between Via and contact	0 λ
r604	Extra metal over via	2 λ
r605	Extra metal2 over via:	2 λ



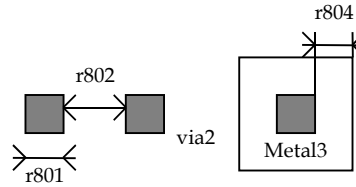
13.12 Metal 2

r701	Metal width::	4 λ
r702	Between two metal2	4 λ
r710	Minimum surface	16 λ ²



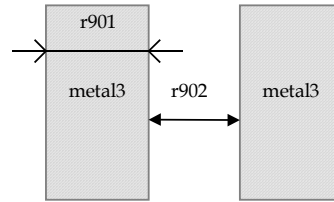
13.13 Via 2

- r801 Via2 width : 2λ
- r802 Between two Via2: 5λ
- r804 Extra metal2 over via2: 2λ
- r805 Extra metal3 over via2: 2λ



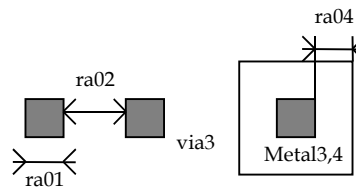
13.14 Metal 3

- r901 Metal3 width: 4λ
- r902 Between two metal3 : 4λ
- r910 Minimum surface : $32 \lambda^2$



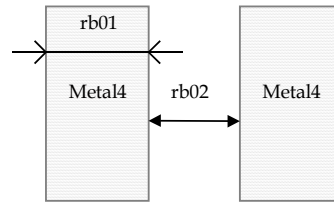
13.15 Via 3

- ra01 Via3 width : 2λ
- ra02 Between two Via3: 5λ
- ra04 Extra metal3 over via3: 2λ
- ra05 Extra metal4 over via3: 2λ



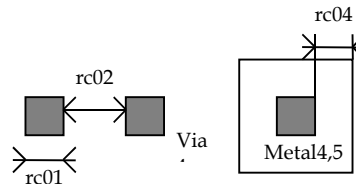
13.16 Metal 4

- rb01 Metal4 width: 4λ
- rb02 Between two metal4 : 4λ
- rb10 Minimum surface : $32 \lambda^2$



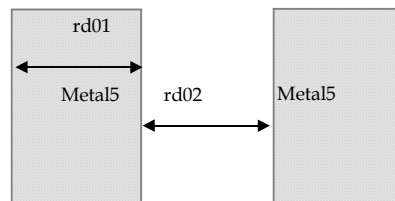
13.17 Via 4

- rc01 Via4 width : 2λ
- rc02 Between two Via4: 5λ
- rc04 Extra metal4 over via2: 3λ
- rc05 Extra metal5 over via2: 3λ



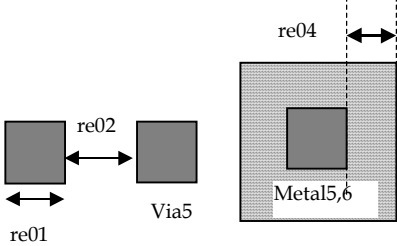
13.18 Metal 5

- rd01 Metal5 width: 8λ
- rd02 Between two metal5 : 8λ
- rd10 Minimum surface : $100 \lambda^2$



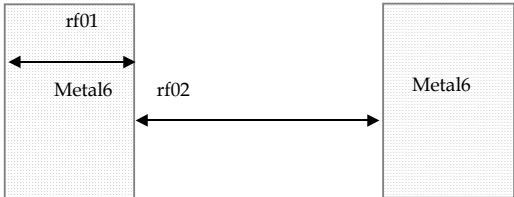
13.19 Via 5

- re01 Via5 width : 4λ
- re02 Between two Via5: 6λ
- re04 Extra metal5 over via5: 3λ
- re05 Extra metal6 over via5: 3λ



13.20 Metal 6

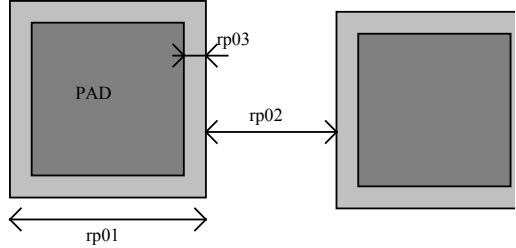
- rf01 Metal6 width: 8λ
- rf02 Between two metal6 : 15λ
- rf10 Minimum surface : $300 \lambda^2$



13.21 Pads

The rules are presented below in μm . In .RUL files, the rules are given in lambda. As the pad size has an almost constant value in μm , each technology gives its own value in λ .

- rp01 Pad width: $100 \mu\text{m}$
- rp02 Between two pads $100 \mu\text{m}$
- rp03 Opening in passivation v.s via : $5 \mu\text{m}$
- rp04 Opening in passivation v.s metals: $5 \mu\text{m}$
- rp05 Between pad and unrelated active area : $20 \mu\text{m}$



13.22 Electrical Extraction Principles

MICROWIND2 includes a built-in extractor from layout to electrical circuit. Worth of interest are the MOS devices, capacitance and resistance. The flow is described in 13-2.

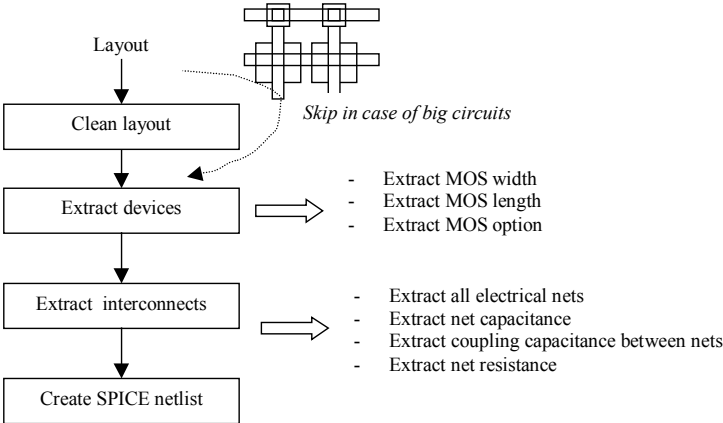


Figure 13-2: Extraction of the electrical circuit from layout

The first step consists in cleaning the layout. Mainly, redundant boxes are removed, overlapping boxes are transformed into non-overlapping boxes. In the case of complex circuits, MICROWIND2 may skip this cleaning step as it required a significant amount of computational time.

