

*Deep-submicron*

# ***CMOS circuit design***

## ***Simulator in hands***

Etienne Sicard  
Sonia Delmas Bendhia

Version December 2003

This book is under consideration for publication by

Brooks/Cole Publishing Company  
3450 South 3650 East Street  
Salt Lake City, Utah 84109, USA  
[www.brookscole.com](http://www.brookscole.com)

(Contact: [Bill.Stenquist@wadsworth.com](mailto:Bill.Stenquist@wadsworth.com))

## Acknowledgements

We would like our early colleagues Jean-Francois Habigand, Kozo Kinoshita, Antonio Rubio for their support throughout the development of the Microwind, Dsch tools. The project of writing a book that seemed initially to be shadowy took form and substance, and led to this present work. We would like to thank Joseph-Georges Ferrante for having faith in our ability to drive ambitious microelectronics research projects, and having provided us a continuous support over the last ten years. Productive technical discussions with Jean-Pierre Schoellkopf, Amaury Soubeyran, Thomas Steinecke, Gert Voland and Jean-Louis Noullet are also gratefully acknowledged.

Special thanks are due to technical contributors to the Dsch and Microwind software (Chen Xi, Jianwen Huang), to our colleagues at INSA who always supported this work, to numerous professors, students and engineers who patiently debugged the technical contents of the book and the software, and gave valuable comments and suggestions. Also, we would like to thank Marie-Agnes Detourbe for having carefully reviewed the manuscript.

Finally we would like to acknowledge our biggest debt to our parents and to our companion for their constant support.

## About the authors



[etienne.sicard@insa-tlse.fr](mailto:etienne.sicard@insa-tlse.fr)

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 scholarship from the Japanese Ministry of Education 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 professor at the INSA Electronic Engineering School of Toulouse. His research interests include several aspects of integrated circuit design including crosstalk fault tolerance, and electromagnetic compatibility of integrated circuits. Etienne is the author of several educational software in the field of microelectronics and sound processing.



[sonia.bendhia@insa-tlse.fr](mailto:sonia.bendhia@insa-tlse.fr)

Sonia DELMAS BENDHIA was born in Toulouse, April 1972, She received an engineering diploma in 1995, and the Ph.D. in Electronic Design from the National Institute of Applied Sciences, Toulouse, France, in 1998. Sonia Bendhia is currently a senior lecturer in the INSA of Toulouse, Department of Electrical and Computer Engineering. Her research interests include signal integrity in deep sub-micron CMOS Ics, analog design and electromagnetic compatibility of systems . Sonia is the author of technical papers concerning signal integrity and EMC.

### About Microwind and Dsch

The present book introduces the design and simulation of CMOS integrated circuits, and makes an extensive use of PC tools Microwind2 and Dsch2. These tools are freeware.

The web link is <http://www.microwind.org>

### In memory...

*In memory of John Uyemura*

# Contents

Chapter	Page
1	<i>Introduction</i> Technology scale down Frequency Improvement Increased layers Reduced power supply
2	<i>The MOS device</i> The MOS Logic simulation of the MOS MOS layout Vertical aspect of the MOS Static MOS characteristics Dynamic MOS behavior Analog simulation Mos options Transmission gate: the perfect switch Layout considerations
3	<i>MOS modeling</i> The MOS model 1 The MOS model 3 The model BSIM4 Temperature effects on the MOS High frequency behavior of the MOS
4	<i>The Inverter</i> The logic Inverter The CMOS inverter (Power, supply, frequency) Layout design (plasma, latchup) Simulation of the inverter Views of the process Buffer 3-state inverter Analog behavior of the inverter Ring oscillator Temperature effects

5	<i>Interconnects</i>
	Signal propagation
	Capacitance load
	Resistance effect
	Inductance effect
	Buffers
	Clock tree
	Supply routing
6	<i>Basic Gates</i>
	Introduction
	From boolean expression to layout
	NAND gate (micron, sub-micron)
	OR3 gate
	XOR
	Complex gates
	Multiplexors (Mux-demux)
	Pulse generator
7	<i>Arithmetics</i>
	Data formats: unsigned, signed fixed
	Half adder gate
	Full adder gate
	4-bit adder
	Comparator
	Multiplier
	ALU
	Low power arithmetics
8	<i>Latches</i>
	RS latch
	D-Latch
	Edge-triggered latch
	Latch optimization (conso, speed, fanout)
	Counter
	Project: programmable pulse generator
9	<i>FPGA</i>
	Goals
	Mux for FPGA
	Configurable logic block
	Look-up table
	Interconnection
	Programmable Interconnection Points
	Propagation delay
10	<i>MEMORIES</i>
	The world of Memories
	Static RAM memory (4T, 6T)
	Decoder (low power)
	Dynamic RAM memory
	Embedded RAM
	Sense ampli
	ROM memory
	EEPROM memory
	FRAM memory
11	<i>Analog Cells</i>

	Diode connected MOS
	Voltage reference
	Current Mirror
	Amplifiers (Class)
	Voltage regulator
	Wide range amplifier
	Charge pump
	Noise
12	<i>RF Analog Cells</i>
	Oscillators
	Inductors
	Sample & Hold
	Mixers
	Voltage-controlled Oscillators
	PLL project
	Power amplifiers
13	<i>Converters</i>
	Introduction
	Converter parameters
	Sample hold
	ADC
	DAC
14	<i>Input/Output Interfacing</i>
	Level shifter
	Pad structure
	Input pad (schmidt, protect, buffer)
	Output pad (log, analog, multi drive)
	Pad ring
	Packages
	IBIS
	LVDS
	High performance I/Os
15	<i>SOI</i>
	Layout improvements
	2D aspects
	SOI model
	Simulation
	Issues
16	<i>Future &amp; Conclusion</i>
Appendix A	<i>Design rules</i>
Appendix B	<i>List of commands Microwind</i>
Appendix C	<i>List of commands Dsch</i>
Appendix D	<i>Quick Reference Sheet Microwind-Dsch</i>
Appendix E	<i>CMOS technology reference Sheet</i>
	0.8 $\mu$ m
	0.6 $\mu$ m
	0.35 $\mu$ m
	0.25 $\mu$ m
	0.18 $\mu$ m
	0.12 $\mu$ m
	90nm
Appendix F	<i>Answer to exercises</i>

## MULTIPLIERS

<i>Value</i>	<i>Name</i>	<i>Standard Notation</i>
$10^{18}$	<b>PETA</b>	P
$10^{15}$	<b>EXA</b>	E
$10^{12}$	<b>TERA</b>	T
$10^9$	<b>GIGA</b>	G
$10^6$	<b>MEGA</b>	M
$10^3$	<b>KILO</b>	K
$10^0$	-	-
$10^{-3}$	<b>MILLI</b>	m
$10^{-6}$	<b>MICRO</b>	u
$10^{-9}$	<b>NANO</b>	n
$10^{-12}$	<b>PICO</b>	p
$10^{-15}$	<b>FEMTO</b>	f
$10^{-18}$	<b>ATTO</b>	a
$10^{-21}$	<b>ZEPTO</b>	z

## PHYSICAL CONSTANTS & PARAMETERS

&lt;verify all &gt;

<i>Name</i>	<i>Value</i>	<i>Description</i>
$\epsilon_0$	$8.85 \times 10^{-12}$ Farad/m	Vacuum dielectric constant
$\epsilon_r \text{ SiO}_2$	3.9 - 4.2	Relative dielectric constant of $\text{SiO}_2$
$\epsilon_r \text{ Si}$	11.8	Relative dielectric constant of silicon
$\epsilon_r \text{ ceramic}$	12	Relative dielectric constant of ceramic
k	$1.381 \times 10^{-23} \text{ J/K}$	Boltzmann's constant
q	$1.6 \times 10^{-19} \text{ Coulomb}$	Electron charge
$\mu_n$	$600 \text{ V.cm}^{-2}$	Mobility of electrons in silicon
$\mu_p$	$270 \text{ V.cm}^{-2}$	Mobility of holes in silicon
$\gamma_{al}$	$36.5 \times 10^6 \text{ S/m}$	Aluminum conductivity
$\gamma_{si}$	$4 \times 10^4 \text{ S/m}$	Silicon conductivity
$n_i$	$1.02 \times 10^{10} \text{ cm}^{-3}$	Intrinsic carrier concentration in silicon at 300K
$\rho_{al}$	$0.0277 \Omega.\mu\text{m}$	Aluminum resistivity
$\gamma_{cu}$	$58 \times 10^6 \text{ S/m}$	Copper conductivity
$\rho_{cu}$	$0.0172 \Omega.\mu\text{m}$	Copper resistivity
$\rho_{tungstène (W)}$	$0.0530 \Omega.\mu\text{m}$	Tungsten resistivity
$\rho_{or (Ag)}$	$0.0220 \Omega.\mu\text{m}$	Gold resistivity
$\mu_0$	$1.257 \times 10^{-6} \text{ H/m}$	Vacuum permeability
T	300°K (27°C)	Operating temperature

## Preface

The present book introduces the design and simulation of CMOS integrated circuits, in an attractive way thanks to user-friendly PC tool Microwind2 given in the companion CD-ROM of this book.

The chapters of this book have been summarized below. Chapter One describes the technology scale down and the major improvements allowed by deep sub-micron technologies. Chapter Two is dedicated to the presentation of the single MOS device, with details on simulation at logic and layout levels. The modeling of the MOS devices is introduced in Chapter Three. Chapter Four presents the CMOS Inverter, the 2D and 3D views, the comparative design in micron and deep-submicron technologies. Chapter Five deals specifically with interconnects, with information on the propagation delay and several parasitic effects. Chapter Six deals with the basic logic gates (AND, OR, XOR, complex gates), Chapter Seven the arithmetic functions (Adder, comparator, multiplier, ALU). The latches and counters are detailed in Chapter Eight, while Chapter Nine introduces the basic concepts of Field programmable Gate Arrays.

As for Chapter Ten, static, dynamic, non-volatile and magnetic memories are described. In Chapter Eleven, analog cells are presented, including voltage references, current mirrors, and the basic architecture of operational amplifiers. Chapter Twelve is dedicated to radio-frequency analog cells, with details on mixers, voltage-controlled oscillators, fast phase-lock-loops and power amplifiers. Chapter Thirteen focuses on analog-to-digital and digital to analog converter principles. The input/output interfacing principles are illustrated in Chapter Fourteen. The last chapter includes an introduction to silicon-insulator technology, before a prospective and a conclusion.

The detailed explanation of the design rules is in appendix A. The details of all commands are given in appendix B for the tool Microwind, and in appendix C for the tool Dsch. Appendix D includes a quick reference sheet for Microwind and Dsch, and Appendix E gives some abstract information about each technology generation, from  $0.7\mu\text{m}$  down to 90nm.

Sonia DELMAS-BENDHIA, Etienne SICARD

*Toulouse, Sept 2003*

# 1 Introduction

The evolution of integrated circuit (IC) fabrication techniques is a unique fact in the history of modern industry. There have been steady improvements in terms of speed, density and cost for more than 30 years. In this chapter, we present some information illustrating the technology scale down.

## 1. GENERAL TRENDS

Inside general purpose electronics systems such as personal computers or cellular phones, we may find numerous integrated circuits (IC), placed together with discrete components on a printed circuit board (PCB), as shown in figure 1-1. The integrated circuits appearing in this figure have various sizes and complexity. The main core consists of a microprocessor, considered as the heart of the system, that includes several millions of transistors on a single chip. The push for smaller size, reduced power supply consumption and enhancement of services, has resulted in continuous technological advances, with possibility for ever higher integration.

*Figure 1-1: Photograph of the internal parts of a cellular phone <Etienne: Or automotive>*

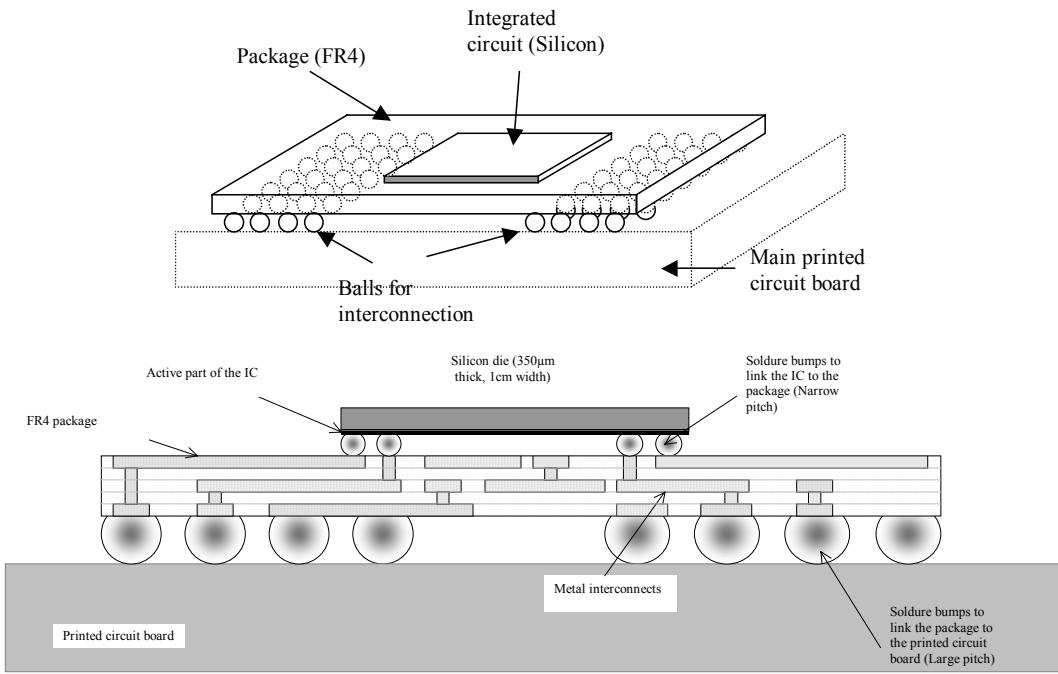
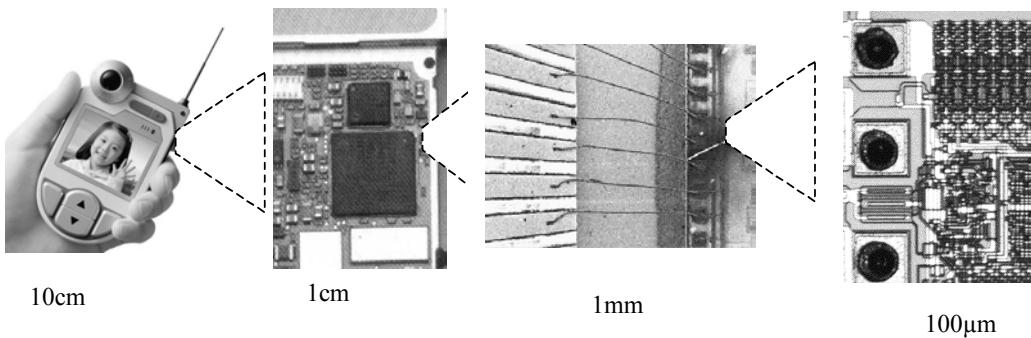


Figure 1-2: Typical structure of an integrated circuit

The integrated circuit consists of a silicon die <Glossary>, with a size usually around 1cmx1cm in the case of microprocessors and memories. The integrated circuit is mounted on a package (Figure 1-2), which is placed on a printed circuit board. The active part of the integrated circuit is only a very thin portion of the silicon die. At the border of the chip, small solder bumps serve as electrical connections between the integrated circuit and the package. The package itself is a sandwich of metal and insulator materials, that convey the electrical signals to large solder bumps, which interface with the printed circuit board.



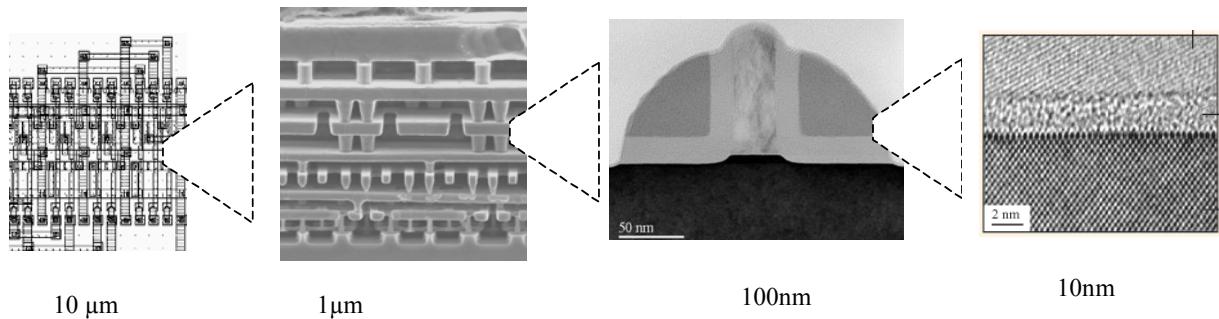


Figure 1-3: Patterns representative of each scale decade from 10cm to 10nm (Courtesy IBM, Fujitsu)

Around eight decades separate the user's equipment (Such as a mobile phone in figure 1-3) and the basic electrical phenomenon, consisting in the attraction of electrons through an oxide. Inside the electronic equipment, we may see integrated circuits and passive elements sharing the same printed circuit board (1 cm scale), wire connections between package and the die (1mm scale), input/output structures of the integrated circuit (100 $\mu$ m scale), the integrated circuit layout (10 $\mu$ m), a vertical cross-section of the process, revealing a complex stack of layers and insulators (1 $\mu$ m scale), the active device itself, called MOS transistor (which stands for Metal oxide semiconductor<glossary>).

Figure 1-4 describes the evolution of the complexity of Intel ® microprocessors in terms of number of devices on the chip [Intel]. The Pentium IV processor produced in 2003 included about 50,000,000 MOS devices integrated on a single piece of silicon no larger than 2x2 cm.

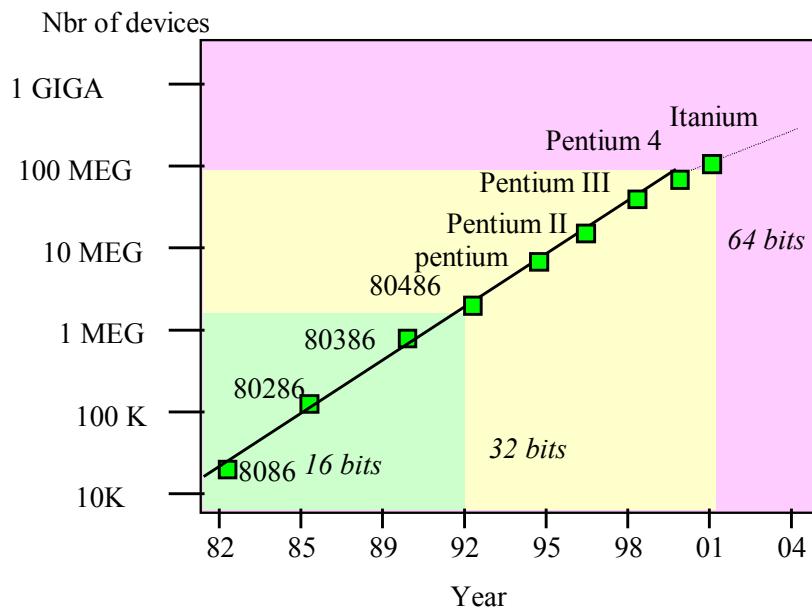


Figure 1-4: Evolution of microprocessors [Intel]

Since the 1 Kilo-byte (Kb) memory produced by Intel in 1971, semiconductor memories have improved both in density and performances, with the production of the 256 Mega-bit (Mb) dynamic memories (DRAM) in 2000, and 1Giga-bit (Gb) memories in 2004 (Figure 1-5). In other words, within around 30 years, the number of memory cells integrated on a single die has been increased by 1,000,000. An other type of memory chip called Flash memory has become very popular, due to its capabilities to retain the information without supply voltage (Non voltaile memories are described in chapter 9). According to the international technology roadmap for semiconductors [Itrs], the DRAM memory complexity is expected to increase up to 16 Giga-byte (Gb) in 2008.

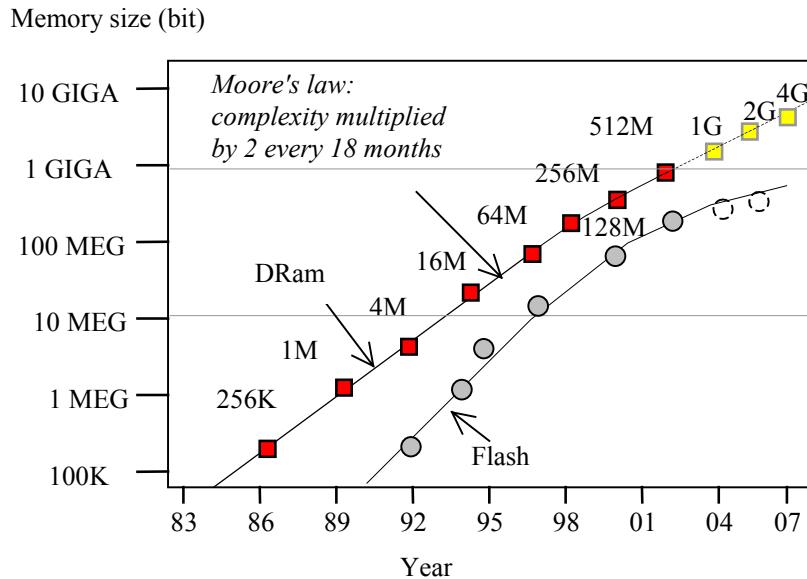


Fig. 1-5: Evolution of Dynamic RAM and Flash semiconductor memories [Itrs][Itoh01]

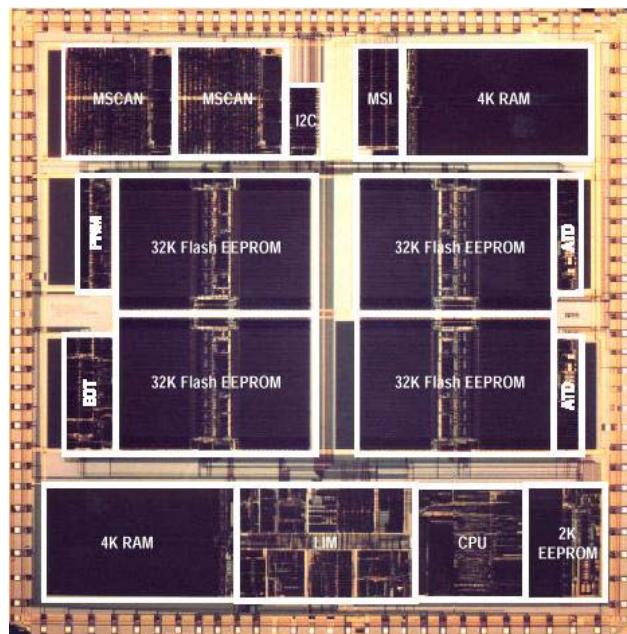


Figure 1-6: Bird's view of a micro-controller die (Courtesy of Motorola Semiconductors)

The layout aspect of the die of an industrial micro-controller is shown in figure 1-6 [Motorola]. This circuit is fabricated in several millions of samples for automotive applications. The micro-controller core is the central process unit (CPU), which uses several types of memory: the Electrically erasable Read-Only Memory (EEPROM), the FLASH memory (Rapidaly erasable Read-Only Memory) and the RAM memory (Random Access Memory). Some controllers are also embedded in the same die: the Control Area Network (MSCAN), the debug interface (MSI), and other functionnal cores (ATD, ETD <Etienne: ask for details to Motorola>).

## 2. THE DEVICE SCALE DOWN

We consider four main generations of integrated circuit technologies: micron, submicron, deep submicron and ultra deep submicron technologies., as illustrated in figure 1-7. The sub-micron era started in 1990 with the  $0.8\mu\text{m}$  technology. The deep submicron technology started in 1995 with the introduction of lithography better than  $0.3\mu\text{m}$ . Ultra deep submicron technology concerns lithography below  $0.1\mu\text{m}$ . In figure 1-7, it is shown that research has always kept around 5 years ahead of mass production. It can also be seen that the trend towards smaller dimensions has been accelerated since 1996. In 2007, the lithography is expected to decrease down to  $0.07\mu\text{m}$ . The lithography expressed in  $\mu\text{m}$  corresponds to the smallest patterns that can be implemented on the surface of the integrated circuit.

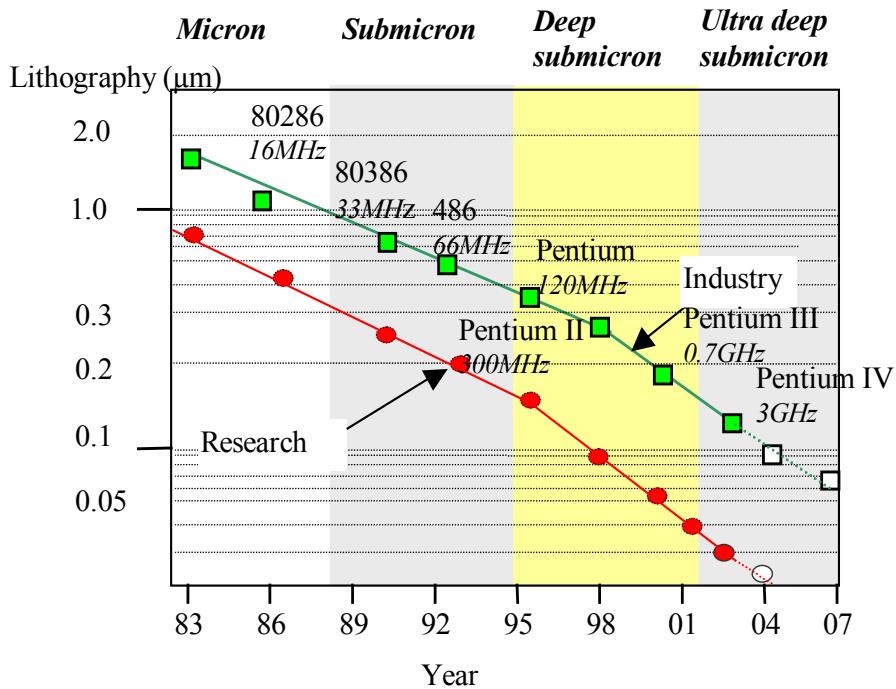


Figure 1-7: Evolution of lithography

### 3. FREQUENCY IMPROVEMENT

Figure 1-8 illustrates the clock frequency increase for high-performance microprocessors and industrial micro-controllers with the technology scale down. The microprocessor roadmap is based on Intel processors used for personal computers [Intel], while the micro-controllers roadmap is based on Motorola micro-controllers [Motorola] used for high performance automotive industry applications. The PC industry requires microprocessors running at the highest frequencies, which entails very high power consumption (30 Watts for the Pentium IV generation). The automotive industry requires embedded controllers with more and more sophisticated on-chip functionalities, larger embedded memories and interfacing protocols. The operating frequency follows a similar trend to that of PC processors, but with a significant shift.

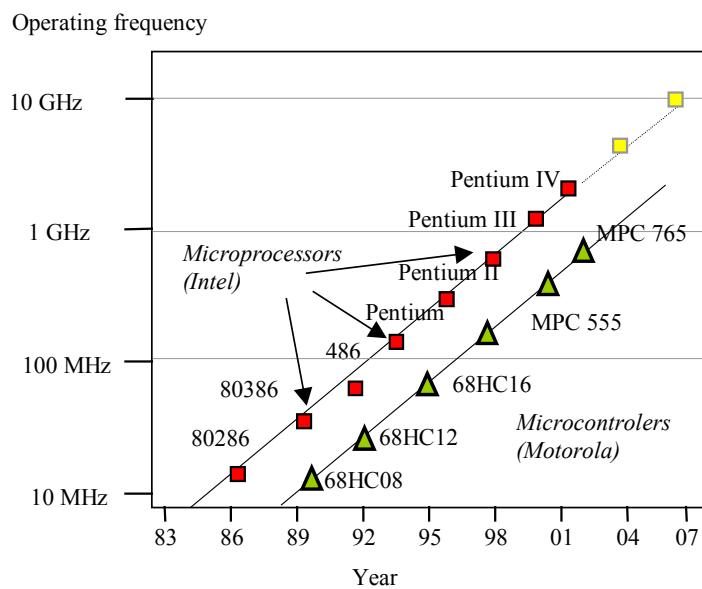


Figure 1-8: Increased operating frequency of microprocessors and micro-controllers

### 4. 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 increase of the size of the die 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

Table 1-1: Evolution of key parameters with the technology scale down [ITRS]

The 1.2µm CMOS process features n-channel and p-channel MOS devices with a minimum channel length of 0.8µm. The Microwind tool may be configured in CMOS 1.2µm technology using the command **File→Select Foundry**, and choosing **cmos12.rul** in the list. Metal interconnects are 2µm wide. The MOS diffusions are around 1µm deep. The two dimensional aspect of this technology is shown in figure 1-9.

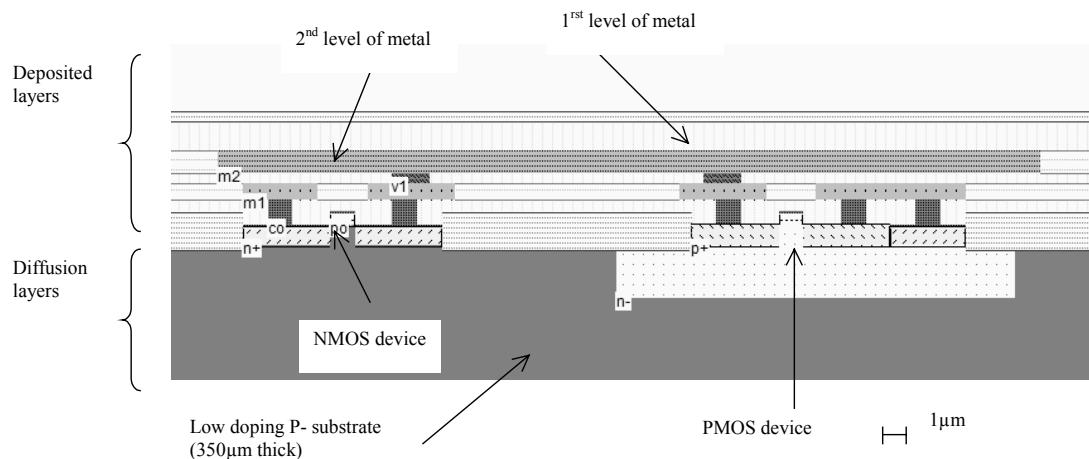


Figure 1-9: Cross-section of the 1.2µm CMOS technology (CMOS.MSK)

The 0.35µm CMOS technology is a five-metal layer process with a minimal MOS device length of 0.35µm. The MOS device includes lateral drain diffusions, with shallow trench oxide isolations. The Microwind tool may be configured in CMOS 0.35µm technology using the command **File→Select Foundry**, and choosing "cmos035.rul" in the list. Metal interconnects are less than 1µm wide. The MOS diffusions are less than 0.5µm deep. The two dimensional aspect of this technology is shown in figure 1-10, using the layout **INV3.MSK**.

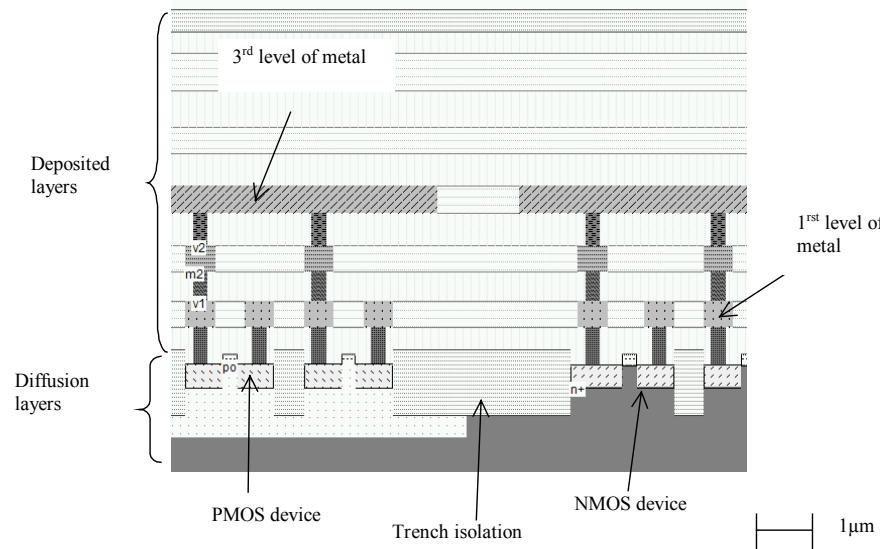


Figure 1-10: Cross-section of the  $0.35\mu\text{m}$  CMOS technology (INV3.MSK)

The Microwind and Dsch tools are configured by default in a CMOS  $0.12\mu\text{m}$  six-metal layer process with a minimal MOS device length of  $0.12\mu\text{m}$ . The metal interconnects are very narrow, around  $0.2\mu\text{m}$ , separated by  $0.2\mu\text{m}$  (Figure 1-11). The MOS device appears very small, below the stacked layers of metal sandwiched between oxides.

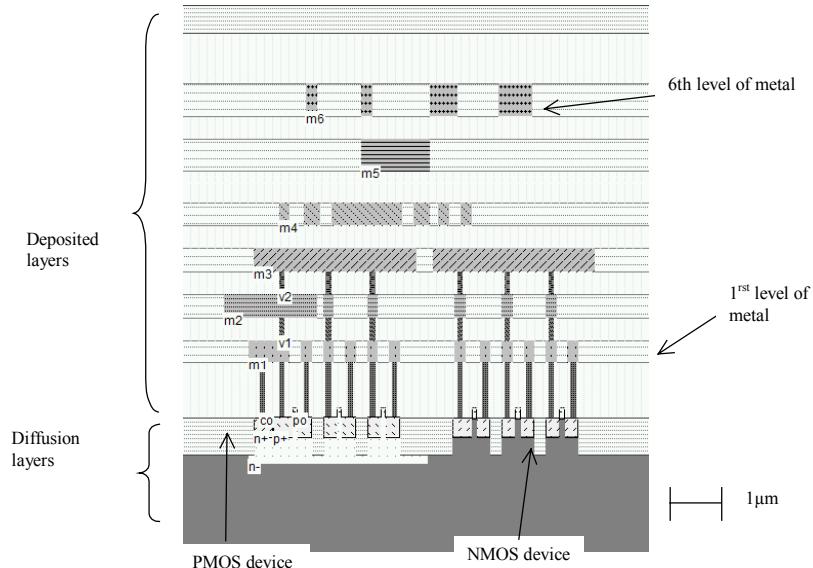


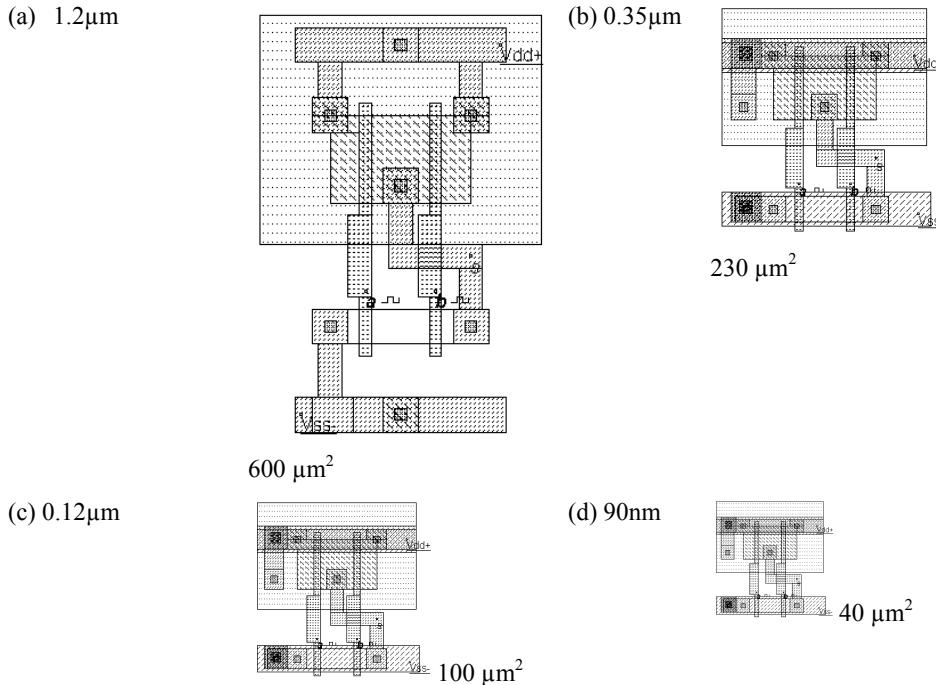
Figure 1-11: 2D View of the  $0.12\mu\text{m}$  process

## 5. DENSITY

The main consequence of improved lithography is the ability to implement an identical function in an ever smaller silicon area. Consequently, more functions can be integrated in the same space. Moreover, 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 to each other if many routing layers are provided (Figure 1-12).

The increased density provides two significant improvements: the reduction of the silicon area goes together with a decrease of parasitic capacitance of junctions and interconnects, thus increasing the switching speed of cells. Secondly, the shorter dimensions of the device itself speeds up the switching, which leads to further operating clock improvements.



*Figure 1-12: The evolution of the silicon area used to implement a NAND gate, which represents 20% of logic gates used in application specific integrated circuits*

Meanwhile, the silicon wafer, on which the chips are manufactured, has constantly increased in size, with the technological advances. A larger diameter means more chips fabricated at the same time, but requires ultra-high cost equipments able to manipulate and process these wafers with an atomic-scale precision. This trend is illustrated in figure 1-13. The wafer diameter for 0.12µm technology is 8 inches or 20cm (One inch is equal to 2.54 cm). Twelve inches wafers (30cm) have been introduced for the 90nm technology generation. The thickness of the wafer varies from 300 to 600µm.

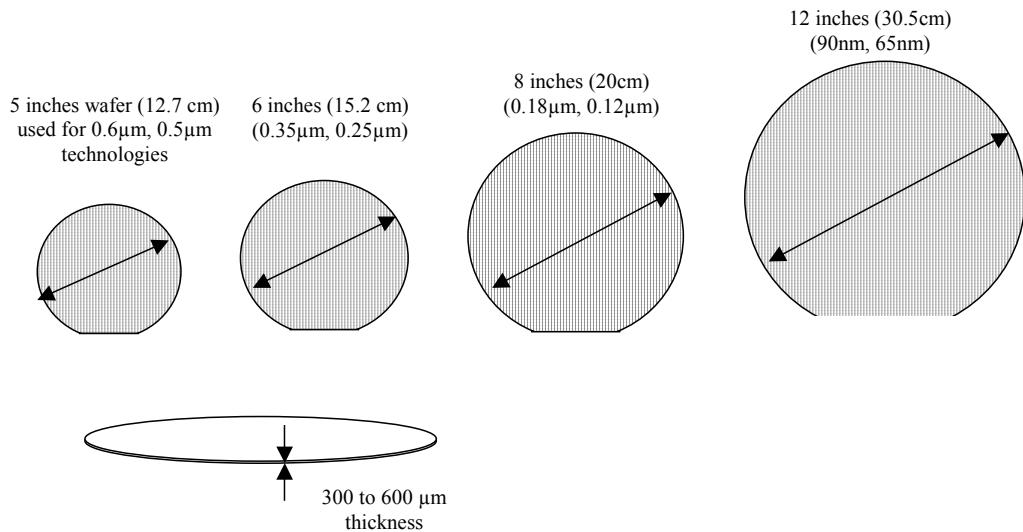


Figure 1-13: The silicon wafer used for patterning the integrated circuits

## 6. 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.

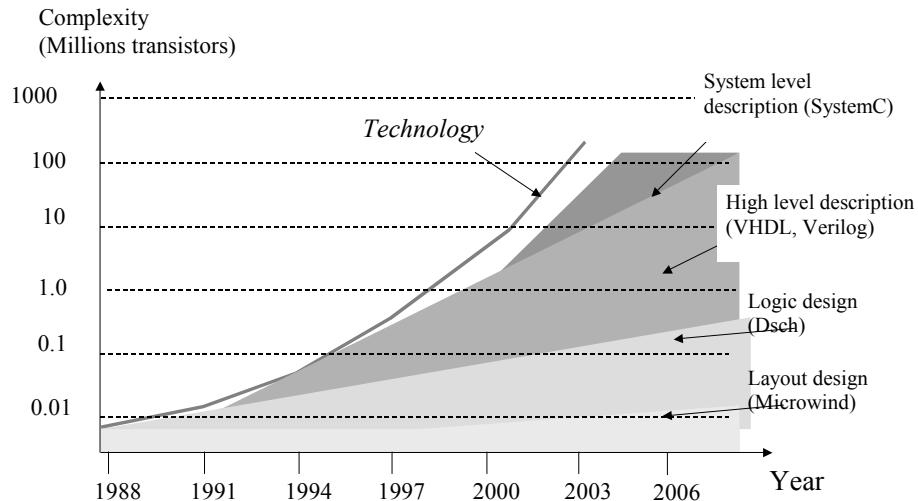


Figure 1-14: The evolution of integrated circuit design techniques, from layout level to system level

The introduction of high level description languages such as VHDL and Verilog [Verilog] have made possible the design of complete systems on a chip (SoC), with complexities ranging from 1million to 10 million transistors (Figure 1-14). Recently, languages for specifying circuit behavior such as SystemC [SystemC] have been made available, which correspond to design complexity between 100 and 1000 million transistors. Notice

that the technology has always been ahead of design capabilities, thanks to tremendous advances in process integration and circuit performances.

## 7. Market

Since the early days of microelectronics, the market has grown exponentially, representing more than 100 billion € in the beginning of the 21<sup>st</sup> century. The average growth in a long term trend is approximately 15%. Recently, two periods of negative growth have been observed: one in 1997-1999, the second one in 2002. Cycles of very high profits (1993-1995) have been followed by violent recession periods.

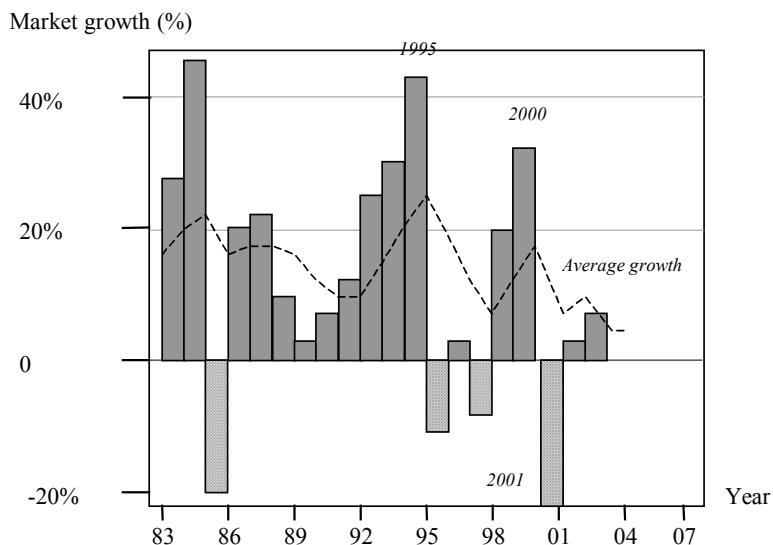


Figure 1-15: The percentage of market growth over the recent years shows a long term growth of 15%

## Conclusion

This chapter has briefly illustrated the technology scale down, the evolution of the microprocessor and micro-controller complexity, as well as some general information about CMOS technology, trends and market. The position of the Microwind layout design tool and Dsch logic design tool has been also described.

## References

- [Moore] G.E Moore, "VLSI: some fundamental challenges", IEEE Spectrum, N° 16 Vol 4, pp 30, 1975
- [SIA] <add web site>
- [Verilog] <add ref>
- [VHDL] <add ref>
- [SystemC] <add ref>

[Intel] <add link>

[Motorola] <add link micro-controller division>

[ITRS] <add link>

[Itoh] K. Itoh "VLSI Memory Chip Design" Springer-Verlag, 2001

## EXERCISES

1. Plot the frequency improvement versus the technology for the CMOSxx 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 date of figure 1-3 to build your answer.
3. From the 2D comparative aspect of  $0.8\mu m$  and  $0.25\mu m$  technologies (Figure 6), what may be the rising problems of using multiple metallization layers?
4. With the technology scale down, the silicon area decreases for the same device (see figure 1.8), but the chip size increases (table 1.1). Can you explain this contradiction?

*Partial answers are provided in Appendix F.*

# 2 The MOS devices and technology

This chapter presents the MOS transistors, their layout, static characteristics and dynamic characteristics. Details on the materials used to build the devices are provided. The vertical aspect of the devices and the three dimensional sketch of the fabrication are also described.

## 1. Properties of Silicon

IA																			0
H 1 Hydrogen	IIA																		He 2 Helium
Li 3 Lithium	Be 4 Beryllium																		B 5 Boron
Na 11 Sodium	Mg 12 Magnesium	IIIB	IVB	VB	VIB	VIIB	VII	VII	VII	IB	IIB	Al 13 Aluminum	Si 14 Silicon	P 15 Phosphorus	S 16 Sulfur	Cl 17 Chlorine	Ar 18 Argon	Ne 10 Neon	
K 19 Potassium	Ca 20 Calcium	Sc 21 Scandium	Ti 22 Titanium	V 23 Vanadium	Cr 24 Chromium	Mn 25 Manganese	Fe 26 Iron	Co 27 Cobalt	Ni 28 Nickel	Cu 29 Copper	Zn 30 Zinc	Ga 31 Gallium	Ge 32 Germanium	As 33 Arsenic	Se 24 Selenium	Br 35 Bromine	Kr 36 Krypton	Xe 54	
Rb 37	Sr 38	Y 39	Ze 40	Nb 41	Mo 42	Tc 43	Ru 44	Rh 45	Pd 46	Ag 47 Silver	Cd 48 Cadmium	In 49 Indium	Sn 50	Sb 51	Te 52	I 53	Xe 54		
Cs 55	Ba 56	La 57	Hf 72	Ta 73	W 74 Tungsten	Re 75	Os 76	Ir 77	Pt 78	Au 79 Gold	Hg 80	Tl 81	Pb 82 Lead	Bi 83	Po 84	At 85	Rn 86		

Figure 2-1: periodic table of elements and position of silicon

The table of figure 2-1 illustrates the table of elements. In CMOS integrated circuits, we mainly focus on Silicon, situated in the column IVA, as the basic material (Also called substrate <glossary>) for all our designs.

The silicon atom has 14 electrons, 2 electrons situated in the first energy level, 8 in the second and 4 in the third. The four electrons in the third energy level are called valence electrons, which are shared with other atoms.

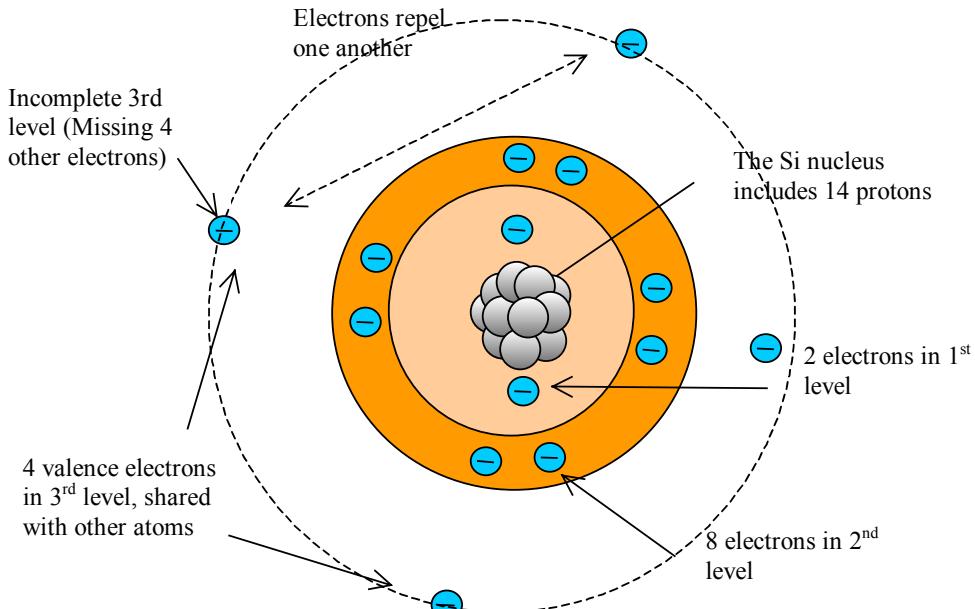


Figure 2-2: The structure of the silicon atom

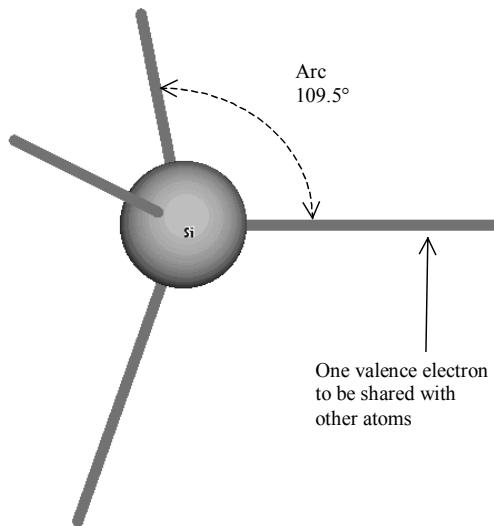


Figure 2-3: The 3D symbol of the silicon atom

The silicon atom has 4 valence electrons, which tend to repel each other. The 3<sup>rd</sup> level would be completed with 8 electrons. The four missing electrons will be shared with other atoms. The position of electrons which minimizes the mutual repulsion is shown in figure 2-3: each valence electron is represented by a line with an angle of 109.5°. In order to complete its valence shell, the silicon atom tends to share its valence electrons with 4 other electrons, by pairs. Each line between Si atoms in figure 2-3

represents a pair of shared valence electrons. The distance between two Si nucleus is 0.235 nm ( $10^{-9}$  m), equivalent to 2.35 Angstrom ( $10^{-10}$  m, also represented by the letter Å).

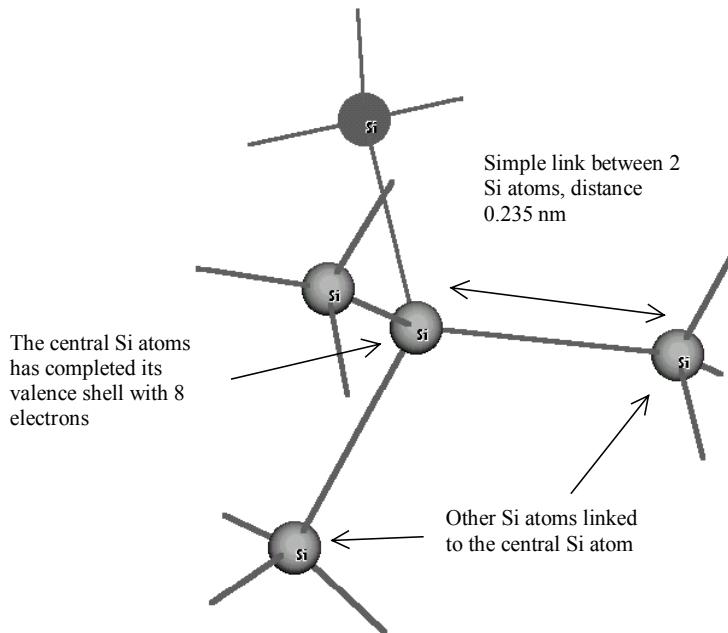


Figure 2- 4: The Si atom has four links, usually to other Si atoms

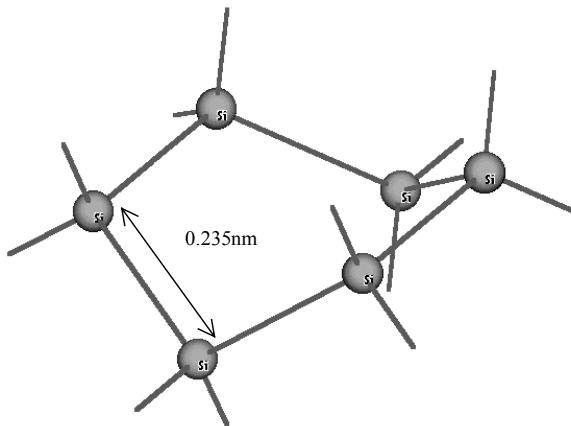


Figure 2-5: The atom arrangement is based on a 6 atom pattern

The Silicon lattice exhibits particular properties in terms of atom arrangements. The crystalline silicon is based on a 6-atom pattern shown in figure 2-5. The structure is repeated infinitely in all directions to form the silicon substrate as used for integrated circuit design. The pure silicon crystal is mechanically very strong and hard, and electrically a very poor conductor, as all valence electrons are shared within the structure (Figure 2-6). The atomic density of a silicon crystal is about  $5 \times 10^{22}$  atoms per cubic centimeter ( $\text{cm}^{-3}$ ).

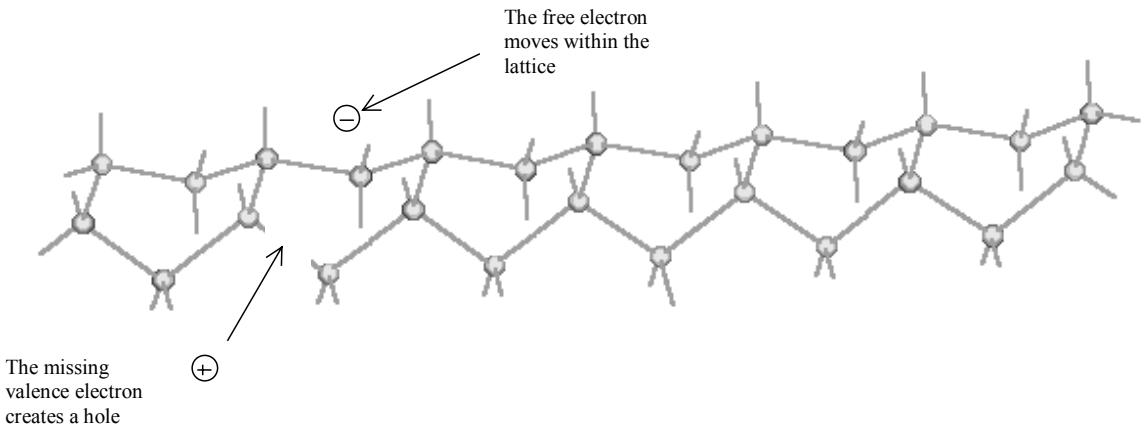


Figure 2-6: The chain of 6 atom pattern creates the silicon lattice

However, the random vibration of the silicon lattice due to thermal agitation may transmit enough energy to some electrons valence for them to leave their position. The electron moves freely within the lattice, and thus participate to the conduction of electricity. The lack of electron is called a hole (Figure 2-6). This is why silicon is not an insulator, nor a good conductor. It is called a semi-conductor <glossary> due to its intermediate electrical properties. The number of electrons which participate to the conduction are called intrinsic carriers <gloss>. The concentration of intrinsic carriers per cubic centimeter, namely  $n_i$ , is around  $1.45 \times 10^{10} \text{ cm}^{-3}$ . When the temperature increases, the intrinsic carrier density also increases. The concentration of free electrons is equal to the concentration of free holes.

## 2. N-type and P-type Silicon

To increase the conductivity of silicon, materials called dopant are introduced into the silicon lattice. To add more electrons in the lattice artificially, phosphorus or arsenic atoms (Group VA) are inserted in small proportions in the silicon crystal (Figure 2-7). As only four valence electrons find room in the lattice, one electron is released and participates to electrical conduction. Consequently, Phosphorus and arsenic are named "electron donors", with an N-type symbol. A very high concentration of donors is coded N++ (Around 1 N-type atom per 10,000 silicon atoms, corresponding to  $10^{18} \text{ atoms per cm}^{-3}$ ). A high concentration of donor is coded N+ (1 N-type atom per 1,000,000 silicon atom, that is  $10^{16} \text{ atoms per cm}^{-3}$ ), while a low concentration of donors is called N- (1 N-type atom per 100,000,000 silicon atom, or  $10^{14} \text{ atoms per cm}^{-3}$ ).

III	IVA	VA
Acceptor		Donor
Add holes		Add electrons
P-type		N-Type

B 5 Boron	C 6 Carbon	N 7 Nitrogen
Al 13 Aluminium	Si 14 Silicium	P 15 Phosphorus
Ga 31 Gallium Germanium	Ge 32 Germanium	As 33 Arsenic

Figure 2-7: Boron, Phosphorus and Arsenic are used as acceptors and donors of electrons to change the electrical properties of silicon

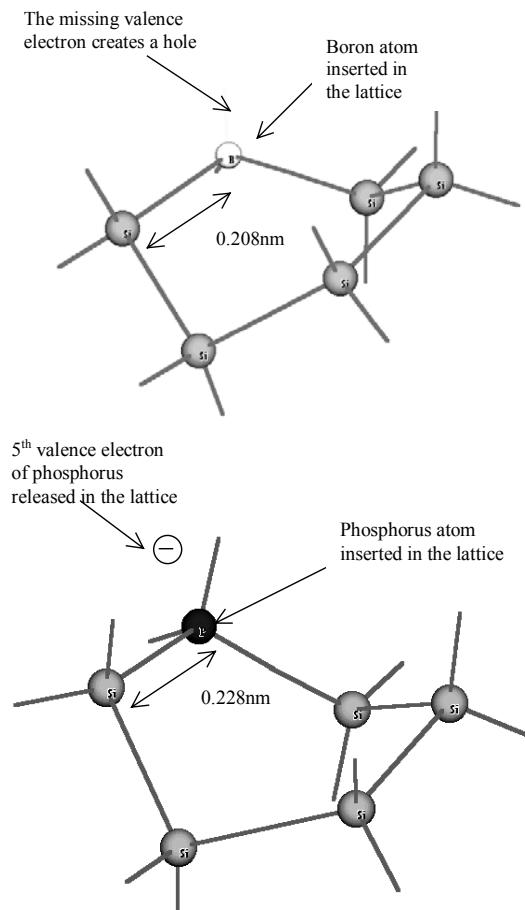
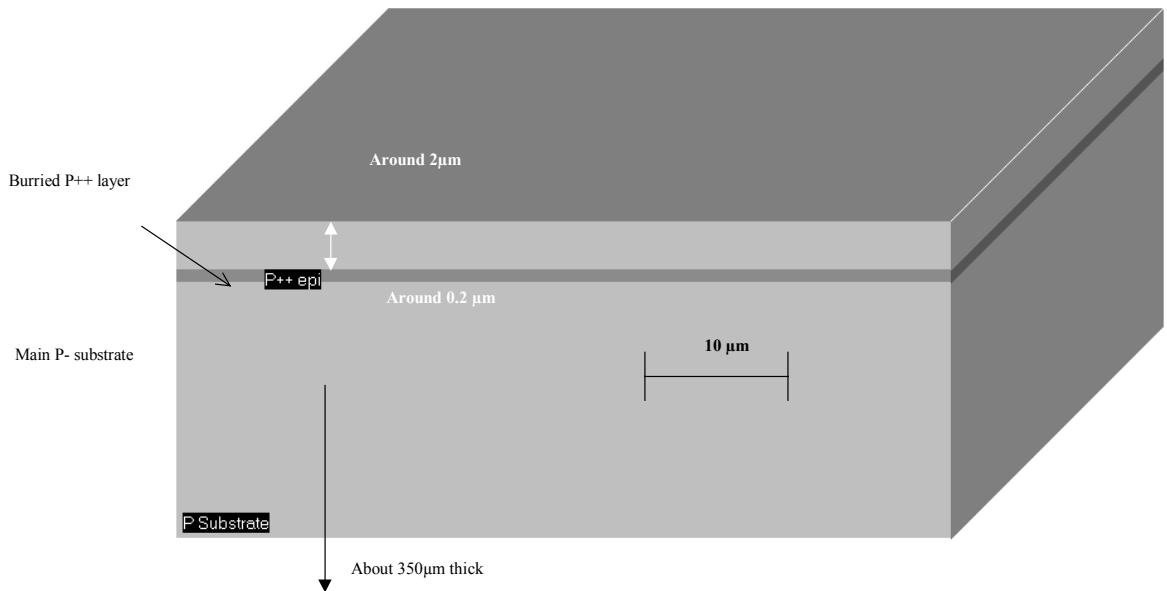


Figure 2-8: Boron added to the lattice creates a hole (P-type property), phosphorus creates a free electron (N-type property)

To increase artificially the number of holes in silicon, boron is injected into the lattice, as shown in figure 2-8. The missing valence link is due to the fact that boron only shares three valence electrons. The electron vacancy creates a hole, which gives the lattice a P-type property. A very high concentration of acceptors is coded P++ ( $10^{18}$  atoms per  $\text{cm}^{-3}$ ), a high concentration of acceptors is coded P+( $10^{16}$  atoms per  $\text{cm}^{-3}$ ), a low concentration of acceptors is called P-( $10^{14}$  atoms per  $\text{cm}^{-3}$ ). The silicon substrate used to manufacture CMOS integrated circuits is lightly doped with boron, characterized by the P- symbol. The

aspect of a small portion of silicon substrate is shown in 3d in figure 2-9. It usually consists of very thick substrate ( $350\mu\text{m}$ ) lightly doped P-. Close from the upper surface, a buried layer saturated with P-type acceptors is usually created, to form a good conductor beneath the active region, connected to the ground voltage.



*Figure 2-9: 3D aspect of a portion of silicon substrate used to manufacture CMOS integrated circuits. The substrate is based on a P- substrate with a buried P++ layer*

### 3. Silicon Dioxide

The natural and most convenient insulator is silicon dioxide, noted  $\text{SiO}_2$ . Its molecular aspect is shown in figure 2-10. Notice that the distance between Si and O atoms is smaller than for Si-Si, which leads to some interface regularity problems. Silicon dioxide is grown on the silicon lattice by high temperature contact with oxygen gas. Oxygen molecules not only combine with surface atoms, but also with underlying atoms. Silicon dioxide has an  $\epsilon_r$  permittivity equal to 3.9. This number quantifies the capacitance effect of the insulator. The permittivity of air is equal to 1, which is the minimum value. The  $\text{SiO}_2$  material is a very high quality insulator that is used extensively in CMOS circuits, both for devices and interconnections between devices. The "O" of CMOS corresponds to "oxide", and refers to  $\text{SiO}_2$ .

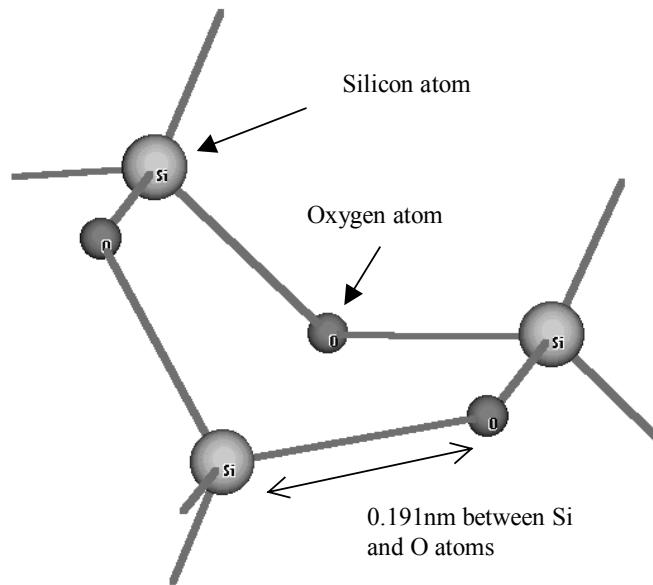


Figure 2- 10: Silicon is linked to Oxygen to form the SiO<sub>2</sub> molecular structure

## 4. Metal materials

Integrated circuits also use several metal materials to build interconnects. Aluminum (III), Tungsten (IVB), Gold (IB), and Copper (IB) are commonly used in the manufacturing of microelectronic circuits.

VIB	VIIB	VII	VII	VII	IB	IIB	Al 13 Aluminum
Cr 24 Chromium	Mn 25 Manganese	Fe 26 Iron	Co 27 Cobalt	Ni 28 Nickel	Cu29 Copper	Zn 30 Zinc	Ga 31 Gallium
Mo 42	Tc 43	Ru 44	Rh 45	Pd 46	Ag 47 Silver	Cd 48 Cadmium	In 49 Indium
W 74 Tungsten	Re 75	Os 76	Ir 77	Pt 78	Au 79 Gold	Hg 80	Tl 81

Table 2-1: Metal materials used in CMOS integrated circuit manufacturing

Metal layers are characterized by their resistivity ( $\sigma$ ). We notice that copper is the best conductor as its resistivity is very low, followed by gold and aluminium (Table 2-2). A highly doped silicon crystal does not exhibit a low resistivity, while the intrinsic silicon crystal is half way between a conductor and an insulator [Hastings].

Material	Symbol	Resistivity $\sigma$ ( $\Omega \cdot \text{cm}$ )
Copper	Cu	$1.72 \times 10^{-6}$

Gold	Au	$2.4 \times 10^{-6}$
Aluminium	Al	$2.7 \times 10^{-6}$
Tungsten	W	$5.3 \times 10^{-6}$
Silicon, N+ doped	N+	0.25
Silicon, intrinsic	Si	$2.5 \times 10^{-5}$

Table 2-2: Conductivity of the most common materials used in CMOS integrated

Conductivity is sometimes used instead of resistivity. In that case, the formulation is as follows:

$$\rho = \frac{1}{\sigma} \quad (\text{Eq. 2-1})$$

with

$\rho$  = conductivity ( $\Omega \cdot \text{cm}$ )<sup>-1</sup>

$\sigma$  = Resistivity ( $\Omega \cdot \text{cm}$ )

## 5. The MOS switch

The MOS transistor (MOS for metal-oxide-semiconductor) is by far the most important basic element of the integrated circuit. The MOS transistor is the integrated version of the electrical switch. When it is on, it allows current to flow, and when it is off, it stops current from flowing. The MOS switch is turned on and off by electricity. Two types of MOS device exist in CMOS technology (Complementary Metal Oxide Semiconductor): the n-channel MOS device (also called nMOS) and the p-channel MOS device (also called pMOS).

### 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		
X	Undefined	X	(Red in logic simulation)	(Gray in simulation)
			(Gray in simulation)	(Gray in simulation)

Table 2-3: the logic levels and their corresponding symbols in Dsch and Microwind tools

## The n-channel MOS switch

Despite its extremely small size (less than 1 $\mu\text{m}$  square), the current that the MOS transistor may switch is sufficient to turn on and off a led, for example. The MOS device consists of two electrical regions called drain and source, separated by a channel. A channel of electrons may exist or not in this channel, depending on a voltage applied to the gate. The gate is a conductor placed on the top of the channel, and electrically isolated by an ultra thin oxide. The MOS is basically a switch between drain and source. A schematic cross-section of the MOS device is given in figure 2-11. 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.

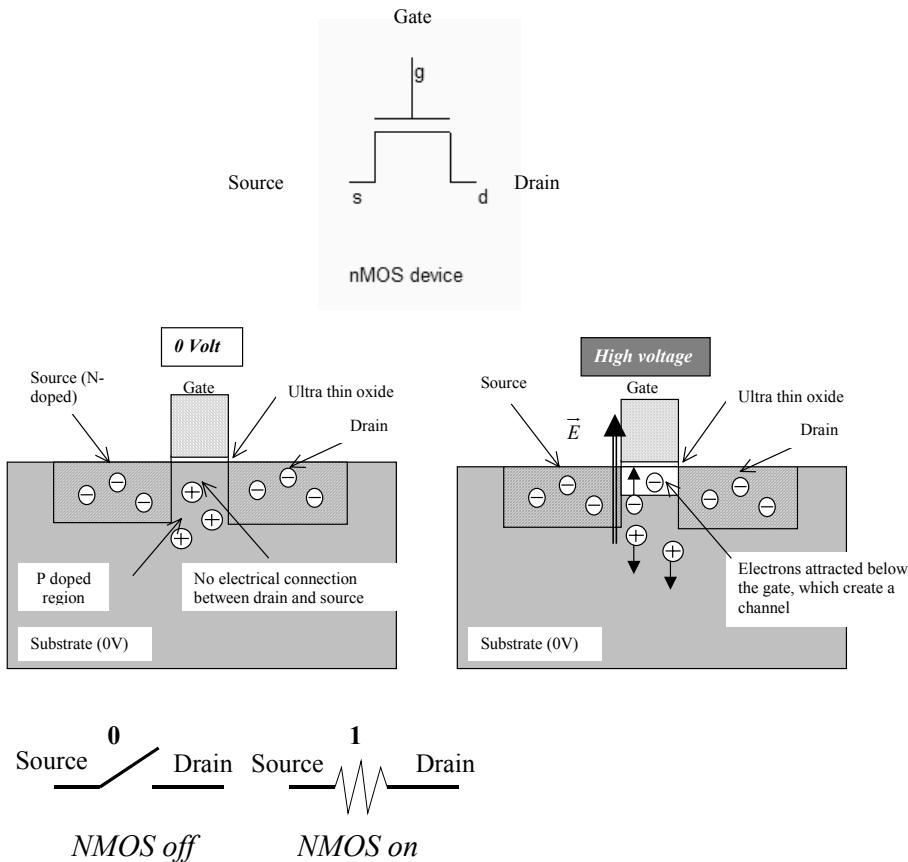


Figure 2-11: Basic principles of a MOS device

When used in logic cell design, it can be *on* or *off*. As illustrated in figure 2-xxx, the n-channel MOS device requires a high supply voltage to be on. When *on*, a current can flow between drain and source. When the MOS device is on, the link between the source and drain is equivalent to a resistor. The resistance may vary from less than 0.1 $\Omega$  to several hundred K $\Omega$ . Low resistance MOS devices are used

for power application, while high resistance MOS devices are widely used in analog low power designs. In logic gate, the  $R_{on}$  resistance is around  $1\text{ K}\Omega$ .

The ‘off’ resistance is considered infinite at first order, but its value is several  $\text{M}\Omega$ . When off, almost no current flow between drain and source. The device is equivalent to an open switch, and the voltage of the floating node (The drain in the case of table 2-xxx) is undetermined. The n-channel MOS logic table can be described as follows.

Gate	Source	Drain
0	0	X
0	1	X
1	0	0
1	1	1

Table 2-5: the n-channel MOS switch truth-table

### The p-channel MOS switch

In contrast, the p-channel MOS device requires a zero voltage supply to be on. The p-channel MOS symbol differs from the n-channel device with a small circle near the gate. The channel carriers for pMOS are holes.

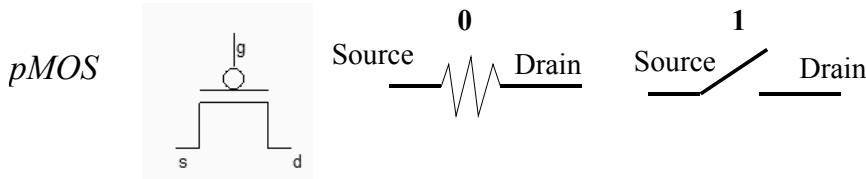


Figure 2-12: the MOS symbol and switch

The p-channel MOS logic table can be described as follows.

Gate	Source	Drain
0	0	0
0	1	1
1	0	X
1	1	X

Table 2-6: the p-channel MOS switch truth-table

For the p-channel MOS, a high voltage disables the channel. Almost no current flows between the source and drain. A zero voltage VDD on the gate, attracts holes below the gate, creates a hole channel and enables current to flow, as shown below.

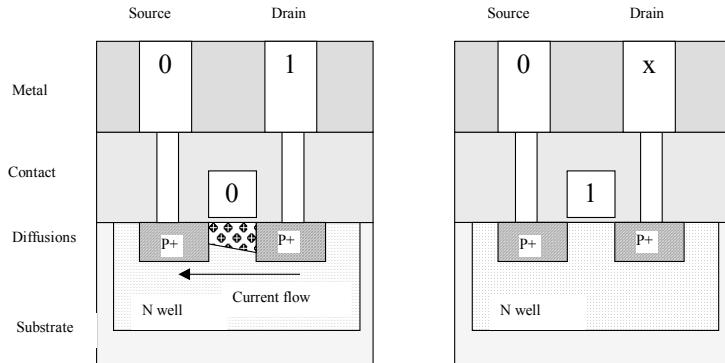


Figure 2-13. The channel generation below the gate in a pMOS device.

## 6. The MOS aspect

The bird's view of the layout including one n-channel MOS device and one p-channel MOS devices placed at the minimum distance is given in figure 2-xxx A two-dimensional zoom at micron scale in the active region of an integrated circuit designed in  $0.12\mu\text{m}$  technology is reported in figure 2-15. This view corresponds to a vertical cross-section of the silicon wafer, in location X-X' in figure 2-14. The Microwind tool has been used to build the layout of the MOS devices (The corresponding file is **allMosDevices.MSK**), and to visualize its cross-section, using the command **Simulate →2D vertical cross-section**.

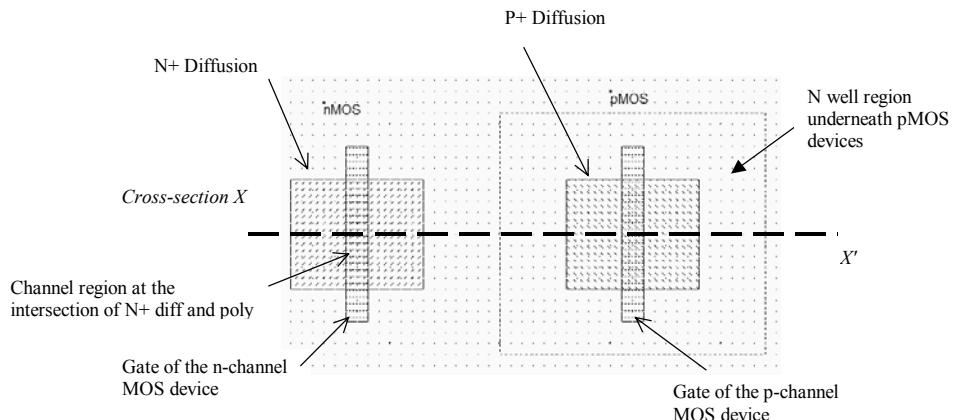


Figure 2-14: Bird's view of the n-channel and p-channel MOS device layout (**allMosDevices.MSK**)

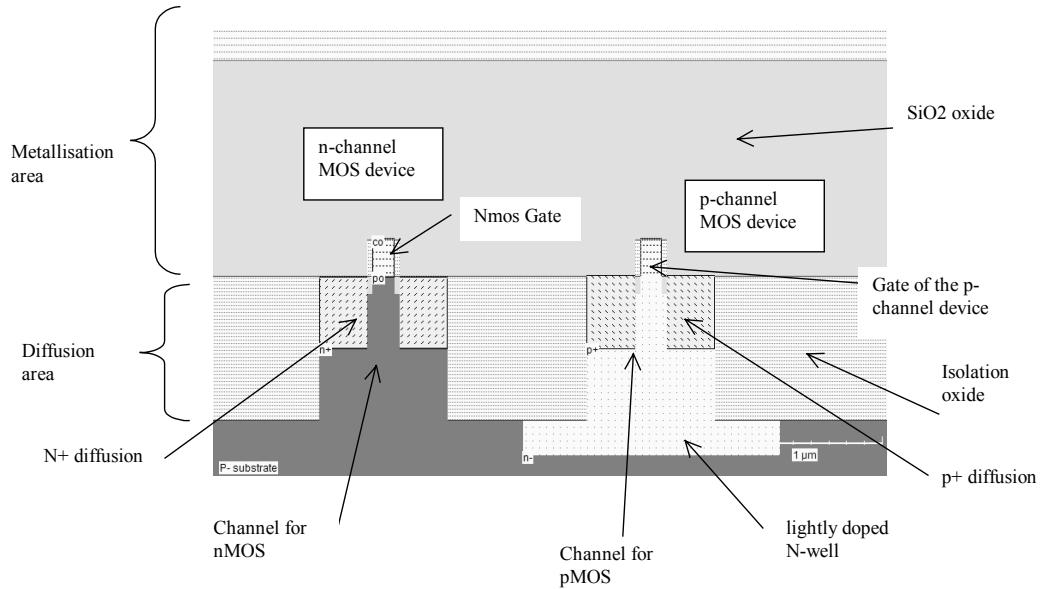


Figure 2-15: Vertical cross-section of an n-channel and p-channel MOS devices in  $0.12\mu\text{m}$  technology  
(allMosDevices.MSK)

The layout of the nMOS and pMOS devices, seen from the top of the circuit, is shown in figure 2-xxx. The MOS is built using a set of layers that are summarized below. CMOS circuits are fabricated on a piece of silicon called wafer, *<gloss>* usually lightly doped with boron, that gives a p-type property to the material.

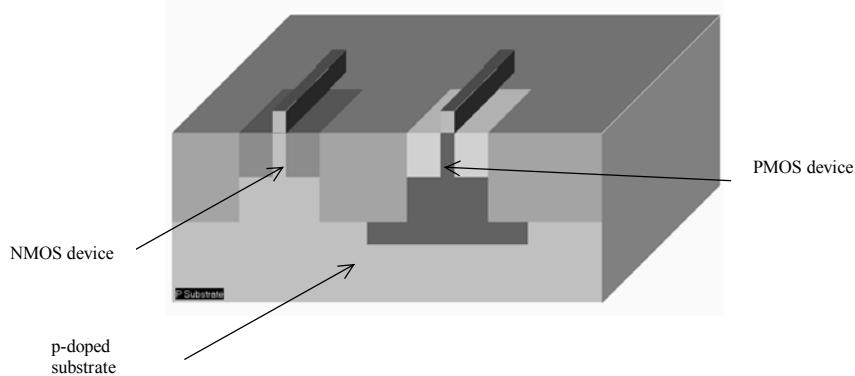


Figure 2-16: 3D view of the n-channel and p-channel MOS devices (AllMosDevices.MSK)

## Zoom at Atomic Scale

When zooming on the gate structure, we distinguish the thin oxide beneath the gate, the low doped diffusion regions on both sides of the channel, the metal deposit on the diffusion surface, and the spacers on each side of the gate, as shown in figure 2-17.

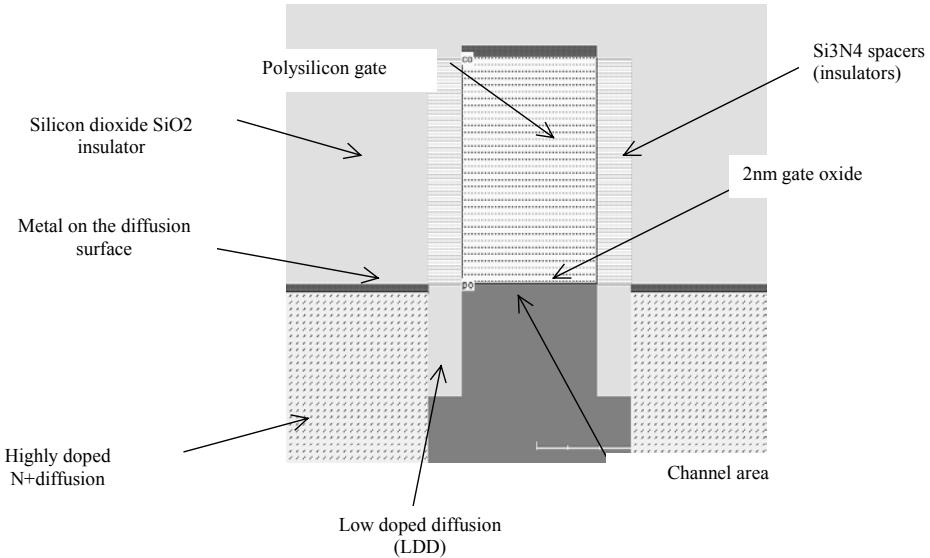


Fig. 2-17: Zoom at the gate oxide for a 0.12μm n-channel MOS device (AllMosDevices.MSK)

When zooming at maximum scale, we see the atomic structure of the transistor. The gate oxide accumulates between 5 and 20 atoms of silicon dioxide. In 0.12μm , the oxide thickness is around 2nm for the core logic, which is equivalent to 8 atoms of SiO<sub>2</sub>.

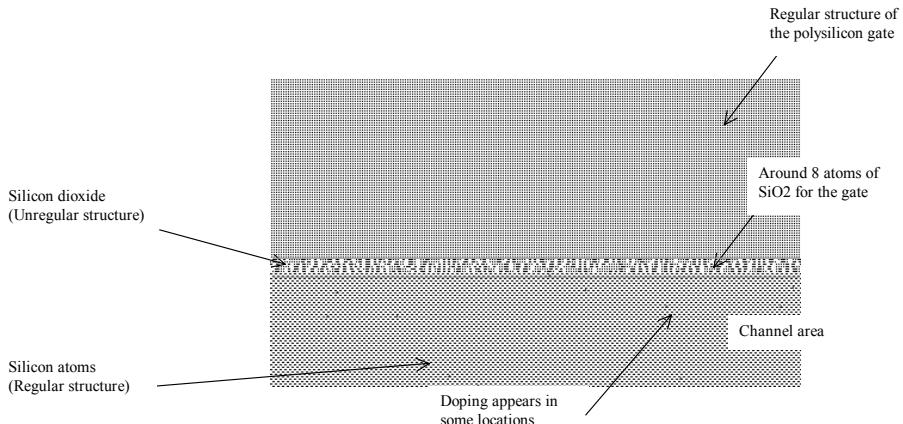


Fig. 2-18: Zoom at atomic scale near the gate oxide for a 0.12μm n-channel MOS device (AllMosDevices.MSK)

## 7. MOS layout

The objective of this paragraph is to draw the n-channel and p-channel MOS devices according to the design rules and usual design practices. The Microwind tool provided in the companion CD-Rom is used to draw the MOS layout and simulate its behavior.

The Microwind main screen shown in figure 2-19 includes two windows: one for the main menu and the layout display, the other for the icon menu and the layer palette. The main layout window features a grid, scaled in lambda ( $\lambda$ ) units. The size of the grid constantly adapts to the layout. In figure 2-19, the grid is 5 lambda. The lambda unit is fixed to half of the minimum available lithography of the technology  $L_{\min}$ . For example, the default technology is a CMOS 6-metal layers 0.12 $\mu\text{m}$  technology, consequently lambda is 0.06 $\mu\text{m}$ .

$$\lambda = \frac{L_{\min}}{2}$$

(Eqn. 2-2)

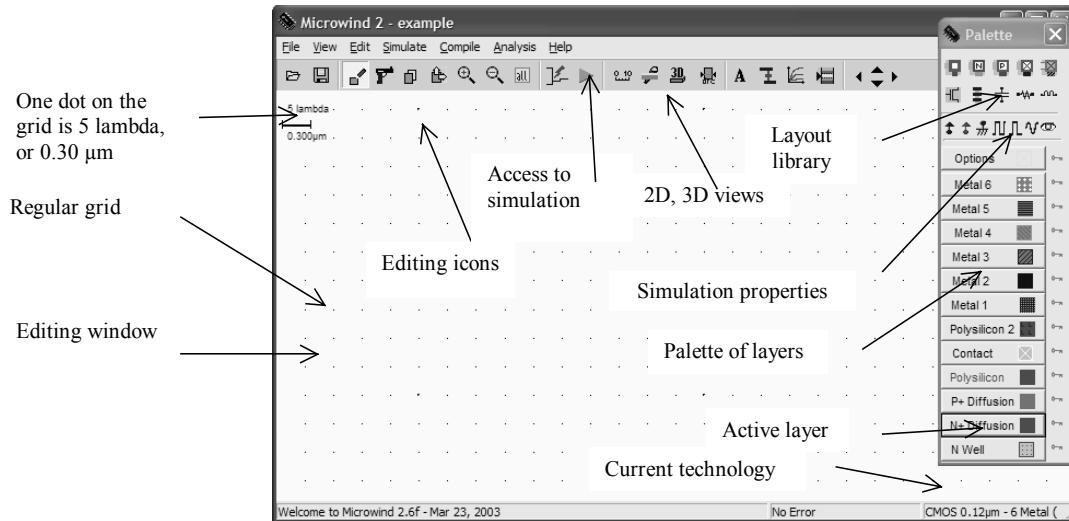


Figure 2-19 The MICROWIND2 window as it appears at the initialization stage

The palette is located in the right corner of the screen. A red color indicates the current layer. Initially the selected layer in the palette is polysilicon.

### n-channel MOS layout

By using the following procedure, you can create a manual design of the n-channel MOS device. The n-channel MOS device consists of a polysilicon gate and a heavily doped diffusion area. Select the "polysilicon" layer in the palette window.

- 1) 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 narrow box in polysilicon layer as shown in Figure 2-20. The box width should not be inferior to  $2\lambda$ , which is the minimum and optimal thickness of the polysilicon gate.

- 2) Change the current layer into N+ diffusion by a click on the palette of the N+ Diffusion 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 2-20. The N+ diffusion should have a minimum of 4  $\lambda$  on both sides of the polysilicon gate. The intersection between diffusion and polysilicon creates the channel of the nMOS device.

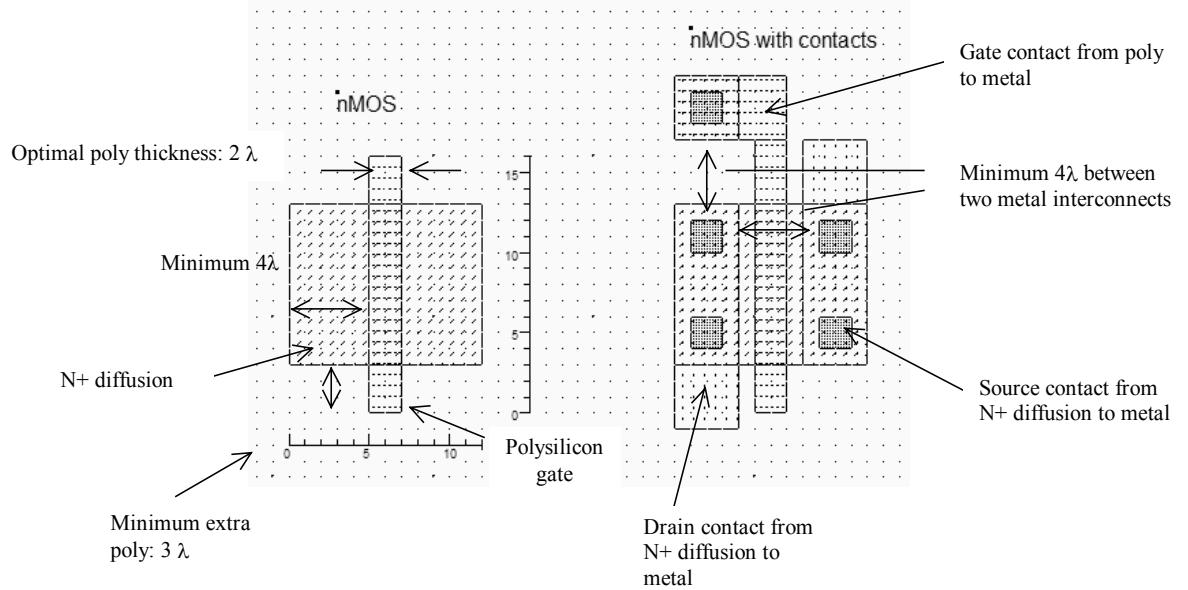


Figure 2-20. Creating the N-channel MOS transistor and adding contacts (*AllMosDevices.MSK*)

Now, we add the metal contacts to enable an electrical access to the source and drain regions. In the palette, such contacts are ready to instantiate on the layout (Figure 2-21). Click on the appropriate icon, and then the appropriate location in the left N+ diffusion. Repeat the process and add an other contact on the right part of the N+ diffusion. The layout aspect should correspond to the layout shown in figure 2-21.

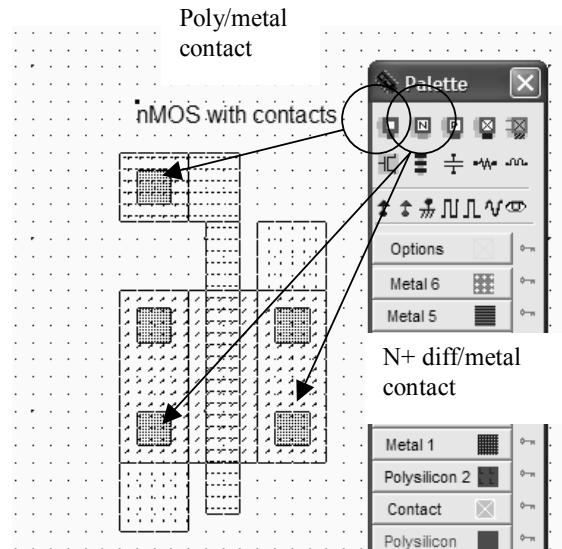


Figure 2-21. Access to the n-diffusion/metal and poly/metal contacts

## Basic layers

The wafer serves as the substrate (or bulk) to N-channel MOS, which can be implemented directly on the p-type substrate. The n-channel MOS device is based on a polysilicon gate, deposited on the surface of the substrate, isolated by an ultra thin oxide (called gate oxide), and an N<sup>+</sup> implantation that forms two electrically separated diffusions, on both side of the gate. The list of layers commonly used for the design of MOS devices is given in table 2-6.

Layer name	Code	Description	Color in Microwind
Polysilicon	Poly	Gate of the n-channel and p-channel MOS devices	Red
N <sup>+</sup> diffusion	Diffn	Delimits the active part of the n-channel device. Also used to polarize the N-well	Dark green
P <sup>+</sup> diffusion	Diffn	Delimits the active part of the p-channel device. Also used to polarize the bulk	Maroon
Contact	Contact	Makes the connection between diffusions and metal for routing. The contact plug is fabricated by drilling a hole in the oxide and filling the hole with metal.	White cross
First level of metal	Metall1	Used to rout devices together, in order to create the logic or analog function	Blue
N well	Nwell	Low doped diffusion used to invert the doping of the substrate. All p-channel MOS are located within N well areas.	Dotted green

Table 2-6: Materials used to build n-channel and p-channel MOS devices

## p-channel MOS layout

The p-channel MOS is built using polysilicon as the gate material and P+ diffusion to build the source and drain. The pMOS device requires the addition of the n-well layer, into which the P+ implant is completely included, in order to work properly (Figure 2-22). By using the following procedure, you can create manually the layout of the p-channel MOS device.

- Select the "polysilicon" layer in the palette window.
- Create a narrow polysilicon box to create the p-channel MOS gate (The minimum value  $2\lambda$  is often used). The material is the same as for the n-channel MOS.
- Change the layer into P+ diffusion. Draw a p-diffusion box at the bottom of the drawing as in Figure 2-22. The P+ diffusion should have a minimum of  $4\lambda$  on both sides of the polysilicon gate.
- Select the n-well layer. Add an n-well region that completely includes the P+ diffusion, with a border of  $6\lambda$ , as illustrated in figure 2-22.

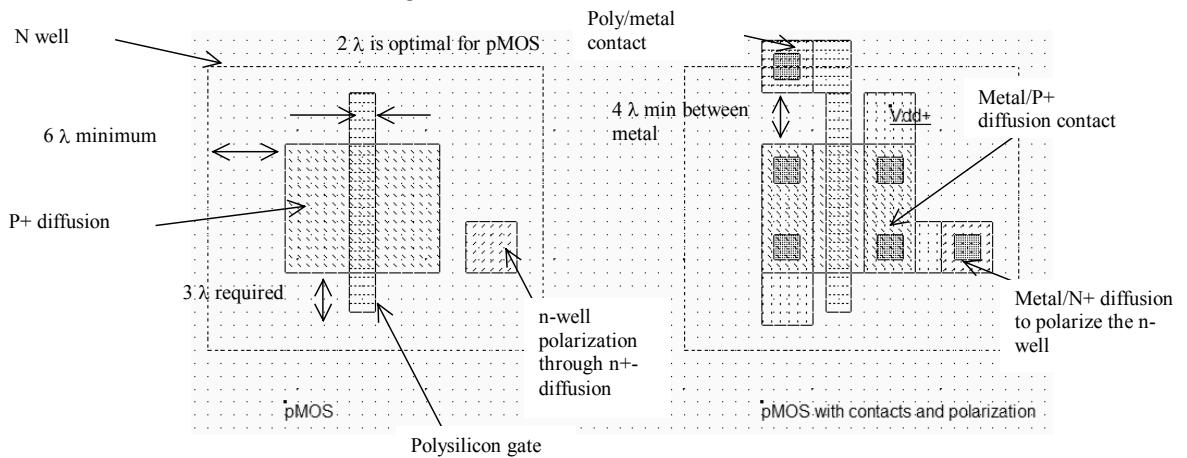


Figure 2-22. Creating the P-channel MOS transistor (AllMosDevices.MSK)

Moreover, the n-well region cannot be kept floating. A specific contact, that can be seen on the right side of the n-well, serves as a permanent connection to high voltage. Why high voltage? Let us consider the two cross sections in figure 2-23. On the left side, the n-well is floating. The risk is that the n-well potential decreases enough to turn on the P+/Nwell diode. This case corresponds to a parasitic PNP device. The consequence may be the generation of a direct path from the VDD supply of the drain to the ground supply of the substrate. In many cases the circuit can be damaged.

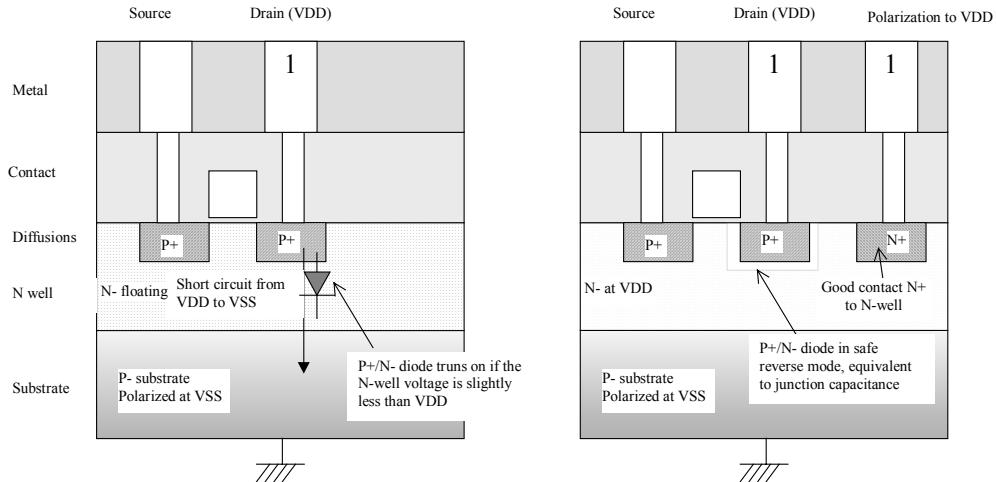


Fig. 2-23: Incorrect and correct polarization of the N-well

The correct approach is indicated in the right part of figure 2-24. A polarization contact carries the VDD supply down to the n-well region, thanks to an N+ diffusion. A direct contact to n-well would generate parasitic electrical effects, consequently, the N+ region embedded in the n-well area is mandatory. There is no more fear of parasitic PNP device effect as the P+/Nwell junctions are in inverted mode, and thus may be considered as junction capacitance.

## Useful Editing Tools

Editing layout is rarely a simple task at the beginning, when cumulating the discovery of the editor user's interface and the application of new microelectronics concepts. The following commands may help you in the layout design and verification processes.

Command	Icon/Short cut	Menu	Description
UNDO	CTRL+U	Edit menu	Cancel the last editing operation
DELETE	CTRL+X	Edit menu	Erase 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	Copy of the layout included in the given area.

Table 2-7: A set of useful editing tools

## 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.

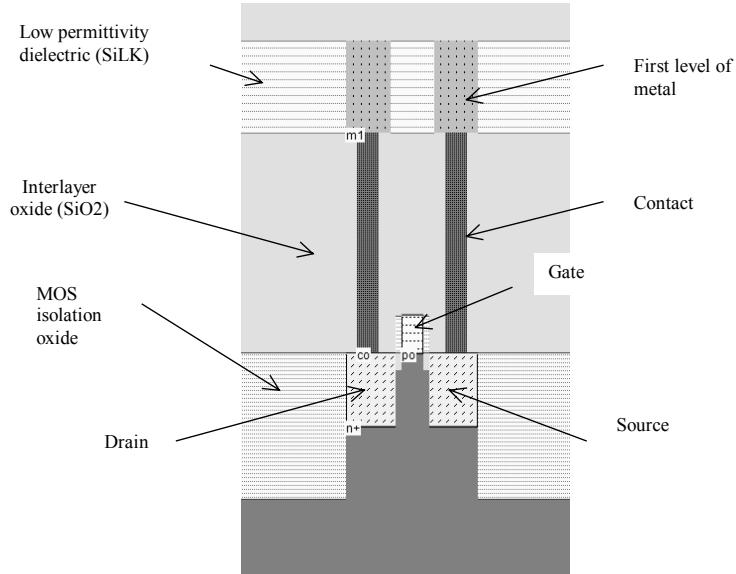


Figure 2-23. The cross-section of the nMOS devices (AllMosDevices.MSK)

In the example of Figure 2-23, 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 lateral drain diffusion (LDD) is a small region of lightly doped diffusion, at the interface between the drain/source and the channel. A light doping reduces the local electrical field at the corner of the drain/source and gate. Electrons accelerated below the gate at maximum electrical field, such as in figure 2-24 acquire sufficient energy to create a pair of electrons and holes in the drain region. Such electrons are called "hot electrons" <gloss>.

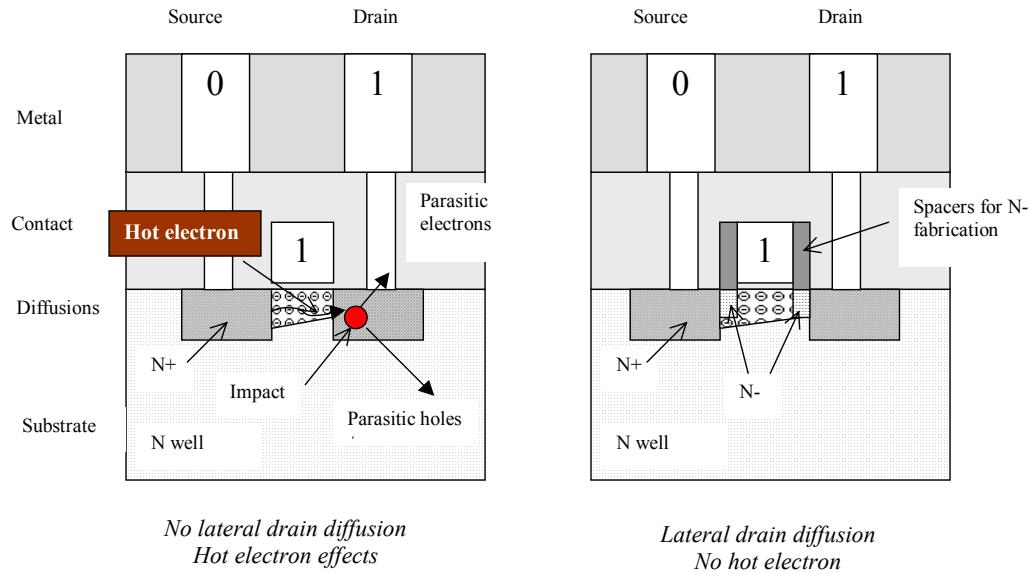


Figure 2-24. Lateral drain diffusion reduces the hot electron effect.