13.23 Node Capacitance extraction

Each deposited layer is separated from the substrate by a SiO2 oxide and generated by a parasitic capacitor. The unit is the aF/μm² (atto = 10⁻¹⁸). Basically all layers generate parasitic capacitors. Diffused layers generate junction capacitors (N+/P-, P+/N). The list of capacitance handled by MICROWIND2 is given below. The name corresponds to the code name used in CMOS012.RUL (CMOS 0.12μm). Surface capacitance refers to the body. Vertical crosstalk capacitance refer to inter-layer coupling capacitance, while lateral crosstalk capacitance refer to adjacent interconnects.

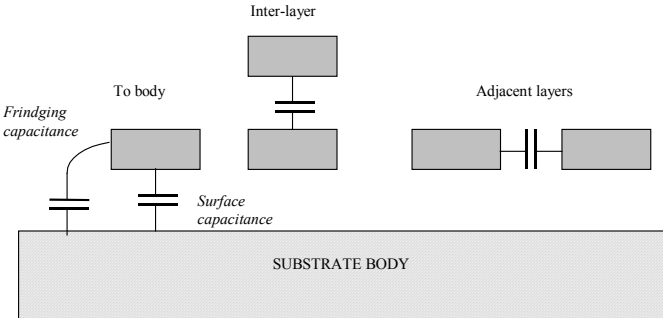


Figure 13-3: Capacitances

13.24 SURFACE CAPACITANCE

NAME	DESCRIPTION	LINEIC (aF/μm)	SURFACE (aF/μm ²)
CpoOxyde	Polysilicon/Thin oxide capacitance	n.c	4600
CpoBody	Polysilicon to substrate capacitance	n.c	80
CMEBody	Metal on thick oxide to substrate	42	28
CM2Body	Metal2 on body	36	13
CM3Body	Metal3 on body	33	10
CM4Body	Metal4 on body	30	6
CM5Body	Metal5 on body	30	5
CM6Body	Metal6 on body	30	4

13.25 INTER-LAYER CROSSTALK CAPACITANCE

NAME	DESCRIPTION	VALUE (aF/μm ²)
CM2Me	Metal2 on metal 1	50
CM3M2	Metal3 on metal 2	50
CM4M3	Metal4 on metal 3	50
CM5M4	Metal5 on metal 4	50
CM6M5	Metal6 on metal 5	50

13.26 LATERAL CROSSTALK CAPACITANCE

NAME	DESCRIPTION	VALUE (aF/μm)
CMeMe	Metal to metal (at 4λ distance, 4λ width)	10
CM2M2	Metal2 to metal 2	10
CM3M3	Metal3 to metal 3	10
CM4M4	Metal4 to metal 4	10
CM5M5	Metal5 to metal 5	10
CM6M6	Metal6 to metal6	10

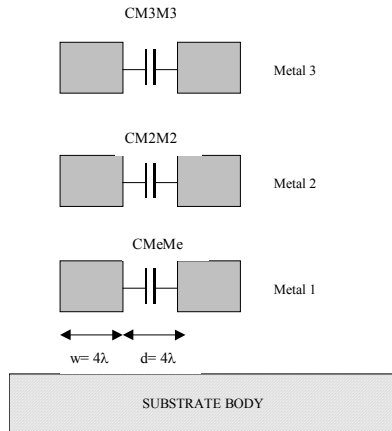


Figure 13-4: Crosstalk capacitance

The crosstalk capacitance value per unit length is given in the design rule file for a predefined interconnect width ($w=4\lambda$) and spacing ($d=4\lambda$).

In Microwind2, the computed crosstalk capacitance is not dependant on the interconnect width w .

The computed crosstalk capacitance value is proportional to $1/d$ where d is the distance between interconnects.

13.27 Parameters for Vertical Aspect of the Technology

The vertical aspect of the layers for a given technology is described in the RUL file after the design rules, using code HE (height) and TH (thickness) for all layers. The figure 13-5 below illustrates the altitude 0, which corresponds to the channel of the MOS. The height of diffused layers can be negative, for P++ EPI layer for example.

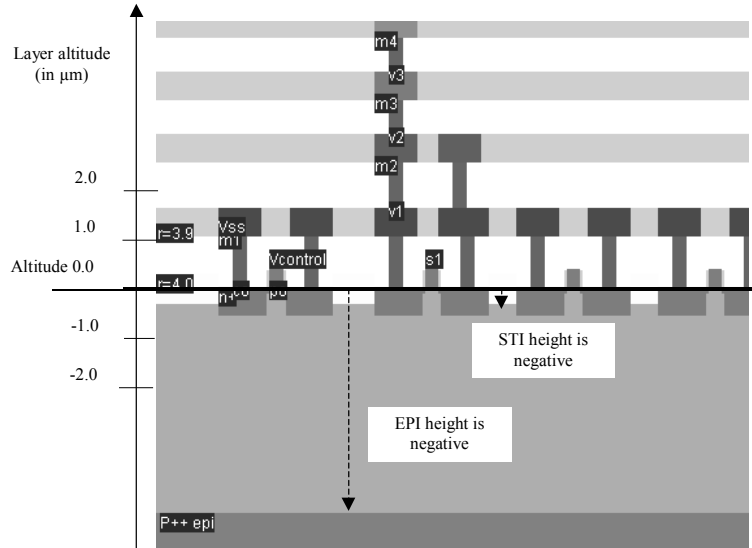


Figure 13-5: Description of the 2D aspect of the CMOS technology

LAYER	DESCRIPTION	PARAMETERS
EPI	Buried layer made of P++ used to create a good ground reference underneath the active area.	HEEPI for height (negative in respect to the origin) THEPI for thickness
STI	Shallow trench isolation used to separate the active areas.	HESTI for height THSTI for thickness
PASSIVATION	Upper SiO2 oxide on the top of the last metal layer	HEPASS for height THPASS for thickness
NITRIDE	Final oxide on the top of the passivation, usually Si3N4.	HENIT for height THNIT for thickness
NISO	Buried N- layer to isolate the Pwell underneath the nMOS devices, to enable forward bias and back bias	HENBURRIED for height THNBURRIED for thickness

13.28 Resistance Extraction

NAME	DESCRIPTION	VALUE (Ω)
RePo	Resistance per square for polysilicon	4
RePu	Resistance per square for unsalicide polysilicon	40
ReP2	Resistance per square for polysilicon2	4
ReDn	Resistance per square for n-diffusion	100
ReDp	Resistance per square for p-diffusion	100
ReMe	Resistance per square for metal	0.05
ReM2	Resistance per square for metal 2 (up to 6)	0.05
ReCo	Resistance for one contact	20
ReVi	Resistance for one via (up to via5)	2

13.29 Dielectrics

Some options are built in Microwind to enable specific features of ultra deep submicron technology. Details are provided in the table below.

CODE	DESCRIPTION	EXAMPLE VALUE
HIGHK	Oxide for interconnects (SiO2)	4.1
GATEK	Gate oxide	4.1
LOWK	Inter-metal oxide	3.0
LK11	Inter-metal1 oxide	3.0
LK22	Inter-metal2 oxide (up to LK66)	3.0
LK21	Metal2-Metal1 oxide	3.0
LK32	Metal3-Metal2 oxide (up to LK65)	3.0
TOX	Normal MOS gate oxide thickness	0.004 μm (40 Å)
HVTOX	High voltage gate oxide thickness	0.007 μm (70 Å)

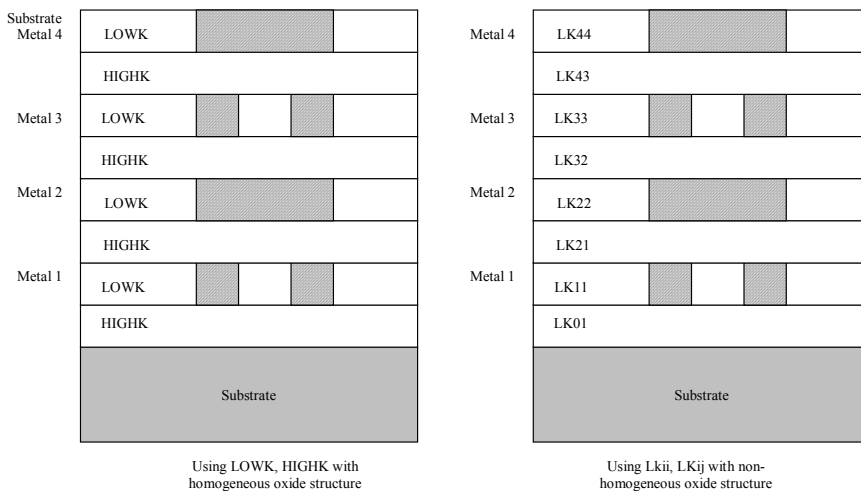


Fig. 13-6: Illustration of the use of LOWK, HIGHK dielectric constants (left figure) or detailed permittivity for each layer (right figure)

13.30 Simulation Parameters

The following list of parameters is used in Microwind2 to configure the simulation.

CODE	DESCRIPTION	TYPICAL VALUE
VDD	Supply voltage of the chip	2.0 V
HVDD	High voltage supply	3.3V
DELTA T	Simulator minimum time step to ensure convergence. You may increase this value to speed up the simulation but instability problems may rise.	0.5e-12 s
TEMPERATURE	Operating temperature of the chip	25 °C

13.31 Models Level1 and Level3 for analog simulation

Four types of MOS devices may be described (Data from SIA, 0.12 μ m CMOS technology). In the rule file, the keyword "MOS1", "MOS2", "MOS3" and "MOS4" are used to declare the device names appearing in menus. In 0.12 μ m technology, three types of MOS devices are declared as follows. Also, NMOS & PMOS keywords are used to select n-channel Mos or p-channel Mos device parameters.

Parameter	MOS1	MOS2	MOS3
Default name	High Speed	Low Leakage	High voltage
Vt (nmos)	0.3	0.5	0.7
Vt (pmos)	-0.3	-0.5	-0.7
KP (nmos)	300	300	200
KP (pmos)	150	150	100

```
* MOS definition
*
MOS1 low leakage
MOS2 high speed
MOS3 high voltage
```

Figure 13-8: Description of MOS options in 0.12 μ m technology (cmos012.RUL)

The list of parameters for level 1 and level 3 is given below:

Parameter	Keyword	Definition	Typical Value 0.25 μ m	
			NMOS	pMOS
VTO	l3vto	Threshold voltage	0.4V	-0.4V
U0	l3u0	Low field mobility	0.06 m ² /V.s	0.025 m ² /V.s
PHI	l3phi	Surface potential at strong inversion	0.3V	0.3V
LD	l3ld	Lateral diffusion into channel	0.01 μ m	0.01 μ m
GAMMA	l3gamma	Bulk threshold parameter	0.4 V ^{0.5}	0.4 V ^{0.5}
KAPPA	l3kappa	Saturation field factor	0.01 V ⁻¹	0.01 V ⁻¹
VMAX	l3vmax	Maximum drift velocity	150Km/s	100Km/s
THETA	l3theta	Mobility degradation factor	0.3 V ⁻¹	0.3 V ⁻¹
NSS	l3nss	Sub-threshold factor	0.07 V ⁻¹	0.07 V ⁻¹
TOX	l3tox	Gate oxide thickness	3nm	3nm
CGSO	L3cgs	Gate to Source lineic capacitance	100.0pF/m	100.0pF/m
CGDO	L3cgd	Gate to drain overlap capacitance	100.0pF/m	100.0pF/m
CGBO	L3cb	Gate to bulk overlap capacitance	1e-10F/m	1e-10F/m
CJSW	L3cj	Side-wall source & drain capacitance	1e-10F/m	1e-10F/m

For MOS2, MOS3 and MOS4, only the threshold voltage, mobility and oxides thickness are user-accessible. All other parameters are identical to MOS1.