Consequently, parasitic currents are generated in the drain region. One part of the current flows down to the substrate, an other part is collected at the drain contact. The lateral drain diffusion <gloss> efficiently reduces this parasitic effect. This technique has been introduced since the  $0.5\mu\text{m}$  process generation.

## 8. Dynamic MOS behavior

In this paragraph, we stimulate the MOS device with variable voltages in order to verify by analog simulation their correct behavior as switches. The proposed simulation setup (Figure 2-25) consists in applying 0 and 1 to the gate and the source, and see the effect on the drain, as outlined in the schematic diagram below.

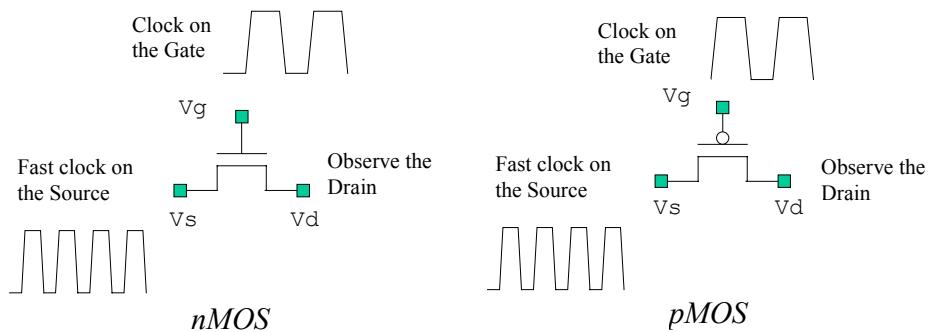


Fig. 2-25. Verification of the MOS switching properties using clocks

### n-channel MOS behavior

The expected behavior of the n-channel MOS device is summarized in figure 2-26. The 0 on the gate should leave the drain floating. The 1 on the gate should link the drain to the source, via a resistive path.

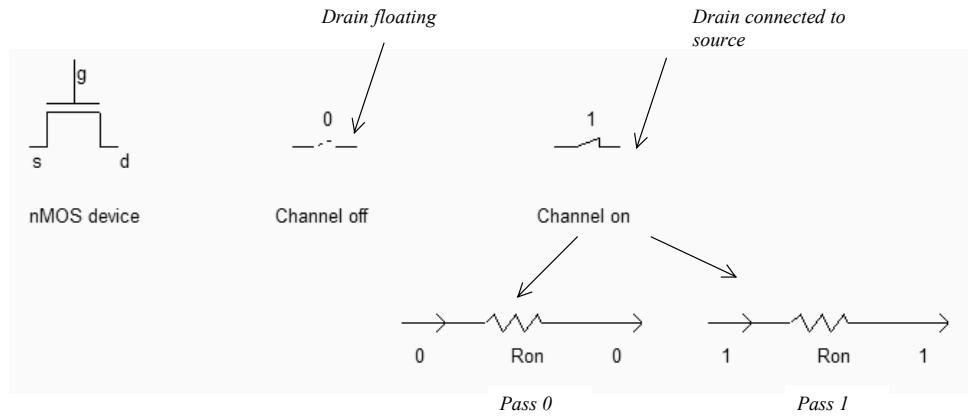


Fig. 2-26. Expected n-channel MOS switching characteristics(MosExplain.SCH)

The most convenient way to operate the MOS is to apply a clock property to the gate, another to the source and to observe the drain. The summary of available properties that can be applied to the layout is reported below.

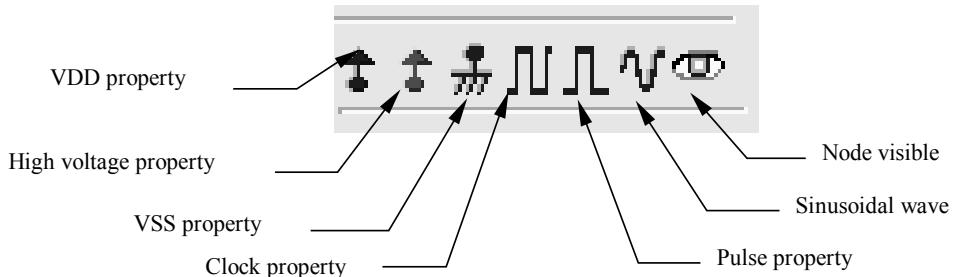


Fig. 2-27. Simulation properties used to conduct the simulation from layout

A clock should be applied to the source, which is situated on the green diffusion area at the left side of the gate. Click on the Clock icon and then, click on the polysilicon gate. The clock menu appears again. Change the name into « Vdrain» and click on OK to apply a clock with 1ns period (0.225ns at 0, 25ps rise, 0.225ns at 1, 25ps fall). The label « Vdrain» appears at the desired location in italic, meaning that the waveform of this node will appear at the next simulation.

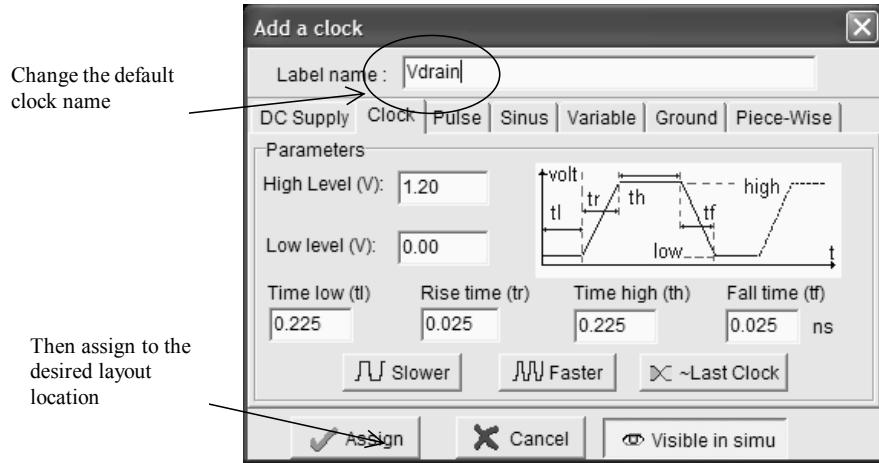


Fig. 2-28. Details on the clock parameters and clock menu.

Now, to apply a clock to the gate, click again on the Clock icon, and click on the polysilicon gate. The Clock menu appears. Notice that the clock parameters "Time low" and "Time high" has been automatically increased by 2, to create a clock with a period twice slower than previously. Change the name into « Vclock » and click on OK. The Clock property is sent to the gate and appears at the right hand side of the label « Vgate ».

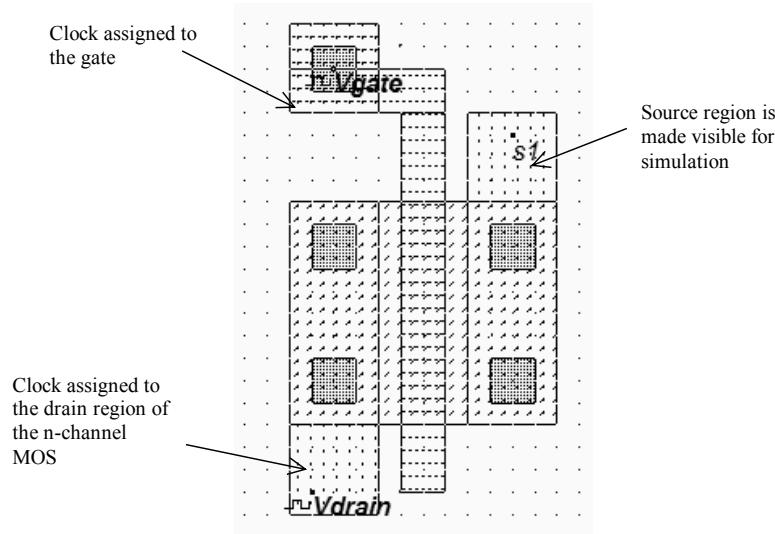


Fig.2- 29. Properties added to the layout for controlling the analog simulation (mosN.MSK)

In order to see the source, click on the eye ("Node Visible") property situated in the palette menu (See fig. 2-30) and click on the right diffusion. Change the label name into "Vout". Click OK. The visible property is then sent to the node. The associated text « Vout » is in italic, meaning that the waveform of this node will appear at the next simulation.

The layout should then appear as illustrated in figure 2-31. The clock properties are situated on the gate and the left N+ diffusion, the "node visible" property is located on the right part of the N+ diffusion. The layout is now ready for analog simulation.

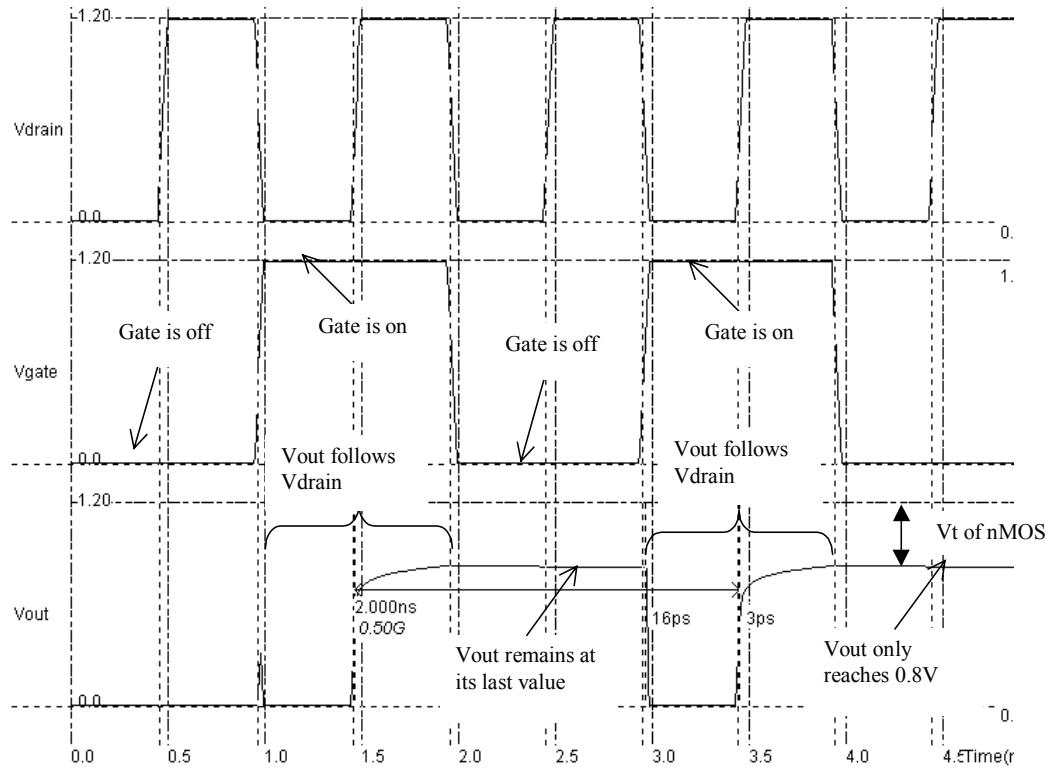


Fig.2- 31. Analog simulation of the n-channel MOS device (MosN.MSK)

Click on **Simulate → Run Simulation → Voltage vs. Time** (Or CTRL+S, or the icon "Run Simulation" in the main iucon menu). The timing diagrams of the nMOS device appear, as shown in Figure 2-31. Most of the logic and analog behavior of the MOS device are summarized in this single figure.

The upper waveform correspond to Vdrain, with a clock from 0 to 1.2V (VDD is 1.2V in  $0.12\mu\text{m}$ ), exhibiting a 1ns period. Below is the gate voltage. The n-channel MOS is off when Vgate is zero, and on if Vgate is 1.2V. The third waveform concerns Vout. Its aspect is very interesting: from 0 to 1ns, its value is zero, which is the default voltage value of all nodes in the layout. The channel is off, consequently nothing happens to the drain. When the gate is on, the channel enables Vout to copy the value of Vdrain (Time 1.0 to 2.0ns).

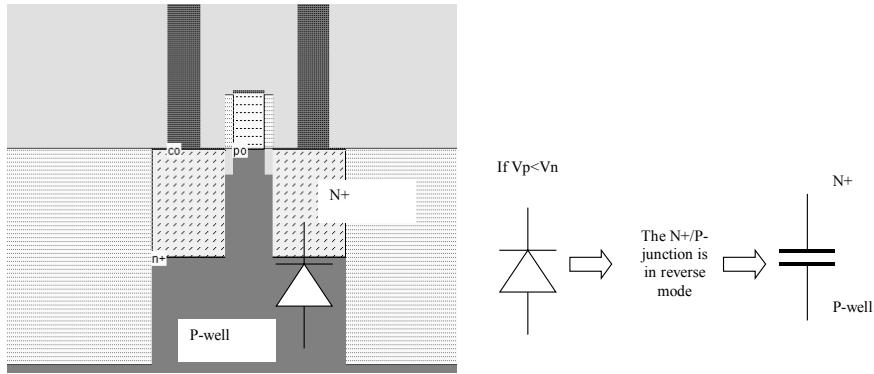


Fig.2- 32. The P/N junction in reverse mode is equivalent to a capacitor that memorizes s the voltage when the channel is off (MosN.MSK)

Then, when the gate is off again (Time 2.0 to 3.0), the voltage remains almost at its last value. The reason is illustrated in figure 2-xxx: the junction P-well/N+ diffusion is in reverse mode and can be considered as a capacitor. The charges are stored in this junction capacitor while the channel is off, which keep its voltage stable, independently of the source fluctuations.

Notice a very small decrease of the voltage, due to parasitic leakage effect <gloss> between source and drain. Click **More** in order to perform more simulations. Click on **Close** to return to the editor.

### Threshold Voltage

You probably noticed that the voltage  $V_{out}$  never reaches 1.2V. It "saturates" to around 0.8V. The reason is the parasitic effect called threshold voltage, which is around 0.4V in the default technology 0.12 $\mu$ m. In summary, the n-channel MOS device behaves as a switch, but when on, it does not pass correctly the high voltages. A zero on one side leads to a good zero, a logic 1 on one side leads to a poor 1. The main reason is the threshold voltage of the MOS.

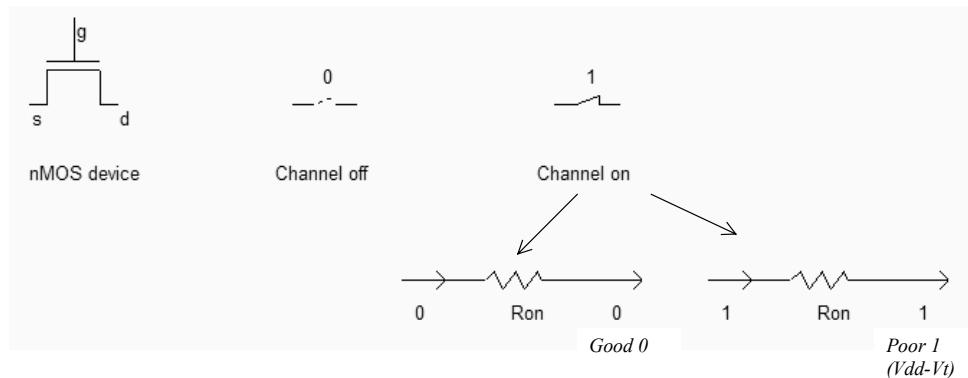


Fig.2- 33. The nMOS device behavior summar (MosExplain.SCH)

### p-channel MOS behavior

The expected behavior of the n-channel MOS device is summarized in figure 2-34. The 0 on the gate should link the drain to the source, via a resistive path. The 1 on the gate should leave the drain floating. In other words, the p-channel transistor simulation features the same functions as the n-channel device, but with opposite voltage control of the gate.

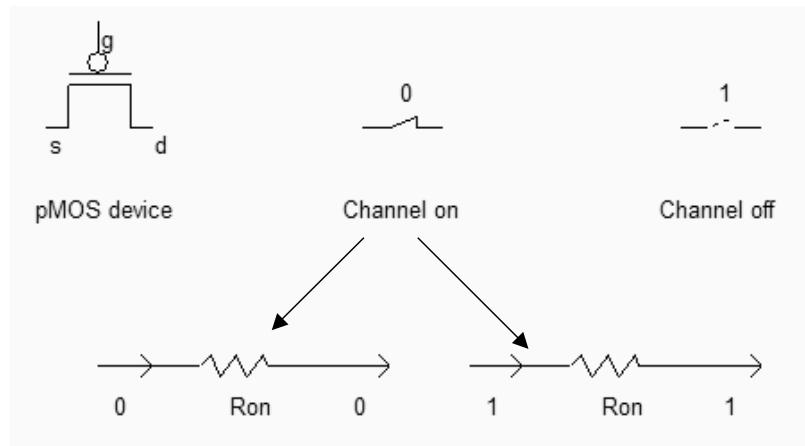


Fig. 2-34. Expected p-channel MOS switching characteristics(MosExplain.SCH)

Again, we operate the p-channel MOS device by using two clocks, one to the gate, another to the source and to observe the drain. Be sure to add the polarization contact inside the n-well region, and add a supplementary VDD property on top of the contact.

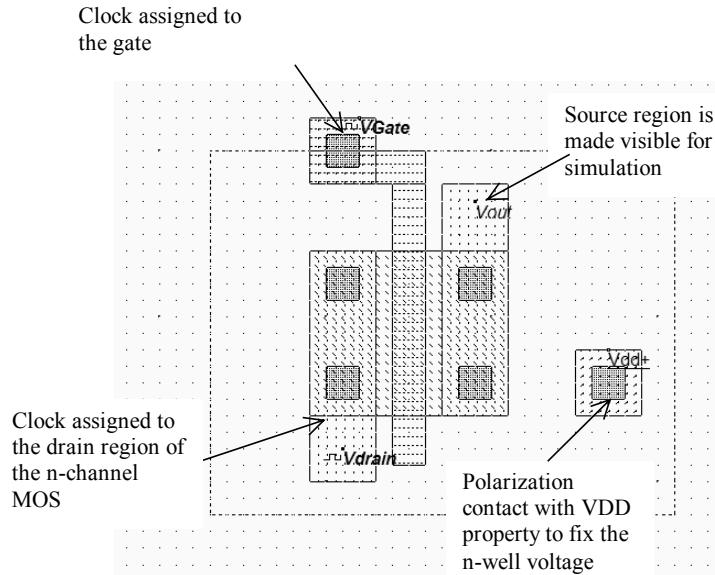


Fig.2- 35. Properties added to the layout for controlling the analog simulation of the p-channel device (Mosp.MSK)

Click on **Simulate → Run Simulation → Voltage vs. Time**. The timing diagrams of the pMOS device appear, as shown in Figure 2-36. The upper waveform correspond to  $V_{drain}$ , with a clock from 0 to 1.2V ( $V_{DD}$  is 1.2V in  $0.12\mu m$ ), exhibiting a 1ns period. Below is the gate voltage. The p-channel MOS is on when  $V_{gate}$  is zero, and off when  $V_{gate}$  is 1.2V. When the gate is on, the channel enables  $V_{out}$  to copy the value of  $V_{drain}$  (Time 0 to 1.0ns). Then, when the gate is off again, the voltage remains almost at its last value. The reason why  $V_{out}$  kept around 1.0V from 1.0 to 2.0ns is that the channel turned off synchronously with a change in the value of  $V_{drain}$ .

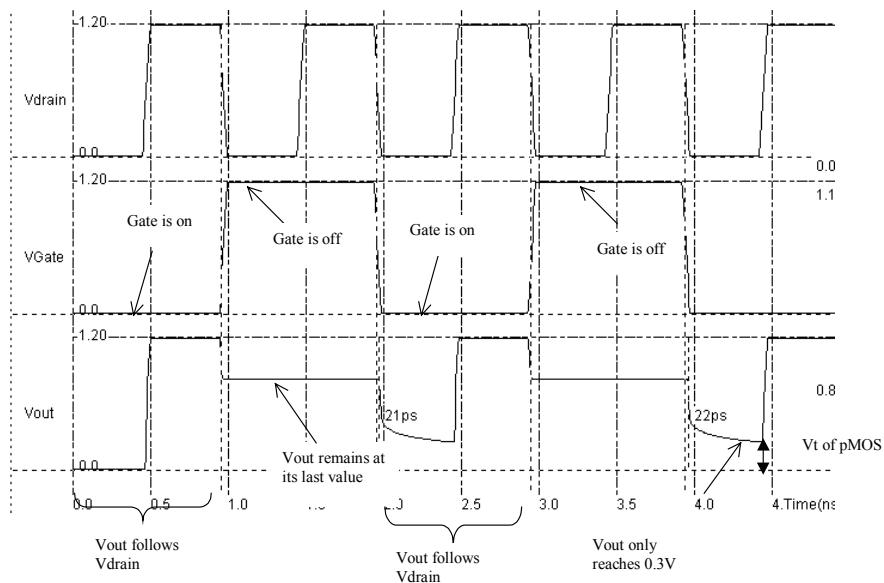


Fig. 2-36. Time domain simulation of the p-channel MOS (Mosp.MSK)

Notice that for the p-channel MOS, the voltage  $V_{out}$  never reaches 0.0V. It "saturates" to around 0.3V, due to the threshold voltage of the device. In summary (Figure 2-37), the p-channel MOS device behaves as a switch, but when on, it does not pass correctly the low voltages. A zero on one side leads to a poor zero, a logic 1 on one side leads to a good 1.

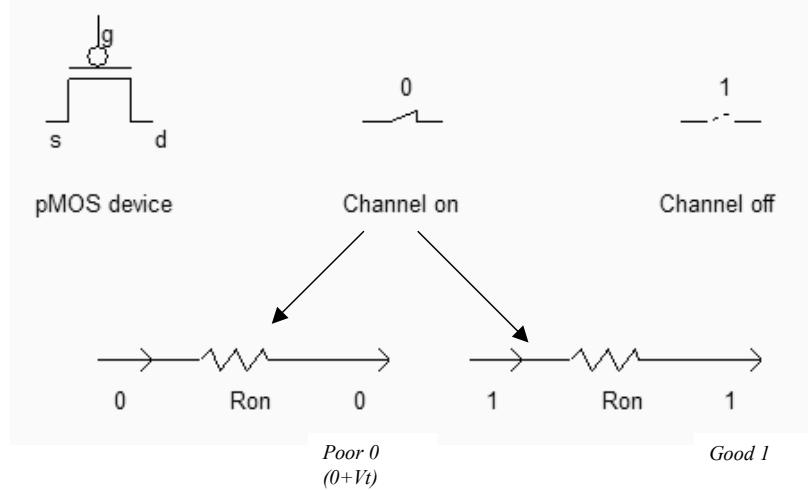
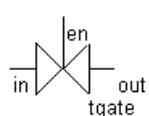
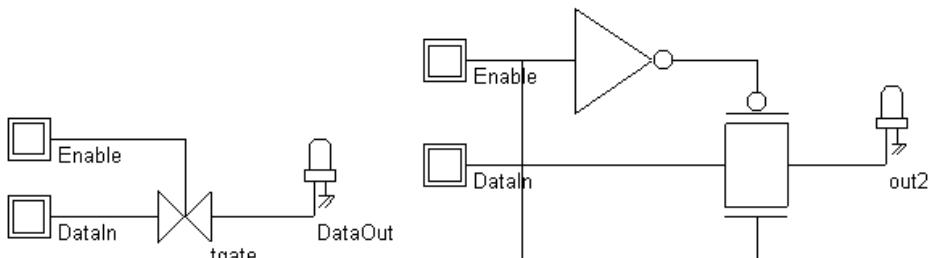


Fig. 2-37. Summary of the performances of a pMOS device

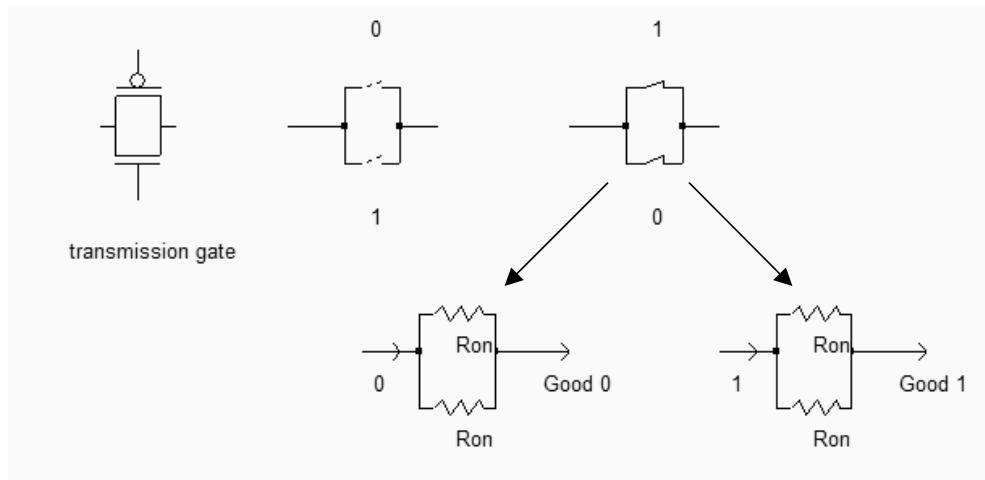
## 9. The Perfect switch

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 2-38, is called the transmission gate <gloss>. In DSCH2, the symbol may be found in the "advance" menu in the palette. The main drawback of the transmission gate is the need for two control signals Enable and /Enable, thus an inverter is required.



*Fig. 2-38. Schematic diagram of the transmission gate (Tgate.SCH)*

The transmission gate let a signal flow if  $en=1$  and  $\sim en=0$ . In that case both the n-channel and p-channel devices are on. The n-channel MOS transmits low voltage signals, while the p-channel device preferably transmits high voltage signals (Figure 2-39).



*Fig. 2-39. The transmission gate used to pass logic signals(Tgate.SCH)*

## Implementation

We need to create an electrical connection between the N+ and P+ regions, in order to comply with the schematic diagram shown in figure 2-39. The most efficient solution consists in using metal and contacts to create a bridge from the N+ region to the P+ region. Figure 2-40 shows the cross-section of the bridge.

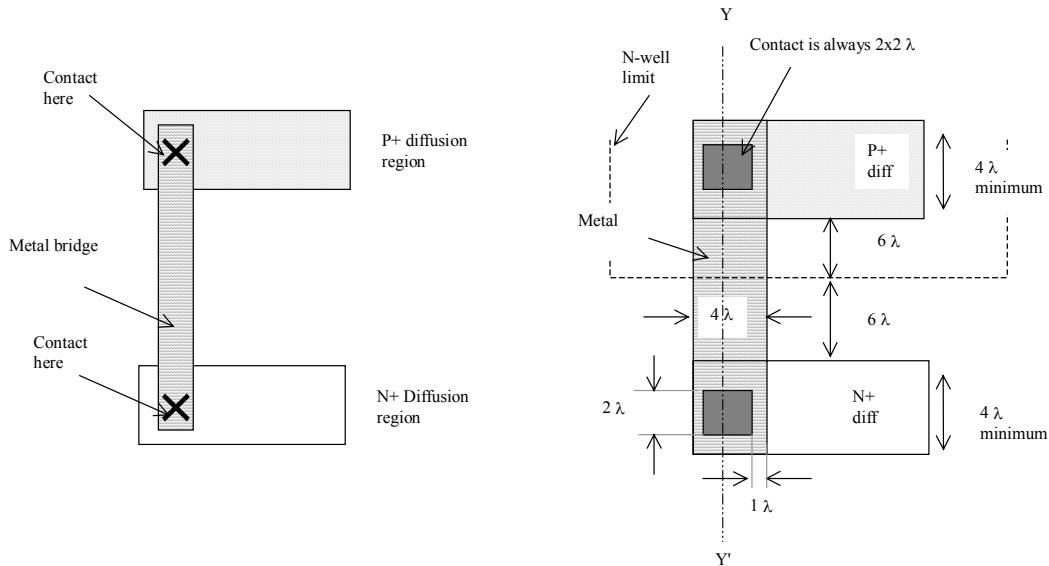


Fig. 2-40. Principles for metal bridge between  $N+$  diffusion and  $P+$  diffusion regions, and associated design rules.

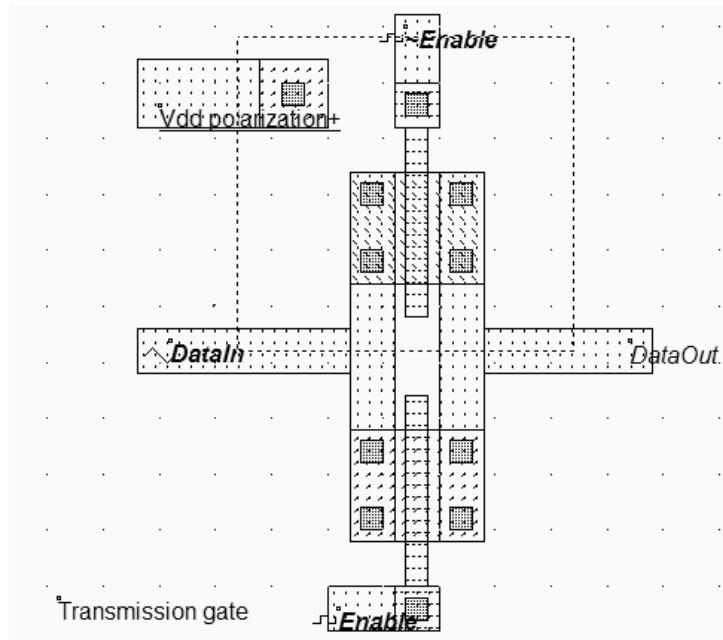


Fig. 2-41: Layout of the transmission gate (TGATE.MSK)

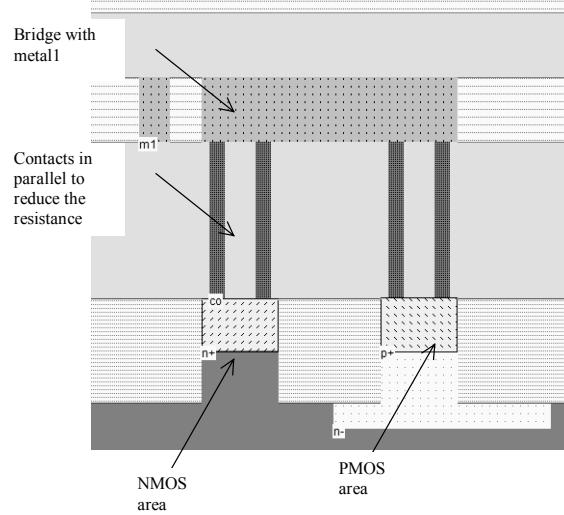


Fig. 2-42. Cross-section at location Y-Y' showing the metal bridge and contacts to N+ diffusion and P+ diffusion regions

The layout of the transmission gate is reported in figure 2-41. The n-channel MOS is situated arbitrarily on the bottom, and the p-channel MOS on the top. Notice that the gate controls are not connected, as  $\sim\text{Enable}$  is the opposite of  $\text{Enable}$ . The operation of the transmission gate is illustrated in figure 2-43. A sinusoidal wave with a frequency of 2GHz is assigned to *DataIn*. The sinusoidal property may be found in the palette of Microwind2, near the clock and pulse properties. With a zero on Enable (And a 1 on  $\sim\text{Enable}$ ), the switch is off, and no signal is transferred. When *Enable* is asserted, the sinusoidal wave appears nearly identical to the output.

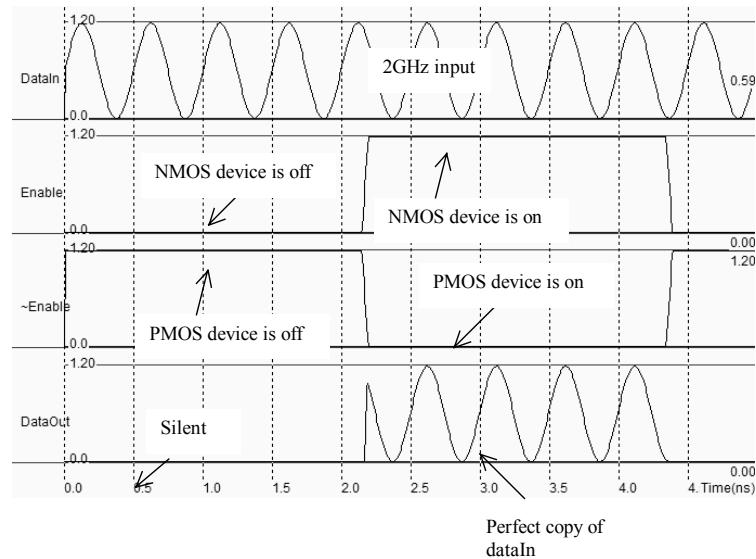


Fig. 2-43. Simulation of the transmission gate (TGATE.MSK)

## 10. Layout considerations

### MOS generation

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 main parameters are the MOS type (either n-channel or p-channel), the width and the length. By default, the proposed MOS is a n-channel device, with a width  $0.6\mu\text{m}$  and minimum length  $0.12\mu\text{m}$ . The maximum current that can flow in the MOS channel is given for information ( $0.262\text{mA}$  in that case). The units for width and length are in  $\mu\text{m}$  by default. You may change the units and use lambda values instead.

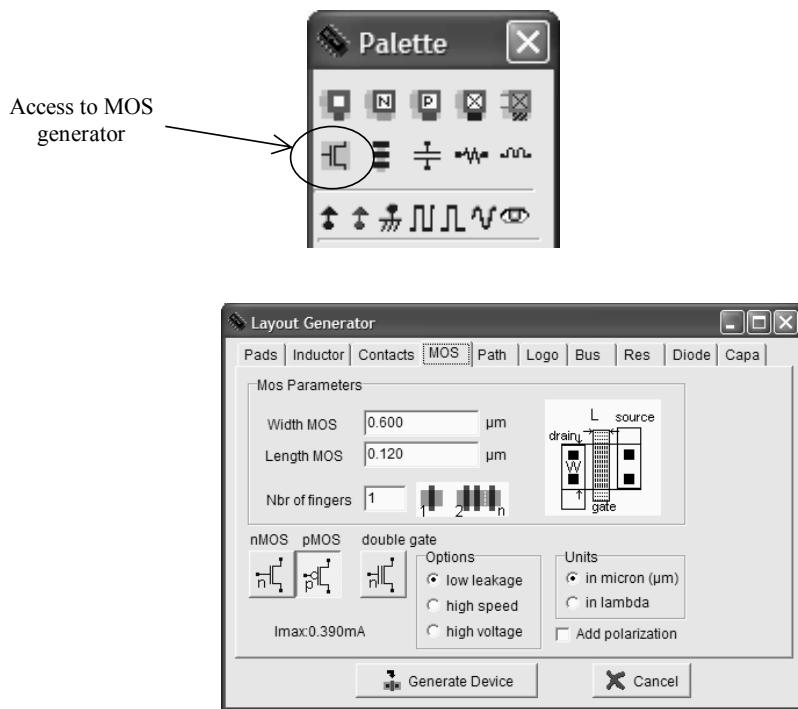


Fig.2-44. Access to the MOS generator menu

Starting  $0.18\mu\text{m}$  technology, three types of MOS devices have been made available: low leakage, high speed and high voltage devices. Those concepts will be developed in the next chapter.

If we increase the width of the MOS device to  $2\mu\text{m}$ , the layout generated by Microwind2 takes the aspect of figure 2-45. Notice the length equal to  $2\lambda$ , that is  $0.12\mu\text{m}$  in the default technology, the width equal to  $2\mu\text{m}$ .

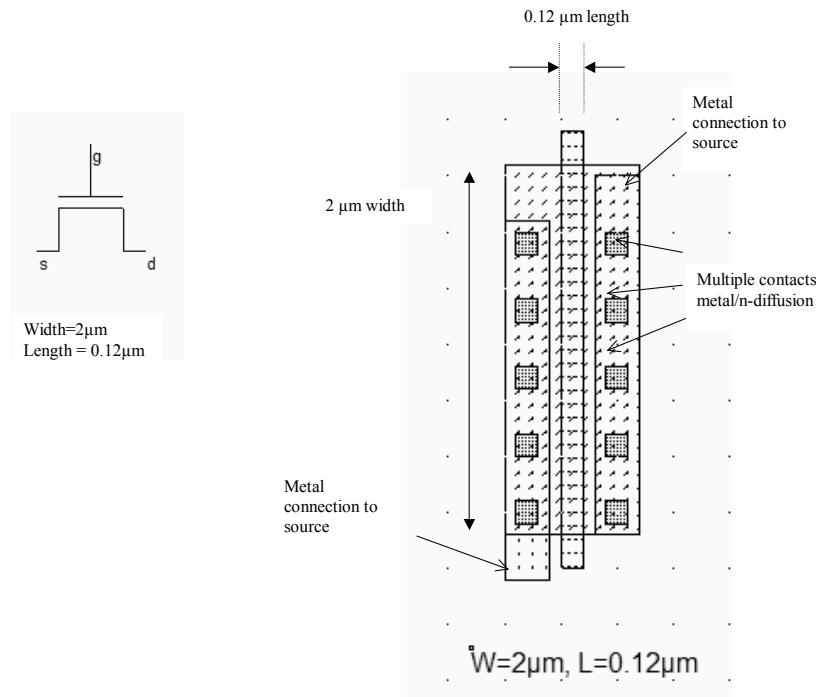


Fig.2-45. A n-channel MOS with  $2\mu\text{m}$  width generated by Microwind2(MosLayout.MSK)

### Multiple contacts

In the layout of figure 2-45, the surprise comes from the multiple contacts on the drain and source regions. The reason for this addition of contacts is due to the intrinsic current limitation of each elementary contact plug, as well as the high resistance of one single contact. One single contact can suffer less than 1mA current without any reliability problem. When the current is stronger 1mA, the contact can be damaged (Figure 2-46). The effect is called electromigration: if too much current flows within the contact, the metal structure starts to change as atoms move inside the conductor. A very strong current such as 10mA would even destroy one lonely contact.

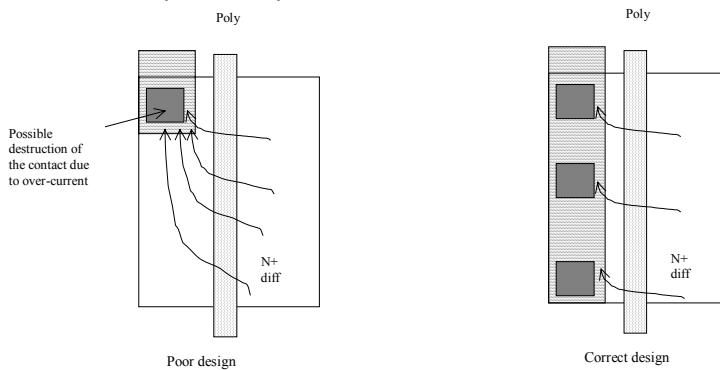
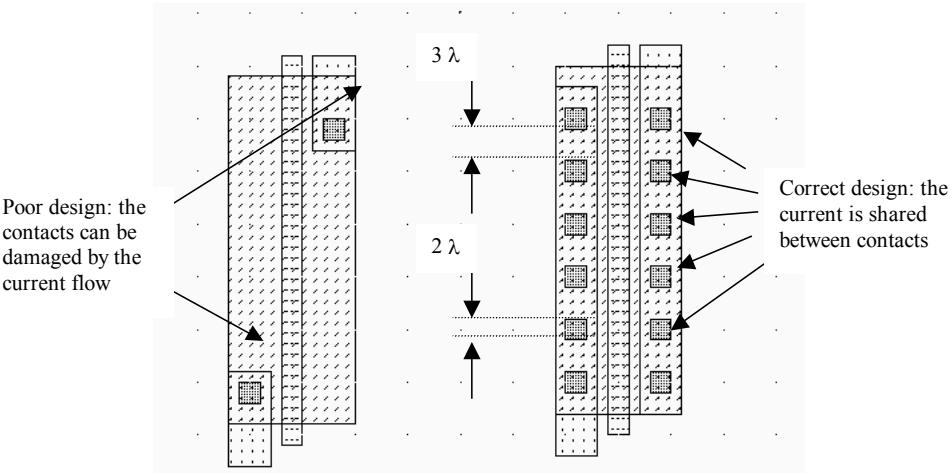
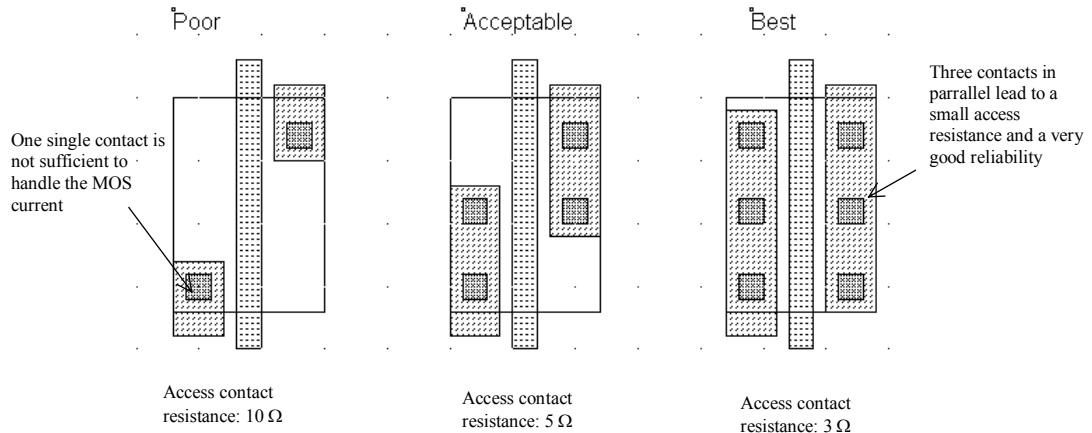


Fig.2-46. A strong current through a single contact could damage the metal structure.



*Fig.2-47. A single contact cannot handle more than  $1mA$ . A series of contacts is preferred (MosLayout.MSK)*

The illustration of this important limitation is given in figure 2-48. Basic rules for contact design are also reported in the figure. The contact is  $2x2 \lambda$ , and the separation is  $4 \lambda$ .



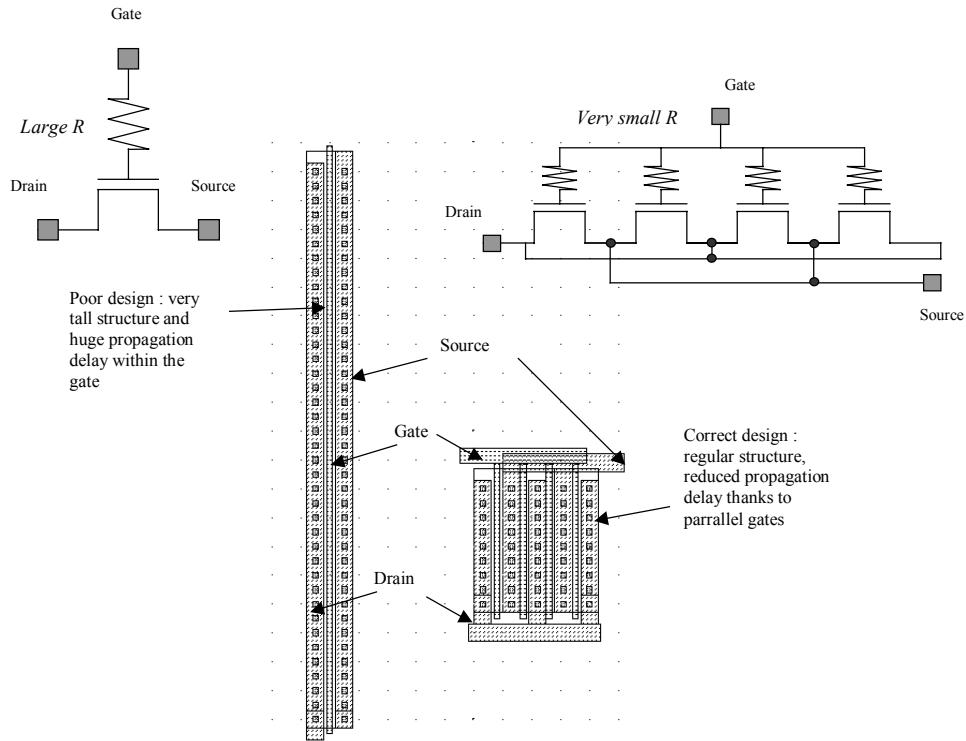
*Fig.2-48. A series of contacts also reduced the serial access resistance(MosContacts.MSK)*

Adding as many contacts as the design rules permit also limits the contact resistance. The equivalent resistance of the access to drain and source regions is reduced proportionally to the number of contacts.

### Multiple gates

The use of MOS devices with long width is very common, for example in analog design and buffer design such as clocks and interface structures. Let us try to design a  $5mA$  MOS switch., which can be found in a

standard output buffer structure. A rapid investigation using the maximum current evaluation in the MOS generator menu leads to the need of a MOS device with  $W=12\mu m$ ,  $L=0.12\mu m$ . The corresponding layout is reported in figure 2-49. The two main drawbacks of this layout are the very unpractical shape of the structure, and the important parasitic resistance along the polysilicon gate, that delays the propagation of the control voltage, thus slows down the switching of the device.



*Fig.2-49. MOS devices with large width must be designed with parallel gates to reduce the delay (MosLayout.MSK)*

The most efficient solution is to connect MOS devices in parallel. Firstly, the polysilicon gate length is divided by four in the case of figure 2-49, secondly, the structures becomes regular, easing the future interconnections. Notice how drains and sources are interleaved to create an equivalent device with the same channel width and length, and consequently the same current ability.

## 11. CMOS process

The process <gloss> steps to fabricate the integrated circuit are illustrated in this paragraph. The starting material is an extremely pure silicon substrate, entering in the foundry as a thin circular wafer. The wafer diameter for  $0.12\mu m$  technology is 8 inches. Most CMOS processes use lightly doped p-type wafers.

## Masks

The complete fabrication includes a series of chemical steps in order to diffuse doping materials (n-well, N+ and P+ diffusions), or to deposit materials (polysilicon, contacts and metals). The chemical attack is driven by optical masks <gloss> on which the patterns are drawn. The masks are listen in figure 2-50.

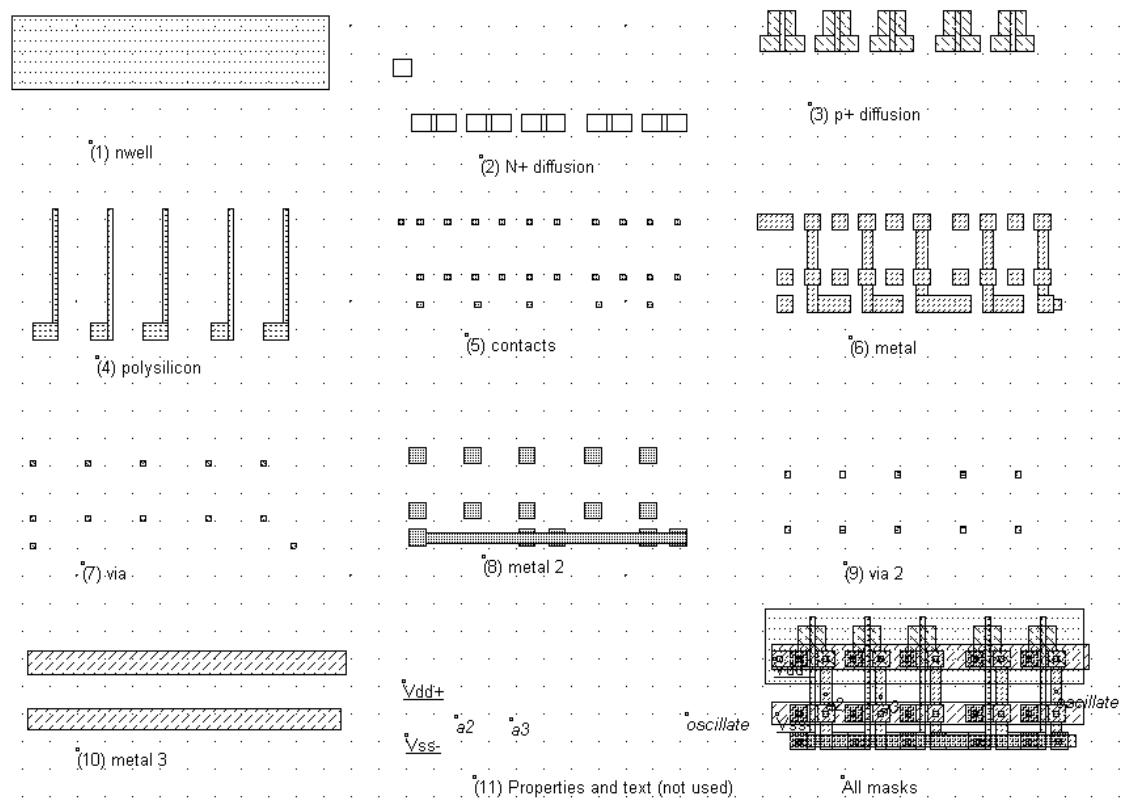


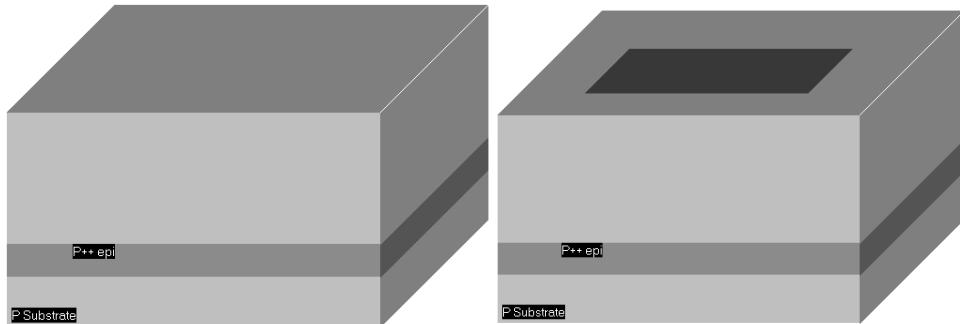
Fig. 2-50: Splitting the layers of the INV5 circuit into separate masks (Inv5Steps.MSK)

In deep submicron technologies, the price of these mask accounts for a significant percentage of the total cost of the chip fabrication. The reason is the extreme precision of each mask, which must have no defect at all, in order to succeed in fabrication the chip correctly. The complete set of masks in 0.12 $\mu$ m is around 100,000 euro. This price should rise significantly in nano-scale technologies (90nm and below), and become the primary limiting factor.

## CMOS process steps

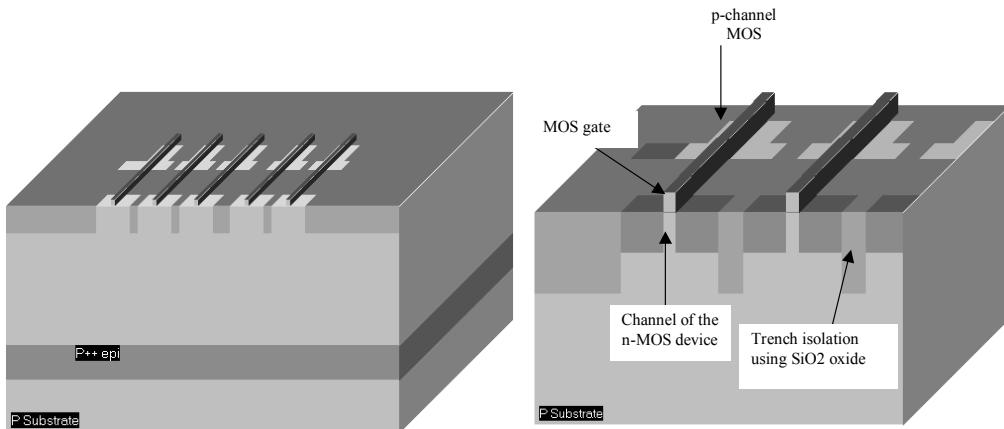
Let us load first the ring oscillator used in Chapter 1 to illustrate the operating frequency increase with the scale down (INV5.MSK). The masks can be split into 11 layers as described in figure 2-50. The

illustration of the process steps piloted by these masks is included in Microwind2. The access command is **Simulate → Process steps in 3D**.



*Fig. 2-51: The fabrication of the nwell (Inv5.MSK)*

A portion of the substrate is shown in figure 2-51. The substrate is a p-type wafer, with a resistivity around  $10 \Omega\text{cm}$ . A P++ layer is situated some microns below the surface of the wafer. Due to its very low resistivity, it serves as a ground plane. Not all CMOS processes use this P++ layer, mainly for cost reasons. In the 3D process window, click "Next step" to skip to the next technological step.

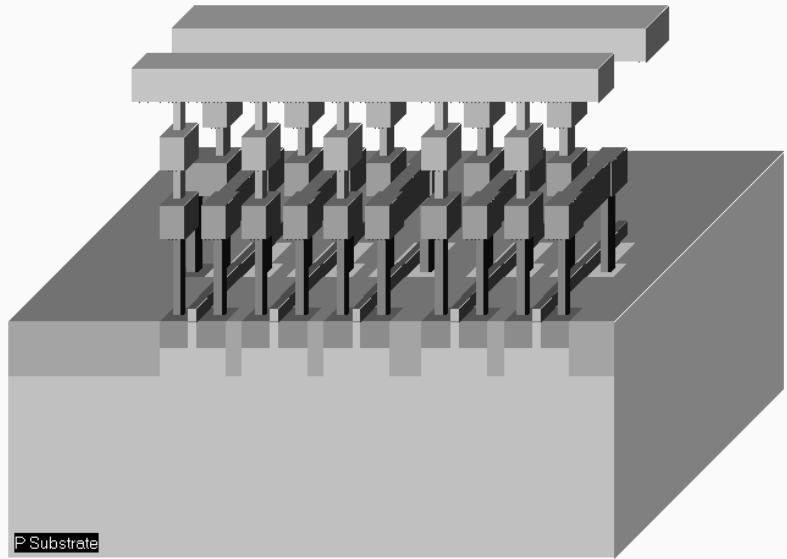


*Fig. 2-52: N+ diffusion, P+ diffusion (left) and polysilicon deposit (right) (Inv5.MSK)*

The n-well mask is used here to build the n-well area, into which p-channel devices will operate. N-channel devices use the native p-type substrate, without the need of a P-well. Next, a thick oxide is created to isolate MOS devices. In  $0.12\mu\text{m}$  this step is called shallow trench isolation <gloss>. Then, a crucial step consists in growing a very high quality, extremely thin oxide that isolates the gate from the channel. Extraordinary precautions are taken to create millions of atomic-scale gate oxides, around  $18 \text{ \AA}$  thick in  $0.12\mu\text{m}$ , that is 8 atoms of  $\text{SiO}_2$ . On the top of that ultra-thin oxide, the polysilicon gate is deposited (Figure 2-52).

Then, n-type dopant ions, usually using arsenic or phosphorus, are implanted to form the drain and source regions of the n-channel MOS devices. The polysilicon gates block the ions and prevent the underlying

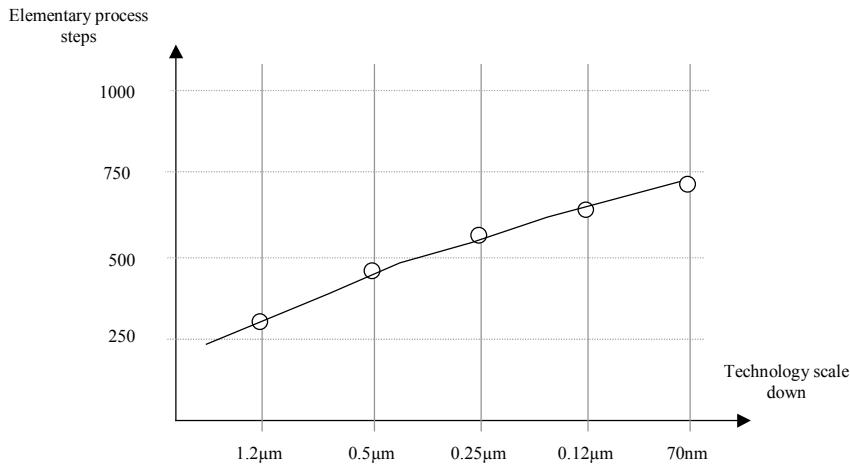
channel from any n+ dopant. The gate serve as a mask to separate the implanted area into two electrically different regions, the source and the drain. Consecutively, p-type boron ions are implanted to form the p-channel drain and sources.



*Fig. 2-53: Fabrication of the metal interconnects (oxide not shown for clarity) (Inv5.MSK)*

The next steps are related to metallization. Up to 6 metal layers can be fabricated, on the top of each other, in  $0.12\mu\text{m}$ . When the contact and metal steps are completed, the chip takes the aspect shown above. Notice that oxides have been removed for clarity (Figure 2-53). You may see the oxide in the window showing the process aspect in 2D, accessible from the Simulation menu of Microwind2.

At the end of the process comes the passivation <gloss>, consisting the growth an oxide, usually Si<sub>3</sub>N<sub>4</sub>. The structure of the oxides that separate the metal layers is not homogeneous. Recent advances in lithography have generalized the use of low dielectric oxides together with the traditional native oxide SiO<sub>2</sub>. The combinational use of SiO<sub>2</sub> and low dielectric oxide will be justified in chapter 5, dedicated to interconnects.



*Fig. 2-54: Increase of elementary process steps with the technology scale down.*

Each process step appearing in the process simulation window of Microwind2 is the sum of elementary chemical, mechanical and optical steps. Consequently, a complete technological process such as  $0.12\mu\text{m}$  is made up with more than 600 elementary steps. The global trend is the increase of technological steps with the technology scale down, as shown in figure 2-54. Therefore, the cost of fabrication is also increased accordingly.

## 12. Conclusion

In this chapter, we described the atomic structure of silicon, and gave some information about P-type and N-type materials. The silicon dioxide has also been investigated. Then, we presented the MOS device from a simple functional point of view, and focused on its switching properties. We presented the two types of MOS devices: n-channel and p-channel MOS. Though analog simulation, we exhibited the good and poor performances of these switches, depending on the logical information. We also proposed a good switch that takes advantage of nMOS and pMOS properties. Finally, we described the MOS layout editing using Microwind, shown how to conduct analog simulation by adding simulation properties such as clocks, supplies, and sinusoidal voltages directly on the layout. The final part of this chapter was dedicated to the CMOS process steps.

## REFERENCES

- [1] R.J Bakcer, H. W.Li, D. E. Boyce "CMOS design, layout and simulation", IEEE Press, 1998, chapters 3&4, [www.ieee.org](http://www.ieee.org)
- [2] M. John, S. Smidh, "Applicaqtion Specific Integrated Circuits", Addison Wesley, 1997, ISBN 0-201-50022-1, chapter 2, [www.awl.com/cseng](http://www.awl.com/cseng)

[3] B. Razavi "Design of Analog CMOS integrated circuits", McGraw Hill, ISBN 0-07-238032-2, 2001,

[www.mhhe.com](http://www.mhhe.com)

[4] C. Y. Chang, S.M. Sze "ULSI technology", McGraw Hill, 1996 ISBN 0-07-063062-3

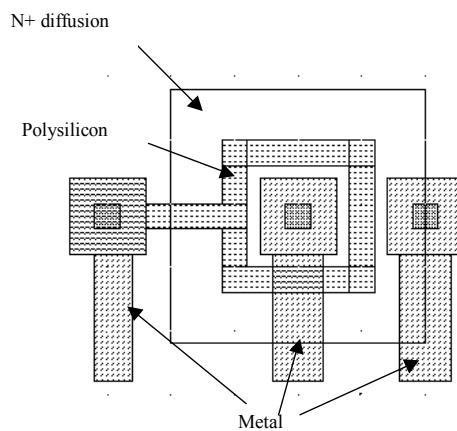
## EXERCISES

2.1 Layout a single gate n-channel MOS with  $W=1.0\mu m$ ,  $L=0.12\mu m$ . What is the maximum current?

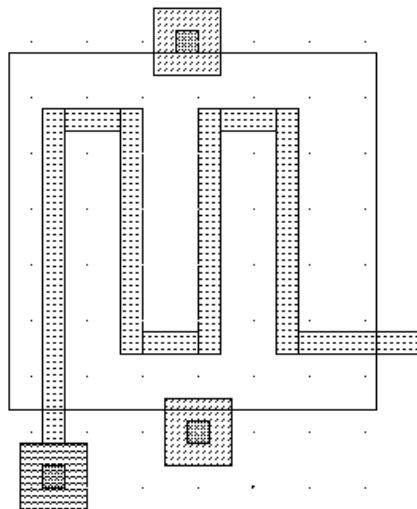
Build the equivalent MOS device with parallel gates.

2.2 Using the current evaluator in the MOS generator menu in Microwind2, find a simple relationship between the maximum current  $I_{max}$  and the width and length of the device.

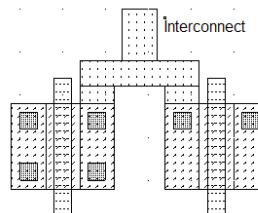
2.3 What is the equivalent width and length of the device drawn in the figure below? What is the possible application of such a design style ?



2.4 What is the width and length of this device? What is the main drawback of this design style?



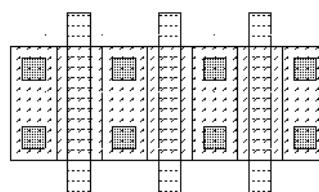
2.5 What is the maximum current which can flow within the interconnect without any damage ?



2.6 Add the appropriate layers to create the biggest MOS you can create with the following structure.

What is the MOS size ?

Find the maximum current which can flow within this MOS without any damage.

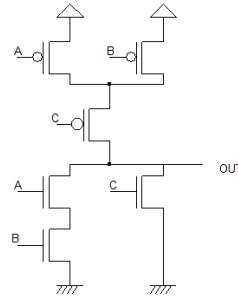


2.7 Consider a sample of silicon (at room temperature) doped pType with a boron density around  $10^{16} \text{ cm}^{-3}$ .

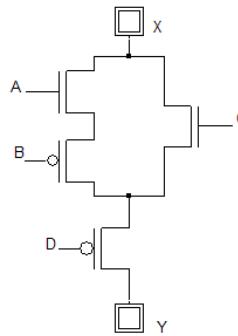
Find The number of minority carrier in this ptype sample.

$$\text{Anw: } N_p = 2.2 \cdot 10^4 \text{ cm}^{-3}$$

2.8 For each transistor NMOS and PMOS show the drain and the source.



2.9 Considering each transistor as a perfect switch, find all the combination allowing the connection between X and Y.



Anw: [ABCD]=[1000], [1010], [1110], [0110], [0010]

# 3 The MOS modeling

This chapter introduces the CMOS transistor modeling. The static characteristics of n-channel and p-channel MOS devices are shown, with details on the maximum current and its relationship with the sizing, the threshold voltage and various 2<sup>nd</sup> order effects. Three generations of MOS device models are introduced. Firstly, the original MOS model 1 is presented, as it was proposed in the early versions of SPICE simulator developed by the University of Berkeley, California. We demonstrate the inaccuracies of this model. Secondly, we introduce the semi-empirical model 3, which is still in use for MOS device simulation with a channel length greater than 1μm. Thirdly, we present a simplified version of the BSMI4 models, developed by the University of Berkeley for MOS devices with channel length down to 90nm.

Details on model parameters are provided for all models. The effects of temperature on the MOS performances are then presented. Finally, the three different MOS that may be found in 0.12μm are introduced: low threshold voltage, high speed and high voltage.

## 1. Introduction to modeling

Modeling the MOS device consists in writing a set of equations that link voltages and currents, in order to simulate and predict the behavior of the single device [Shockley] and consequently the behavior of a complete circuit. A considerable research and development effort has been dedicated in the past years to modeling MOS devices in an accurate way. Many books have been published over the years about the semiconductor physics and semiconductor device modeling. The most common references are [Tsividis], [Sze], [Lee] and recently [Liu]. For MOS devices, one of the key objectives of the model is to evaluate the current  $I_{ds}$  which flows between the drain and the source, depending on the supply voltages  $V_d, V_g, V_s$  and  $V_b$ .

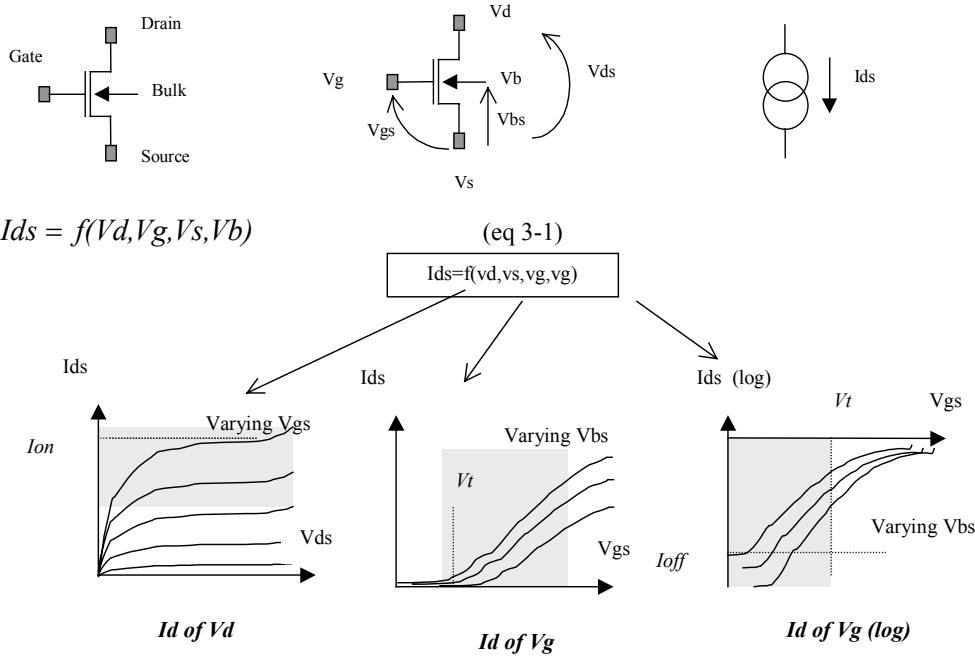


Figure 3-1: Useful representations of the MOS device characteristics

From the equation (3-1), we may represent the variation of the current  $Ids$  versus voltages in three different ways, as illustrated in figure 3-1. The graphs are usually called  $Id/Vd$ ,  $Id/Vg$ , and  $Id(\log)/Vg$ . For simplicity's sake, we consider that the voltage  $Vs$  is grounded.

In the  $Id/Vd$  curve, the current  $Ids$  is plotted for varying gate voltage  $Vgs$ , from 0 to VDD. The parameter  $Ion$  gives the maximum available current, corresponding to maximum voltage  $Vds$  and  $Vgs$ .  $Ion$  is a very important parameter for signal switching, for example in logic gates.

In the  $Id/Vg$  curve, we extract the threshold voltage. In the previous chapter we observed the parasitic effects due to this threshold. Analog design is much concerned by an accurate prediction of the threshold voltage.

Then, the curve  $Id(\log)/Vg$  is convenient to illustrate the current  $Ids$  for small values of the gate control. One of the most important parameters is the  $Ioff$  current, when  $Vg=0$ , that has a direct impact on standby power consumption.

A second objective of MOS models is to estimate the value of parasitic capacitances, mainly  $Cgs$ ,  $Cgd$  and  $Cgb$  (Figure 3-2). Those capacitance prove to vary with the voltage  $Vs$ ,  $Vd$ ,  $Vg$  and  $Vb$ . Although not considered in the static simulations  $Id/Vd$  and  $Id/Vg$ , the variation of the capacitance must be computed at each iteration of the analog simulation, for accurate prediction of the switching delay.

$$Cgs = f_1(Vd, Vg, Vs, Vb) \quad (\text{eq 3-2})$$

$$Cgd = f_2(Vd, Vg, Vs, Vb) \quad (\text{eq 3-3})$$

$$Cgb = f_3(Vd, Vg, Vs, Vb) \quad (\text{eq 3-4})$$

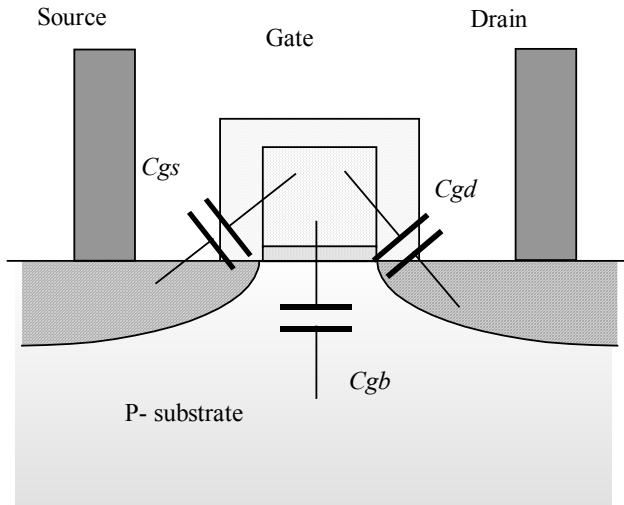


Figure 3-2: Capacitance between the gate and the source, drain, or substrate

A long list of MOS models have been developed for analog simulators. We choose to implement in Microwind2 three of those: the model 1, the model 3 and the model BSIM4. Details on those three models and their physical basis are provided in the next paragraphs.

The complete set of parameters for a given technology is called the model card. The procedure to build an accurate MOS model is quite complex, as it is based on a large set of measurements and sophisticated optimization procedures. The experimental data concerning a MOS device with large width and large length is used first, to fix basic parameters. Then the MOS model is tuned for small channel device measurements, and then for several sizes.

## 2. MOS Model 1

### Equations

Historically, the MOS model 1 was the first to be proposed by Shockley, in 1952 [Shockley]. The equations of the MOS level 1 are provided in the next paragraphs. The evaluation of the current  $I_{ds}$  between the drain and the source as a function of  $V_d, V_g$  and  $V_s$  is summarized in equations 3-5, 3-6 and 3-7. The model parameters appearing in the user interface of Microwind2 are written using COURRIER font. The device operation is divided into three regions: cut-off, linear and saturated.

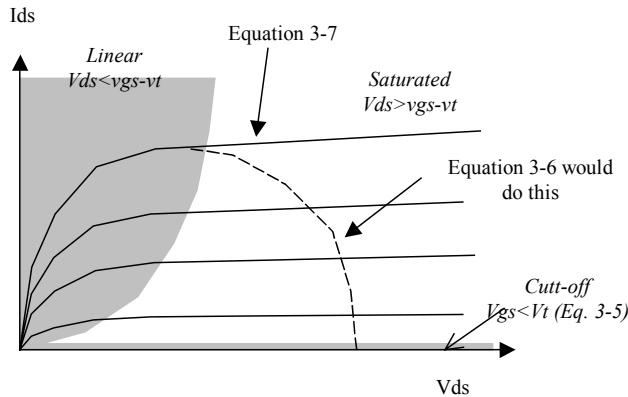


Figure 3-3: Two main domains are considered in the model: the linear area and the saturated area.

IF  $V_{gs} < 0$ , the device is in cut-off mode.

$$I_{ds} = 0 \quad (3-5)$$

IF  $V_{ds} < V_{gs} - V_{TO}$ , the device is in linear mode

$$I_{ds} = U_0 \frac{\epsilon_0 \epsilon_r}{T_{OX}} \cdot \frac{W}{L} ((V_{gs} - V_t) \cdot V_{ds} - \frac{(V_{ds})^2}{2}) \quad (3-6)$$

IF  $V_{ds} > V_{gs} - V_{TO}$ , the device is in saturated mode:

$$I_{ds} = U_0 \frac{\epsilon_0 \epsilon_r}{T_{OX}} \cdot \frac{W}{L} (V_{gs} - V_t)^2 \quad (3-7)$$

With:

$$V_t = V_{TO} + \text{GAMMA}(\sqrt{(\text{PHI} - v_{bs})} - \sqrt{\text{PHI}}) \quad (3-8)$$

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

$\epsilon_r$  = relative permittivity, equal to 3.9 in the case of SiO<sub>2</sub> (no unit)

Mos Model 1 parameters			
Parameter	Definition	Typical Value 0.12μm	
		NMOS	PMOS
V <sub>TO</sub>	Threshold voltage	0.4V	-0.4V
U <sub>0</sub>	Carrier mobility	0.06m <sup>2</sup> /V-s	0.02m <sup>2</sup> /V-s
T <sub>OX</sub>	Gate oxide thickness	2nm	2nm
PHI	Surface potential at strong inversion	0.3V	0.3V
GAMMA	Bulk threshold parameter	0.4 V <sup>0.5</sup>	0.4 V <sup>0.5</sup>
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

### Implementation in Microwind

The static characteristics of the MOS model 1 may be obtained using the command **Simulate → Mos characteristics** available in the main menu of Microwind.

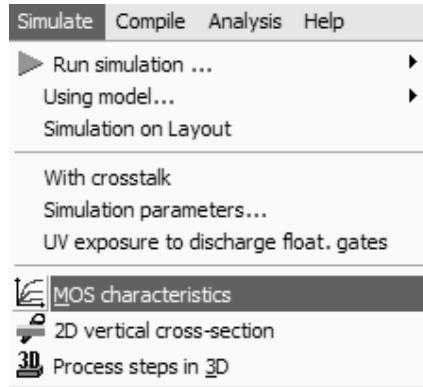


Figure 3-4: Access to the static MOS characteristics

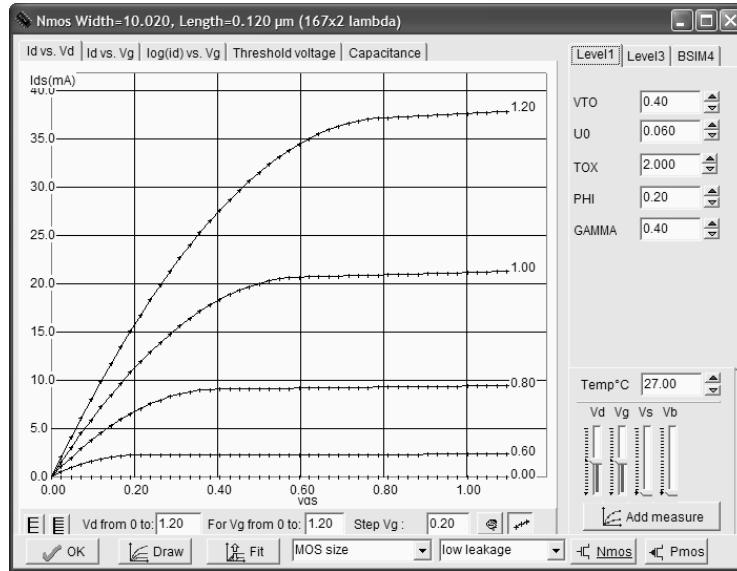


Figure 3-5: The screen used to simulate the static characteristics of the MOS with model 1 within Microwind2

In the top right part of the window, select the item "Level 1". The variation of  $Ids$  versus the voltage  $Vds$ , for varying gate voltage  $Vgs$ , is shown by default. The device width is 10 $\mu m$  by default, the channel length is 0.12 $\mu m$ . The parameters  $VTO$ ,  $U_0$ ,  $TOX$ ,  $PHI$  and  $GAMMA$  are listed in the right part of the window.

### Mismatch between simulation and measurements

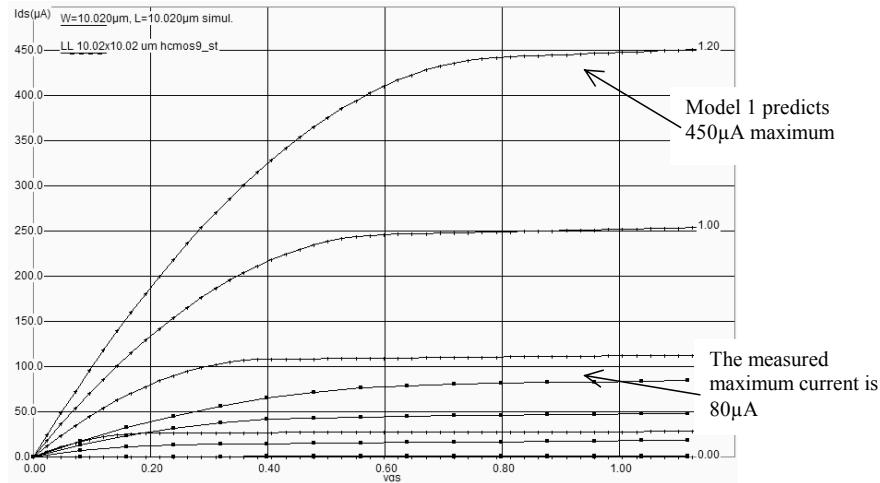


Figure 3-6: The model 1 predict a current 5 times higher than the measurement in the case of a large channel MOS device ( $L=10\mu\text{m}$ ).

These old equations (1968, in [Shichman]) are not acceptable in  $0.12\mu\text{m}$ . If we consider MOS devices with very long lengths ( $L>10\mu\text{m}$ ), the mismatch between the simulation and the measurement is of the order of a factor of five. Let us compare the simulation and the measurement, for a device with a width  $W=10\mu\text{m}$ , and a long channel length  $L=10\mu\text{m}$ , fabricated in  $0.12\mu\text{m}$  CMOS technology, as presented in figure 3-6. The measurement *Ne10x10.MES* was downloaded using the button **Load Measurement**. This measurement corresponds to an n-channel MOS device with a  $10\mu\text{m}$  channel width and  $10\mu\text{m}$  length, fabricated in CMOS  $0.12\mu\text{m}$  from ST-microelectronics.

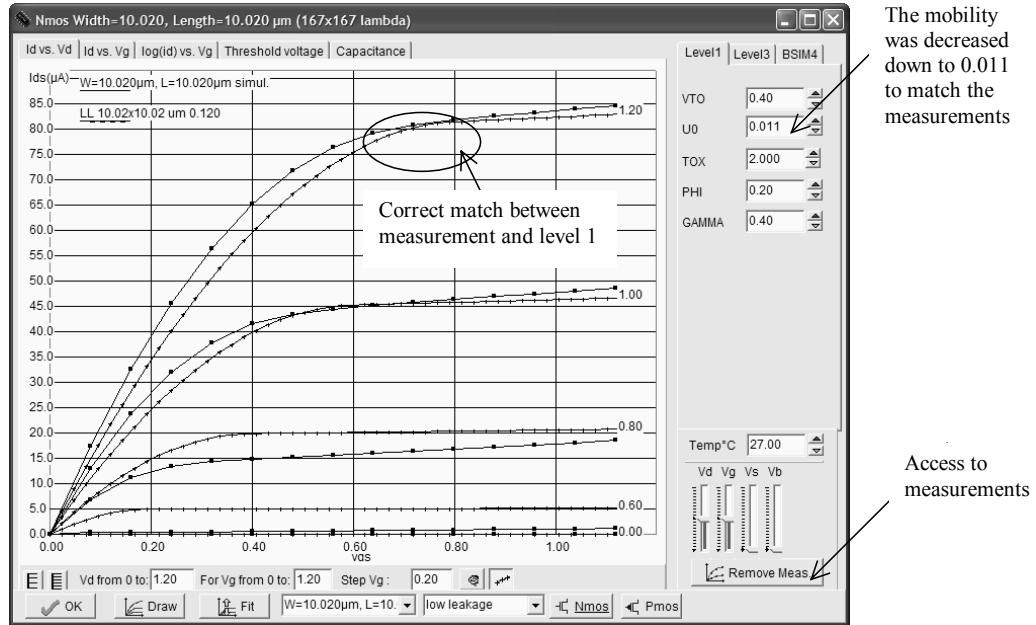


Figure 3-7: Comparing measured  $Id/Vd$  and level 1 simulations for a  $10 \times 10 \mu\text{m}$  device result in a surprising similarity (Ne10x10.MES).

Initially, the simulation and measurement do not correspond at all. The mobility  $U_0$  needs to be decreased from its initial value 0.06 down to 0.01. The curves are fitted at the price of an unrealistic change in the mobility parameter.

When dealing with sub-micron technology, the current predicted by model "Level 1" is several times higher than the real-case measurements. This means that several parasitic effects appeared with the technology scale down, most of them tending to reduce the effective current compared to the early modeling equations of model 1.

### 3. MOS Model 3

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

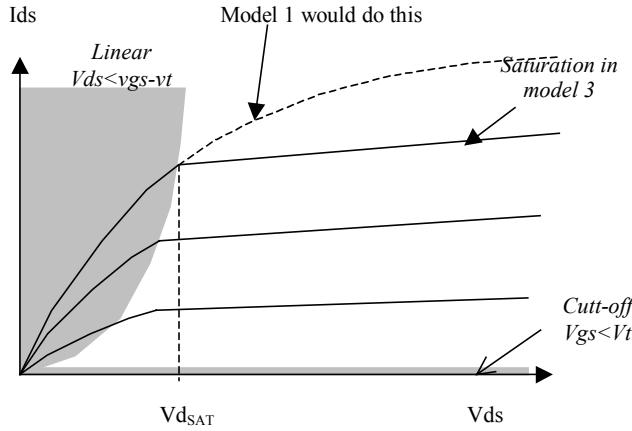


Figure 3-8: Introduction of the saturation voltage  $V_{dSAT}$  which truncates the equations issued from model 1

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. This saturation effect is significant for small channel length. The main LEVEL3 equations are listed below.

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

$$I_{ds} = 0 \quad (3-9)$$

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

$$I_{ds} = K_{eff} \frac{W}{L_{eff}} (1 + \text{KAPPA} \cdot V_{ds}) V_{de} ((V_{gs} - V_{th}) - \frac{V_{de}}{2}) \quad (3-10)$$

with

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

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

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

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

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

$$V_c = \text{VMAX} \frac{L_{eff}}{0.06}$$

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

The formulation of the effective factor  $K_{eff}$  (Equation 3-11) includes a mobility degradation factor  $\Theta$ , which tends to reduce the mobility at high  $V_{gs}$ . The consequence is a reduction of the current  $I_{ds}$  as compared to LEVEL1.

$$K_{eff} = \frac{\epsilon \sigma \epsilon_r}{TOX} \frac{U_0}{(1 + \Theta(V_{gs} - V_{th}))} \quad (3-11)$$

In sub-threshold mode, that is for a gate voltage less than the threshold voltage,  $V_{ds}$  is replaced by  $V_{on}$  in the above equations. An exponential dependence of the current with  $V_{gs}$  is introduced by using the equation 3-12. Notice the temperature effect introduced in the denominator  $nkT$ .

Without any voltage applied to the gate, the current is no more equal to zero. The current of  $I_{ds}$  for  $V_{gs}=0$  is called the  $I_{off}$  current (Figure 3-9). Its value in  $0.12\mu m$  is around  $10^{-10} A$ . In contrast, for  $V_{gs}=VDD$ , the maximum current  $I_{on}$  is of the order of several mA ( $10^{-3} A$ ).

$$I_{ds} = I_{ds}(V_{on}, V_{ds}) \exp\left(\frac{q(V_{gs} - V_{on})}{nkT}\right) \quad (3-12)$$

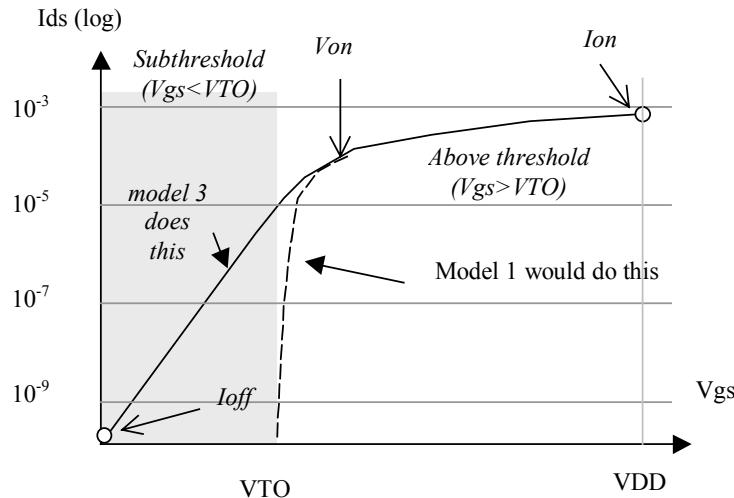


Figure 3-9: Introduction of an exponential law to model the sub-threshold behavior of the current

## TEMPERATURE EFFECTS

The MOS device is sensitive to temperature. Three main parameters are concerned: the threshold voltage  $VTO$ , the mobility  $U_0$  and the slope in sub-threshold mode dependent on  $kT/q$ . Both  $VTO$  and  $U_0$  decrease when the temperature increases. The physical background is the degradation of the mobility of electrons

and holes when the temperature increase, due to a higher atomic volume of the crystal underneath the gate, and consequently less space for the current carriers. The modeling of the temperature effect is as follows:

$$U_0 = U_{0(T=27)} \left( \frac{T+273}{300} \right)^{-1.5} \quad (\text{eq. 3-13})$$

$$V_T = V_{T0(T=27)} - 0.002(T - 300) \quad (\text{eq. 3-14})$$

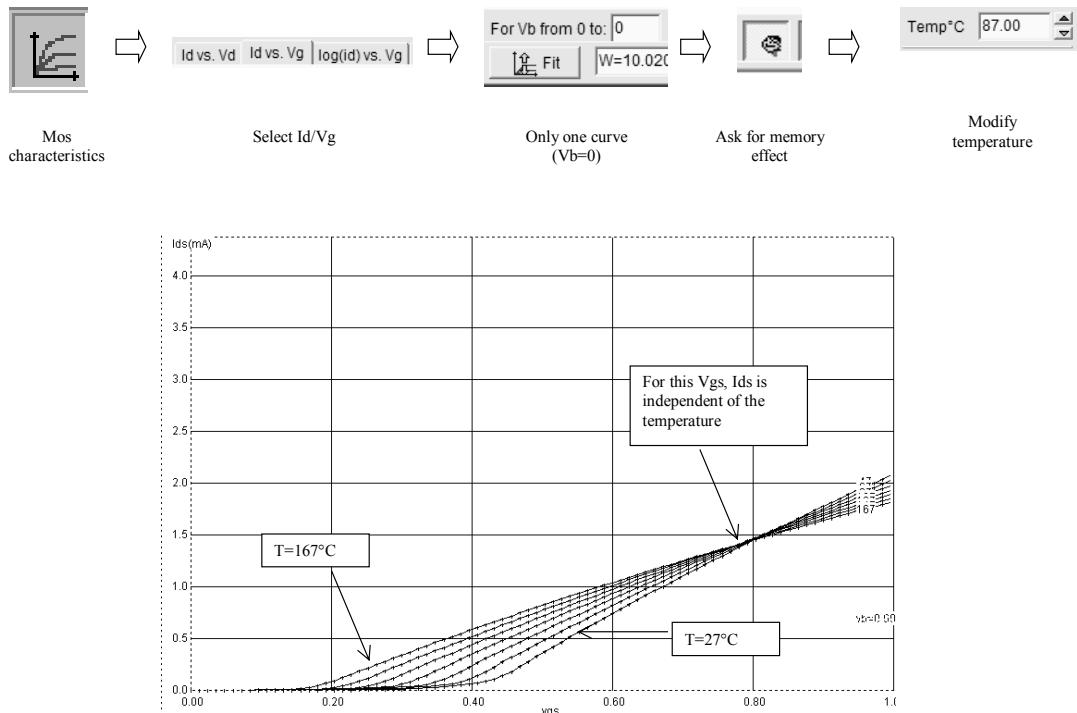


Figure 3-10 The effect of temperature on the MOS characteristics. In Id/Vg mode, a specific Vds makes the current independent of the temperature.

To obtain the curve of figure 3-10, click the icon MOS characteristics, select the curve *Id/Vg*, and enter the value "0" for the upper limit of *Vb*, so as to draw only one single curve. Enable the screen memory mode by a click on the icon **Enable Memory**. When you change the temperature, the change in the slope and the temperature-independent point appear, as shown in figure 3-10.

Mos Model 3 parameters			
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 <sup>2</sup> /V.s	0.025 m <sup>2</sup> /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	$0.01\mu\text{m}$	$0.01\mu\text{m}$
GAMMA	Bulk threshold parameter	$0.4\text{ V}^{0.5}$	$0.4\text{ V}^{0.5}$
KAPPA	Saturation field factor	$0.01\text{ V}^{-1}$	$0.01\text{ V}^{-1}$
VMAX	Maximum drift velocity	$150\text{ Km/s}$	$100\text{ Km/s}$
THETA	Mobility degradation factor	$0.3\text{ V}^{-1}$	$0.3\text{ V}^{-1}$
NSS	Subthreshold factor	$0.07\text{ V}^{-1}$	$0.07\text{ V}^{-1}$
W	MOS channel width	$0.5\text{-}20\mu\text{m}$	$0.5\text{-}40\mu\text{m}$
L	MOS channel length	$0.12\mu\text{m}$	$0.12\mu\text{m}$

Table 3-xxx: list of parameters used in the implementation of the LEVEL3 model in Microwind2

### Microwind User's Interface

You may understand the action of each parameter by using the screen reported in figure 3-11. Each parameter may be changed interactively using cursors, or by entering with the keyboard the appropriate value.

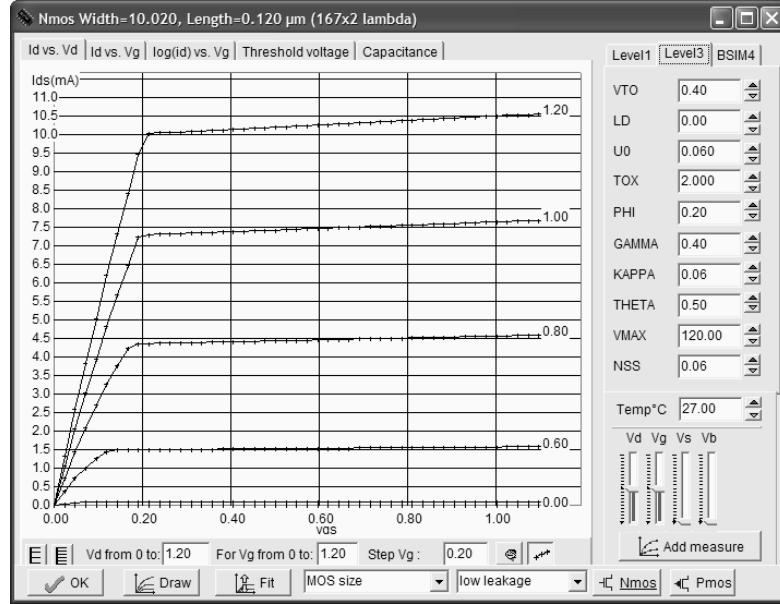


Fig. 3-11. The user interface to investigate the effect of each parameter on the current  $Id_s$  ( $W=10\mu\text{m}$ ,  $L=0.12\mu\text{m}$ )

Several screens may be proposed:

- $Id_s$  vs.  $V_d$ , for varying  $V_g$ . This is the default screen. Its main interest is the characterization of the Ion current, the maximum current available in the device, for  $V_d$  and  $V_g$  set to VDD.
- $Id_s$  vs.  $V_g$ , for varying  $V_b$ . In this screen, the threshold voltage  $V_t$  (VTO) is characterized, as well as its dependence on the bulk polarization.
- $Id_s$  vs.  $V_g$ , in logarithmic scale. This screen is mandatory to characterize the MOS device in sub-threshold mode, that is for  $V_{gs} < V_t$ . Two of the important parameters are the slope of the current vs.

$V_{gs}$ , and the  $I_{off}$  current. The  $I_{off}$  current is the standby current appearing between drain and source for  $V_{gs}=0$ .

- Threshold voltage  $V_t$  vs. Length. This screen has been added to illustrate advances in the modeling of deep-submicron effects. With LEVEL 3,  $V_t$  is constant for varying length, but is impacted by the bulk voltage.
- Capacitance vs.  $V_{ds}$ . This screen illustrates the variation of  $C_{gs}$  and  $C_{gd}$  versus the drain-source voltage.

### Current versus drain-source voltage

Using the display mode **Id vs. Vd**, you may see the effect of parameters  $U_0$ ,  $TOX$ ,  $KAPPA$  and  $VMAX$ . Basically, the carrier mobility  $U_0$  moves the whole curve, as it impacts the current  $Ids$  in an almost linear way. As  $U_0$  is nearly a physical constant, a significant change of mobility has no physical meaning. The oxide thickness  $TOX$  does the same but in an opposite way.

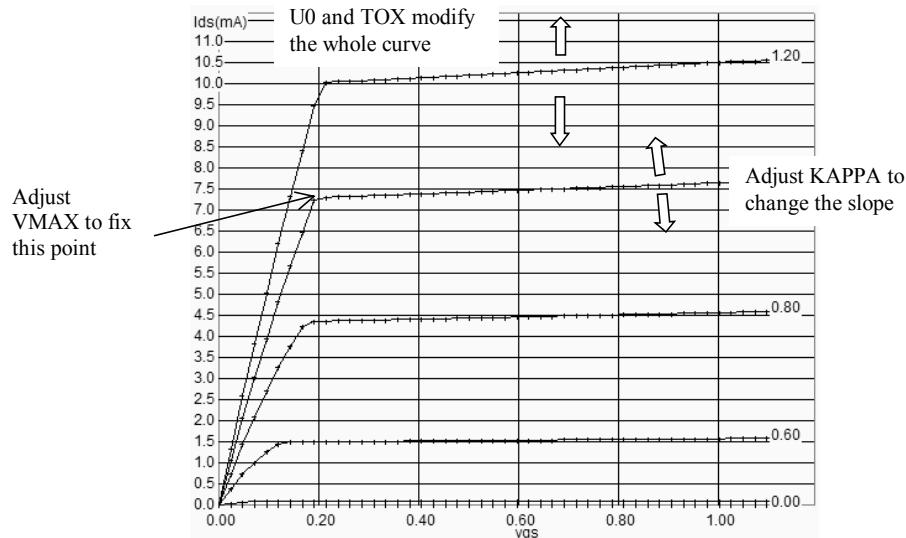


Fig. 3-12. Demonstration of the role of  $U_0$ ,  $KAPPA$  and  $VMAX$  in  $Id/Vd$  ( $W=10\mu m$ ,  $L=0.12\mu m$ )

A  $TOX$  increase leads to a less efficient device, with less current.  $KAPPA$  changes the slope of the current when  $V_{ds}$  is high, corresponding to the saturation region. Finally,  $VMAX$  truncates the curves for low values of  $V_{ds}$ , to fit the transition point between the linear and the saturated region (Figure 3-12).

### Current versus gate voltage

The role of VTO and GAMMA can be observed in the figure 3-13, using the display mode **Id vs. Vg**. If we use a long channel device, that is a length much greater than the minimum length, the second order effects are minimized. Act on VTO cursors in order to shift the curves right or left, and GAMMA to fit the spacing between curves. Parameters U0 and TOX also have a direct impact on the slope for high  $V_{gs}$ .

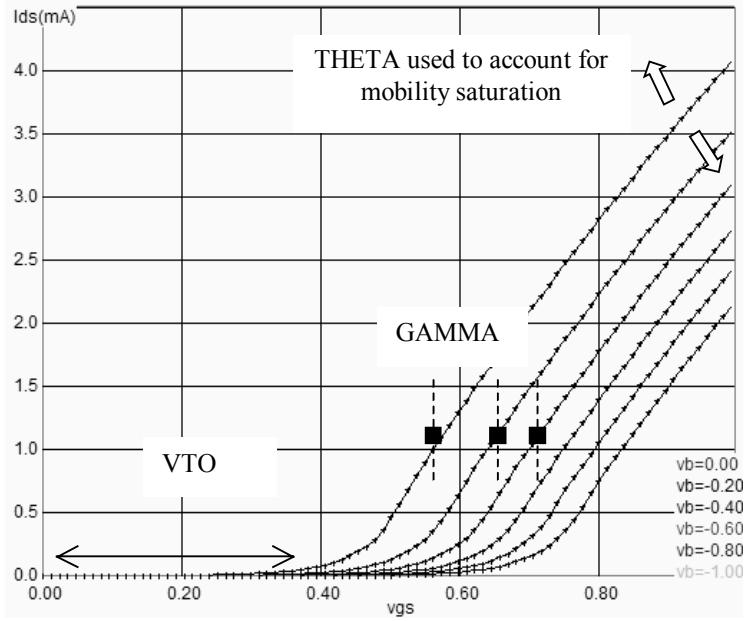


Fig. 3-13. The effects of VTO and GAMMA are illustrated in Id/Vg mode voltage ( $W=10\mu m$ ,  $L=0.12\mu m$ )

Now we focus on a short channel MOS device, for example  $W=2\mu m$ ,  $L=0.12\mu m$ . Using the same display mode **Id vs. Vg**, we obtain similar curves as for long-channel device. We observe that the shape of the current is bent. This modification is due to short channel parasitic effects. The parameter THETA is used to bend the current curves at high  $V_{gs}$ . The MOS model 3 do not provide parameters to account for the VTO dependence with length.

### Current vs Vg in logarithmic scale

We finally illustrate the role of NSS in the display mode **Id(log)/Vg** (Figure 3-14). The parameter NSS has a direct impact on the slope in sub-threshold mode, that is for  $V_{gs} < V_t$ .