Parameter	Keyword	Definition	Typical Value 0.25 μ m	
			NMOS	pMOS
VTO Mos2	l3v2to	Threshold voltage for MOS2	0.5V	-0.5V
VTO Mos3	l3v3to	Threshold voltage for MOS3	0.7V	-0.7V
U0 Mos2	l3u2	Mobility for MOS2	0.06	0.025
U0 Mos3	l3u3	Mobility for MOS3	0.06	0.025
TOX Mos 2	l3t2ox	Thin oxide thickness for MOS2	3nm	3nm
TOX Mos 3	l3t3ox	Thin oxide thickness for MOS3	7nm	7nm

13.32 BSIM4 Model for analog simulation

The list of parameters for BSIM4 is given below:

Parameter	Keyword	Description	NMOS value in 0.12 μ m	PMOS value in 0.12 μ m
VTHO	b4vtho	Long channel threshold voltage at V _{bs} = 0V	0.3V	0.3V
VFB	b4vfb	Flat-band voltage	-0.9	-0.9
K1	b4k1	First-order body bias coefficient	0.45 V ^{1/2}	0.45 V ^{1/2}
K2	b4k2	Second-order body bias coefficient	0.1	0.1
DVT0	b4d0vt	First coefficient of short-channel effect on threshold voltage	2.2	2.2
DVT1	b4d1vt	Second coefficient of short-channel effect on V _{th}	0.53	0.53
ETA0	b4et	Drain induced barrier lowering coefficient	0.08	0.08
NFACTOR	B4nf	Sub-threshold turn-on swing factor. Controls the exponential increase of current with V _{gs} .	1	1
U0	b4u0	Low-field mobility	0.060 m ² /Vs	0.025 m ² /Vs
UA	b4ua	Coefficient of first-order mobility degradation due to vertical field	11.0e-15 m/V	11.0e-15 m/V
UC	b4uc	Coefficient of mobility degradation due to body-bias effect	-0.04650e-15 V ⁻¹	-0.04650e-15 V ⁻¹
VSAT	b4vsat	Saturation velocity	8.0e4 m/s	8.0e4 m/s
WINT	b4wint	Channel-width offset parameter	0.01 ^{e-6} μ m	0.01 ^{e-6} μ m
LINT	b4lint	Channel-length offset parameter	0.01 ^{e-6} μ m	0.01 ^{e-6} μ m
PSCBE1	b4pscbe1	First substrate current induced body-effect mobility reduction	4.24e8 V/m	4.24e8 V/m
PSCBE2	b4pscbe2	Second substrate current induced body-effect mobility	4.24e8 V/m	4.24e8 V/m

		reduction		
KT1	b4kt1	Temperature coefficient of the threshold voltage.	-0.1V	-0.1V
UTE	b4ute	Temperature coefficient for the zero-field mobility U0.	-1.5	-1.5
VOFF	b4voff	Offset voltage in subthreshold region.	-0.08V	-0.08V
PCLM	b4pclm	Parameter for channel length modulation	1.2	1.2
TOXE	b4toxe	Gate oxide thickness	3.5e-9m	3.5e-9m
NDEP	b4ndep		0.54	0.54
XJ	b4xj	Junction depth	1.5e-7	1.5e-7

For MOS2, MOS3 and MOS4, only the threshold voltage, mobility and oxides thickness are user-accessible. All other parameters are identical to MOS1.

13.33 Technology files for DSCH2

The logic simulator includes a current evaluator. To run this evaluation, the following parameters are proposed in a TEC file (example: cmos012.TEC):

```

DSCH 2.0 - technology file
NAME "CMOS 0.12um"
VERSION 14.12.2001
* Time unit for simulation
TIMEUNIT = 0.01
* Supply voltage
VDD = 1.2
* Typical gate delay in ns
TDelay = 0.02
* Typical wire delay in ns
TWireDelay = 0.07
* Typical current in mA
TCurrent = 0.5
* Default MOS length and width
ML = "0.12u"
MNW = "1.0u"
MPW = "2.0u"

```

13.34 Design Rule File

The default design rule file used by Microwind2 corresponds to a CMOS 0.12 μ m technology. All its parameters are listed below.

```

MICROWIND 2.0
*
* Rule File for CMOS 0.18um
* Date : 18 May 98 by Etienne Sicard
* Date : 27 April 99 By Etienne/Fabrice
* 16 May 99 r603 dist via/contact
* 23 Jun 99 KOR mm9
* 04 Jan 00 smaller dT
* 19 Feb 00 STI, Niso, LL, high VT,
LIL
*
* status : preliminary
*
NAME CMOS 0.18um - 6 Metal
*
lambda = 0.1 (Lambda is set to half the
gate size)
metalLayers = 6 (Number of metal layers :
6)
lowK = 4.0 (inter-metal oxide)

```

```

lil = 1          (local interconnect layer
l=enable, 0= disable)
tox = 0.004     (fast MOS oxide in µm
0.0=disable)
hvtox= 0.007   (high voltage MOS oxide)
salicide = 0   (Enable salicide l=enable
0= disable)
*
* Design rules associated to each layer
*
* Well (Gds2 level 1)
r101 = 10      (well width)
r102 = 11      (well spacing)
*
* Diffusion (N+ 16, P+ 17, active 2)
*
r201 = 4       (diffusion width)
r202 = 4       (diffusion spacing)
r203 = 6       (border of nwell on diffp)
r204 = 6       (nwell to next diffn)
*
* Poly (13)
*
r301 = 2       (poly width)
r302 = 2       (ngate width)
r303 = 2       (pgate width)
r304 = 3       (poly spacing)
r305 = 1       (spacing poly and unrelated
diff)
r306 = 4       (width of drain and source
diff)
r307 = 2       (extra gate poly)
* Contact (19)
r401 = 2       (contact width)
r402 = 3       (contact spacing)
r403 = 2       (metal border for contact)
r404 = 2       (poly border for contact)
r405 = 2       (diff border for contact)