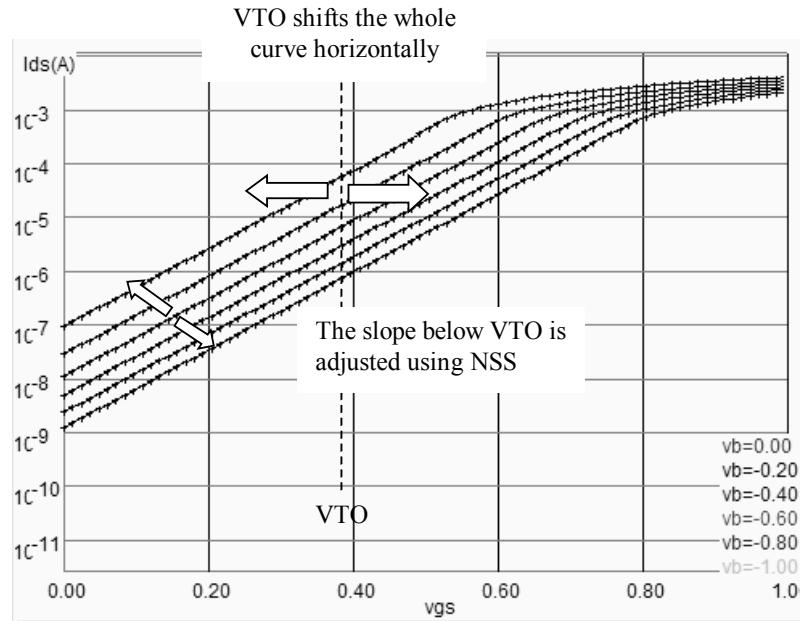


Figure 3-14. In sub-threshold region, the  $Id$  dependence on  $V_{gs}$  is exponential. The slope is tuned by parameter NSS. The whole curve is shifted using VTO voltage ( $W=10\mu\text{m}$ ,  $L=0.12\mu\text{m}$ )

### Capacitance vs. Vds

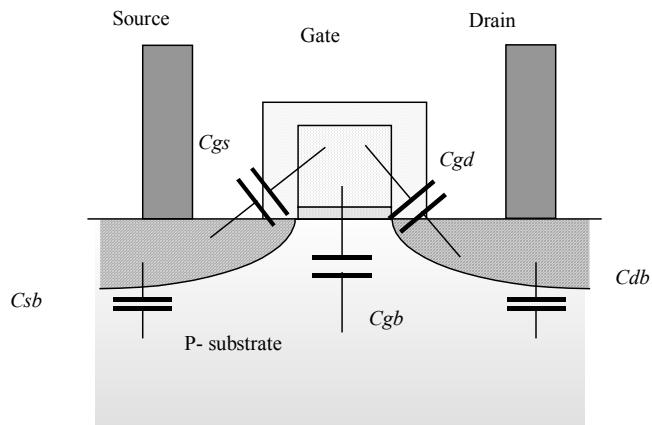


Figure 3-15: The MOS capacitance considered in MOS model 3

The five main capacitors considered in our implementation of MOS model 3 are the gate to bulk capacitance  $C_{gb}$ , the gate to source capacitance  $C_{gs}$ , the gate-to-drain capacitance  $C_{gd}$ , the junction capacitance between source and bulk  $C_{sb}$  and the junction capacitance between drain and bulk  $C_{db}$ .

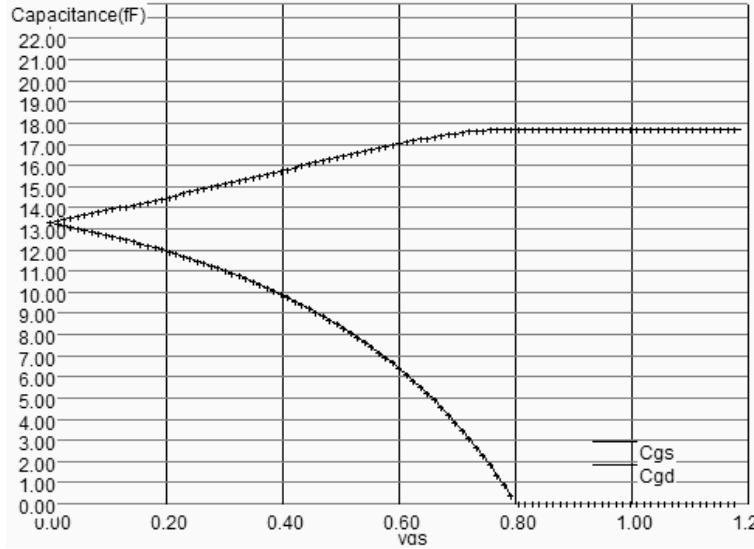


Figure 3-16: The evolution of MOS capacitance with the drain voltage ( $W=10\mu m$ ,  $L=0.12\mu m$ )

The variation of the capacitance must be computed at each iteration of the analog simulation, for accurate prediction of the switching delay. In our implementation of MOS level 3, we use the following model, based on the formulations given in [Fjedly]. The parameter  $V_{dsat}$  was given by equation 3-10.

$$C_{GS} = \frac{2}{3} C_i \left[ 1 - \left( \frac{V_{GS} - V_t - V_{dsat}}{2(V_{GS} - V_t) - V_{dsat}} \right)^2 \right] \quad (3-15)$$

$$C_{GD} = \frac{2}{3} C_i \left[ 1 - \left( \frac{V_{GS} - V_t}{2(V_{GS} - V_t) - V_{dsat}} \right)^2 \right] \quad (3-16)$$

$$C_{GB} = 0 \quad (3-17)$$

with

$$C_i = W \cdot L \cdot \frac{\epsilon_0 \epsilon_r}{TOX} \quad (3-18)$$

$W$  = width of the MOS device (m)

$L$  = length of the MOS device (m)

$TOX$  = oxide thickness (m)

The two remaining capacitance  $C_{DB}$  and  $C_{SB}$  are junction capacitance. Their model is given by equations 3-19 and 3-20.

$$C_{DB} = W \cdot L_{drain} \frac{C_J}{\left( 1 - \frac{V_{BD}}{P_B} \right)^{M_J}} \quad (3-19)$$

$$C_{SB} = W \cdot L_{source} \frac{C_J}{\left(1 - \frac{V_{BS}}{P_B}\right)^{M_J}} \quad (3-20)$$

where

$W$  is the channel width (m)

$L_{drain}$  is the drain length, according to figure 3-17 (m)

$C_J$  is around  $3 \times 10^{-4}$  F/m<sup>2</sup>

$P_B$  is the built-in potential of the junction (around 0.8V)

$M_J$  is the grading coefficient of the junction (around 0.5)

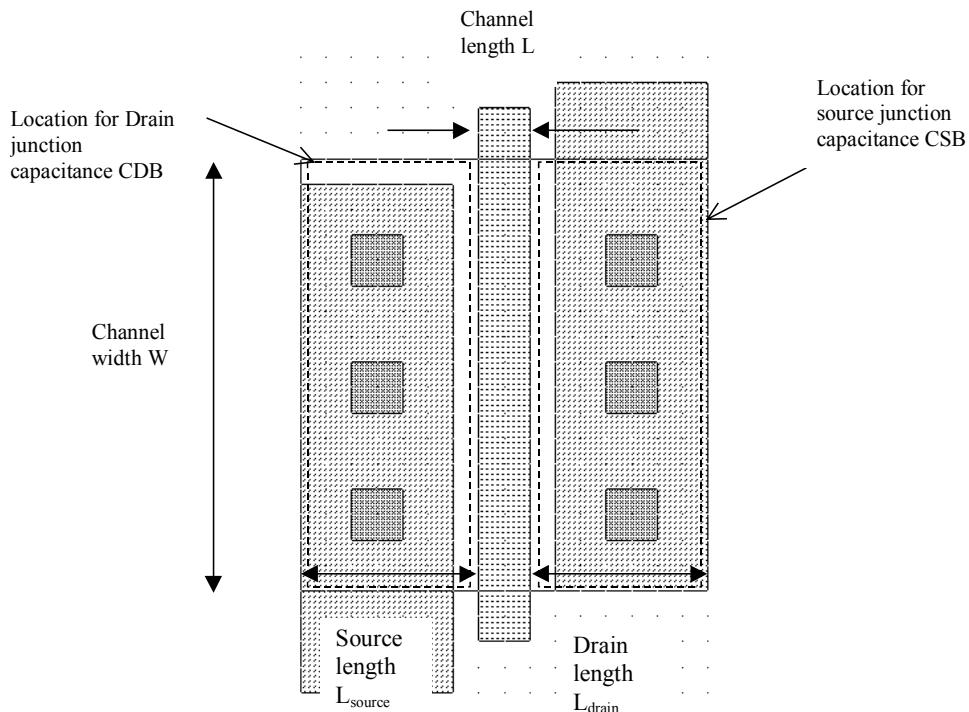


Figure 3-17: The junction capacitance for drain and source contributes significantly to the MOS capacitance

## 4. The BSIM4 MOS Model

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

A new MOS model, called BSIM4 [Bsim4], was introduced in 2000. A simplified version of this model is supported by Microwind2, and recommended for ultra-deep submicron technology simulation. The complete details on BSIM4 are provided in the excellent book [Liu]. 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 (Figure 3-18). In that region,  $V_{ds}$  is very high, over the nominal supply voltage  $VDD$ . One of the key features of BSIM4 is the use of one single equation to build the current, valid for all operating modes. Smoothing functions ensure a nice continuity between operating domains.

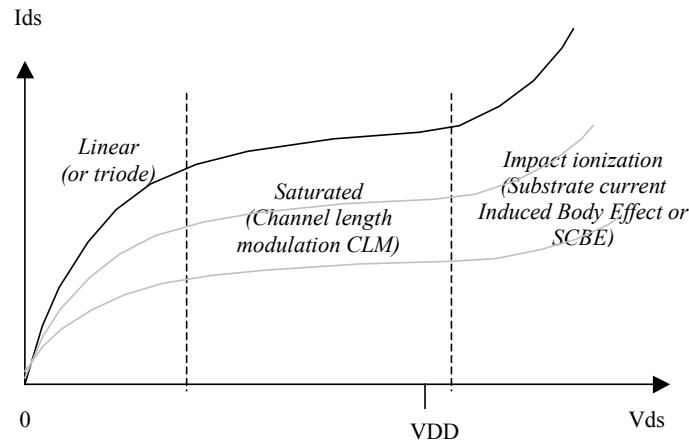


Figure 3-18: The three regions considered in our simplified version of BSIM4

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 purposes. The set of parameters is reduced to around 30.

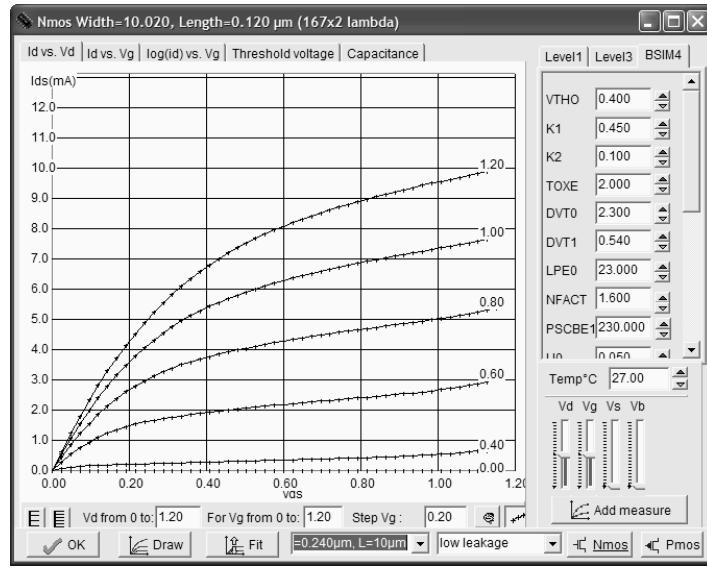


Figure 3-19: Implementation of BSIM4 within Microwind2, based on [Liu]

### Effective Channel Length and Width

Once fabricated, the physical length  $L_{eff}$  and width  $W_{eff}$  of the MOS device do not correspond exactly to the initial length L and width W drawn using Microwind2 (Figure 3-20). The parameters  $L_{INT}$  and  $W_{INT}$  have been introduced for that purpose, with equations 3-21 and 3-22.

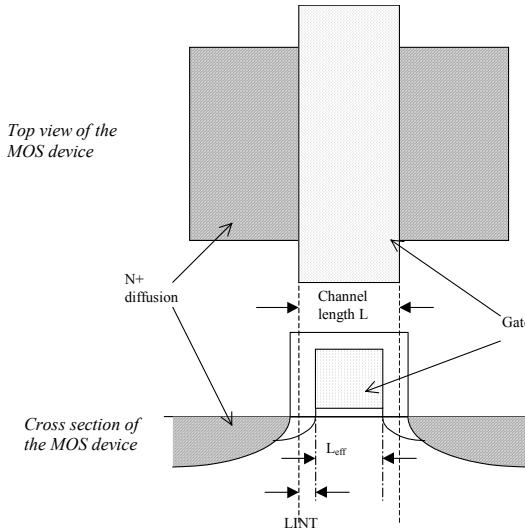


Figure 3-20: Illustration of the effective channel length  $L_{eff}$

$$L_{eff} = L - 2.LINT \quad (3-21)$$

$$W_{eff} = W - 2.WINT \quad (3-22)$$

### Surface potential and junction depth

The surface potential  $\Phi_s$  and junction depth are basic parameters taken into account in the evaluation of the threshold voltage and the global current. The surface potential  $\Phi_s$  is defined by equation 3-23.

$$\Phi_s = 0.4 + vt \cdot \ln\left(\frac{NDEP}{ni}\right) \quad (3-23)$$

where  $vt$  the thermal voltage given by equation 3-24,  $NDEP$  is the channel doping concentration for zero body bias (Around  $10^{17}\text{cm}^{-3}$  in practice), and  $ni$  is the intrinsic carrier concentration of silicon ( $ni=1.02 \times 10^{10}\text{cm}^{-3}$  at  $300^\circ\text{K}$ ). Consequently, the surface potential  $\Phi_s$  in deep-submicron CMOS process is around 0.85V.

The thermal voltage is

$$vt = \frac{k_B T}{q} \quad (3-24)$$

$k_B$  = Boltzmann constant =  $1.38 \times 10^{-23}$  J/K

$T$  = temperature ( $300^\circ\text{K}$  by default)

$q$  = Electronic charge =  $1.60 \times 10^{-19}$  C

The built-in voltage of the source/drain junctions is given by equation 3-25.

$$V_{bi} = vt \cdot \ln\left(\frac{NDEP.NSD}{ni^2}\right) \quad (3-25)$$

where  $vt$  the thermal voltage given by equation 3-24,  $NDEP$  is the channel doping concentration for zero body bias (Around  $10^{17}\text{cm}^{-3}$  in practice),  $NSD$  is the source/drain doping concentration (Around  $10^{20}\text{cm}^{-3}$  in practice), and  $ni$  is the intrinsic carrier concentration of silicon ( $ni=1.02 \times 10^{10}\text{cm}^{-3}$  at  $300^\circ\text{K}$ ). Consequently, the built-in voltage  $V_{bi}$  in deep-submicron CMOS process is around 1.0V.

The depletion depth  $X_{dep}$  is computed by equation 3-26. It corresponds to the thickness of the region near the N+/P- junction interfaces, as illustrated in figure 3-21.

$$X_{dep} = \sqrt{\frac{2\epsilon_{rsi} \cdot \epsilon_0 (\Phi_s - V_{bs})}{q \cdot NDEP}} \quad (3-26)$$

where  $\epsilon_{rsi}$  is the dielectric constant of silicon (11.7)  $\epsilon_0$  is the permittivity in vacuum ( $8.854 \times 10^{-12} \text{ F/m}$ ),  $\Phi_s$  is the surface potential given by equation 3-23,  $NDEP$  is the channel doping concentration for zero body bias,  $q$  is the electronic charge ( $1.60 \times 10^{-19} \text{ C}$ ), and  $vbs$  is the bulk-source potential. The typical value of  $Xdep$  is  $0.5 \mu\text{m}$  (Figure 3-21).

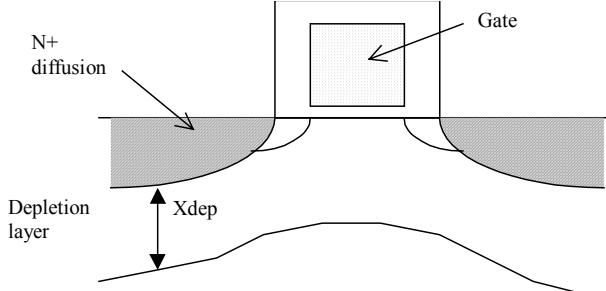


Figure 3-21: Illustration of the depletion depth  $Xdep$

### Threshold voltage

The main impact of the threshold voltage  $Vt$  is the  $Ioff$  parasitic current, that exhibits an exponential dependence with  $1/Vt$ . A high threshold voltage  $Vt$  leads to a small  $Ioff$  current, at the price of a low  $Ion$  current. Low threshold MOS devices consume a very high standby current, which impacts the power consumption of the whole circuit. An accurate prediction of the threshold voltage is a key issue for low power integrated circuit design. The general equation of the threshold voltage is presented in equation 3-27.

$$vth = VTHO + K1 \cdot \sqrt{(\Phi_s - Vbs) - \sqrt{\Phi_s}} - K2 \cdot Vbs + \Delta Vt_{SCE} + \Delta Vt_{NULD} + \Delta Vt_{DIBL} \quad (3-27)$$

where  $VTHO$  is the long channel threshold voltage at  $Vbs=0$  (Around 0.5V),  $K1$  is the first order body bias coefficient ( $0.5 \text{ V}^{1/2}$ ),  $\Phi_s$  is the surface potential given by equation 3-23,  $Vbs$  is the bulk-source voltage,  $K2$  is the second order body bias coefficient,  $\Delta Vt_{SCE}$  is the short channel effect (SCE <gloss>) on  $Vt$  (Detailed in equation 3-28),  $\Delta Vt_{NULD}$  is the non-uniform lateral doping effect (NULD <gloss>) explained in equation 3-29, and  $\Delta Vt_{DIBL}$  is the drain-induced barrier lowering (DIBL <gloss>) effect of short channel on  $Vt$  (Detailed in equation 3-30).

### Short channel effect

The threshold voltage is not the same for all MOS devices. There is a complex dependence between the threshold voltage and the effective length of the channel. For small channel, the threshold value tends to decrease. The equation 3-28 is proposed, based on an hyperbolic cosine function.

$$\Delta Vt_{SCE} = -\frac{0.5 \cdot DVT0}{\cosh(DVT1 \cdot \frac{L_{eff}}{l_t} - 1)} (Vbi - \Phi_s) \quad (3-28)$$

where  $DVTO$  is the first coefficient of short-channel effect on the threshold voltage (2.2 by default),  $DVT1$  is the second coefficient of short-channel effect on the threshold voltage (0.53 by default),  $L_{eff}$  is the effective channel length given in equation 3-21, and  $l_t$  is the characteristic length, approximated in our implementation to 1/4 of the minimum channel length (0.03  $\mu\text{m}$  for a 0.12 $\mu\text{m}$  ),  $V_{bi}$  is defined in equation 3-25, and  $\Phi_s$  is the surface potential given by equation 3-23.

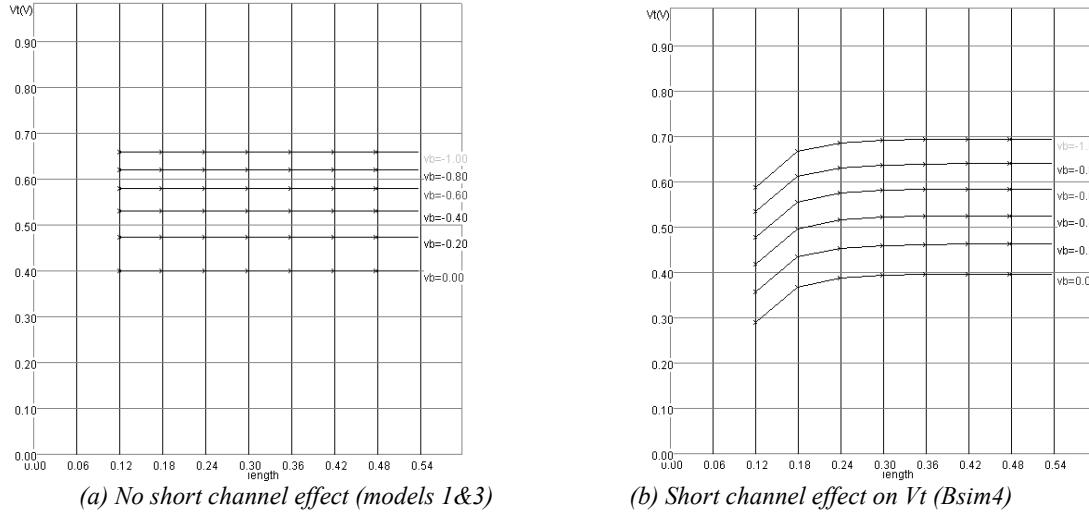


Fig. 3-22: Short channel effect (SCE) on the threshold voltage

The illustration of the effect of  $\Delta V_{t,SCE}$  is proposed in figure 3-22. Without taking into account the short-channel effect, the threshold voltage is only dependent on  $V_{bs}$ . It can be seen that  $V_t$  increases when  $V_{bs}$  decreases. There is no dependence on the length. When we add the contribution of the short-channel effect expressed by equation 3-28, the threshold voltage is decreased significantly for small length values.

### Non-uniform lateral doping

The lateral drain diffusion (LDD) is a technique introduced in recent technologies to reduce the peak channel fields in the MOS channel. The location for high-field parasitic effects is illustrated in the process section of figure 3-23 (a).

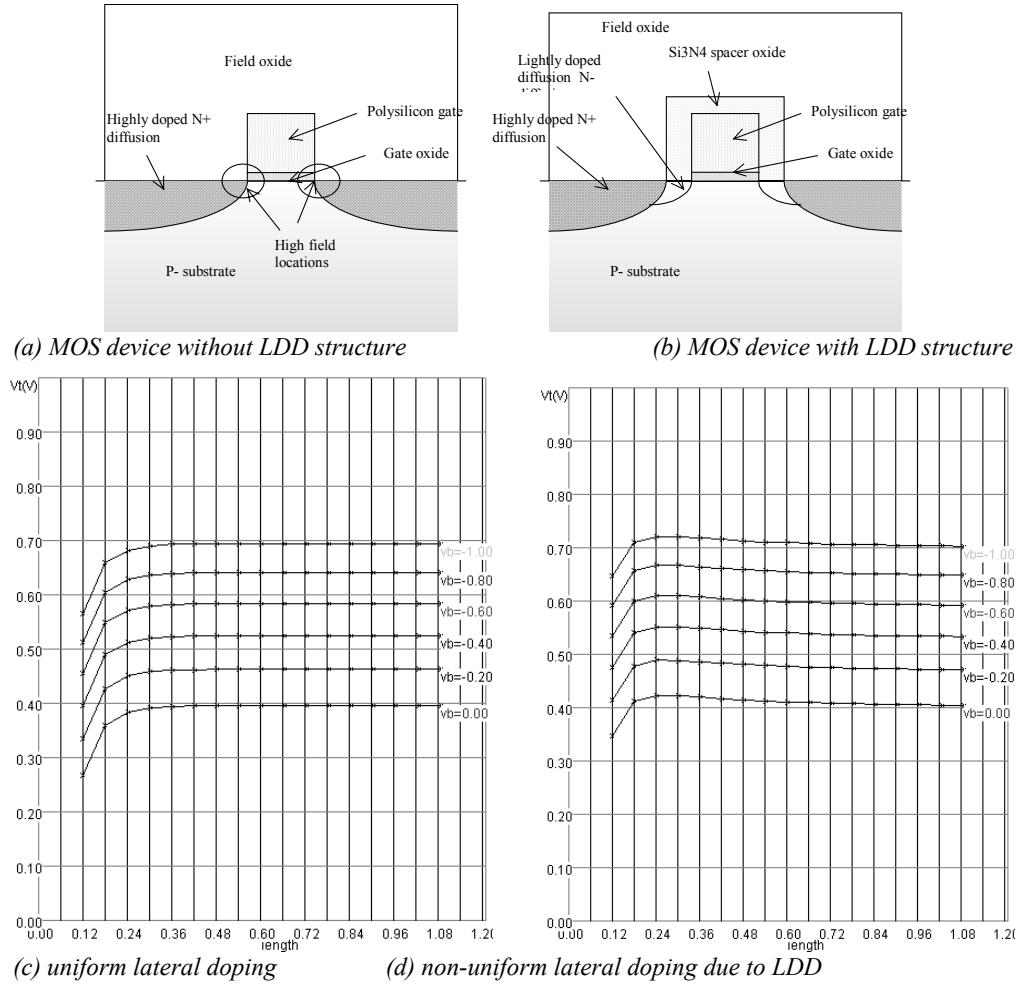


Fig. 3-23: Effect of non-uniform lateral doping on the threshold dependence with the channel length

In the cross-section of figure 3-23 (b), the doping concentration at the corner of the gate is reduced thanks to a lightly doped N-type implantation. The Si<sub>3</sub>N<sub>4</sub> spacer is grown over the gate before the N<sup>+</sup> highly doped implantation is performed. Consequently, the high-field effects are moderated. Unfortunately, the threshold voltage exhibits a complex dependence on the channel length, as illustrated in figure 3-23 (d), compared to (c). For a decreasing length, the threshold voltage tends to increase first (due to  $\Delta V_{t_{NULD}}$ ), before decreasing rapidly due to the short channel effect  $\Delta V_{t_{SCE}}$  described in formula 3-28.

A simple formulation of the non-uniform lateral doping is  $\Delta V_{t_{NULD}}$  given below:

$$\Delta V_{t_{NULD}} = K_1 \left( \sqrt{1 + \frac{LPE_0}{L_{eff}}} - 1 \right) \cdot \sqrt{\Phi_s} \quad (3-29)$$

### Drain induced barrier lowering

When we apply a positive voltage on the drain of a long-channel n-MOS device, we observe no significant change in the value of  $V_t$ . When we do the same for a short-channel n-MOS device, we observe a decrease of the threshold voltage. The physical origin of DIBL is the increase of the depletion layer due to a high value of  $V_{ds}$  that reduces the equivalent channel length, and consequently decreases the threshold voltage [Liu].

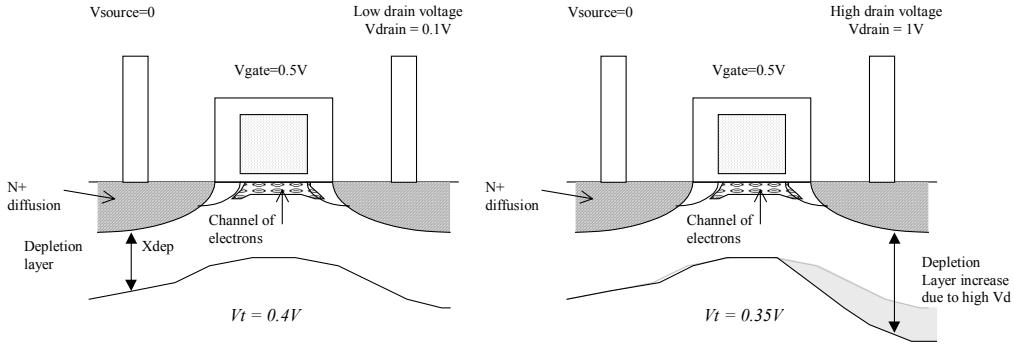


Fig. 3-24: Illustration of the depletion depth  $X_{dep}$

$$\Delta V_{t_{DIBL}} = -0.5 \cdot \text{ETA0} \cdot V_{ds} \quad (3-30)$$

A simplified model of the DIBL effect on the threshold voltage is proposed in equation 3-30. The parameter  $\text{ETA0}$  is the DIBL coefficient in sub-threshold region (default value 0.08), and  $V_{ds}$  is the drain-source voltage.

## Mobility

In this paragraph, we introduce the formulations for mobility of channel carriers. The generic parameter is  $U_0$ , the mobility of electrons and holes. The effective mobility  $\mu_{eff}$  is reduced due to several effects: the bulk polarization, and the gate voltage. The equation implemented in Microwind2 is one of the mobility models proposed in BSIM4 (Equation 3-31).

$$\mu_{eff} = \frac{U_0}{1 + (UA + UC \cdot V_{BS}) \left( \frac{V_{gsteff} + 2(V_{TH0} - V_{fb} - \phi_s)}{TOXE} \right)^{\text{EU}}} \quad (3-31)$$

where

$U_0$  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  $\text{m}/\text{V}$ . Its default value is around  $10^{-15}$ .

$UC$  is the body-effect coefficient of mobility degradation, in  $\text{m}/\text{V}^2$ . Its default value is  $-0.045 \times 10^{-15}$ .

$V_{FB}$  is the flat band voltage, in V. It is computed using equation 3-32, where  $\Phi_S$  is derived from equation 3-23. Its value is around 0.8V.

$$V_{FB} = VTO - \Phi_S - K1 \sqrt{\Phi_S} \quad (3-32)$$

$TOXE$  is the oxide thickness, in m. A typical value for  $TOXE$  in  $0.12\mu\text{m}$  is  $2\text{nm}$  ( $2.10^{-9}\text{m}$ ).

$V_{BS}$  is the voltage difference between the bulk and the source (V).

$EU$  is a coefficient equal to 1.67 for n-channel MOS, and 1.0 for p-channel MOS.

The parameter  $V_{gsteff}$  is a smoothing function, to ensure continuity between the subthreshold region and the linear region.

$$V_{gsteff} = \max(VOFF, \frac{n.vt \ln(1 + \exp(\frac{(Vgs - Vth)}{n.vt}))}{1 + n \exp(\frac{-(Vgs - Vth)}{n.vt})}) \quad (3-33)$$

$$n = I + NFACTO \quad (3-34)$$

A specific parameter  $VOFF$  is introduced to account for a specific effect appearing in short-channel device when  $Vgs$  is negative. Conventional models predict that the current decrease with an exponential law down to zero with decreasing  $Vgs$ . For  $Vgs < 0$ ,  $Ids$  is supposed to be 0.

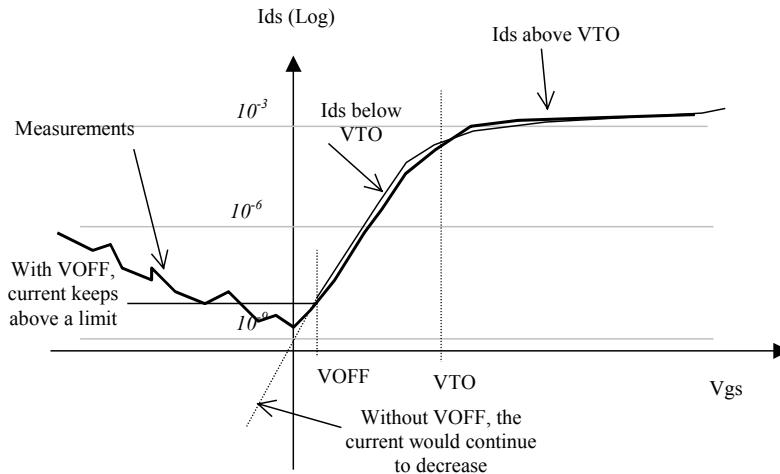


Figure 3-25: Illustration of the gate-induced drain leakage (GIDL) for negative  $Vgs$

In reality,  $Ids$  stops decreasing near zero  $Vgs$ , and then tends to increase with negative  $Vgs$  (Figure 3-25). This effect is called gate-induced drain leakage (GIDL). Consequently, the leakage current  $Ioff$  can be

significant when  $V_{gs}$  is negative (Quite frequent in logic cells). The  $VOFF$  parameter stops the  $Ids$  at a certain value, a simplified version of the BSIM4 modeling of the so-called gate-induced leakage current (More info may be found in [Liu]).

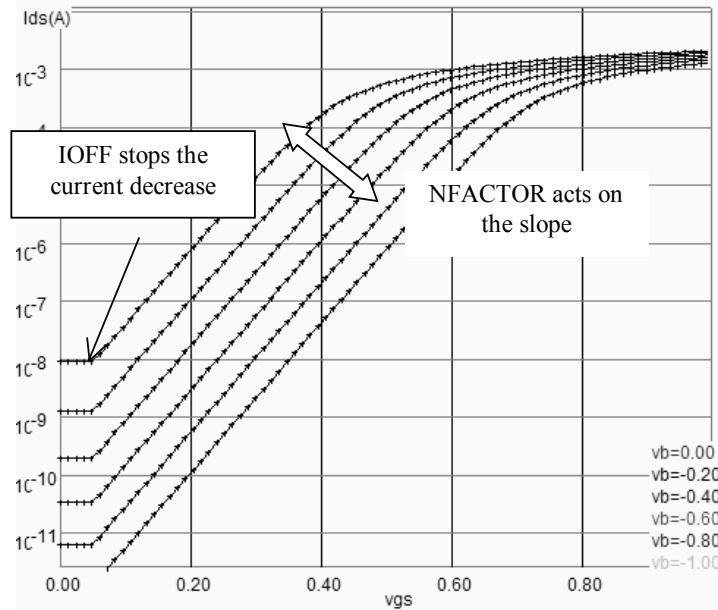


Figure 3-26: Illustration of the effects of IOFF and NFACTOR in sub-threshold mode

The parameter NFACTOR is usually close to 1, meaning that  $n$  is close from 2 (Equation 3-34). The effect on NFACTOR is illustrated in the display mode **Id.** vs. **Vg**, in logarithmic scale, as illustrated in figure 3-26.

$$Esat = 2 \frac{V_{sat}}{\mu_{eff}} \quad (3-35)$$

$$V_{dsat} = Esat \cdot L \frac{(V_{gsteff} + 2.vt)}{(Esat \cdot L + V_{gstEff} + 2.vt)} \quad (3-36)$$

Again,  $V_{dsEff}$  is defined so as to smooth the evolution from  $V_{ds}$  to the saturation voltage  $V_{dsat}$  (Equation 3-37). The parameter  $DELTA$  is fixed to 0.01. The effect of  $DELTA$  is shown in figure 3-27. With a small value of  $DELTA$  (0.001 for example), the transition between linear and saturated region leads to a discontinuity. Experimental measurements show a gradual transition, that is well approximated when  $DELTA=0.01$ . A higher value of  $DELTA$  would lead to an  $Ids$  curve significantly lower than measurements.

$$V_{dseff} = V_{dssat} - 0.5(V_{dsat} - V_{ds} - \delta + \sqrt{(V_{dsat} - V_{ds} - \delta)^2 + 4\delta \cdot V_{dsat}}) \quad (3-37)$$

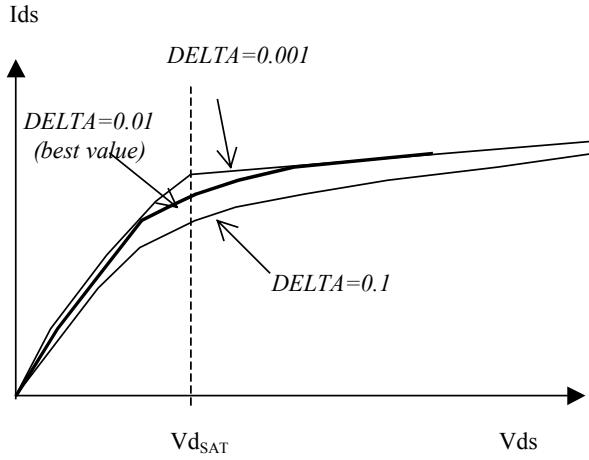


Figure 3-27: The smoothing function between linear and saturated regions can be modulated by  $\text{DELTA}$ . In Microwind2,  $\text{DELTA}$  is fixed to 0.01.

### Current $Ids$

The current  $Ids$  is computed using one single equation, as described below.

$$Ids0 = \frac{W_{eff}}{L_{eff}} \mu_{eff} \frac{\varepsilon_0 \varepsilon_r}{TOXE} V_{gsteff} \left(1 - \frac{A_{bulk} V_{dseff}}{(2V_{gsteff} + 4.vt)}\right) \frac{V_{dseff}}{\left(1 + \frac{V_{dseff}}{\varepsilon_{sat} L_{eff}}\right)} \quad (3-38)$$

In our implementation of BSIM4 in Microwind, the parameter  $A_{bulk}$  is fixed to 1. The final current  $Ids$  used in analog simulation is computed by equation 3-39:

$$Ids = Ids0 \left(1 + \frac{(V_{ds} - V_{dseff})}{V_{ascbe}}\right) \left(1 + \frac{1}{C_{clm}} \ln \left(\frac{V_{ASAT} + V_{ACLM}}{V_{ASAT}}\right)\right) \quad (3-39)$$

Two new terms appear after  $Ids0$ . The second term of the current equation accounts for impact ionization. It corresponds to a parasitic current at very high  $Vds$ , created by hot electrons and generating supplementary pairs or electrons and holes, when hitting the drain region after acquiring a high energy level inside the device channel. The parameter  $V_{ascbe}$  is a voltage below which the impact ionization becomes significant. If  $V_{ascbe}$  is large,  $Ids$  is almost equal to  $Ids0$ , meaning that there is no impact ionization effect. If  $V_{ascbe}$  is small, the shape of  $Ids$  is changed for high  $Vds$ , as illustrated in figure 3-28.

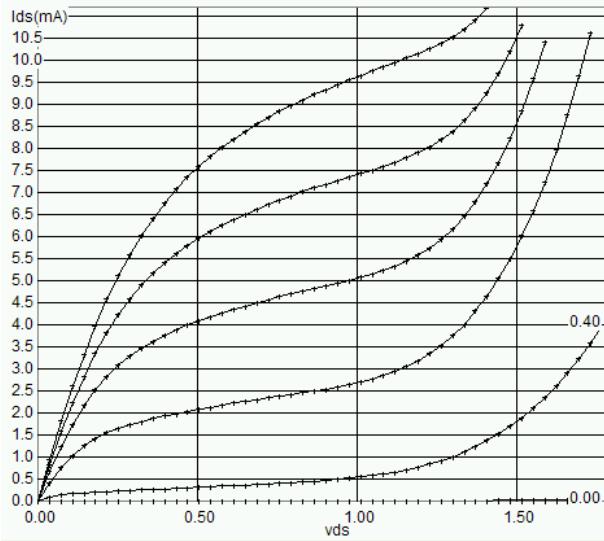


Figure 3-28: Effect of impact ionization at large  $V_{ds}$  ( $W=10\mu m$ ,  $L=0.12\mu m$ ).

Two parameters affect the shape of the ionization current : PSCBE1 and PSCBE2. The first parameter can be changed interactively on the screen. The voltage  $V_{asbe}$  is determined thanks to the following equations:

$$V_{asbe} = \frac{L_{eff}}{PSCBE2} \exp(PSCBE1 \frac{litl}{(V_{ds} - V_{dseff})}) \quad (3-40)$$

with

$$Litl = \sqrt{XJ \frac{\epsilon_{rsi}}{\epsilon_{rsiO2}} TOXE} \quad (3-41)$$

$XJ$  is the source/drain junction depth, around  $0.1\mu m$  ( $10^{-7}m$ )

$TOXE$  is the oxide thickness, in m (Around  $3nm$  in  $0.12\mu m$ )

$\epsilon_{rsi}$  = relative permittivity of silicon (11.7)

$\epsilon_{rsiO2}$  = relative permittivity of silicon oxide (3.9)

The third term of equation 3-39 accounts for the channel length modulation. An illustration of this phenomenon is provided in figure 3-29. It represents the  $Ids$  increase with large  $V_{ds}$ . Graphically,  $V_{ACLM}$  is equivalent to an Early voltage, i.e. the value for which the  $Ids$  slope would cross the horizontal axis for negative  $V_{ds}$ . For long channel devices ( $L=1\mu m$  for example), the effect of channel length modulation effect is small, so  $V_{ACLM}$  has a very high value (10V). For very short channels ( $0.12\mu m$  in the case of figure 3-30), a significant channel length modulation effect is observed, and  $V_{ACLM}$  is small.

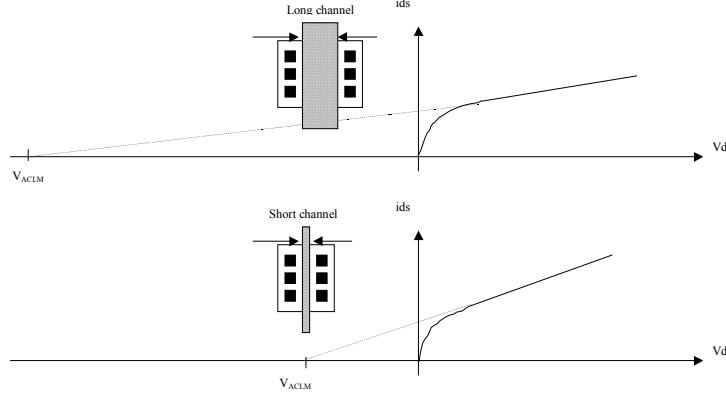


Figure 3-29: The channel length modulation is significant for short channel devices, and corresponds to the  $I_{ds}$  increase at high  $V_{ds}$ .

Only one new parameter, PCLM, is introduced in the equations. The parameters  $V_{ASAT}$  and  $V_{ACLM}$  are detailed below. The original equations from BSIM4 have been significantly simplified, and some fitting parameters have been ignored. See [Liu] for a description and relevant comments about the original equations.

$$C_{clm} = \frac{1}{PCLM \cdot litl} \left( l_{eff} + \frac{V_{dsat}}{\epsilon_{sat}} \right) \quad (3-42)$$

$$V_{ACLM} = C_{clm} (V_{ds} - V_{dseff}) \quad (3-43)$$

$$V_{ASAT} = (\epsilon_{sat} \cdot l_{eff} + V_{DSSat}) \left( 1 - \frac{A_{bulk} V_{dsat}}{2(V_{gsteff} + 2Vt)} \right) \quad (3-44)$$

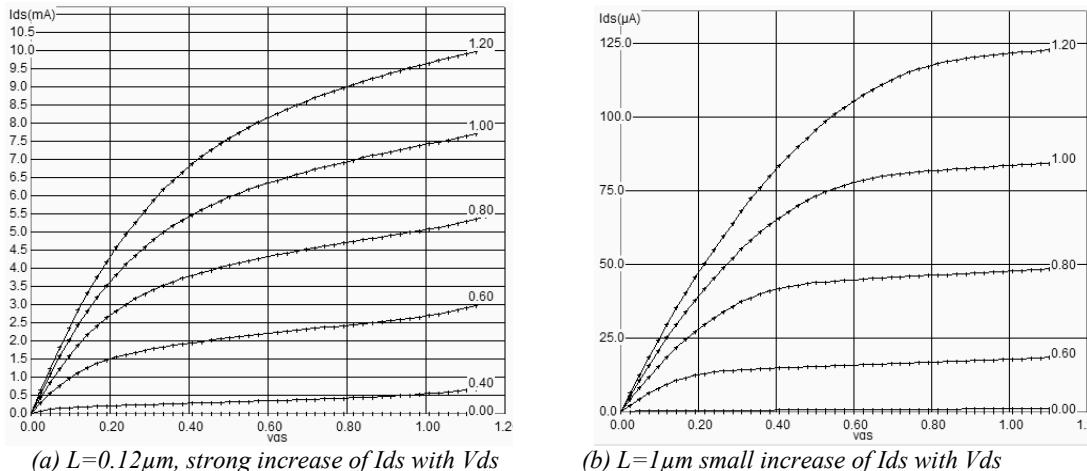


Figure 3-30: The channel length modulation effect is significant for short channel devices in  $0.12\mu m$  technology

## Temperature Effects

Three main parameters are concerned by the sensitivity to temperature: the threshold voltage  $VTO$ , the mobility  $U0$  and the slope in sub-threshold mode. Both  $VTO$  and  $U0$  decrease when the temperature increases. The modeling of the temperature effect in BSIM4 is as follows. In Microwind2,  $TNOM$  is fixed to 300°K, equivalent to 27°C.  $UTE$  is negative, and set to -1.8 in 0.12μm CMOS technology, while  $KT1$  is set to -0.06 by default.

$$U0 = U0_{(T=27)} \left( \frac{T + 273}{TNOM} \right)^{UTE} \quad (3-45)$$

$$VT = VT0_{(T=27)} + KT1 \left( \frac{T + 273}{TNOM} - 1 \right) \quad (3-46)$$

A higher temperature leads to a reduced mobility, as  $UTE$  is negative. Consequently, at a higher temperature, the current  $Ids$  is lowered. This trend is clearly illustrated in figure 3-31. The reduction of the maximum current is 40% between -30°C and 100°C.

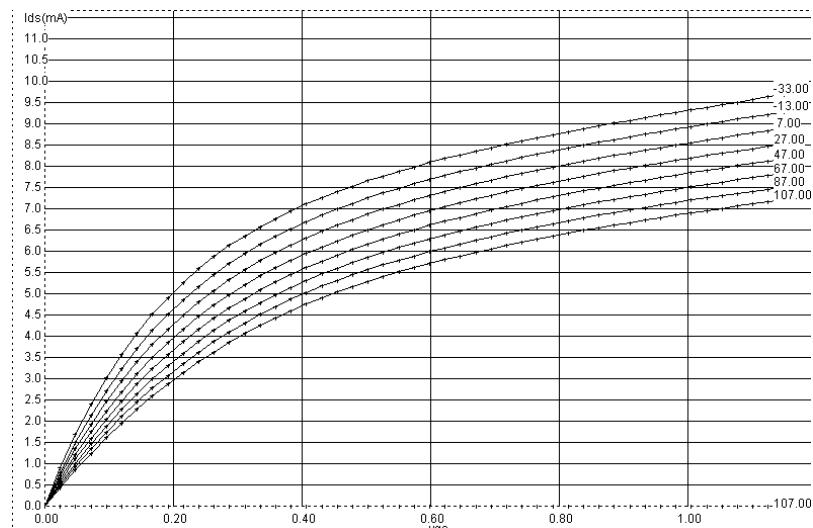


Figure 3-31: The effect of temperature on the peak  $Ids$  current, showing a degradation of current with rising temperature

For a short channel n-channel MOS device ( $L=0.12\mu m$ ), the result of the parametric analysis illustrates the same trend (Figure 3-33). The parametric analysis is conducted as follows: the layout **MosTemperature.MSK** is loaded first. The MOS is polarized with a gate always on, the drain at  $VSS$ , and the source at  $VDD$ . The parametric analysis is launched. In the new window, select the temperature (Upper menu) and the maximum current (Lower menu). We observe in figure 3-33 a significant decrease of the Ion current with the temperature.

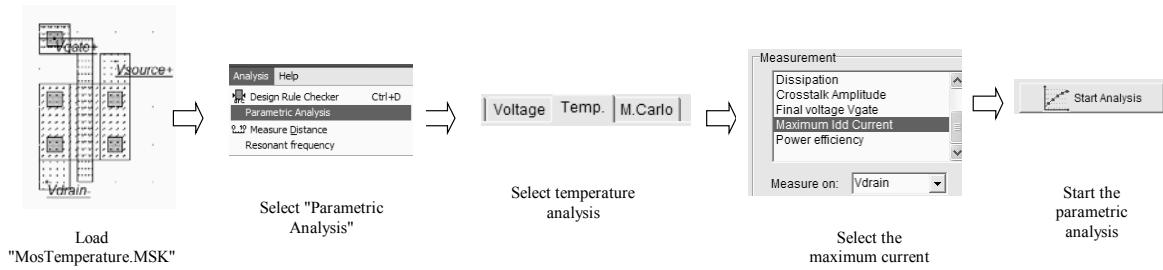


Figure 3-32: Configuring Microwind to display the variations of  $Id_s$  vs. temperature

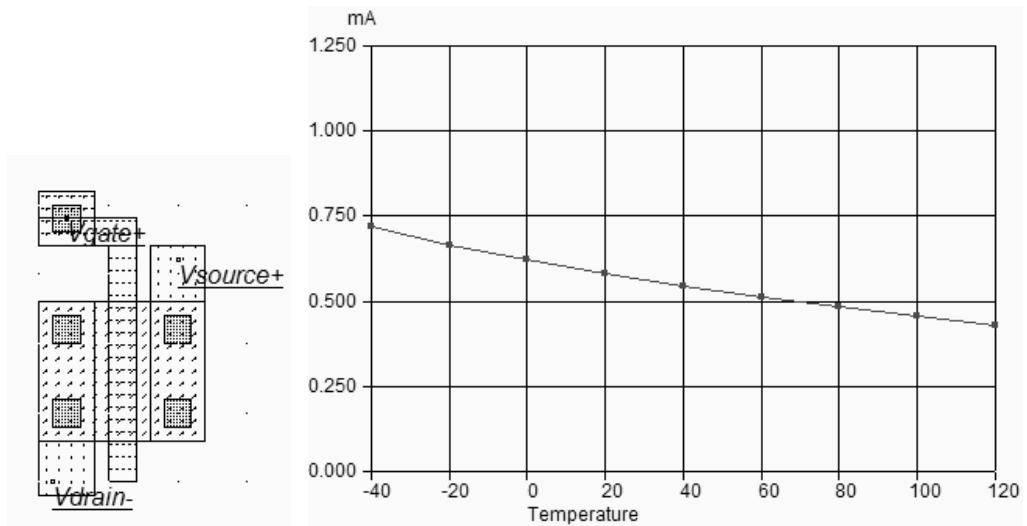


Figure 3-33: The parametric analysis reveals an important decrease of the maximum current  $Id_s$  with temperature (MosTemperature.MSK)

Meanwhile, in an opposite trend, the threshold voltage is decreased, as  $KT_1$  is negative (Figure 3-34). Therefore, there exists a remarkable operating point where the  $Id_s$  current is almost constant and independent of temperature variation. In  $0.12\mu m$  CMOS, the  $V_{ds}$  voltage with zero temperature coefficient (ZTC) <Gloss> is around 0.9V, as shown in figure 3-34.