```

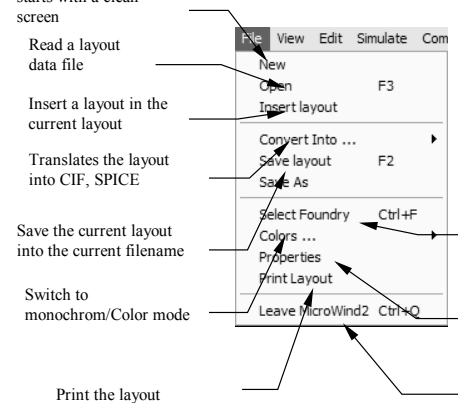
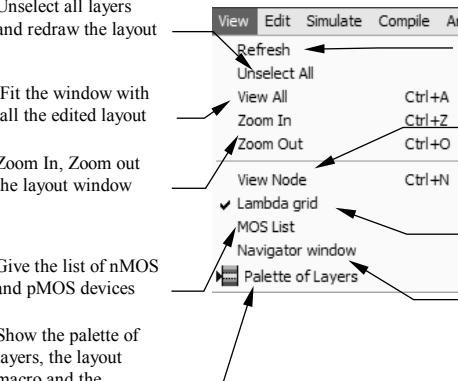
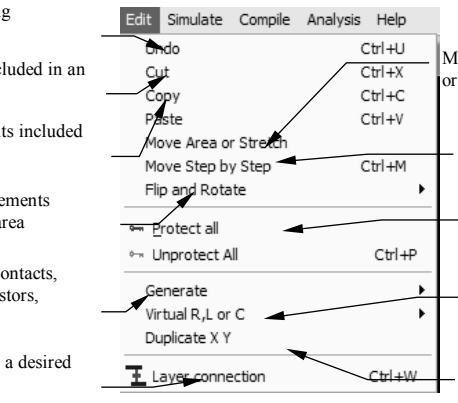
```


* metal (23)
r501 = 3       (metal width)
r502 = 4       (metal spacing)
* via (25)
r601 = 3       (Via width)
r602 = 4       (Spacing)
r603 = 0       (via/contact)
r604 = 2       (border of metal&metal2)
* metal 2 (27)
r701 = 3       (Metal 2 width)
r702 = 4       (spacing)
* via 2 (32)
r801 = 3       (Via width)
r802 = 4       (Spacing)
r804 = 2       (border of metal2&metal3)
* metal 3 (34)
r901 = 3       (width)
r902 = 4       (spacing)
* via 3 (35)
ra01 = 3       (Via width)
ra02 = 4       (Spacing)
ra04 = 2       (border of metal3&metal4)
* metal 4 (36)
rb01 = 3       (width)
rb02 = 4       (spacing)
* via 4 (52)
rc01 = 3       (Via width)
rc02 = 4       (Spacing)
rc04 = 2       (border of metal4&metal5)
* metal 5 (53)
rd01 = 8       (width)

```

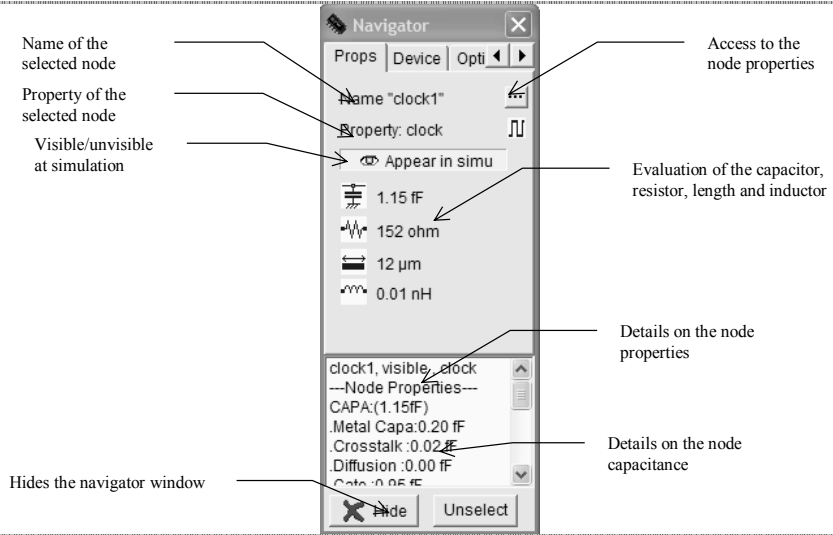
<section>

14 Microwind2 Menus





















<p>14.1 FILE MENU</p>	 <p>Reset the program and starts with a clean screen</p> <p>Read a layout data file</p> <p>Insert a layout in the current layout</p> <p>Translates the layout into CIF, SPICE</p> <p>Save the current layout into the current filename</p> <p>Switch to monochrom/Color mode</p> <p>Print the layout</p> <p>Configure Microwind2 to a foundry</p> <p>Layout properties : number of box, devices, size, etc...</p> <p>Quit Microwind2 and returns to Windows</p>
<p>14.2 VIEW MENU</p>	 <p>Unselect all layers and redraw the layout</p> <p>Fit the window with all the edited layout</p> <p>Zoom In, Zoom out the layout window</p> <p>Give the list of nMOS and pMOS devices</p> <p>Show the palette of layers, the layout macro and the simulation properties</p> <p>Redraw the screen</p> <p>Extract the electrical node starting at the cursor location</p> <p>Show/Hide the lambda grid</p> <p>Show the navigator window to display the node properties</p>
<p>14.3 EDIT MENU</p>	 <p>Cancel last editing command</p> <p>Cut elements included in an area</p> <p>Duplicate elements included in an area</p> <p>Flip or rotate elements included in an area</p> <p>Generate MOS, contacts, pads, diodes, resistors, capacitors, etc...</p> <p>Connect layers at a desired location</p> <p>Move elements included in an area or stretch the selected box border</p> <p>Move step by step a selection of elements</p> <p>Protect and unprotect layers from copying, moving, erasing</p> <p>Add a virtual R,L,C for simulation purpose</p> <p>Duplicate in X and Y a selection of elements</p>

<p>14.4 SIMULATE MENU</p>	<p>Run the simulation and choose the appropriate mode V(t), I(t), V/V, F(t), etc</p> <p>Simulate directly on the layout, with a palette of colors representing voltage</p> <p>Include crosstalk effects in simulation</p> <p>View the process steps of the layout fabrication in 3D</p>
<p>14.5 COMPILE MENU</p>	<p>Compile one single line (on-line)</p>
<p>14.6 ANALYSIS MENU</p>	<p>Verifies the layout and highlight the design rule violations</p> <p>Measure the distance in the layout window, in μm and lambda</p>
<p>14.7 PALETTE</p> 	

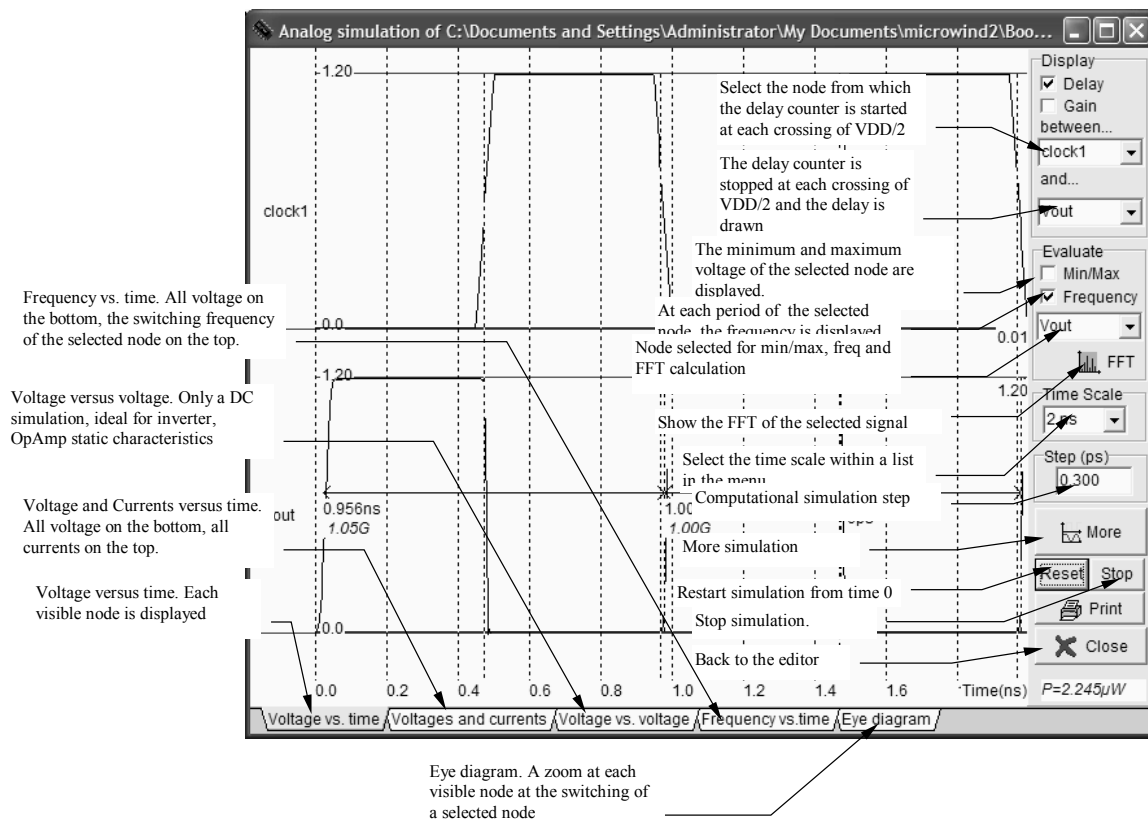
14.8 NAVIGATOR WINDOW



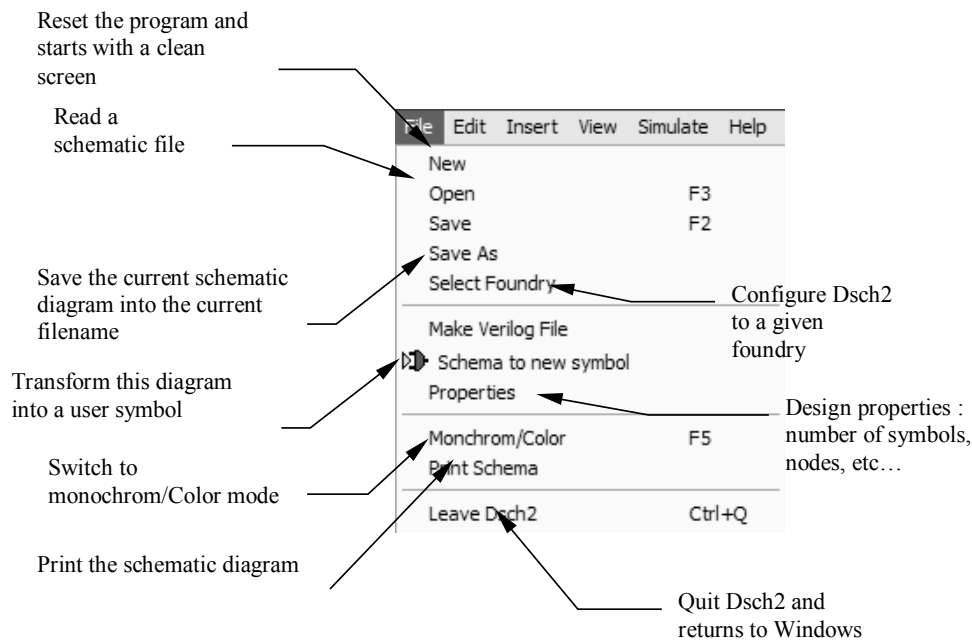
14.9 LIST OF ICONS

	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

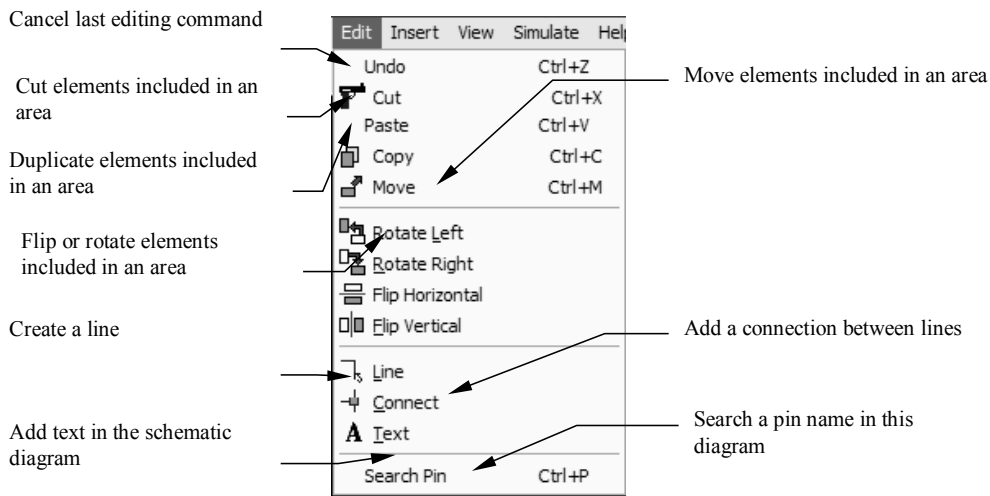
14.10 Microwind2 Simulation menu



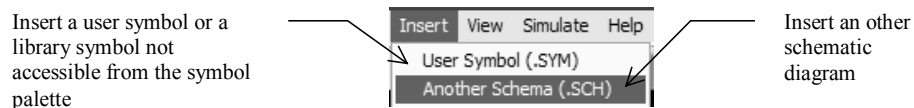
14.11 Dsch2 Menus



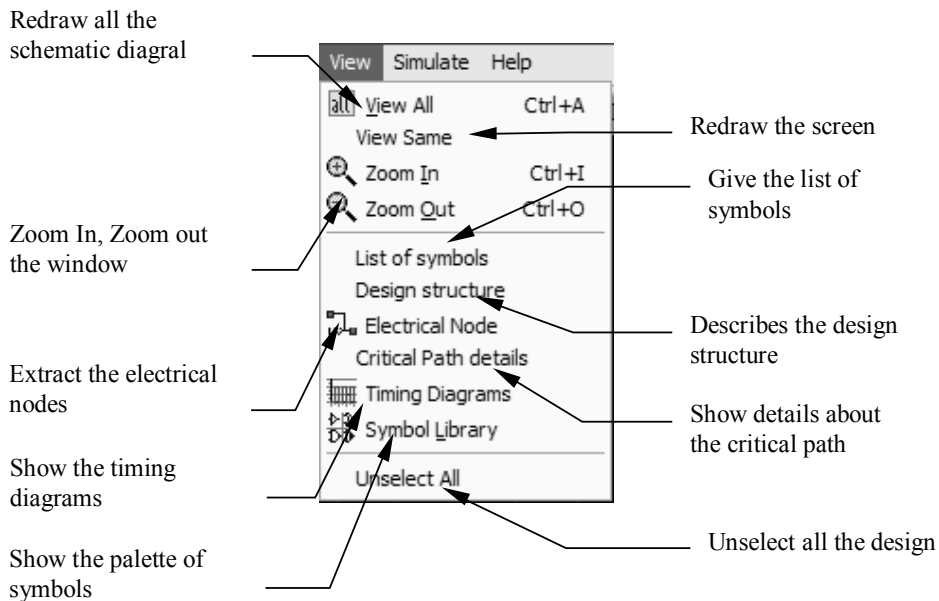
14.12 Edit Menu



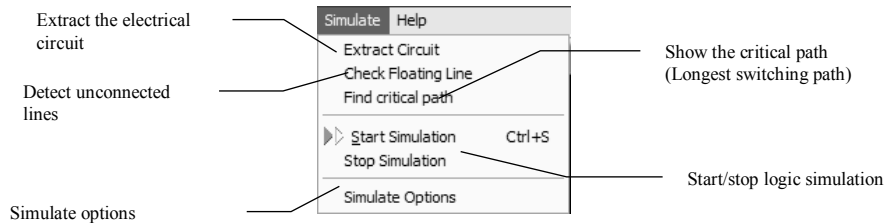
14.13 Insert Menu



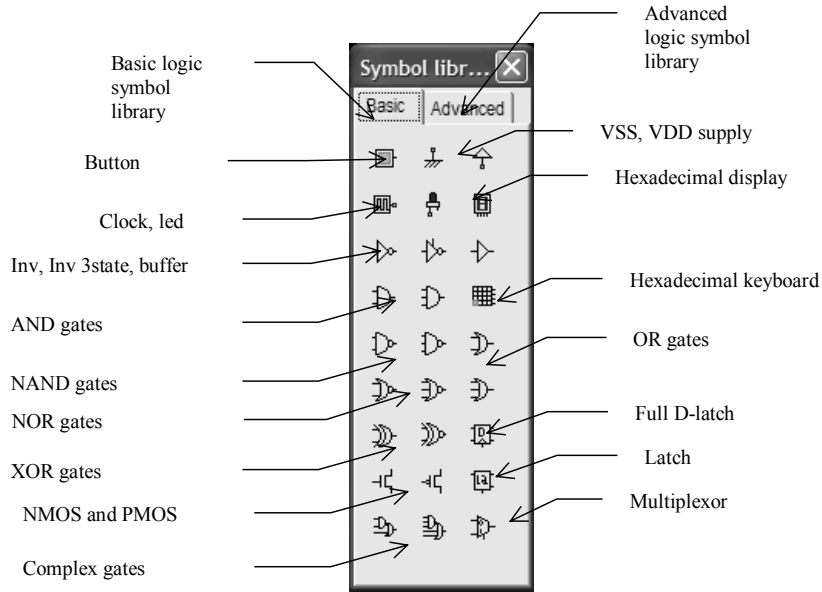
14.14 View Menu



14.15 Simulate Menu



14.16 Symbol Palette



14.17 List of Files

FILE	DESCRIPTION
MICROWIND2.EXE	Microwind2 executable file
DSCH2.EXE	Dsch2 executable file
MICROWIND2.HTML DSCH2.HTML	Help manuals for Microwind2 and Dsch2
*.RUL	TECHNOLOGY FILES. The MICROWIND2 program reads the rule file to update the simulator parameters, the design rules and parasitic capacitor values. A detailed description of the .RUL file is reported at the end of Appendix A.
*.MSK	LAYOUT FILES. The MICROWIND2 software creates data files with the appendix .MSK. Those files are simple text files containing the list of boxes and layers, and the list of text declarations.
*.CIR	The command File -> Make SPICE File generates a SPICE compatible text description.
*.MES	MOS I/V Measurements
*.TXT	Verilog text inputs
*.TEC	TECHNOLOGY FILES. The DSCH2 program reads the rule file to update the simulator parameters. A detailed description of the .TEC file is reported at the end of Appendix A.
*.SCH	Schematic diagram created by Dsch2
*.SYM	Symbols generated and used by Dsch2