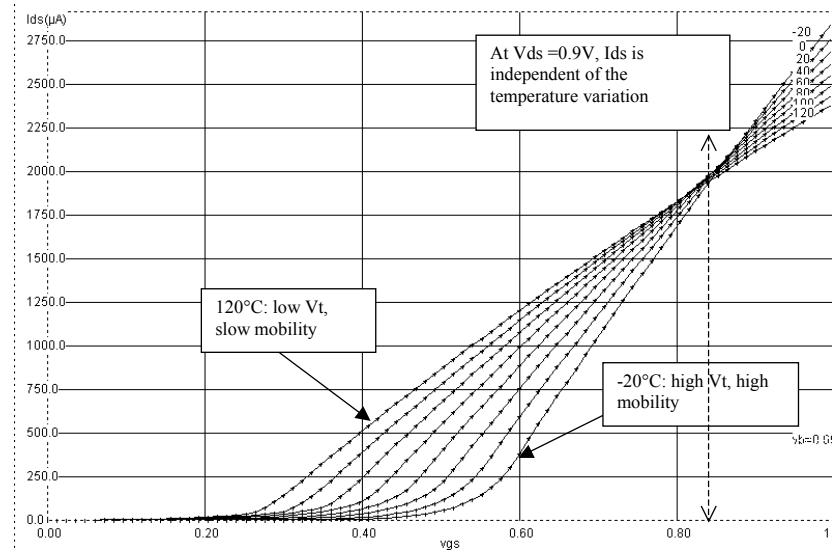


Figure 3-34: The effect of temperature on the  $Ids$  current, showing a zero temperature coefficient (ZTC) operating point

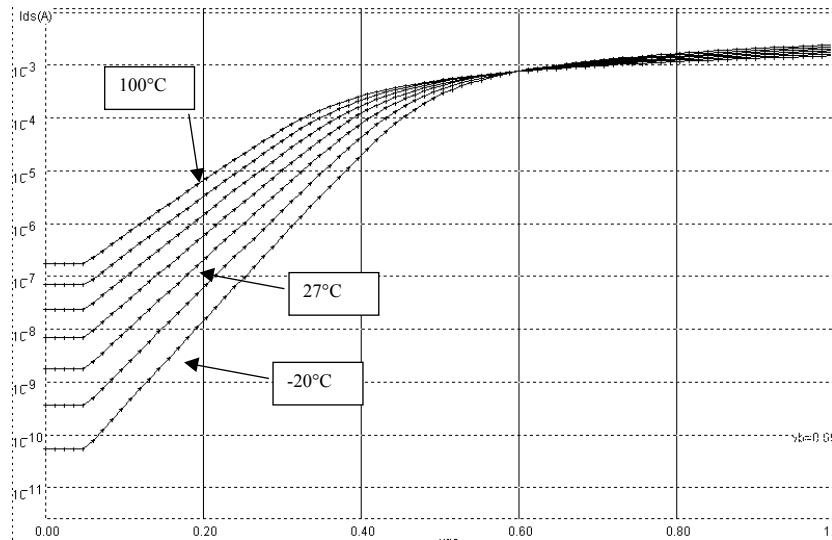


Figure 3-35 The effect of temperature on the MOS characteristics.

In the sub-threshold region, the impact of temperature is extremely important, as demonstrated in figure 3-35. At low temperature the current  $Ids$  decreased rapidly down to  $10\text{nA}$ , corresponding to a small off leakage current. In contrast, at high temperature, not only the threshold voltage is reduced but the sub-threshold slope is flattened, which means an exponential increase of the  $I_{off}$  leakage current (figure 3-35).

Parameter	Description	NMOS value in 0.12μm	NMOS value in 0.12μm	Name in RUL file
DVT0	First coefficient of short-channel effect on threshold voltage	2.2	2.2	B4D0VT
DVT1	Second coefficient of short-channel effect on Vth	0.53	0.53	B4D1VT
ETA0	Drain induced barrier lowering coefficient	0.08	0.08	B4ETA0
LINT	Channel-length offset parameter	0.01 <sup>e</sup> -6μm	0.01 <sup>e</sup> -6μm	B4LINT
LPE0	Lateral non-uniform doping parameter at Vbs = 0	2.3 <sup>e</sup> -10	2.3 <sup>e</sup> -10	B4LPE
NFACTOR	Sub-threshold turn-on swing factor. Controls the exponential increase of current with Vgs.	1	1	B4NFACTOR
PSCBE1	First substrate current induced body-effect mobility reduction	4.24e8 V/m	4.24e8 V/m	B4PSCBE1
PSCBE2	Second substrate current induced body-effect mobility reduction	4.24e8 V/m	4.24e8 V/m	B4PSCBE2
K1	First-order body bias coefficient	0.45 V1/2	0.45 V1/2	B4K1
K2	Second-order body bias coefficient	0.1	0.1	B4K2
KT1	Temperature coefficient of the threshold voltage.	-0.06V	-0.06V	B4KT1
NDEP	Channel doping concentration	1.7 <sup>e</sup> 17 cm <sup>-3</sup>	1.7 <sup>e</sup> 17 cm <sup>-3</sup>	B4NDEP
PCLM	Parameter for channel length modulation	1.2	1.2	B4PCLM
TOX	Gate oxide thickness	100nm	100nm	B4TOX
UA	Coefficient of first-order mobility degradation due to vertical field	11.0e-15 m/V	11.0e-15 m/V	B4UA
UC	Coefficient of mobility degradation due to body-bias effect	-0.04650e-15 V-1	-0.04650e-15 V-1	B4UC
U0	Low-field mobility	0.060 m <sup>2</sup> /Vs	0.025 m <sup>2</sup> /Vs	B4U0
UTE	Temperature coefficient for the zero-field mobility U0.	-1.8	-1.8	B4UTE
VFB	Flat-band voltage	-0.9	-0.9	B4VFB
VOFF	Offset voltage in subthreshold region.	-0.08V	-0.08V	B4VOFF
VSAT	Saturation velocity	8.0e4 m/s	8.0e4 m/s	B4VSAT
VTH0	Long channel threshold voltage at Vbs = 0V	0.3V	0.3V	B4VTH0
WINT	Channel-width offset parameter	0.01 <sup>e</sup> -6μm	0.01 <sup>e</sup> -6μm	B4WINT
XJ	Source/Drain junction depth	1.5 <sup>e</sup> -7m	1.5 <sup>e</sup> -7m	B4XJ

Table 3-4 List of user-accessible parameters in the BSIM4 implementation in Microwind.

## 5. Specific MOS devices

New kinds of MOS devices have been introduced in deep submicron technologies, starting the 0.18μm CMOS process generation. These MOS devices have specific characteristics which are described in this section.

### Low leakage MOS

The main objective of the low leakage MOS is to reduce the  $I_{off}$  current significantly, that is the small current that flows between drain and source with a zero gate voltage. The price to pay is a reduced  $I_{on}$  current. The designer has the possibility to use high speed MOS devices, which have high  $I_{off}$  leakages but large  $I_{on}$  drive currents. The symbols of the low leakage MOS and the high speed MOS are given in figure 3-36.

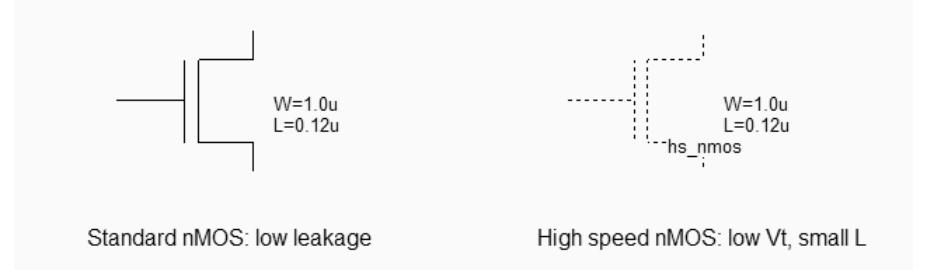


Figure 3-36: The low leakage MOS symbol (left) and the high speed MOS symbol (right) (*MosOptions.SCH*)

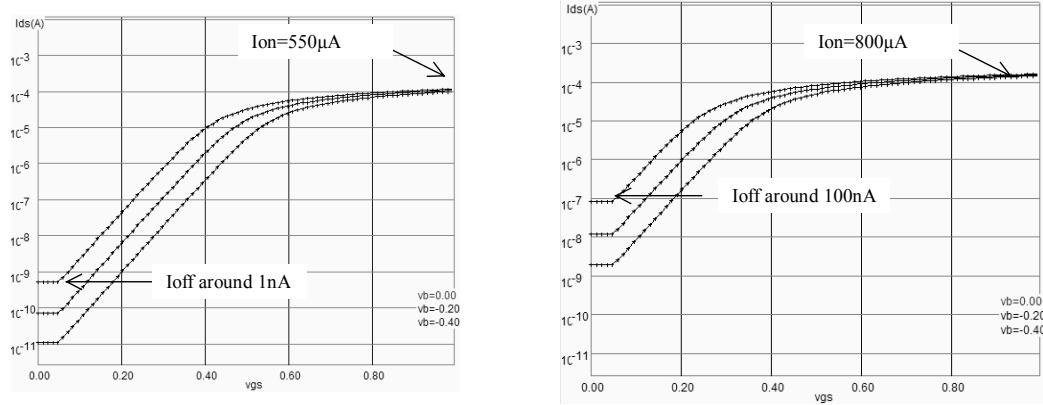


Fig. 3-37: The low leakage MOS offers a low  $I_{off}$  current ( $1nA$ ) but a reduced  $I_{on}$  current ( $550\mu A$ ) as compared to the high speed MOS

In figure 3-37, the low leakage MOS device (left side) has an  $I_{off}$  current reduced nearly by a factor 100, thanks to a higher threshold voltage (0.4V rather than 0.3V) and larger effective channel length (120nm) compared to the high speed MOS (100nm, see figure 3-30). By default, the MOS device is in low leakage option, to encourage low power design. The  $I_{on}$  difference is around 30%. This means that an high speed MOS device is 30% faster than the low leakage MOS. Its use is justified in circuits where speed is critical.

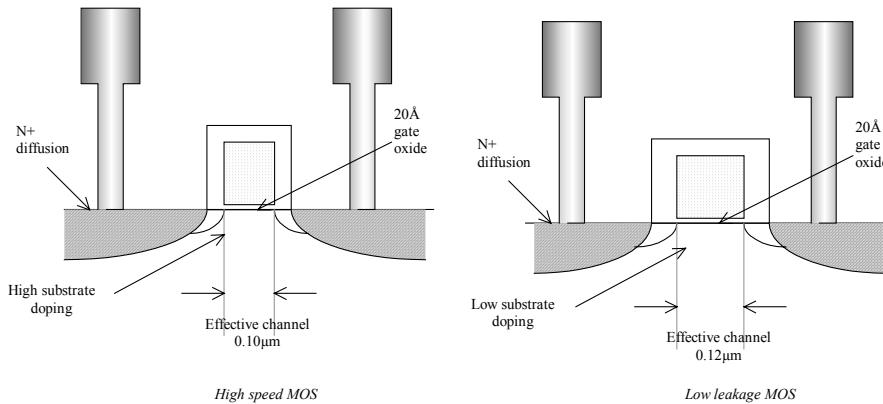


Fig. 3-38: Process section of the high speed (left) and low leakage (right) MOS devices

High speed MOS devices may be found in clock trees, data bus interfaces, central processing units, while low leakage MOS are used whenever possible, for all nodes where a maximum switching speed is not mandatory.

### Mos options in Microwind

A specific layer, called option layer, is used to configure the MOS device option. The layer is situated in the upper part of the palette of layers. The bird's view of the standard MOS is identical to the high speed MOS, except for the added option layer which surrounds the MOS device. The p-channel MOS device includes an option layer together with the n-well layer, as seen in figure 3-39.

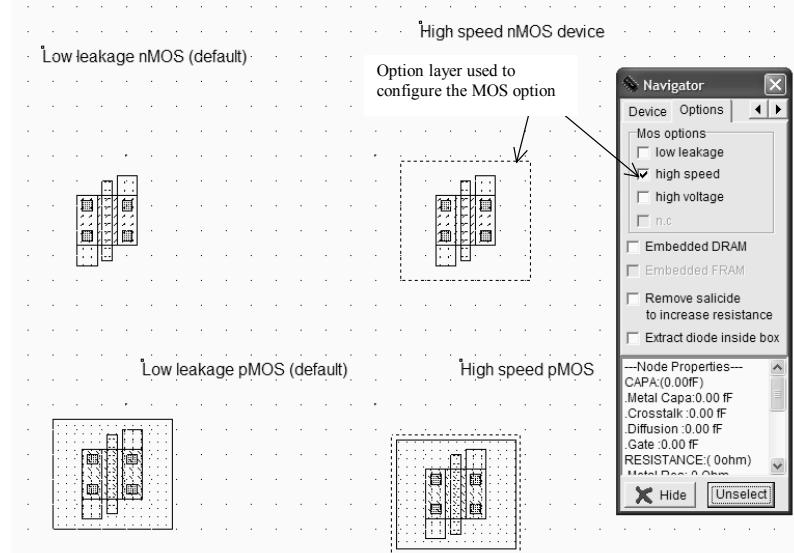


Fig. 3-39: High speed and Low leakage MOS layout. The only difference is the option layer configured for the low leakage option

An "Ultra-high speed" MOS has been introduced, together with the 90nm technology. This MOS device has a very narrow channel, nearly half of the technology, which increases significantly the Ion current at the price of a very high  $I_{off}$  parasitic leakage current (Figure 3-40).

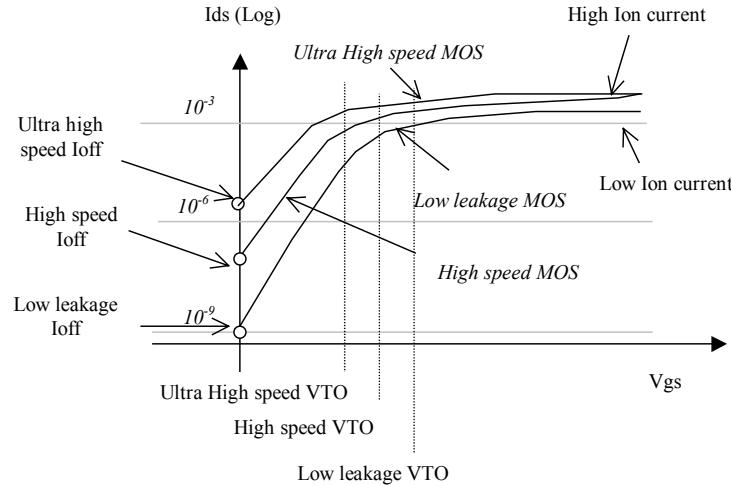


Fig. 3-40: Three types of MOS with different VTO threshold voltage are available in 90nm technology.

### 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).

Figure 3-41 shows the evolution of the supply voltage with the technology generation. The internal supply voltage is continuously decreasing. For compatibility reasons, the chip interface is kept at standard voltages, depending on the target application. Consequently, the input/output structures work at high voltage thanks to specific MOS devices with thick oxide called "High Voltage MOS", while the internal devices work at low voltage for optimum performances.

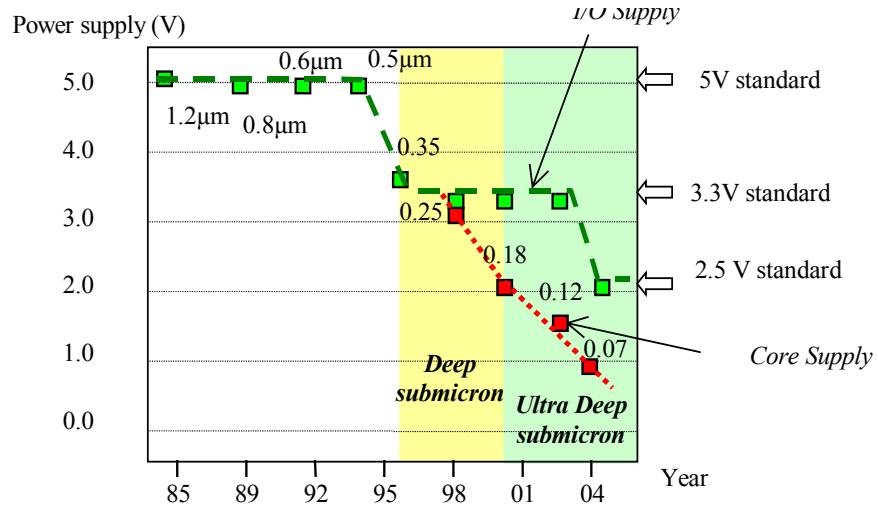


Fig. 3-41: The technology scale down leads to decreased core supply while keeping I/O interfacing compatible with 5V, 3.3V and 2.5V standards

For I/Os operating at high voltage, the high voltage MOS devices are commonly used. High-speed or low leakage devices would be dangerous to use because of their ultra-thin oxide: a 3V voltage applied to the gate of a core MOS device would damage the poly/substrate oxide. 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 (Figure 3-42). Furthermore, the length of the channel is 0.25 $\mu\text{m}$  minimum, that is twice the minimum length of core MOS. The cross-section of the three types of MOS (Low leakage, high speed and high voltage) is given in figure 3-43.

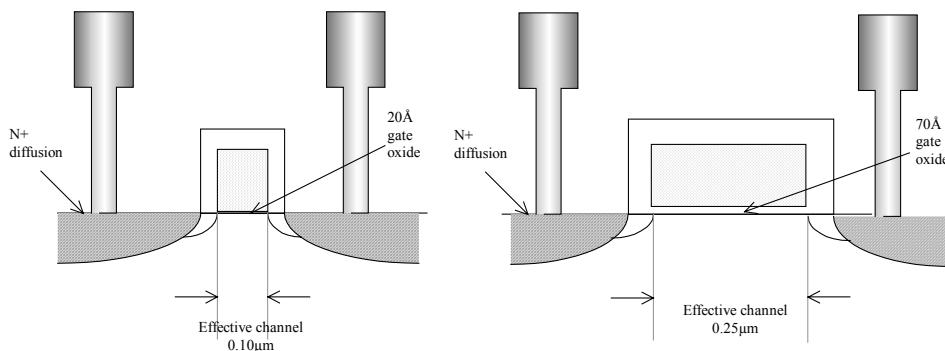


Fig. 3-42: Process section of the high speed and high voltage MOS devices

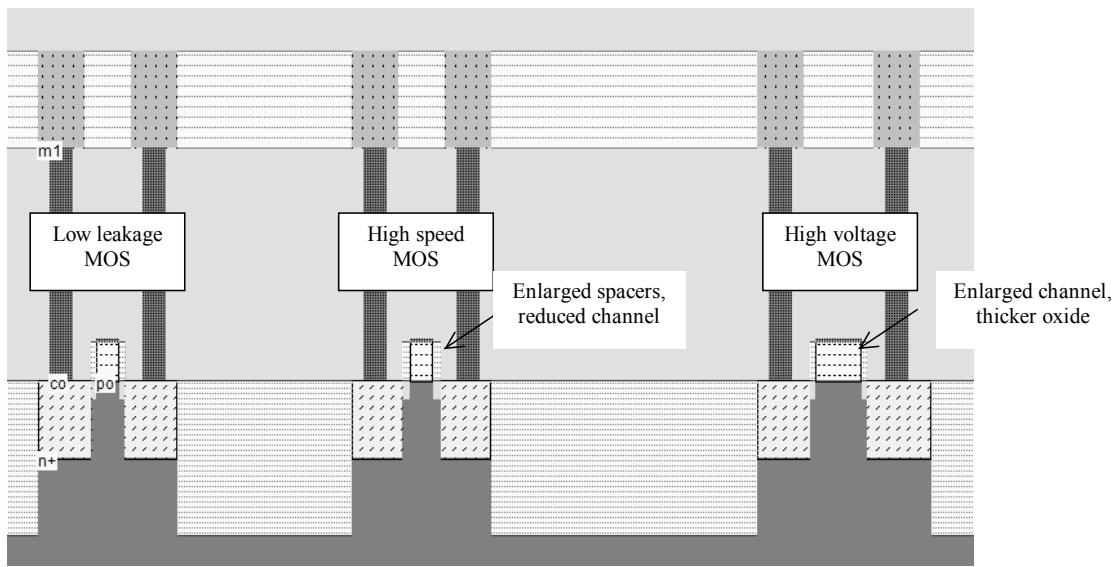


Fig. 3-43. The cross-section of the 3 n-channel MOS options: standard, high speed, and high voltage (*lddExplain.MSK*)

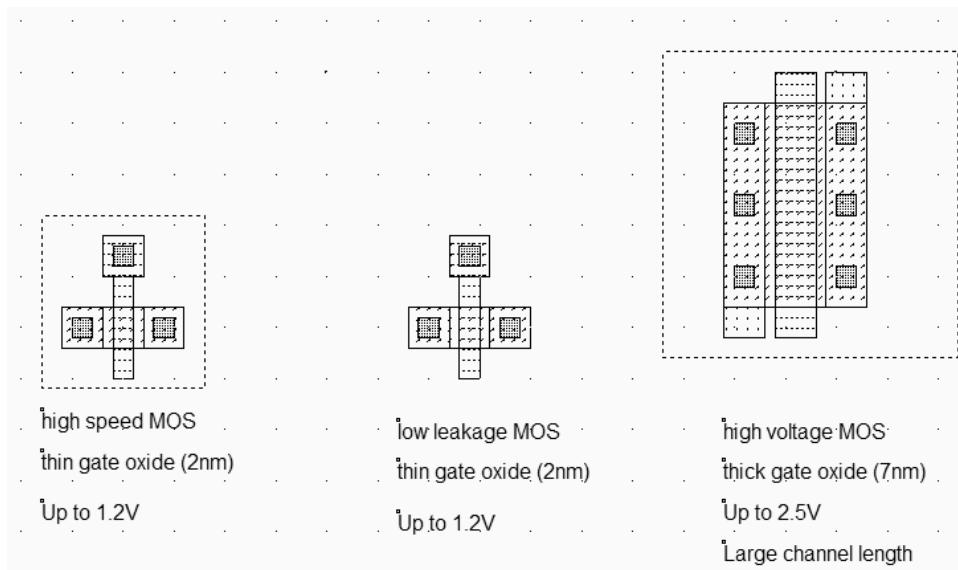


Fig. 3-44: High speed, low leakage and high voltage MOS (*MosHighVoltage.MSK*)

There is no difference between the high-speed MOS and the low leakage MOS from a layout point of view (Figure 3-44), except the option layer for the high-speed option. The High voltage MOS has a significantly different layout, due to the enlarged channel length and width.

The I/V Characteristics of the high voltage MOS are plotted in figure 3-45, for  $V_{gs}$  and  $V_{ds}$  up to 2.5V. The channel length is 0.25μm and the channel width is 1.2μm. Due to a large channel length, the current drive is less efficient.

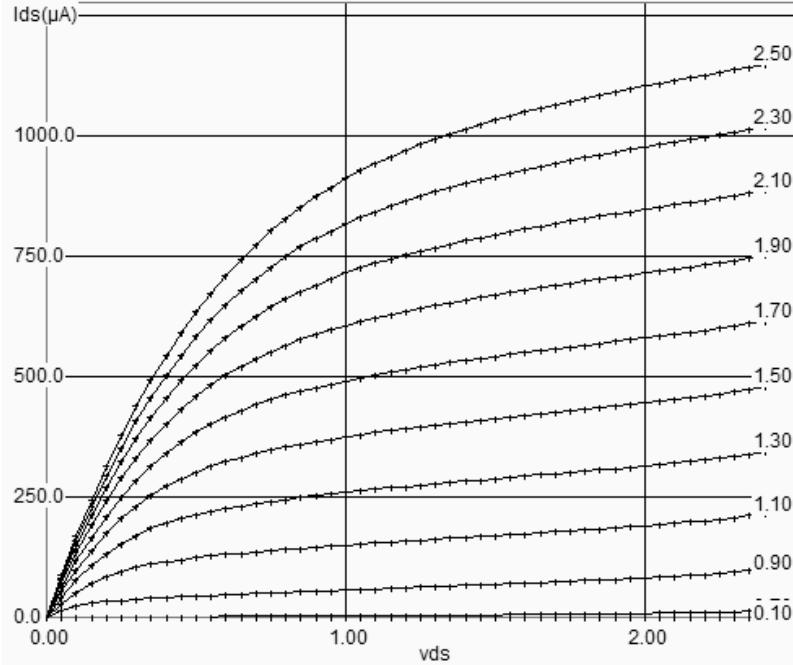


Fig. 3-45:  $Ids/Vds$  characteristics of the high voltage MOS.

There are two main reasons to keep a low-voltage supply for the core of the integrated circuit. The first one is low-power consumption, which is of key importance for integrated circuits used in cellular phones or any portable devices. Low supply strongly reduces power consumption by reducing the amplitude of signals, thus reducing the charge and discharge of each elementary node of the circuit. The equation 3-xxx gives an approximation of the power consumption. We deduce that even a small reduction of  $V_{dd}$  has a very positive impact on the reduction of the power consumption.

$$P = k \cdot C \cdot f \cdot V_{DD}^2 \quad (3-47)$$

where

P= power consumption (Watts)

K=technology factor, close to 0.5

C= total active capacitance of electrical nodes (F) (Not taking into account decoupling capacitance)

f=operational frequency of the integrated circuit (Hz)

$V_{dd}$ = supply voltage (V)

## Oxide Breakdown

The second reason for internal low voltage operation is the oxide breakdown. Increased switching performances have been achieved by a continuous reduction of the gate oxide thickness. In  $0.12\mu\text{m}$  technology, the MOS device has an ultra thin gate oxide, around  $0.002\mu\text{m}$ , that is  $2\text{nm}$  or  $20\text{\AA}$ . Knowing that the molecular distance of  $\text{SiO}_2$  oxide is around  $2\text{\AA}$ ,  $20\text{\AA}$  means 10 atoms. The oxide may be destroyed by a voltage higher than a maximum limit  $V_{crit}$ , called oxide breakdown voltage. A first order estimation is  $0.1\text{V}/\text{\AA}$  [Wang], which is expressed by equation 3-47.

$$V_{crit} = \frac{K}{tox} \quad (3-47)$$

With

$K$ =breakdown coefficient (Close to  $1\text{ V}\cdot\text{nm}$ )

$Tox$  = oxide thickness in nm

$V_{crit}$ =critical breakdown voltage (V)

Consequently, in  $0.12\mu\text{m}$ , the breakdown voltage is around  $2.0\text{V}$ , that is less than twice the nominal  $VDD$  ( $1.2\text{V}$ ). An illustration of the breakdown voltage is proposed in figure 3-xxx. If we display the  $Id/Vd$  characteristics with  $Vg$  higher than  $VDD$ , (for example  $2.5\text{V}$  instead of  $1.2\text{V}$ ), the oxide damage is represented by dotted lines (Here for  $Vg > 2.0\text{V}$ ). The MOS polarization should always be fixed in such a way that the gate voltage is lower than the breakdown voltage limit.

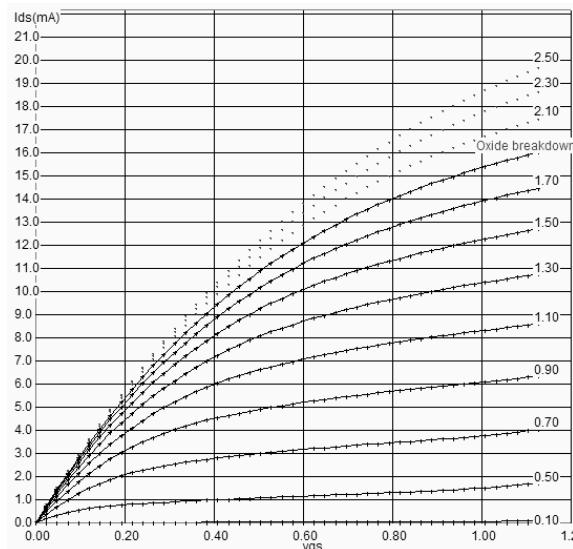


Fig. 3-46. Illustration of the breakdown voltage for a low leakage nMOS device, with a very high voltage applied on the gate

The oxide may be damaged by a 2V gate voltage. If the gate voltage is further increased, the physical destruction of the oxide may be observed, which usually results in a permanent conductive path between the gate and the source (Figure 3-47).

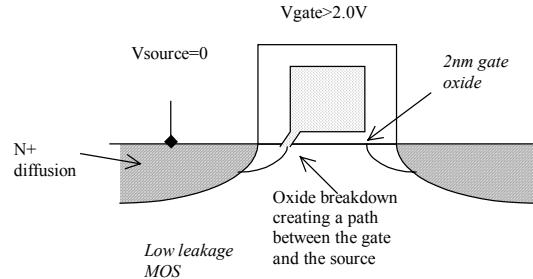


Fig. 3-47: oxide breakdown appears for gate voltage significantly higher than the nominal supply voltage

### Microwind Configuration

A set of specific parameters are used for each MOS option to configure the BSIM4 and LEVEL3 models. The industrial approach usually consists in describing each MOS device in a completely separated set of model parameters. Consequently, MOS model cards may include several thousands of parameters. We are trying to be as practical and didactic as possible, at the cost of a poor matching between measured and simulated MOS characteristics. In table 3-5, the list of the main varying parameters includes the gate oxide, the effective channel length parameter, and the threshold voltage.

Parameter	Description	NMOS value in 0.12µm	NMOS value in 0.12µm	Name in RUL file
TOX	Gate oxide thickness (low leakage) (high speed) (high voltage)	20Å 20 70	20Å 20 70	B4TOX B4T2OX B4T3OX
LINT	Channel-length offset parameter (low leakage) (high speed) (high voltage)	0.0 nm 10 0.0	0.0 nm 10 0.0	B4LINT B4L2INT B4L3INT
VTHO	Long channel threshold voltage (low leakage) (high speed) (high voltage)	0.40 V 0.30 0.50	0.40 V 0.30 0.50	B4VTHO B4V2THO B4V3THO

Table 3-5: BSIM4 parameters variation depending on the MOS option

## 6. 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, and up to 30% difference between separate dies. One varying parameter is the effective channel length. In figure 3-48, although both devices have been designed with a drawn 2 lambda, the result is a  $0.11\mu\text{m}$  length for the MOS situated on the left side, and  $0.13\mu\text{m}$  for the MOS situated on the right side.

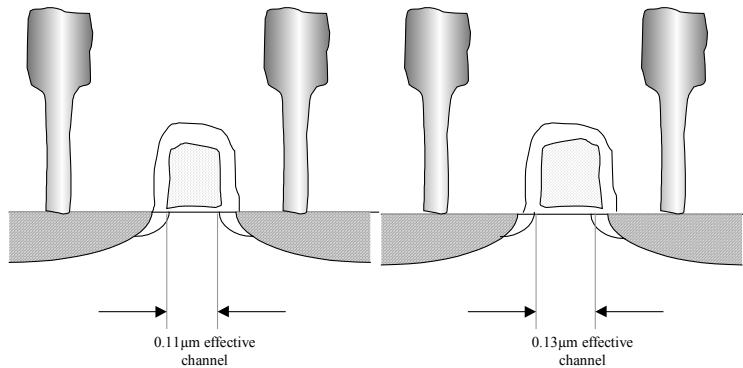


Fig. 3-48: The same MOS device may be fabricated with an important effective channel variation

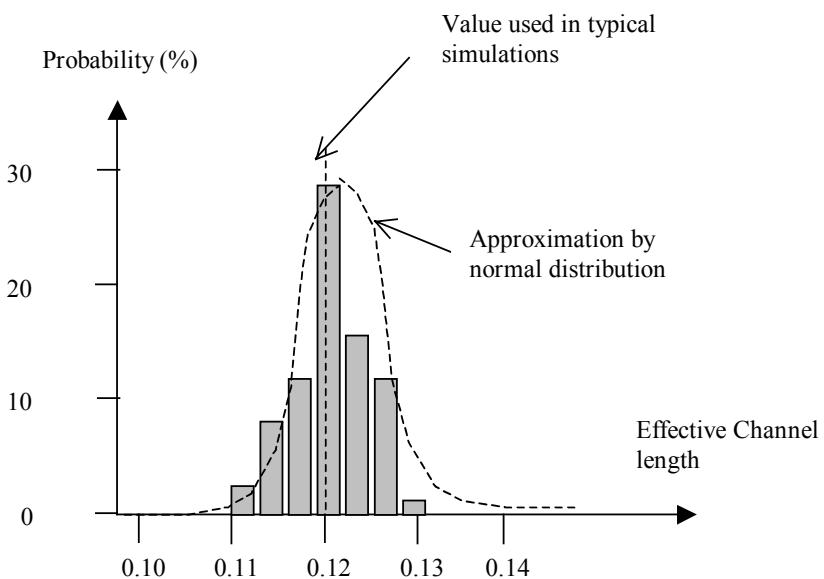


Fig. 3-49: The effective channel length may vary significantly with the process

If we cumulate several measurements on a wide number of devices, we can plot the probability of occurrence versus the measured effective length. The curve is usually a normal distribution with a center close to the default parameter given in the electrical rules (Figure 3-49).

### Simulation with Microwind

The menu **Simulate → Simulation parameters** gives a simple access to minimum/typical/maximum parameter sets (Figure 3-50). The industrial approach usually consists in providing a separate set of model parameters for each case, which represents a huge amount of model parameters. In Microwind, the approach has consisted in altering two main parameters: the threshold voltage (20% variation) and the mobility (20% variation). All other parameters are supposed to be constant.

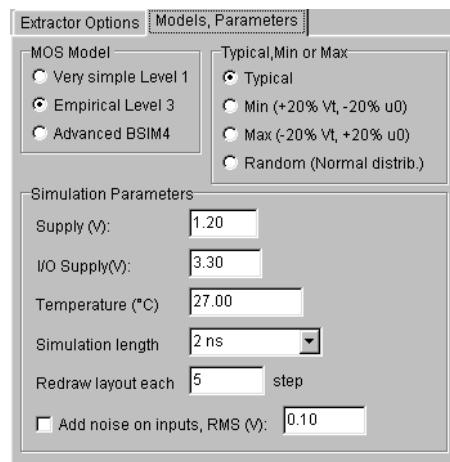


Fig. 3-50: Access to minimum, typical, maximum model parameters or Random simulation

A comparative simulation of the  $Id/Vd$  curve in typical, maximum and minimum scenarios shows a very large variation of performances (Figure 3-51). 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).

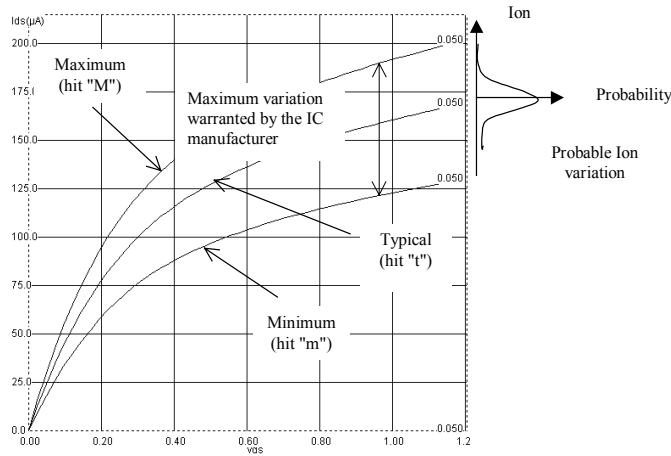


Fig. 3-51: The MOS  $Id/vd$  curve in Min, Typ, Max modes.

To superimpose the three curves, click the small brain icon (Enable Memory), and increase the **Step Vg** to 1.2 to draw only one curve for each mode. Notice the important variation between the minimum Ion and maximum Ion (From  $125\mu A$  to  $200\mu A$ ). In reality, the MOS characteristics vary in a normal distribution around the typical case. Consequently, the Ion current of this MOS device is very likely to reach the typical value. The min/max simulation is very interesting to validate the design in extreme situations. The min/max simulation should also consider the temperature: the worst current is obtained at high temperature and with a minimum set of parameters, while the highest current is obtained at low temperature and with a maximum set of parameters.

## 7. Concluding remarks

The number of parameters required for various MOS models is reported in figure 3-52. It can be seen that the trend is to increase the number of parameters, in order to take into account various effects linked to the device scale down.

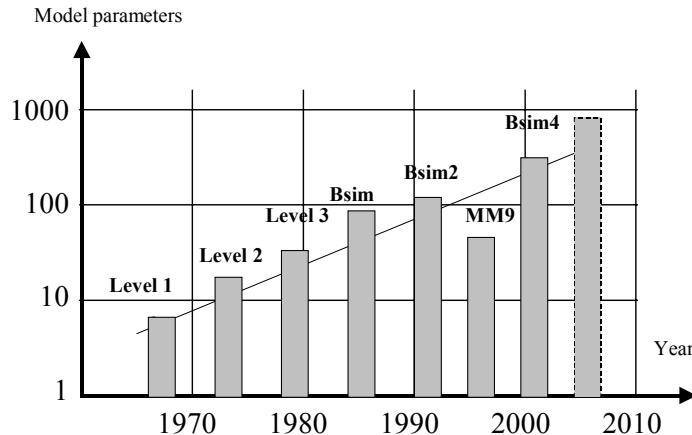


Fig. 3-52. Increased number of parameters in the MOS models

Even with advanced models, the resulting models may not fit well in all operating regions, for all device sizes. This is why the industrial approach for building model parameters is based on optimization mathematical algorithms. In deep submicron technology, the model parameters have a strong variation with the device size. For example the threshold voltage and mobility vary significantly with the device length, and the equations cannot always handle these dependencies properly. One solution is called binning. It consists in breaking the width-length space into several regions, as illustrated in figure 3-53. In each region, a specific set of model parameters is setup and optimized.

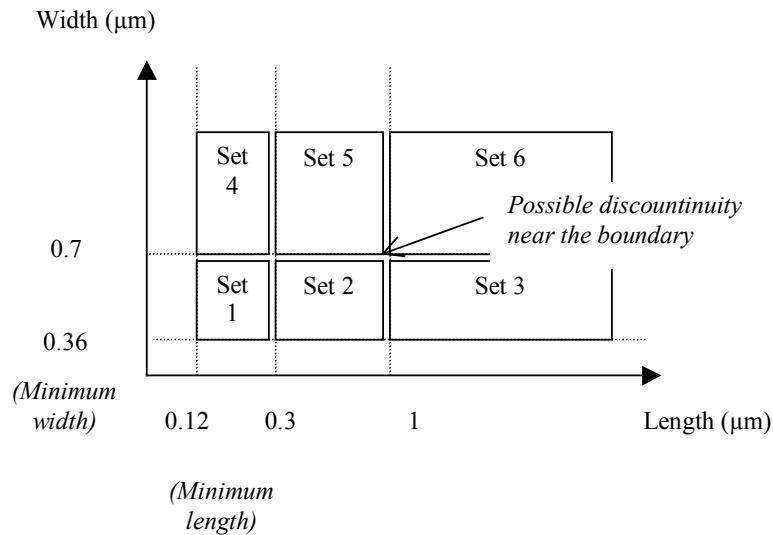


Figure 3-53: Using 6 sets of parameters to accurately cover the whole range of width and length

Binning severely complicates the process of parameters extraction. In the case of figure 3-xxx, 6 sets of parameters are required. Notice that set 1 covers small length and small width. Set n°2 covers a wider length

interval, while set n°3 is valid for any length greater than  $1\mu\text{m}$ . This is because long length devices are easier to model than short length devices, where many second order effects appear. Binning is used in industry to increase the analog simulation accuracy, at the cost of several drawbacks: the simulation time cost due to model complexity, and discontinuities in the current prediction that may be observed at the boundary of two sets. These limitations are the fuel for constructing more complex models that fit well for the whole range of width and length. In the future, nano-scale technologies may require MOS models with up to 1000 parameters, requiring a degree of expertise never attained up to now.

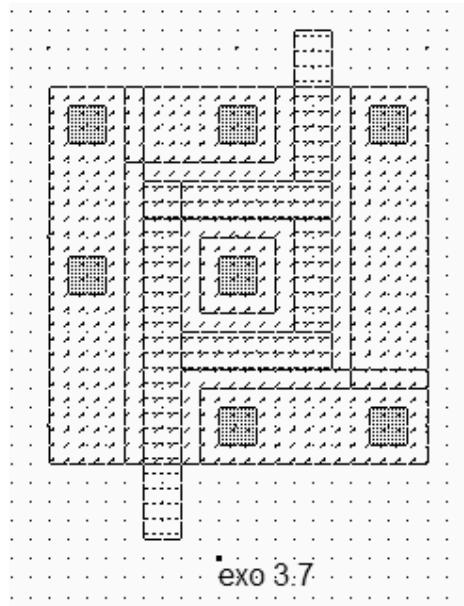
## REFERENCES

- [Shockley] W. Shockley "A Unipolar field effect transistor", proceedings of IRE, vol 40, Nov 1952, PP. 1365-1376
- [EIA] <http://www.ei.org/eig/CMC>
- [Shichman] H. Shichman, D. Hodges, "Modeling and simulation of insulated-gate field effect transistor switching circuits", IEEE J. Solid State Circuits, vol 3, pp 285-289, 1968
- [Tsividis] Y. P. Tsividis "Operating and Modeling of the MOS transistor", McGraw-Hill, 1987, ISBN 0-07-065381-X
- [Sze] S. M Sze "Physics of Semiconductor devices", John-Wiley, 1981, ISBN 0-471-05661-8
- [Cheng] Y. Cheng, C. Hu "MOSFET Modeling & BSIM3 user's guide", Kluwer Academic Publishers, 1999
- [Bsim4] BSIM4 web site [www-device.eecs.berkeley.edu](http://www-device.eecs.berkeley.edu)
- [Weste] N. Weste, K. Eshraghian "Principles of CMOS VLSI design", Addison Wesley, ISBN 0-201-53376-6, 1993
- [Lee] K. Lee, M. Shur, T.A Fjeldly, T. Ytterdal "Semiconductor Device Modeling for VLIS", Prentice Hall, 1993, ISBN 0-13-805656-0
- [Liu] W. Liu "Mosfet Models for SPICE simulation including Bsim3v3 and BSIM4", Wiley & Sons, 2001, ISBN 0-471-39697-4
- [Wang] Albert Z.H. Wang "On-Chip ESD protection for Integrated Circuits", An IC Design Perspective, Kluwer Academic Publishers, 2002, ISBN 0-7923-7647-1
- [TSMC] <http://www.tsmc.com>

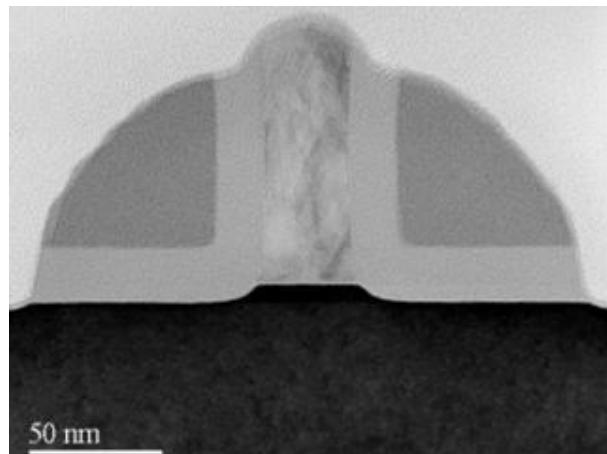
## EXERCISES

- 3.1 Configure Microwind in  $0.35\mu\text{m}$  technology (File **cmos035.RUL**) using the command **File → Select Foundry**. Compare simulation and measurement in  $0.35\mu\text{m}$  for a nMOS device,  $W=10\mu\text{m}$ ,  $L=0.4\mu\text{m}$  (File **Na10x0.4.MES**). Evaluate the mismatch between level 1, level 3 and BSIM4. In which domain is model 3 poorly fitted?

- 3.2 Configure Microwind in  $0.18\mu\text{m}$  technology (File **cmos018.RUL**). Compare simulation and measurement in  $0.18\mu\text{m}$  for a nMOS device,  $W=4\mu\text{m}$ ,  $L=0.2\mu\text{m}$ , low leakage option (File **Nc4x0.2.mes**). Evaluate  $R_{on}$ ,  $I_{on}$ ,  $I_{off}$ , and  $V_t$ . Perform the same evaluation for the high speed MOS, same size (File **NcHS4x0.2.mes**). Perform the same evaluation for the high voltage MOS, same size (File **NcHV4x0.2.mes**).
- 3.3 In low power applications, the n-well can be connected to a voltage  $V_{well}$  different from VDD. This non usual polarization aims at modifying the threshold of the p-MOS transistor. Under which condition for  $V_{well}$  the threshold of the PMOS transistor (Low Leakage option) is decreased to  $0.2\text{V}$  in  $0.12\mu\text{m}$ ?
- 3.4 Design a MOS device with  $I_{on}=10\text{mA}$ , minimum gate length in  $0.12\mu\text{m}$ . What is  $I_{off}$ ? How to obtain a MOS with  $I_{on}=10\text{mA}$  but twice less  $I_{off}$ ?
- 3.5 Compare the value of  $V_{GS\_T0}$  (The value of  $V_{gs}$  for which  $I_{ds}$  is independent of temperature) between MOS options in  $0.12\mu\text{m}$ .
- 3.6 Show that four MOS devices ( $W_n, L_n$ ) connected in parallel have approximately the same  $I_{on}$  than a single MOS device with a width equal to  $4 W_n$ . What is the origin of the mismatch?
- 3.7 What is the channel size of the following MOS device? Does Microwind correctly extract the channel size of that device?



- 3.8 In the following picture of a  $50\text{nm}$  MOS device, locate the gate, drain, and source, field oxide, gate oxide .



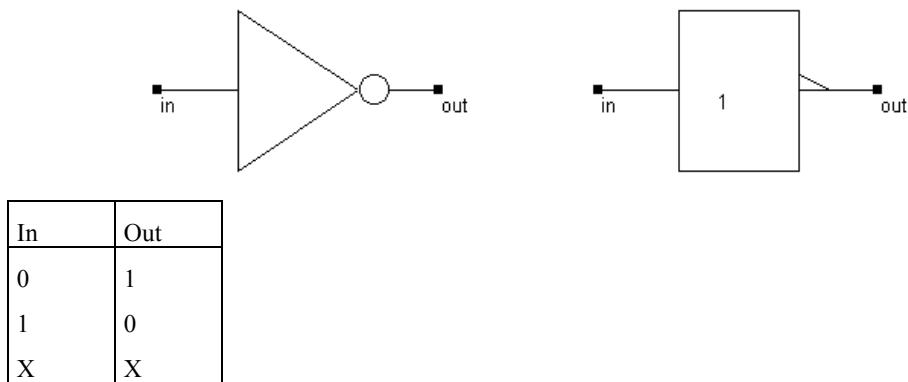
# 4

## The Inverter

The inverter is probably the most important basic logic cell in circuit design. This chapter introduces the logical concepts of the inverter, its layout implementation, the link between the transistor size and the static and analog characteristics. The manual design of the inverter is detailed. The performances of the inverter are analyzed in terms of static transfer function, switching speed, MOS options influence, and power consumption.

### 1. Logic symbol

Two logic symbols are often used to represent the inverter: the "old style" inverter (Left of figure 4-1), and the IEEE symbol (right of figure 4-1). In DSCH, we preferably use traditional symbol layout. As the logic truth table of figure 4-1 shows, the cell inverts the logic value of the input *In* into an output *Out*.



*Fig. 4-1: Symbols used to represent the logic inverter*

In the truth table, the symbol 0 represents 0.0V while 1 represents the logic supply, which is 1.2V in 0.12 $\mu$ m. The symbol X means "undefined". This state is equivalent to an undefined voltage, just like with a floating input node without any input connection. The undefined state appears in gray in the simulations and chronograms.

## 2. CMOS Inverter

The CMOS inverter design is detailed in the figure 4-2. Here one p-channel MOS and one n-channel MOS transistors are used as switches. Notice that the size of each device is plotted (W accounts for the width, L for the length). The channel width for pMOS devices is set to twice the channel width for nMOS devices. The reason is described in details in the next chapters.

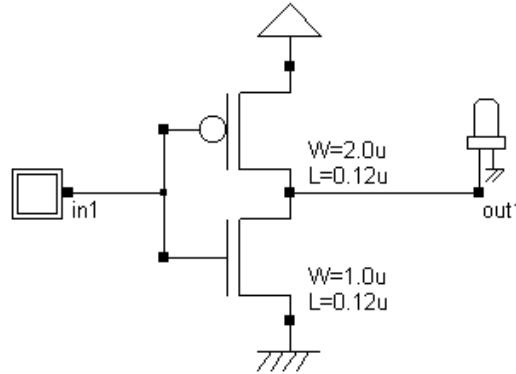


Fig. 4-2: The CMOS inverter is based on one n-channel and one p-channel MOS device

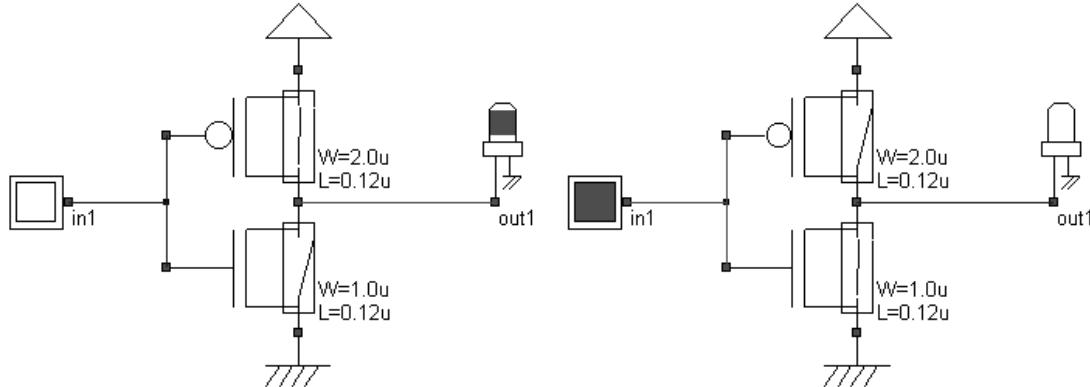


Fig. 4-3: Logic simulation of the CMOS inverter (CmosInv.sch)

When the input signal is logic 0 (Fig. 4-3 left), the nMOS is switched off while the pMOS passes VDD through the output, which turns to 1. When the input signal is logic 1 (Fig. 4-3 b), the pMOS is switched off while the nMOS passes VSS to the output which goes back to 0. In that simulation, the MOS is considered as a simple switch. The n-channel MOS symbol is a device that allows the current to flow between the source and the drain when the gate voltage is "1".

To simulate the inverter at logic level, start the software DSCH2, load the file “CmosInv.SCH”, and launch the simulation by the command **Simulate → Start Simulate**. Click inside the button *in1*. The result is displayed on the output *out1*. The red value indicates logic 1, the black value means a logic 0.

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

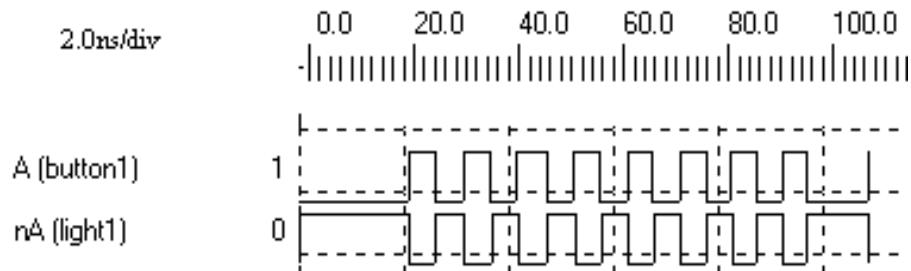


Fig. 4-4 Chronograms of the inverter simulation (CmosInv.SCH)

### 3. Inverter Layout

In this paragraph, details on the layout of a CMOS inverter are provided. The simplest way to create a CMOS inverter is to generate both n-channel MOS and p-channel MOS devices using the cell generator provided by Microwind. The advantage of this approach is to avoid any design rule error. The corresponding menu is reported below. You can generate an n-channel or p-channel device. A double gate device may also be created for EEPROM memory devices (See chapter 10). By default the proposed length is the minimum length available in the technology (2 lambda), and the width is 10 lambda. In 0.12 $\mu$ m technology, where lambda is 0.06 $\mu$ m, the corresponding size is 0.12 $\mu$ m for the length and 0.6 $\mu$ m for the width.

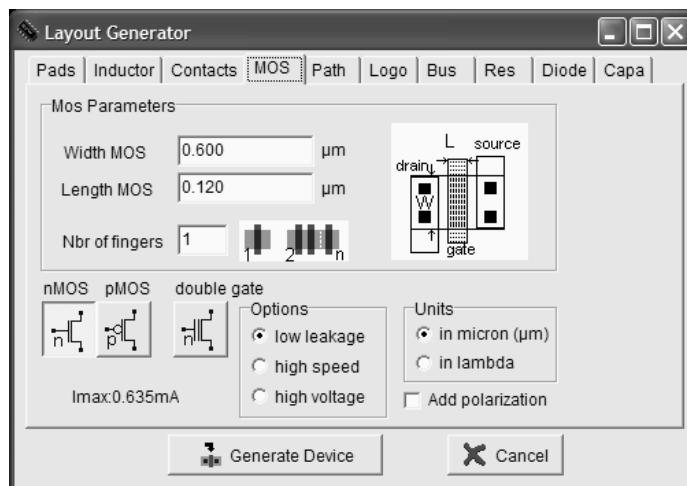


Fig. 4-5 Using the MOS generator to add n-channel and p-channel MOS devices on the layout

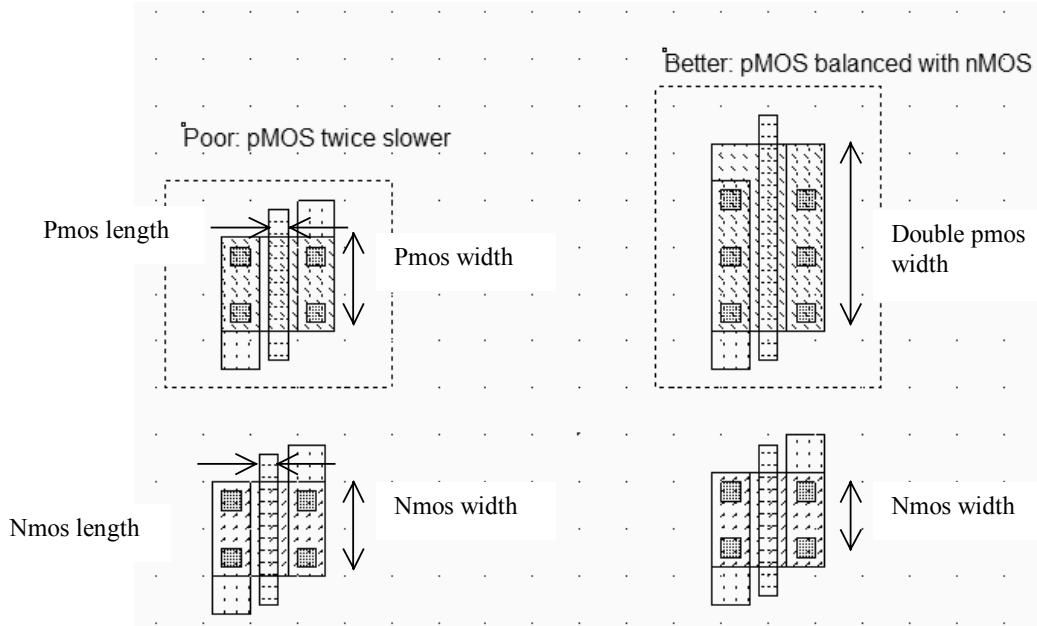


Fig. 4-6 The layout of one nMOS and one pMOS to build the CMOS inverter (invSizing.MSK)

The design starts with the implementation of one nMOS and one pMOS, as shown in figure 4-6. Using the same default channel width ( $0.6\mu\text{m}$  in CMOS  $0.12\mu\text{m}$ ) for nMOS and pMOS is not the best idea, as the p-channel MOS switches half the current of the n-channel MOS. The origin of this mismatch can be seen in the general expression of the current delivered by n-channel MOS devices (equation 4-1) and p-channel MOS devices (equation 4-2).

$$I_{ds}(Nmos) \approx \varepsilon_0 \varepsilon_r \frac{\mu_n}{TOX} \frac{W_{Nmos}}{L_{Nmos}} f(Vd, Vg, Vs, Vb) \quad (\text{Equ. 4-1})$$

$$I_{ds}(Pmos) \approx \varepsilon_0 \varepsilon_r \frac{\mu_p}{TOX} \frac{W_{Pmos}}{L_{Pmos}} f(Vd, Vg, Vs, Vb) \quad (\text{Equ. 4-2})$$

If  $W_{nmos}=W_{pmos}$  and  $L_{nmos}=L_{pmos}$ ,  $I_{ds}(Nmos)$  is proportional to  $\mu_n$  while  $I_{ds}(Pmos)$  is proportional to  $\mu_p$ . Typical mobility values are:

$$\mu_n = 0.068 m^2 / v.s \quad \text{for electrons}$$

$$\mu_p = 0.025 m^2 / v.s \quad \text{for holes}$$

Consequently, the current delivered by the n-channel MOS device is more than twice that of the p-channel MOS. Usually, the inverter is designed with balanced currents to avoid significant switching discrepancies. In other words, switching from 0 to 1 should take approximately the same time as switching from 1 to 0. Therefore, balanced current performances are required.

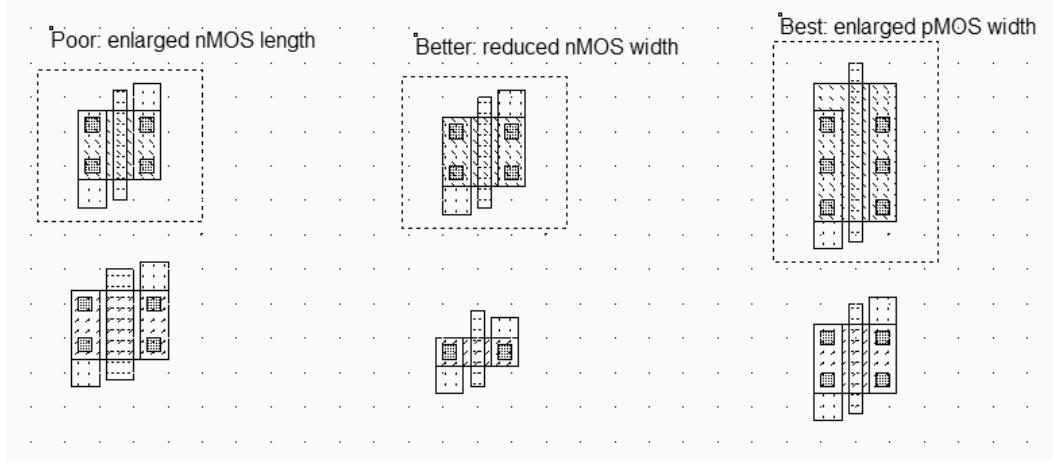


Fig. 4-7 Three techniques to compensate the poor hole mobility (invSizing.MSK)

There are several techniques to counterbalance the intrinsic mobility difference: increase the nMOS channel length (left of figure 4-7), decrease the nMOS channel width (middle), or increase the pMOS channel width. The main drawback of the design of figure 4-7(left) is the spared silicon area. The design in the middle is equivalent, but consumes less silicon space. However, reducing the nMOS width slows down the switching. The best approach (right) consists in enlarging the pMOS width. Its  $I_{on}$  current is doubled, and becomes comparable to the nMOS current. The behavior will be balanced in terms of switching speed.

## Connection between Devices

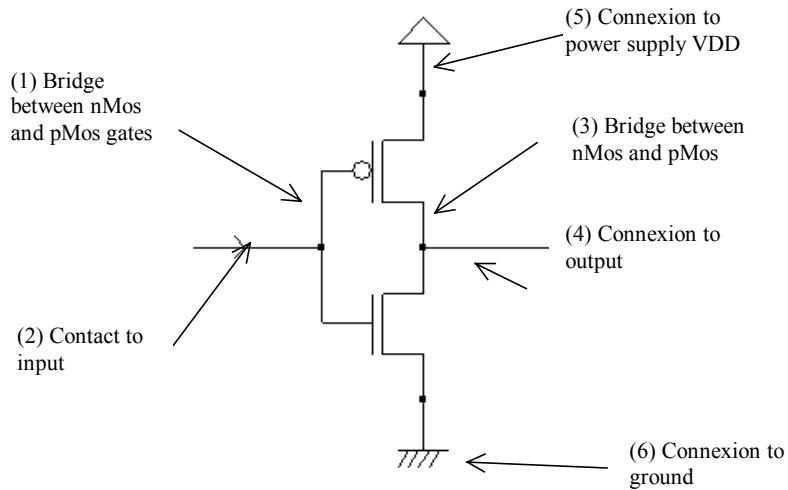


Fig. 4-8 Connections required to build the inverter (CmosInv.SCH)

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 4-8. Polysilicon is rarely used for long interconnects, except if a huge resistance value is expected.

In the layout shown in figure 4-9, 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.

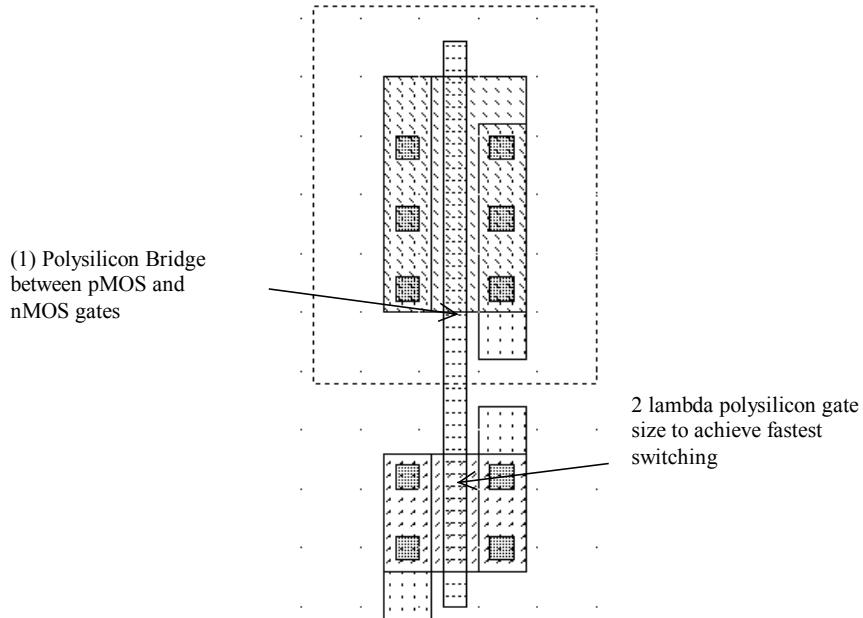


Fig. 4-9 Polysilicon bridge between nMOS and pMOS devices (*InvSteps.MSK*)

## Useful Editing Tools

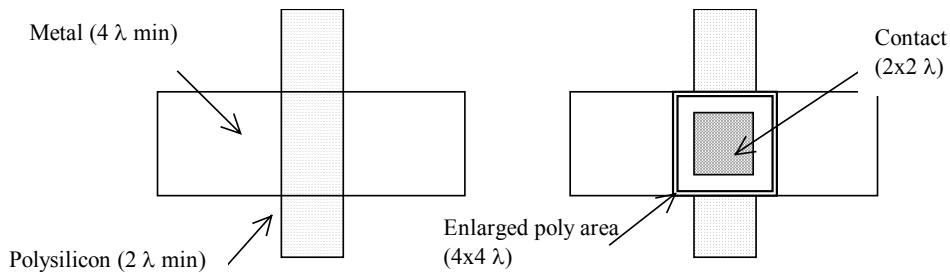
The following commands may help you in the layout design and verification processes.

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.

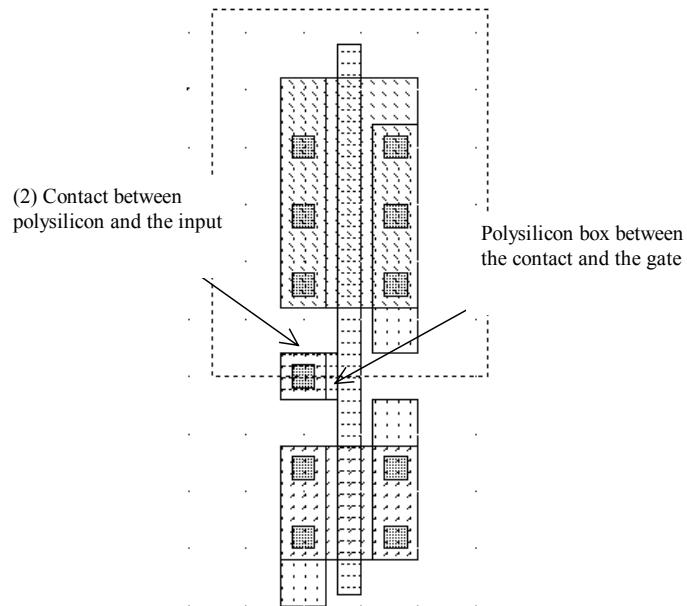
Table 4-1: A set of useful editing tools

## Metal-to-poly

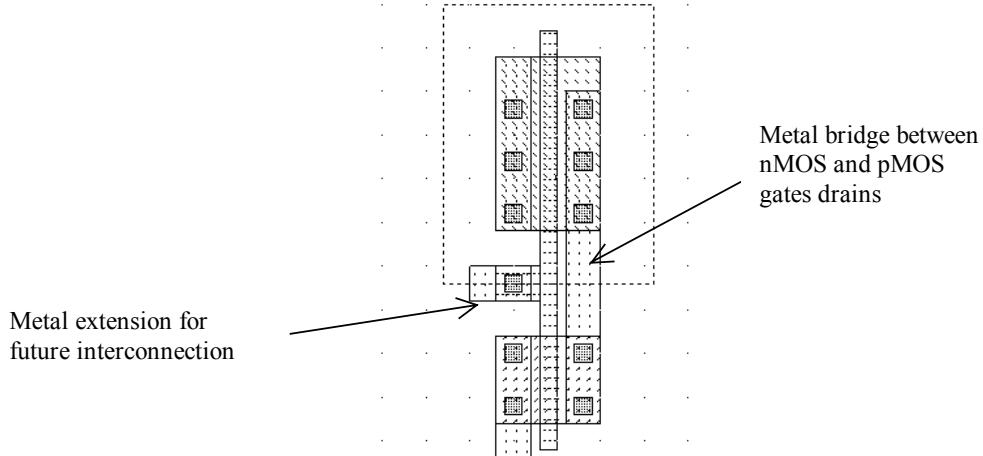
As polysilicon is a poor conductor, metal is preferred to interconnect signals and supplies. Consequently, the input connection of the inverter is made with metal. Metal and polysilicon are separated by an oxide which prevents electrical connections. Therefore, a box of metal drawn across a box of polysilicon does not allow an electrical connection (Figure 4-10). 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. 4-10 Physical contact between metal and polysilicon*

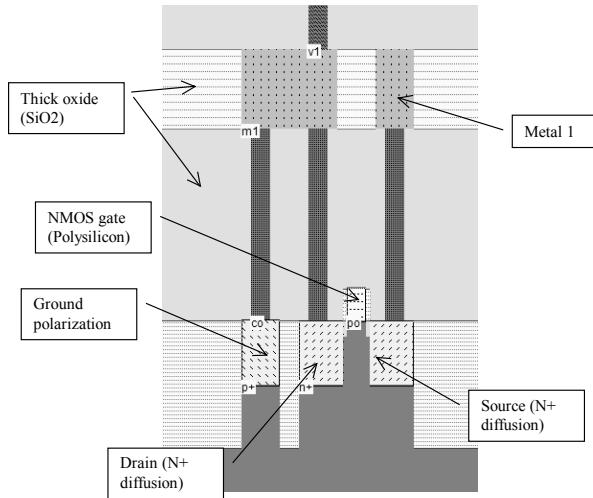


*Fig. 4-11 Physical contact between metal and polysilicon (InvSteps.MSK)*



*Fig. 4-12 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 4-13.

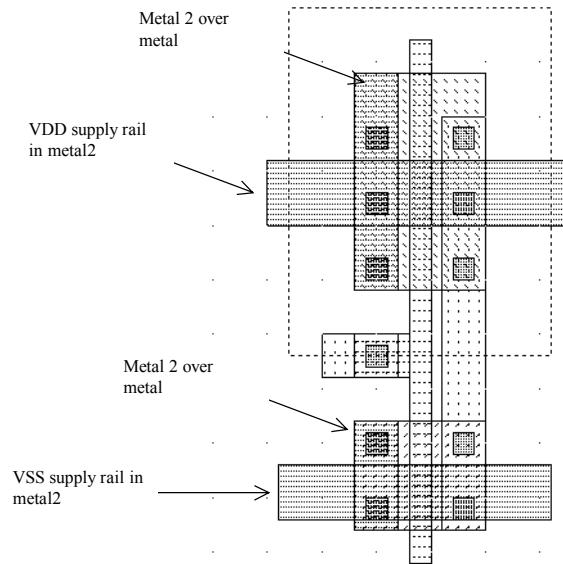


*Fig.4-13 The 2D process section of the inverter circuit near the nMOS device (InvSteps.MSK)*

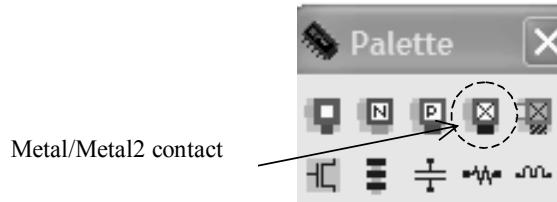
## Supply Connections

The next design step consists in adding supply connections, that is the positive supply VDD and the ground supply VSS. In figure 4-14, 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 called electromigration (More details are given in chapter 5 dedicated to interconnects).

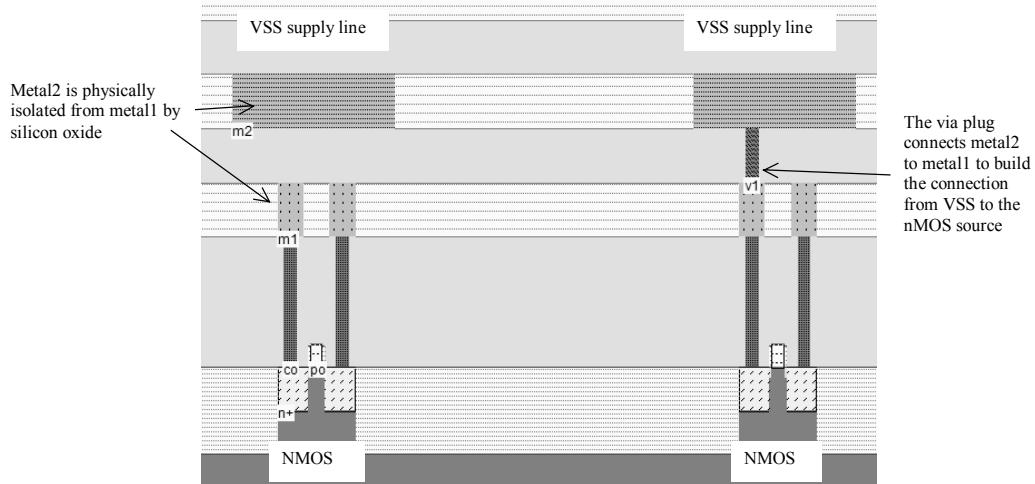


*Fig.4-14 Adding metal2 supply lines and the appropriate vias (InvSteps.MSK)*



*Fig.4-15 The metal/Metal2 contact in the palette*

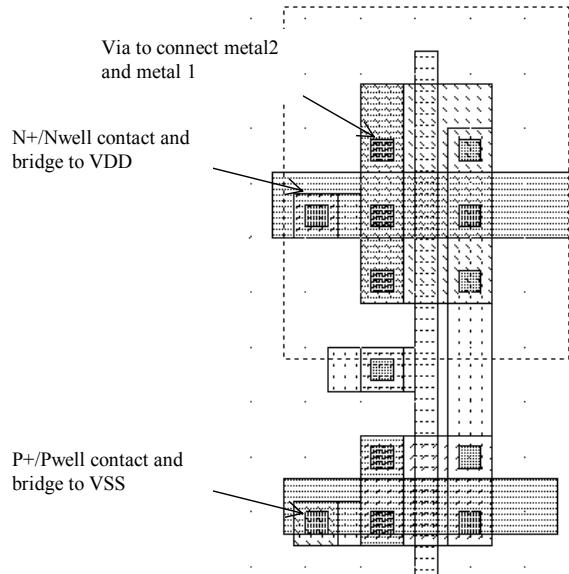
The metal layers are electrically isolated by a SiO<sub>2</sub> dielectric. Consequently, the metal2 supply line floats over the inverter cell and no physical connection exists down to the MOS source region. The simplest way to build the physical connection is to add a metal/Metal2 contact that may be found in the palette (Figure 4-15).



*Fig.4-16 2-D view of the connection built near the nMOS region to connect the source to the VSS supply line*

As seen in figure 4-16, 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. We have seen that the MOS behavior is influenced by the bulk polarization. For example,  $I_{on}$  is directly dependent on  $V_{BS}$ , which represents the voltage difference between the bulk and the source (See for example equation 3-31). See also the characteristics  $Ids$  versus  $Vgs$  for varying bulk voltage like in figure 3-26. If we ensure a clean supply polarization near each device (VSS for nMOS, VDD for pMOS), we avoid such variations. Remember that the n-well region should always be polarized to a high voltage to avoid short-circuit between VDD and VSS.



*Fig.4-17 Adding polarization contacts*

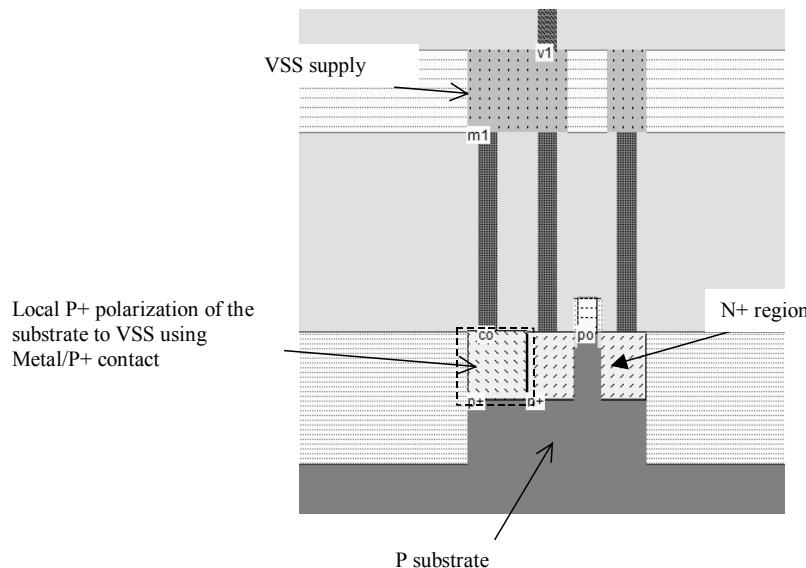


Fig.4-18 2-D view of the VSS polarization built near the nMOS source

More details about the vertical aspect of the VSS polarization is given in figure 4-18. When adding the metal/P+ contact, we create a VSS supply path to the P substrate. Consequently, the surrounding of the n-MOS device is firmly tied to VSS supply voltage. We also illustrate the VDD polarization near the pMOS channel in figure 4-19. The n-well region cannot be left without polarization.

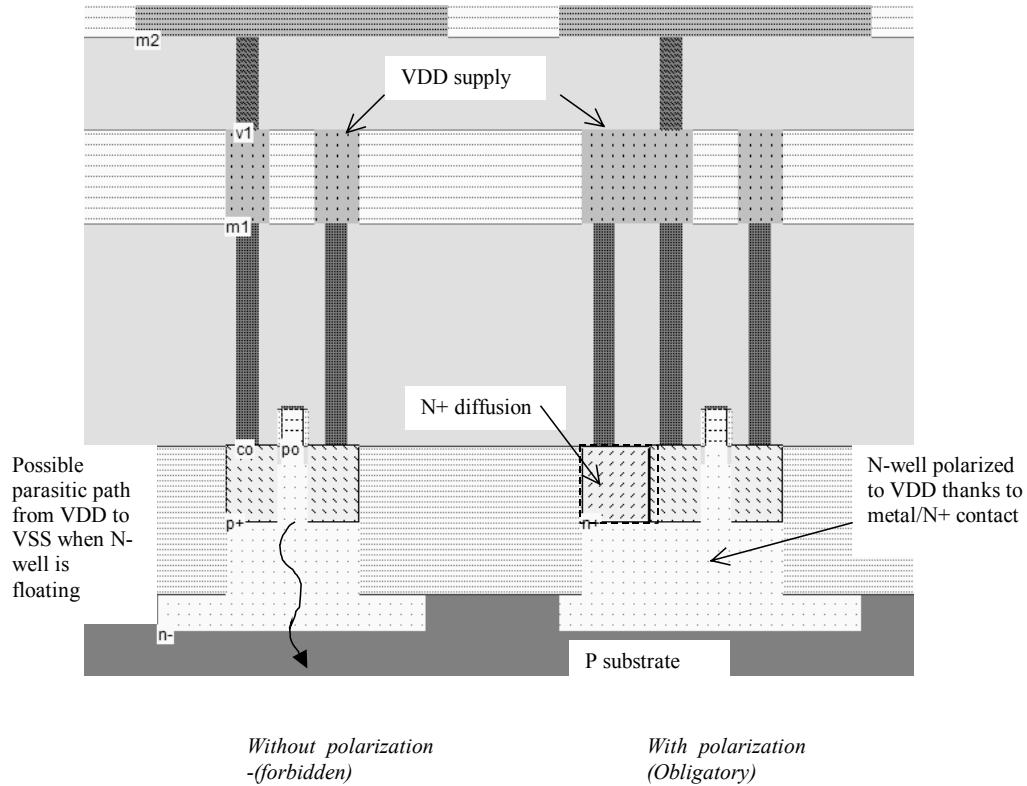


Fig.4-19 2-D view of the VDD polarization built near the pMOS source

Adding the VDD polarization in the n-well region is a very strict rule. The local polarization built with a metal/N+ diffusion contact, as shown in figure 4-19, is efficient to avoid a floating n-well region, which may result in parasitic current path from the PMOS source down to the P substrate usually tied to VSS. The current path may be strong enough to damage the chip. This effect is called latchup <Gloss>.

### 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 4-20, 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.

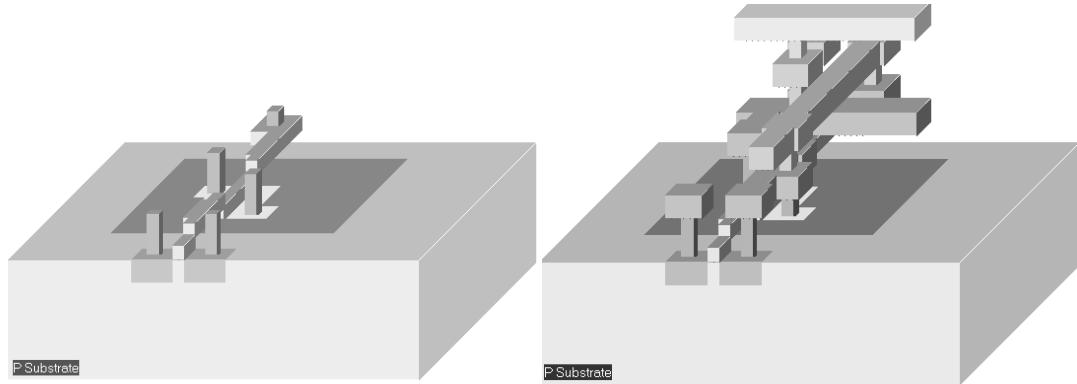


Fig.4-20 The step-by-step fabrication of the Inverter circuit (InvSteps.MSK)

## 4. 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 (The default name *out1* has been changed into *Vout*).

The expected behavior is shown in figure 4-22. The basic phenomenon is the charge and discharge of the output parasitic capacitor *Cout*, which is the sum of junction and wire capacitance. When *In1* is equal to 0, the pMOS device is on, and the capacitor *Cout* is charged until its voltage rises to VDD. When *In1* is equal to 1, the nMOS device is on, and the capacitor *Cout* is discharged until its voltage reaches VSS.

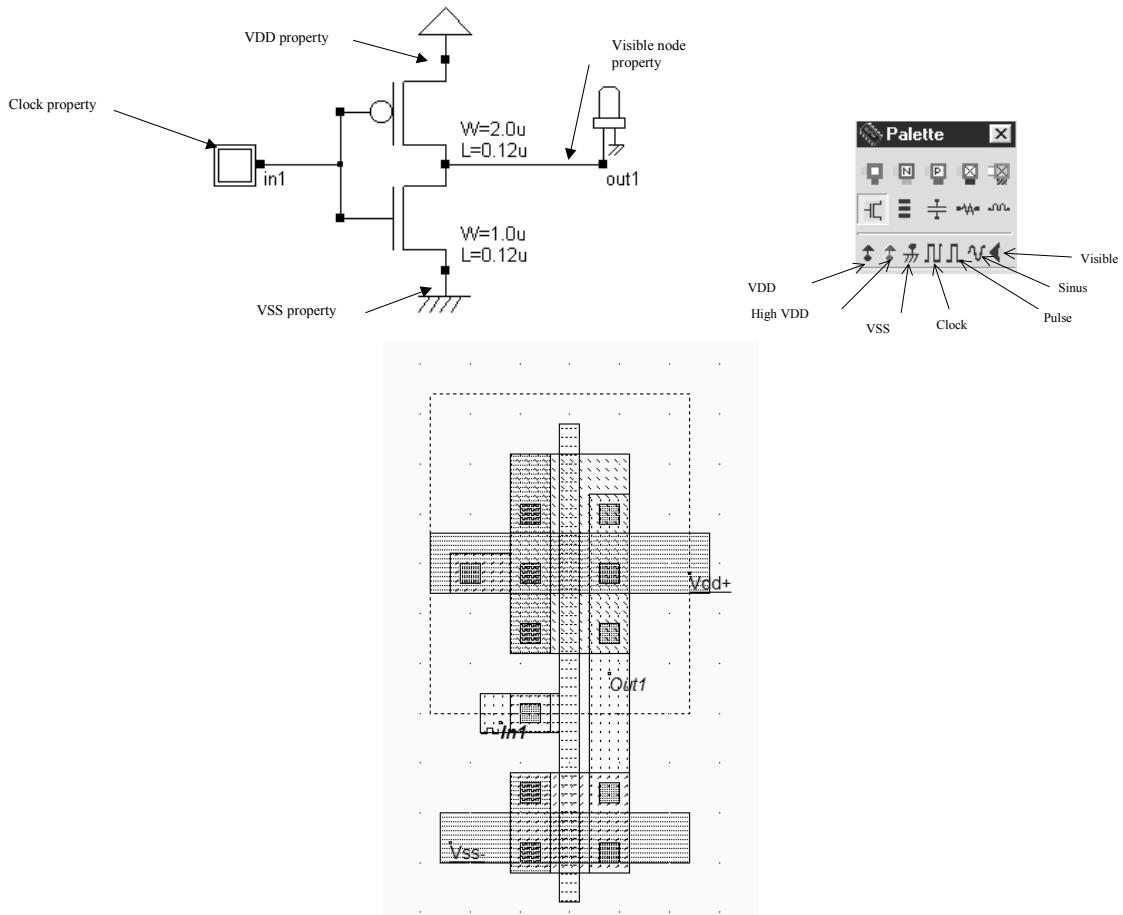


Fig.4-21 Adding simulation properties (InvSteps.MSK)

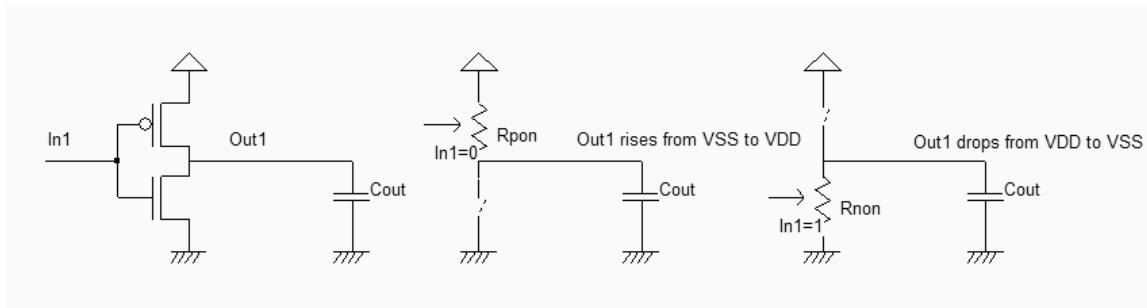


Fig.4-22 Expected behavior of the CMOS inverte (InverterLoad.SCH)

## Starting Simulation

The command **Simulate → Run Simulation** gives access to four simulation modes: **Voltage vs. time**, **Voltage and current vs. Time**, **Voltage vs. voltage** and **Frequency vs. time**. All these simulation modes are applicable to the inverter simulation.

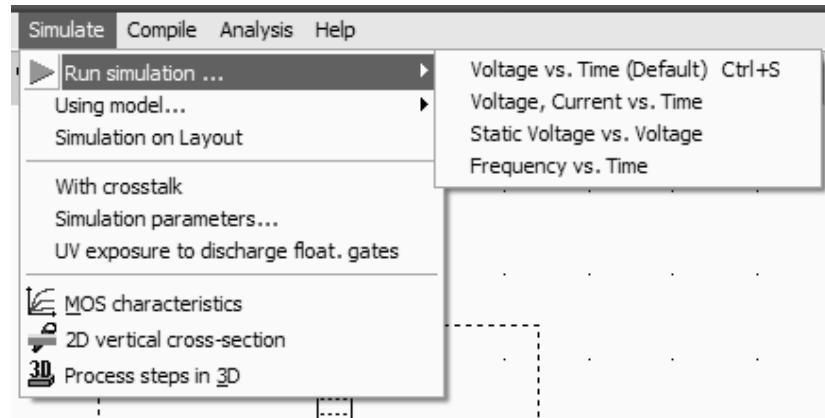


Fig.4-23 The four simulation modes in Microwind

Due to the fact that the layout **InvSteps.MSK** not only includes the inverter correctly polarized, but also several other mos devices without any simulation properties, a warning window appears prior to the analog simulation, as shown in figure 4-24. In this case, you may click **Simulate as it**. In normal cases, all n-well regions should be stuck at VDD.

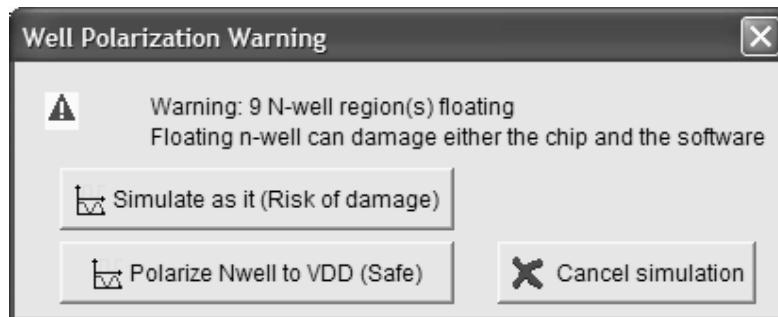


Fig.4-24 Missing polarization in n-well regions provoke a warning prior to simulation (*InvSteps.MSK*)

## Voltage vs. Time

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 4-25.

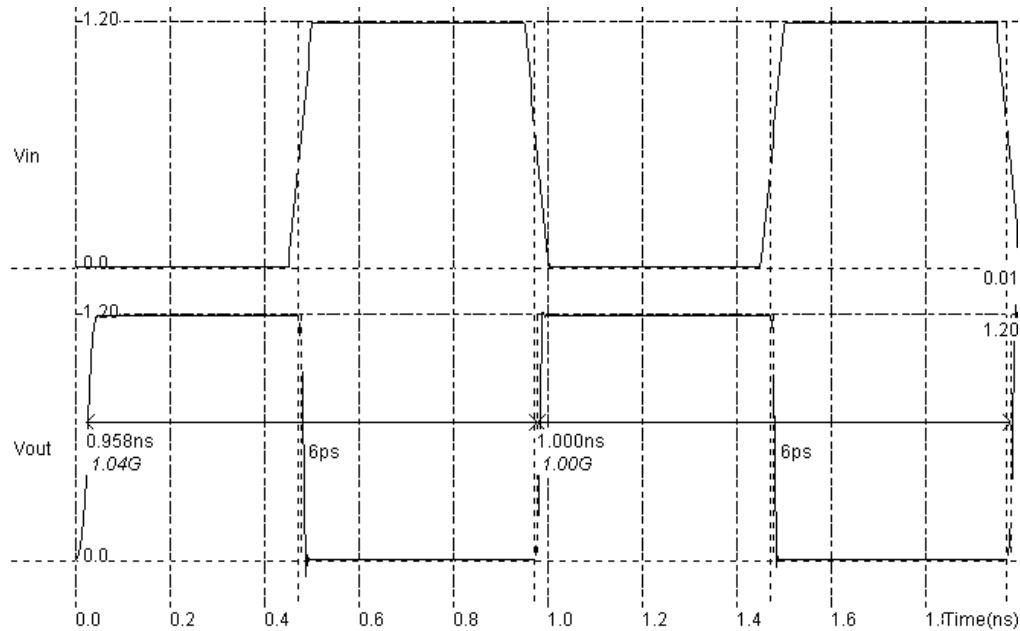


Fig.4-25 Transient simulation of the CMOS inverter (InvSteps.MSK)

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).

In	Out
0	1
1	0

Logic table

In (V)	Out (V)
0.0	1.2
1.2	0.0

Analog voltage table

## Current vs. Time

The inverter consumes power during transitions, due to two separate effects. The first is short circuit power arising from momentary short-circuit current that flows from  $VDD$  to  $VSS$  when the transistor functions in the incomplete-on/off state (Figure 4-26). The second is the charging/discharging power, which depends on the output wire capacitance. With small loading, the short circuit power loss is dominant. With a huge loading, that is a large output node capacitance, the loading power is dominant.

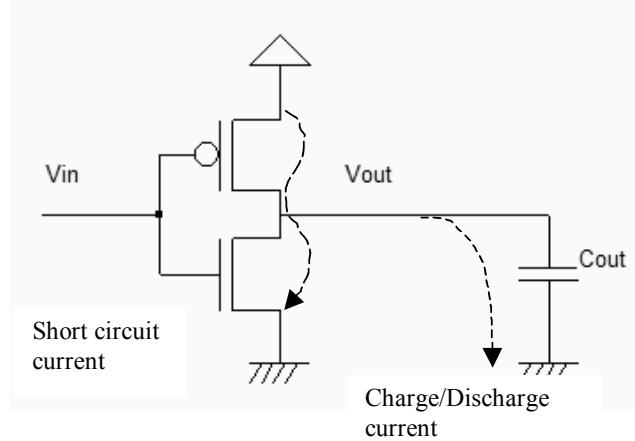


Fig. 4-26: Short circuit current in CMOS inverters (InverterLoad.SCH)

The power consumption occurs briefly during transitions of the output, either from 0 to 1 or from 1 to 0 (Fig. 4-27). 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.

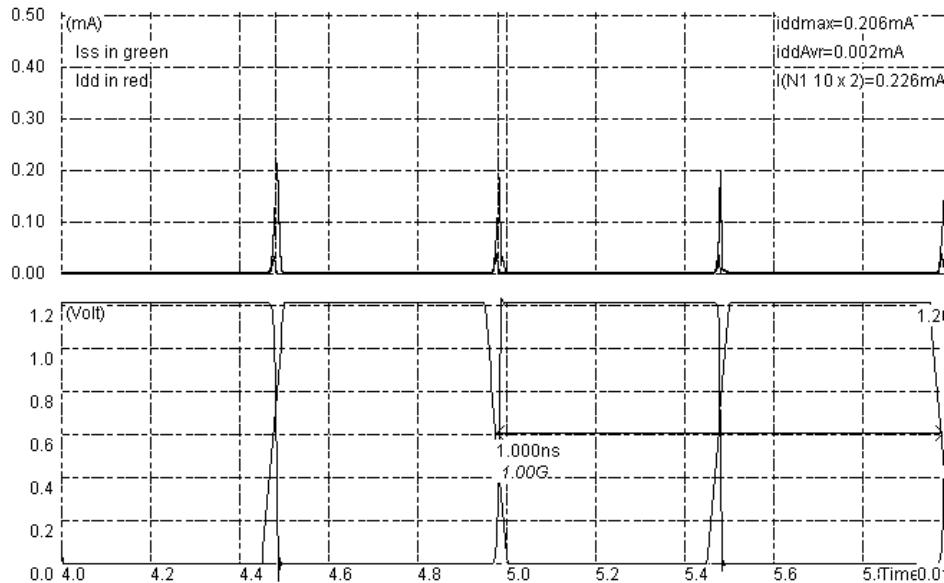


Fig. 4-27: Simulation of the current peaks appearing between  $VDD$  and  $VSS$  in the CMOS inverter at each output transition (InvSteps.MSK)

## Inverter Delay

As the number of gates connected to the inverter output node increase, the load capacitance increases. The fanout <glossary> corresponds to the number of gates connected to the cell output. Physically, a large fanout means a large number of connections (Figure 4-28), that is a large load capacitance.

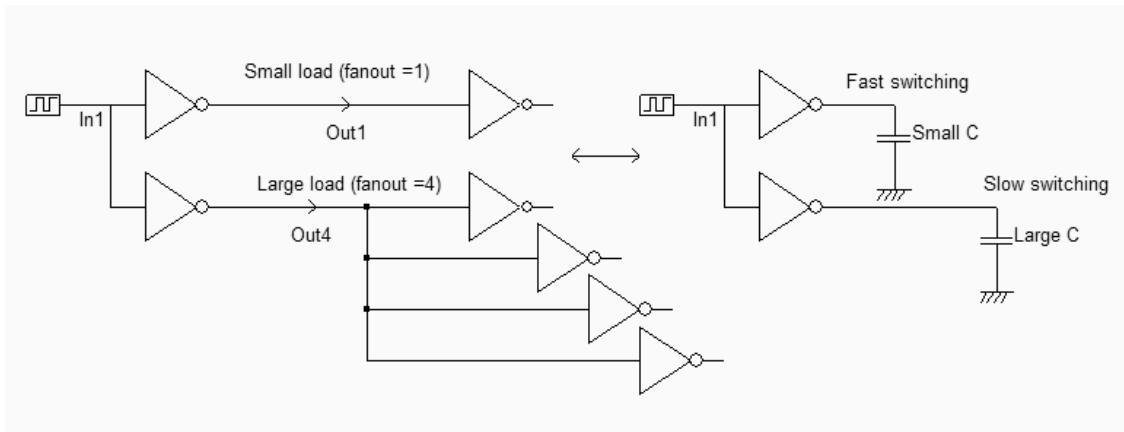


Fig. 4-28 One inverter connected either to a single inverter or to 4 inverters in parallel (InverterLoad.SCH)

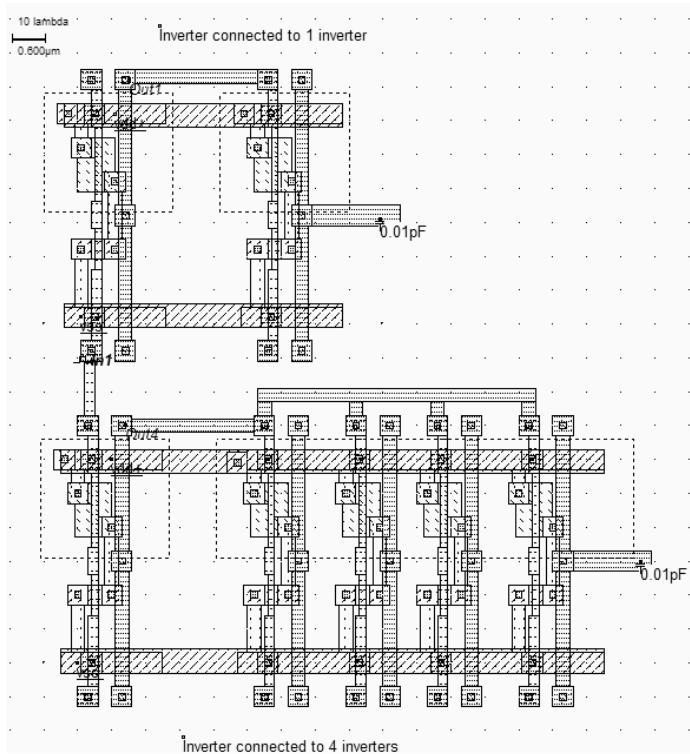


Fig. 4-29 One inverter connected either to a single inverter or to 4 inverters in parallel (InvFanout.MSK)

An inverter circuit is simulated using different clock, fanout and supply conditions. The initial configuration is based on one inverter controlled by a 2GHz clock, with its output connected either to a single inverter or to four inverters (Fig. 4-30). The supply voltage is 1.2V, with a 0.12μm CMOS technology.

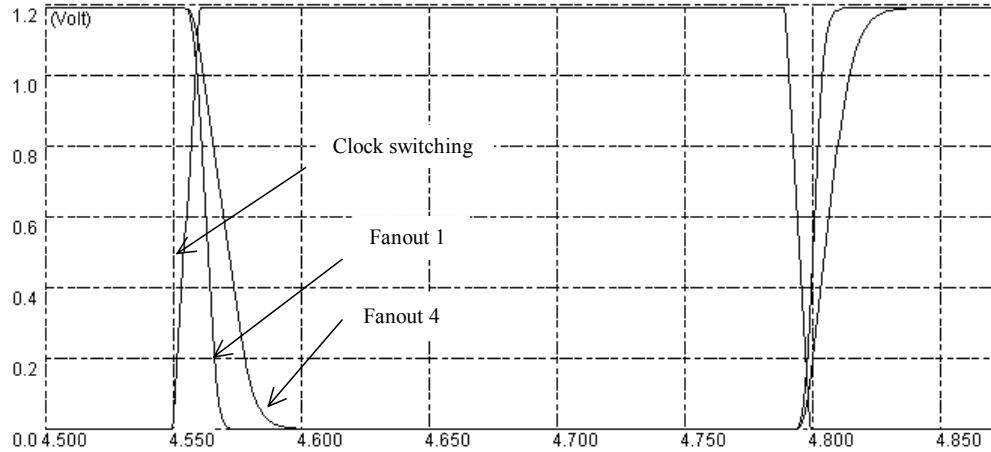


Fig. 4-30: Influence of the output capacitance on the current and switching response (InvFanout.MSK)

Now, we connect 4 inverter circuits to the output node, thus increasing the charge capacitance. In the simulation chronograms reported in figure 4-30, the inverter delay is significantly increased. When we investigate the delay variation with the output capacitance load, we observe the curve reported in figure 4-31. It can be seen that the gate delay variation with the loading capacitance is quite linear. A 100fF load leads to around 300ps delay in CMOS 0.12μm technology.