14.18 List of Measurement Files

In the package, a selection of measurements are proposed, for comparison between real case measurements and models. Four chips have been fabricated and measured in our laboratory:

Chip « a » (0.35 μ m ST)

Chip « b » (0.25 μ m ST)

Chip « c » (0.18 μ m ST)

Chip "d" (0.8 μ m ATMEL-ES2)

Measurement files	Description
0.35 μ m CMOS Na10x0,4.mes Na10x10.mes Na10x2.mes Na1x0,4.mes Na80x0,4.mes Pa10x0,4.mes Pa10x10.mes Pa10x2.mes Pa1x0,4.mes Pa80x0,4.mes	Nmos W=10 μ m, L=0.4 μ m (0.35 effective) Nmos W=10 μ m, L= 10 μ m Nmos W=10 μ m, L= 2 μ m Nmos W=1 μ m, L= 0.4 μ m Nmos W=80 μ m, L= 0.4 μ m Pmos W=10 μ m, L= 0.4 μ m Pmos W=10 μ m, L= 10 μ m Pmos W=10 μ m, L= 2 μ m Pmos W=1 μ m, L= 0.4 μ m Pmos W=80 μ m, L= 0.4 μ m
0.25 μ m CMOS Nb10x0,25.mes Nb10x10.mes	Nmos W=10 μ m, L=0.25 μ m Nmos W=10 μ m, L=10 μ m
0.18 μ m CMOS Nc10x10.mes NcHS4x0.2.mes NcHV4x0.2.mes Nc4x0.2.mes	Nmos W=10 μ m, L=10 μ m Nmos W=4 μ m, L=0.2 μ m high speed option Nmos W=4 μ m, L=0.2 μ m high voltage option Nmos W=4 μ m, L=0.2 μ m normal (low leakage)
0.8 μ m CMOS Nd20x20.mes Nd20x0,8.mes	Nmos W=20 μ m, L=20 μ m Nmos W=20 μ m, L=0.8 μ m

14.19 Measurement file example

```
Measure v3.0 - 4 May 00
LL 10x10um cmos018
NMOS 10.0 10.0
IDVd 5 0.0 2.0 0.5
41 0
0.0 0.0 0.0 0.0 0.0 0.0

5.0000E-02 2.0226E-12 6.7235E-07 6.4727E-06 1.1584E-05 1.5635E-05
1.0000E-01 2.3586E-12 7.7469E-07 1.2143E-05 2.2459E-05 3.0636E-05
1.5000E-01 2.4540E-12 7.8616E-07 1.7012E-05 3.2623E-05 4.5003E-05
2.0000E-01 2.5151E-12 7.8892E-07 2.1089E-05 4.2079E-05 5.8736E-05
2.5000E-01 2.5716E-12 7.9037E-07 2.4386E-05 5.0826E-05 7.1835E-05
3.0000E-01 2.6275E-12 7.9158E-07 2.6929E-05 5.8867E-05 8.4301E-05
3.5000E-01 2.6834E-12 7.9273E-07 2.8768E-05 6.6202E-05 9.6133E-05
4.0000E-01 2.7393E-12 7.9387E-07 2.9992E-05 7.2834E-05 1.0733E-04
```



```
...
2.0000E+00  4.5477E-12  8.3166E-07  3.2021E-05  1.0562E-04  2.0746E-04
IdVg  4      0.0    -1.5   -0.5   0.05
21      0
0.0    2.0226E-12  5.7123E-13  1.0630E-12  1.5644E-12
1.0000E-01  3.4083E-11  7.6465E-13  1.0653E-12  1.5644E-12
2.0000E-01  5.7669E-10  4.6877E-12  1.1181E-12  1.5655E-12
3.0000E-01  8.9864E-09  8.3526E-11  2.2992E-12  1.5922E-12
...
1.9000E+00  1.4895E-05  1.3132E-05  1.1692E-05  1.0455E-05
2.0000E+00  1.5635E-05  1.3899E-05  1.2479E-05  1.1259E-05
```

15 References

- [1] R.J. Bakker, H. W. Li, D. E. Boyce "CMOS design, layout and simulation", IEEE Press, 1998, www.ieee.org
- [2] M. John, S. Smith, "Application Specific Integrated Circuits", Addison Wesley, 1997, ISBN 0-201-50022-1, www.awl.com/cseng
- [3] B. Razavi "Design of Analog CMOS integrated circuits", McGraw Hill, ISBN 0-07-238032-2, 2001, www.mhhe.com
- [4] Y. P. Tsividis "Operating and Modeling of the MOS transistor", McGraw-Hill, 1987, ISBN 0-07-065381-X
- [5] S. M. Sze "Physics of Semiconductor devices", John-Wiley, 1981, ISBN 0-471-05661-8
- [6] Y. Cheng, C. Hu "MOSFET Modeling & BSIM3 user's guide", Kluwer Academic Publishers, 1999
- [7] A. Hastings "The Art of Analog Layout", Prentice-Hall, 2001, ISBN 0-13-087061-7
- [8] N. Weste, K. Eshraghian "Principles of CMOS VLSI design", Addison Wesley, ISBN 0-201-53376-6, 1993
- [9] K. Lee, M. Shur, T.A. Fjeldly, T. Ytterdal "Semiconductor Device Modeling for VLSI", Prentice Hall, 1993, ISBN 0-13-805656-0
- [10] W. Liu "MOSFET Models for SPICE simulation including Bsim3v3 and BSIM4", Wiley & Sons, 2001, ISBN 0-471-39697-4
- [11] C. Motchenbacher, J. A. Connelly "Low Noise Electronic System Design", Wiley & sons, 1993, ISBN 0-471-57742-1
- [12] A.K. Sharma "Semiconductor Memories", IEEE Press, 1996, ISBN 0-7803-1000-4