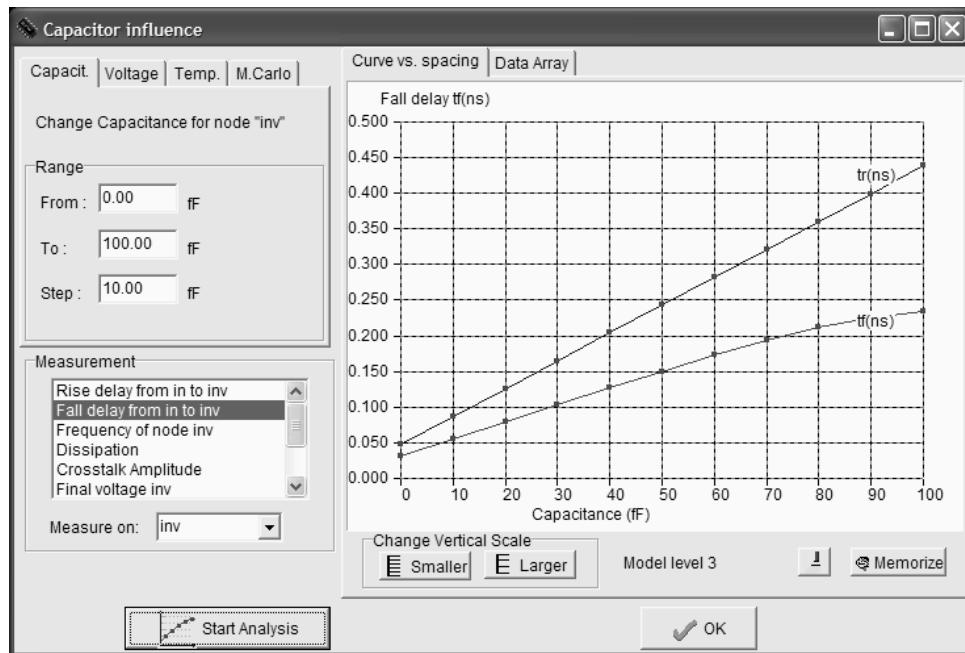


Fig. 4-31: Inverter delay increases with the output capacitance (InvCapa.MSK)

In Microwind2, we may obtain directly this type of screen thanks to the command **Parametric Analysis**. Load the file InvCapa.MSK, invoke the command **Parametric Analysis**, click in the output node, and click **Start Analysis**. By default, the capacitance of the output node is increased step by step from its default value  $C_{def}$  to  $C_{def}+100\text{fF}$ . For each value of the output capacitance, the analog simulation is performed, and the last computed rise time is plotted, appearing as one single red dot in the graphs. The complete graph is built once all analog simulations have been

completed. The memory button enables to store one curve (evaluation of the rise time for example) prior to a new parametric simulation, for comparison purposes. Three main parameters may vary in the parametric analysis: the capacitance as in figure 4-31, voltage, or temperature. Several analog parameters may be monitored: rise and fall delay, oscillating frequency, power consumption, final voltage of a node, crosstalk, etc..

## 5. Power Consumption

The power consumption  $P$  is computed by Microwind as the average product of the supply voltage  $VDD$  and the supply current  $IDD$ , computed at each iteration step. In other words:

$$P = \frac{\sum I_{DD} \cdot V_{DD}}{steps} \quad (\text{Eq. 4-3})$$

Three main factors contribute to the power consumption  $P$ : the load capacitance  $C$ , the supply voltage  $VDD$  and the clock frequency  $f$ . For a CMOS inverter, this relation is usually represented by the first-order approximation below. The equation 4-4 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  $VDD$ .

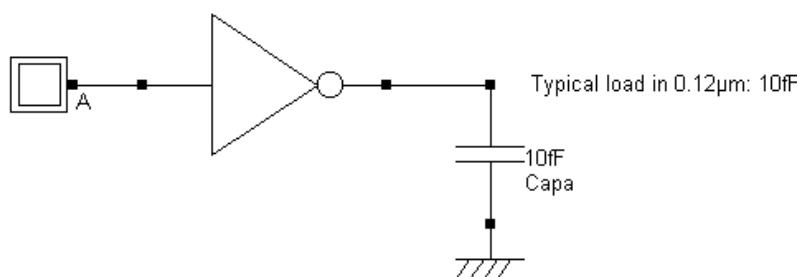
$$P = \frac{1}{2} \eta \cdot C \cdot V_{DD}^2 \cdot f \quad (\text{Eq. 4-4})$$

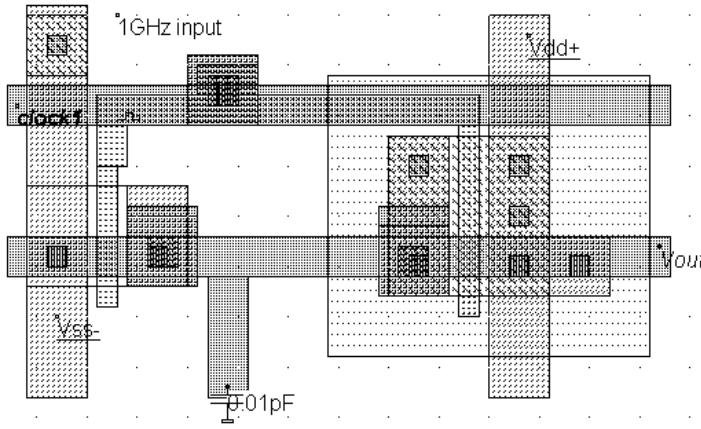
Where:

- $k$ : technological factor (close to 1)
- $C$ : Output load capacitance (Farad)
- $VDD$ : supply voltage (V)
- $f$ : Clock frequency (Hz)
- $\eta$ : switching activity factor (Between 0 and 1)

## Frequency dependence

We can verify the linear dependence of the power consumption with the operating frequency by simulating a CMOS inverter circuit. At each time-domain analog simulation, we get a value of the power consumption, which is computed by Microwind as the average product of the supply voltage  $VDD$  and the supply current  $IDD$  (Equation 4-3).

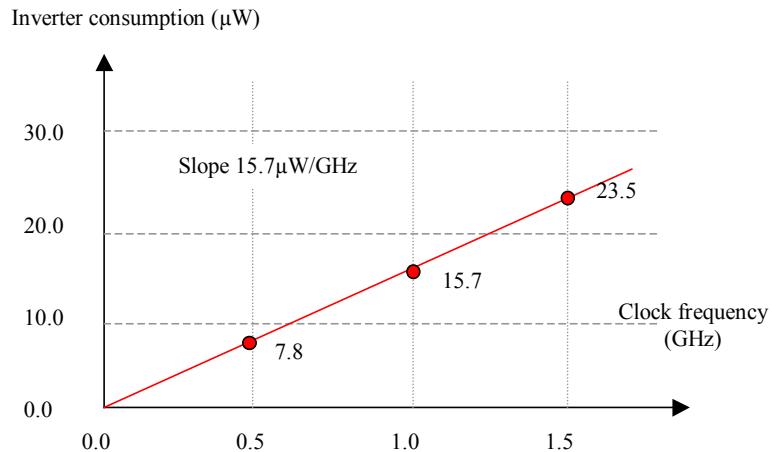




A typical load of 10fF is added to the output

*Fig. 4-32: CMOS inverter setup used to simulate the effect of the clock frequency on the power consumption. A 10fF load is added to the output to represent a typical loading condition in 0.12μm (CmosLoad.MSK)*

In the case of figure 4-32, a 1GHz switching of the inverter induces a circuit power dissipation of  $15.7\mu\text{W}$ . When we change the frequency, we observe a linear increase of P with the clock frequency, as forecast in equation 4-4 (Figure 4-33).



*Fig. 4-33: Power consumption increase with the clock frequency, for an inverter with a 10fF load, in 0.12μm CMOS technology (CmosLoad.MSK)*

As the power consumption is linearly proportional to the clock frequency, a usual metric found in most cell libraries is the  $\mu\text{W}/\text{GHz}$ . In the case of the simple inverter and its 10fF load, we get  $15.7\mu\text{W}/\text{GHz}$ .

## Supply Voltage dependence

It can be considered, as a first-order approximation that the average power consumption is proportional to  $VDD^2$  (Equation 4-4). We use the parametric analysis tool in Microwind to control the incremental change of the supply voltage, from 0.5 to 2.0V. The supply voltage step is 0.1V. In the measurement window, the item "Dissipation" is selected. The result plotted in the figure 4-44 shows a non-linear dependence of the power dissipation with VDD. The square law fits with the experimental data from 0.8 to 1.5V. We notice a very important rise of the power consumption over 1.5V, due to the avalanche effects in n-channel MOS devices. This simulation demonstrates the interest for a minimum supply operation to achieve optimum low-power operation.

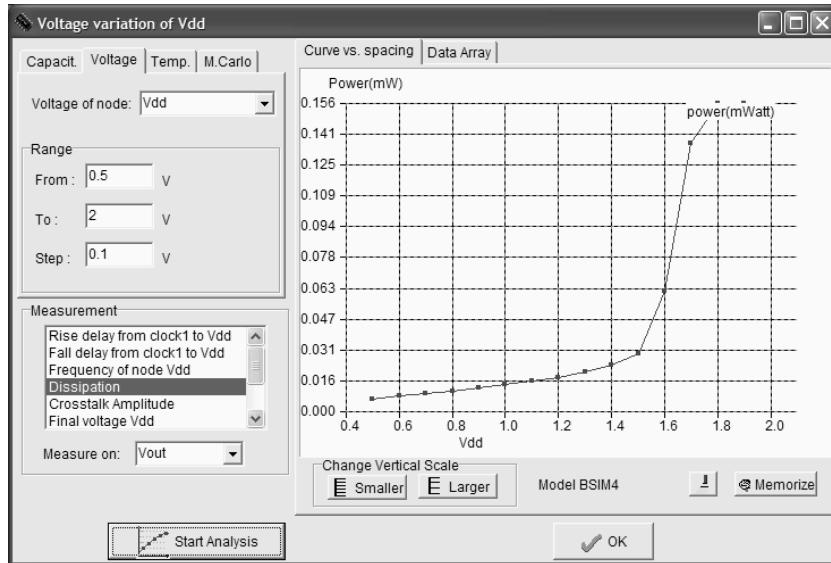


Fig. 4-44: Analysis of the power consumption increase with the supply voltage VDD (CmosLoad.MSK)

## Minimum Supply Voltage

The question is: what is the supply voltage below which the inverter does not work anymore? The answer can be given by the parametric analysis, focusing this time on the inverter delay dependence versus the supply voltage. Load the file **CmosLoad.MSK** for this study. Invoke the command **Parametric Analysis** of the Analysis menu. Click the layout region corresponding to the node VDD. Verify that the Voltage menu is selected in the parametric analysis window. Verify that the node "VDD" is selected. Modify the VDD voltage range from 0.5 to 1.5V, step 0.1V. Finally, in the measurement menu, select the item **Rise delay** and click **Start Analysis**.

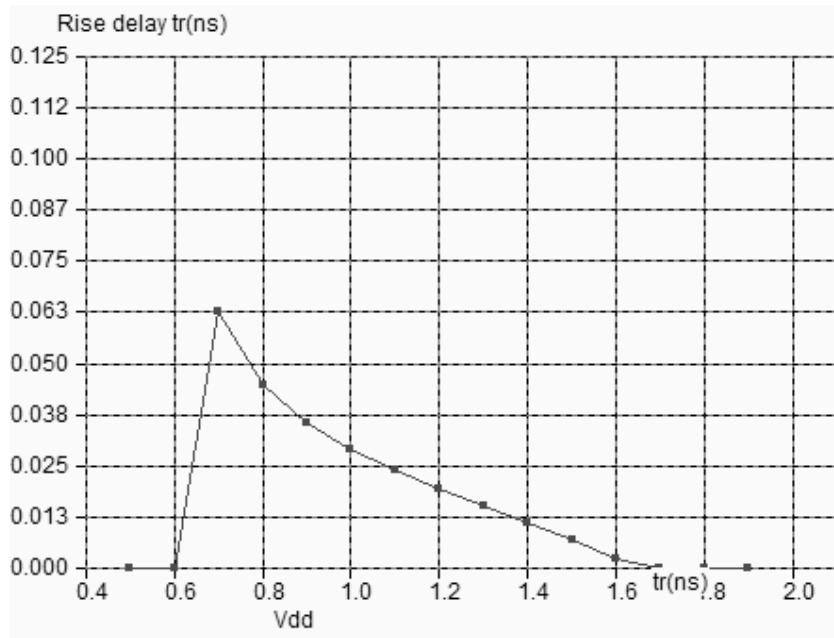


Fig. 4-45: Switching delay dependence with the supply voltage VDD (CmosLoad.MSK)

We observe that the delay is significantly increased as we decrease VDD from its nominal value 1.2V down to 0.6V. Below 0.7V, the inverter delay is higher than the default transient simulation time (10ns) so that the delay evaluator does not work anymore.

## 6. Static Characteristics

The static characteristics of the inverter correspond to the variation plot of the output voltage versus the input voltage. The simulation involves a step by step increase of  $V_{in}$ , and the monitoring of  $V_{out}$ . In the simulation window, the static characteristics are obtained by a click on the item **Voltage vs. Voltage** situated in the selection menu, at the bottom of the chronograms. The curve shown in figure 4-46 appears.

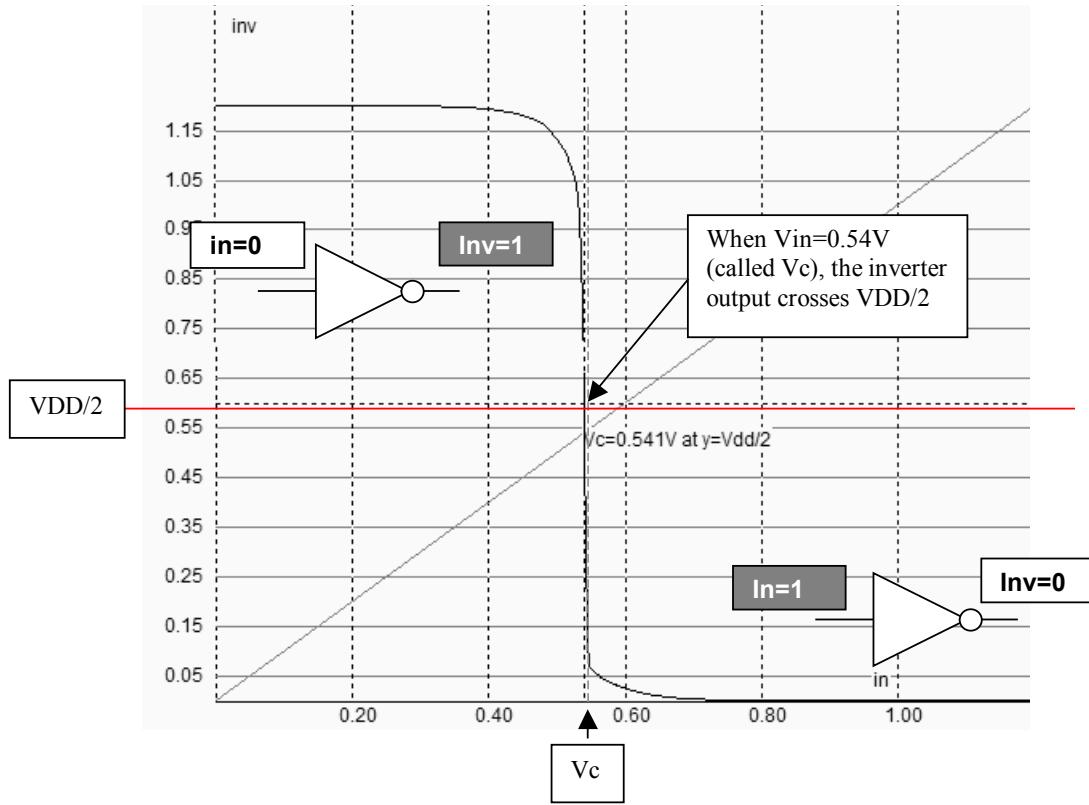


Fig. 4-46: The static characteristics of the inverter (Inv.MSK)

When  $V_{in}$  is low,  $V_{out}$  is high, which corresponds to one logic state of the inverter. When  $V_{in}$  increases,  $V_{out}$  starts to decrease slowly, and suddenly crosses the  $V_{DD}/2$  boundary. At that point, the value of  $V_{in}$  is the commutation point <Glossary> of the inverter, called  $V_c$ . Then, when  $V_{in}$  rises to  $V_{DD}$ ,  $V_{out}$  reaches 0, which corresponds to the other logic state of the inverter.

### Modify the commutation point

Several theoretical formulations of the commutation voltage versus layout parameters exist. A simple formula derived from MOS model 1 is reported in equation 4-5 [Baker]. Although based on an obsolete model, this formulation may be applied for first order hand calculations. The verification must be performed by simulation.

$$V_c = \frac{K \cdot V_{TN} + V_{DD} - V_{TP}}{1 + K} \quad (\text{Eq. 4-5})$$

with

$$K = \sqrt{\frac{\mu_n}{\mu_p} \frac{W_n}{L_n} \frac{W_p}{L_p}}$$

$\mu_n$ = mobility of electrons ( $600 \text{ V.cm}^{-2}$ )

$\mu_p$ = mobility of holes ( $270 \text{ V.cm}^{-2}$ )

$W_n$  = n-channel MOS width (in  $\mu\text{m}$ )

$L_n$  = n-channel MOS length (in  $\mu\text{m}$ )

$W_p$  = p-channel MOS width (in  $\mu\text{m}$ )

$L_p$  = p-channel MOS length (in  $\mu\text{m}$ )

$V_{DD}$  = supply voltage (1.2V)

$V_{TN}$  = threshold voltage of n-channel device (0.30V)

$V_{TP}$  = threshold voltage of p-channel device (0.30V)

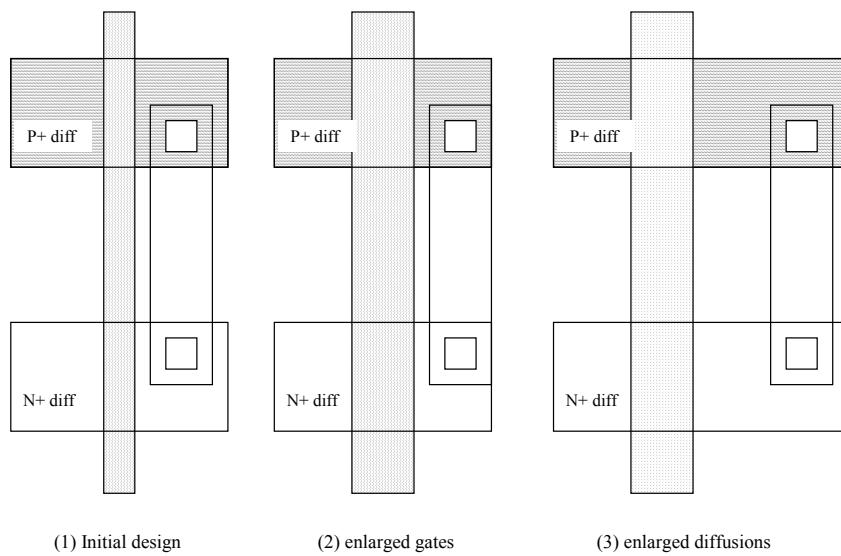


Figure 4-47: Layout modifications that do not change the commutation point

As predicted by the formulation, the sizing of the n-channel and p-channel MOS devices has a strong influence on the commutation point  $V_c$ . Enlarging both the nMOS and pMOS channels does not change the commutation, nor a supplementary diffusion area (Figure 4-47). As the ratio between the nMOS and pMOS sizes has an effect on  $V_c$ , only one device should be modified.

In figure 4-48, we have designed three inverters with almost identical characteristics. The only change is the n-channel or p-channel sizing. The inverter on the left uses the default MOS size, that is  $W_p=16$ ,  $L_p=2$  lambda,  $W_n=6$ ,  $L_n=2$  lambda. The large width for the pMOS device compensates the low mobility of holes compared to electrons, in order to achieve a balanced inverter in terms of switching performances. As a result, the static characteristics are almost symmetrical. In other words, when  $V_{in}$  is  $VDD/2$ ,  $V_{out}$  is nearly  $VDD/2$  (Curve inv\_1 in figure 4-49).

For the inverter situated in the middle of the layout, the width and length of the n-channel MOS are identical to those of the p-channel MOS. The result is a lower commutation point, as shown in curve inv\_2 of figure 4-49. Thanks to this modification, the nMOS device can drive stronger currents and moves the whole curve towards lower voltages. Now, if we enlarge the n-channel MOS channel to reduce its current, the opposite result is achieved, with a commutation point shifted to higher voltages (Curve inv\_3).

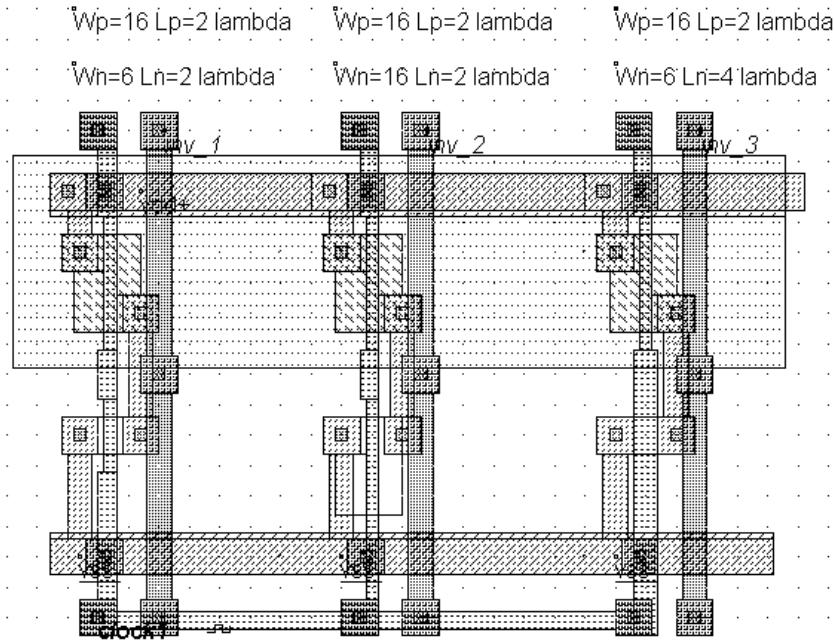


Fig. 4-48: Three different inverter sizing used to investigate its influence on the commutation point  $V_c$  (InvSizing.MSK)

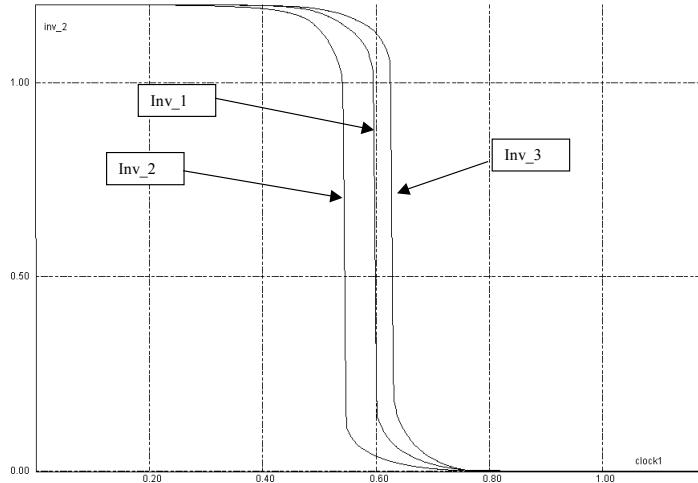


Fig. 4-49: Influence of the inverter sizing on the commutation point (InvSizing.MSK)

## Influence of the model

Using the analog simulation with various models, we may obtain significantly different estimations of the switching characteristics. In figure 4-50, we superimpose the static characteristics of the same inverter using model 3 and BSIM4. While the simulation with model 3 gives  $V_c=0.6V$ , the simulation with BSIM4 gives  $V_c=0.63V$ . This difference is not significant as far as logic behavior is concerned, but may lead to wrong performance estimation in the case of analog design.

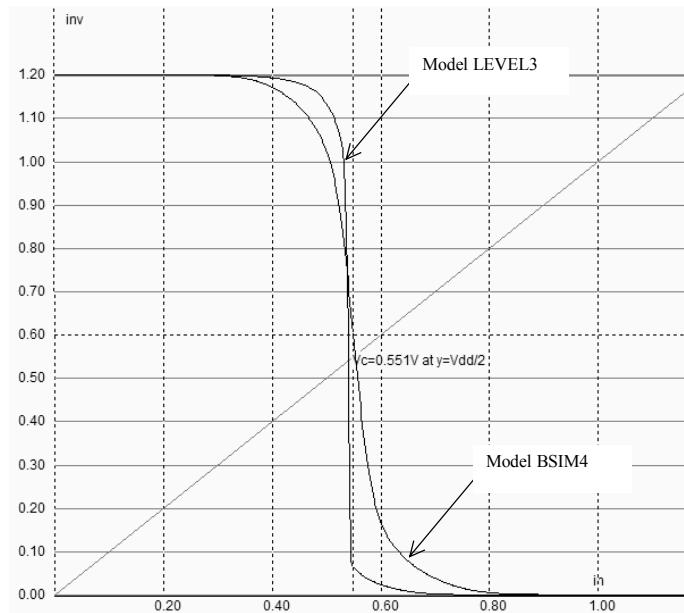


Fig. 4-50: Influence of the model on the simulation (Inv.MSK)

## 7. Random simulation

As explained in chapter 3, unavoidable process variations may occur during the integrated circuit fabrication, which may impact the static and dynamic characteristics of the inverter. In the menu **Simulate → Simulation parameters** the default set of parameters corresponds to the "typical" case. We may simulate the inverter in "minimum" or "maximum" case, as we did for the MOS device. An interesting alternative consists in using the "random" mode, also called "Monte-Carlo" analysis, where the threshold voltage and the mobility are chosen in a random way, as illustrated in figure 4-51. There is a high probability that VTO is close to the typical value, and almost no chance that VTO is higher than 0.44V or lower than 0.36V.

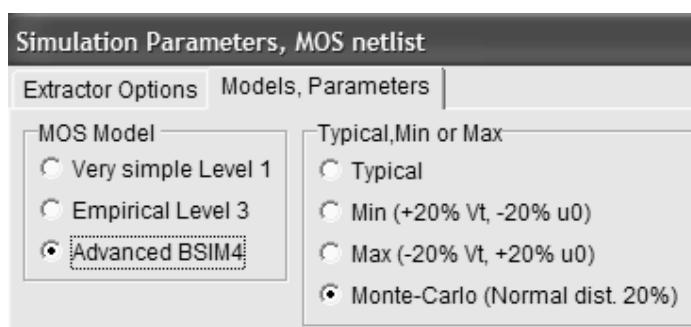


Fig. 4-51: Access to random simulation using an arbitrary set of MOS model parameters (Inv.MSK)

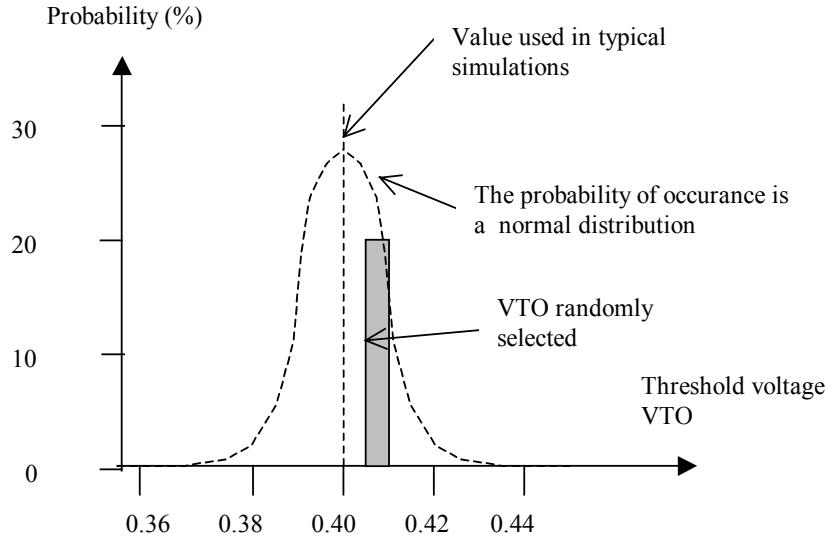


Fig. 4-52: Random selection of  $V_t$ , with a normal probability

The simulations of the transient response may be cumulated by a press of the **Reset** button. A new button **Memory** appears in the simulation window, at the right lower corner. Press this button to draw all simulations together without refreshing the grid. Each time the Reset button is activated, a new set of threshold and mobility parameters is used to conduct the simulation. The accumulation of ten successive transient simulations is represented in figure 4-53.

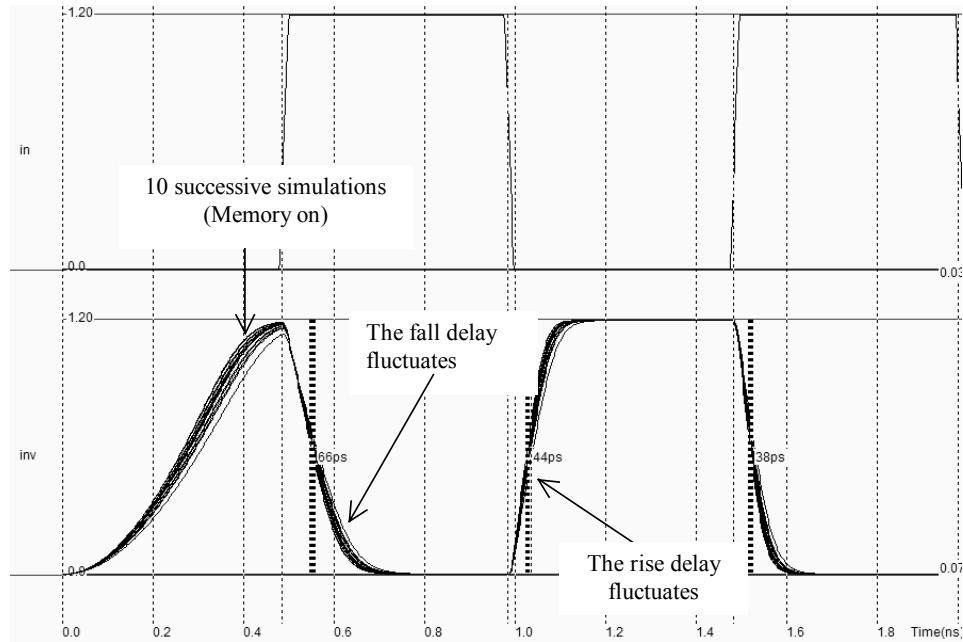


Fig. 4-53: The Monte-Carlo simulation of the inverter transient characteristics, using random VTO and UO parameters (Inv.MSK).

The inverter has a strong probability to behave close to the typical value. In some rare cases, the switching performances vary significantly. The min/max simulation is also very interesting to simulate the inverter in extreme situations.

## 8. The Inverter as a library cell

Generally speaking, the integrated circuit design relies on a library of basic cells. In this library, each basic cell is described in a very detailed way. The layout information and several static and dynamic aspects are usually included. Such details are important to choose the appropriate cell, to evaluate the circuit size, standby parasitic current and switching performances.

The data-sheet of the inverter usually looks like figure 4-54. Firstly, the header gives the cell name. The mask level file and symbol files are also listed. The truth table recalls the logic behavior of the cell. The symbol is also provided. In the case of complex cells such as latches, where numerous versions and options co-exist beyond the same name, the truth-table is of key importance. The node capacitance is useful for propagation delay prediction, as the switching performance is linked with the capacitance load.

The operating point recalls the value of the supply with which the characterization has been conducted. Some information is also given for low supply voltage (See also the switching characteristics at 0.9V).

Name: INVERTER

Technology: 0.12 $\mu$ m CMOS

Layout: INV.MSK

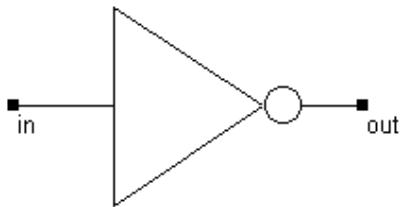
Symbol: NOT.SYM

Operating point: VDD=1.2V, Temperature=25°C

TruthTable:

In	Out
0	1
1	0

Symbol:



Capacitance:

- Input: 0.5fF
- Output 0.5fF

Drive: 1x

Cell Area : 1.26 $\mu$ m $\times$ 4.3 $\mu$ m (5.14 $\mu$ m $^2$ )Power consumption : 1.02 $\mu$ W/MHz typical

Standby current: 100pA

Inverter	Rise time (ps)				Fall time (ps)			
	0.01ns (fast)		0.1ns (slow)		0.01ns (fast)		0.1ns (slow)	
Input slope								
Load (fF)	10fF	100fF	10fF	100fF	10fF	100fF	10fF	100fF
Delay In→Out	42	340	61	416	35	288	49	338
Delay In→Out (VDD=0.9V)			86				65	
Delay In→Out (max, -40°)			70				50	
Delay In→Out (min, 120°C)			122				98	

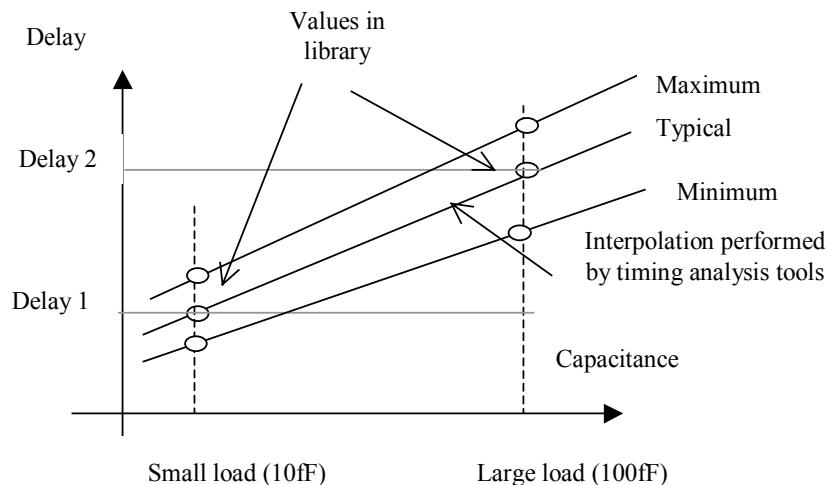
Inverter	Peak current ( $\mu$ A)			
Input slope	0.01ns (fast)		0.1ns (slow)	
Load (fF)	10fF	100fF	10fF	100fF
Peak current (typ)			138	
Peak current (max, -40°)			189	
Peak current (min, 120°C)			105	
Peak current (typ, VDD=0.9V)			79	

Fig. 4-54: The library information for the basic inverter (Inv.MSK).

The power consumption is usually described in  $\mu\text{W}/\text{MHz}$ . To characterize this value, a 1MHz clock is connected to the input and the total power consumption is computed from the integral of the current. In Microwind, we preferably use a 1GHz clock, and consequently divide the power estimation by 1000. The cell consumption increase linearly with frequency. The standby current is a key information in low-power circuits, where the standby parasitic current should be as small as possible. Depending on the MOS option (normal, high-speed, low leakage), the standby current may vary in a very significant way.

The keyword "1x" refers to the inverter strength. A cell with 1x strength is designed with small output MOS devices, usually close to the minimum length and width. A cell with 2x has medium size MOS devices, a cell with drive 4x is used for high speed signals. Cells with 8x drive or even 16x drive may exist, to propagate very fast signals such as clocks and bus. However, using cells with high drive means a high power consumption and high risks of signal integrity problems.

The delay between signals *in* and *out* is strongly dependant on the slope of the input signal, the capacitance connected to the output signal, the temperature, the power supply and the process variations. The goal of the switching delay table is to summarize the delay in typical conditions as well as in extreme conditions. Several design tools such as the timing analyzer and the power consumption extractor will use these data to guess all possible cases of loading conditions, temperature variation, etc.

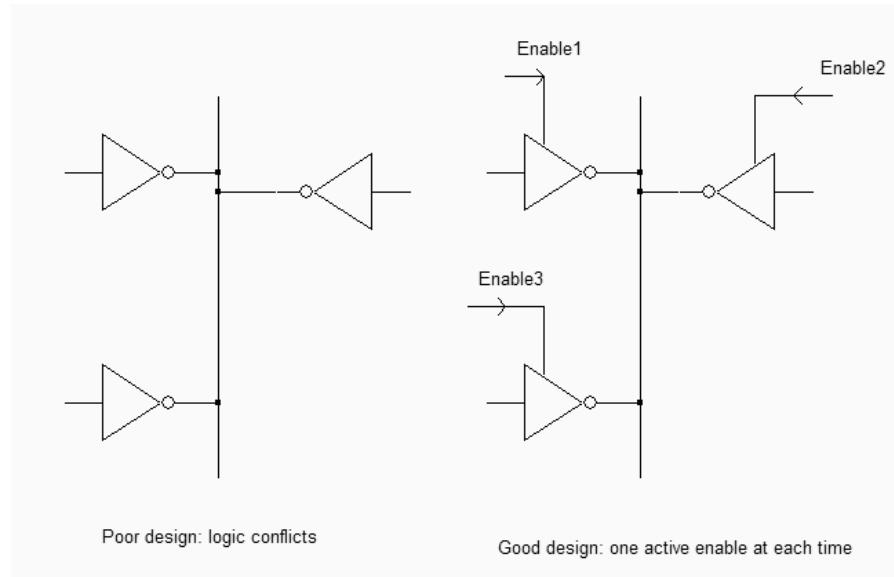


*Fig. 4-55: Delay parameters are used by timing analysis tools to predict the cell switching performances for any capacitance*

The current peak details are used to evaluate the power consumption of the circuit. The value of the current changes significantly depending on the loading conditions, temperature and supply voltage, as expected.

## 9. 3-State Inverter

Until now all the symbols produced the value logic ‘0’ and logic ‘1’. However, if several inverters share the same node, such as bus structures (Figure 4-56), conflicts will rise. In order to avoid multiple access at the same time, specific circuits called 3-state inverters are used, featuring the possibility to remain in a ‘high impedance’ state when access is not required.



*Figure 4-56: If multiple access is required on a single node, 3-state inverters are used for interfacing (Inv3state.SCH)*

The 3-state inverter symbol consists of the logic inverter and an enable control circuit. The output remains in ‘high impedance’ (Logic symbol ‘X’) as long as the enable *En* is set to level ‘0’. The truth table is reported below.

NOTIF1		
In	En	Out
0	0	X
0	1	1
1	0	X
1	1	0
x	0 or 1	X
0 or 1	X	X

The internal structure of the 3-state inverter is shown in figure 4-57. The basic CMOS inverter is no more connected to the supply lines VDD and VSS directly. In contrary, pass nMOS and pMOS devices are inserted to disconnect the inverter when the cell is disabled.

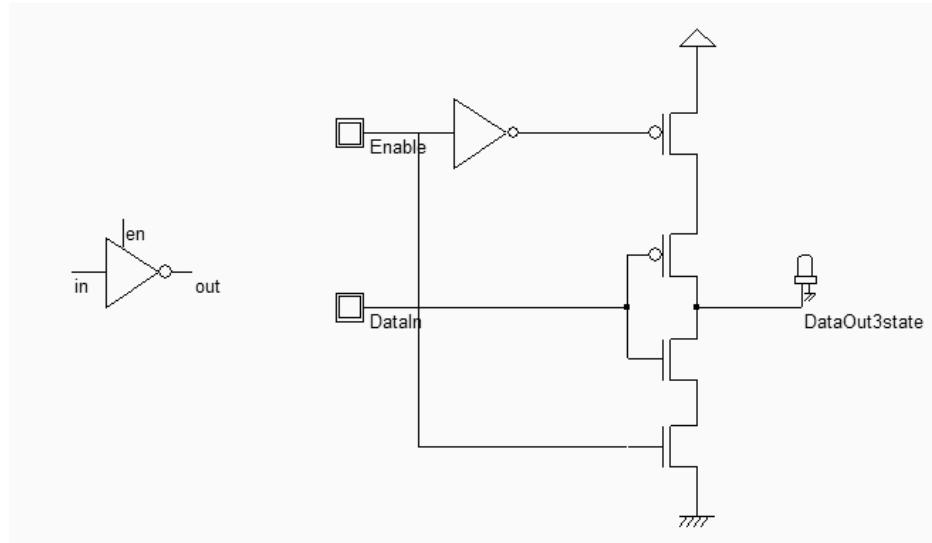


Figure 4-57: Schematic diagram and logic symbol for the 3-state inverter (CmosInv3State.SCH)

Unfortunately, a supplementary inverter is needed to generate the /enable signal required to control the pMOS device. The logic simulation reported in figure 4-58 illustrates two basic situations: one where *enable* is inactive, and the output is in high-impedance state <glossary> as no path exists to VDD or VSS, the other where the circuit is equivalent to an inverter, as the upper and lower pass transistors are enabled.

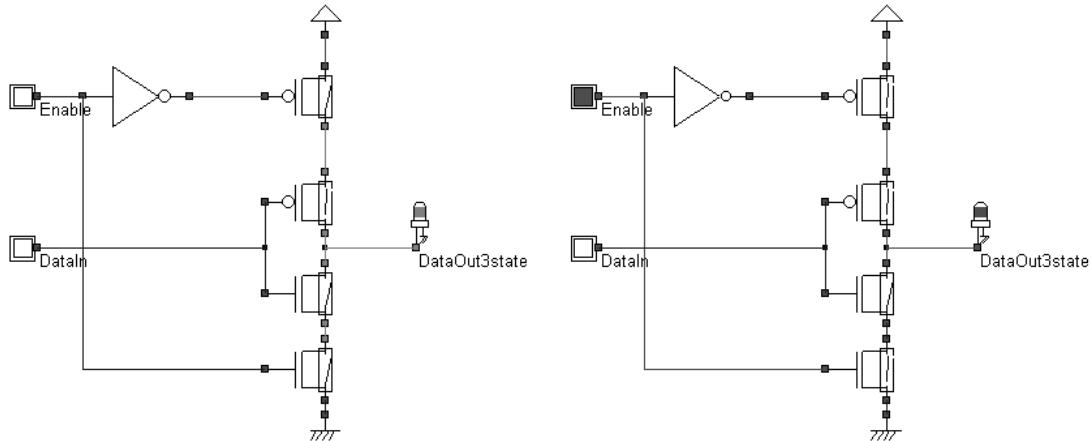


Figure 4-58: Simulation of the 3-state inverter (CmosInv3State.SCH)

Two versions of layout are proposed in figure 4-59, and they correspond to the same design. The cell situated on the left is the direct implementation of the schematic diagram of the 3-state inverter. The layout implementation is not optimal as we loose some silicon area due to severe diffusion design rules which require a 4 lambda spacing. The new arrangement, shown on the right of the figure, is significantly more compact, thanks to horizontal flip of the *Enable* inverter, and the sharing of the ground and supply contacts, as illustrated in figure 4-60. Continuous diffusions always lead to more compact and faster designs.

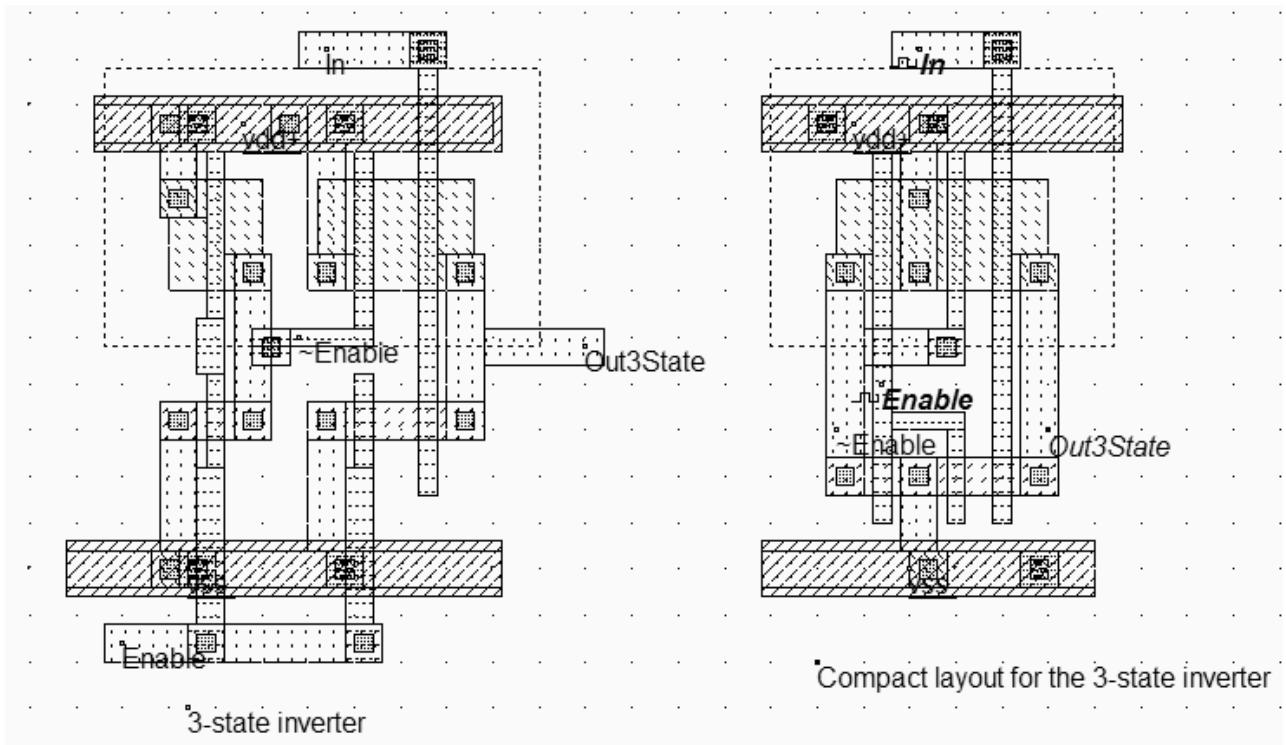


Figure 4-59: The layout of the 3-state inverter (Inv3State.MSK)

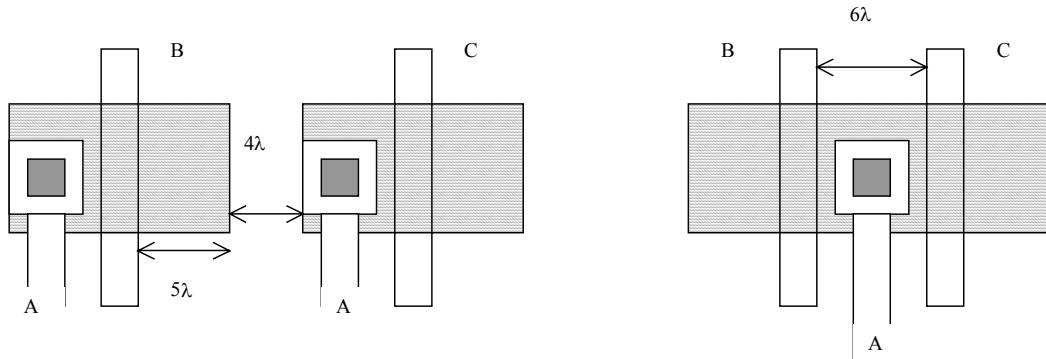


Figure 4-60: A permutation technique to achieve more compact layout

The analog simulation reported in figure 4-61 gives an interesting view of the high impedance state. From the chronograms, we see that when *Enable*=1 the cell acts as a regular CMOS inverter, while when *Enable*=0 the output "floats" in an unpredictable voltage value, which tends to fluctuate at the switching of the input, mainly due to parasitic leakage and couplings.

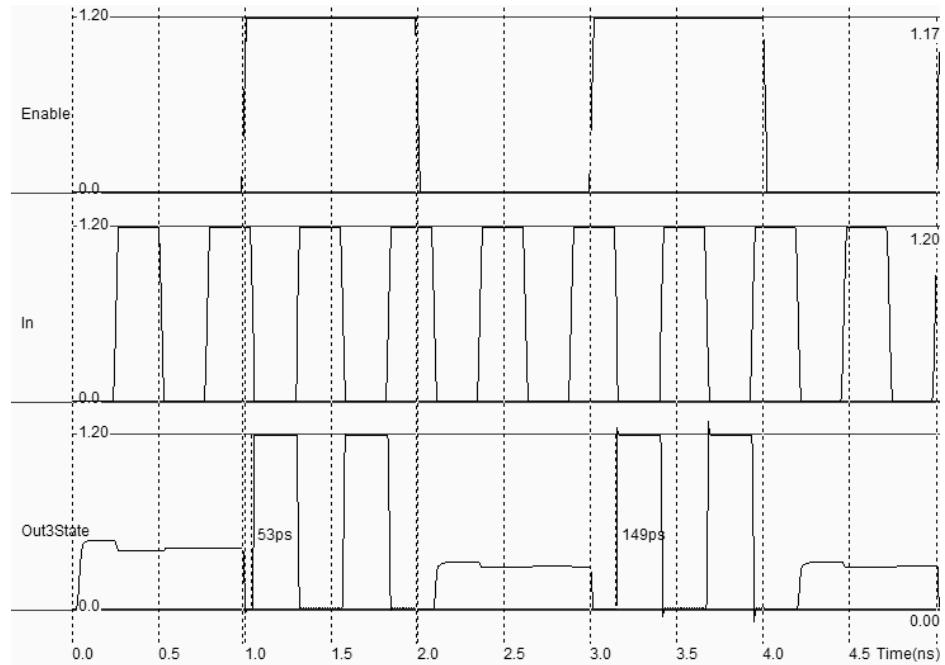


Figure 4-61: Analog simulation of the 3-state inverter (INV3STATE.MSK)

## 10. All nMOS Inverters

Several other circuits exist to build the logic inverter function [Baker]. One popular inverter is shown in figure 4-62. It consists of a normal n-channel MOS device  $N1$  and of another n-channel MOS device  $N2$  connected as a simple load. Due to the permanent connection of the gate to VDD, the nMOS device  $N2$  is equivalent to a resistance  $R_{on\_N2}$ . When  $Clock=0$ , a path exists to rise the output through the resistance. The final value  $V_{high}$  is  $VDD-V_{tn}$ , where  $V_{tn}$  is the threshold of  $N2$ . When  $Clock=1$ , the circuit is equivalent to a voltage divider where  $N2$  still conducts, and  $N1$  creates a path to ground. An approximation of  $V_{low}$  is given by equation 4-7.

$$V_{low} = VDD \frac{R_{on\_N2}}{R_{on\_N1} + R_{on\_N2}} \quad (\text{Eqn. 4-7})$$

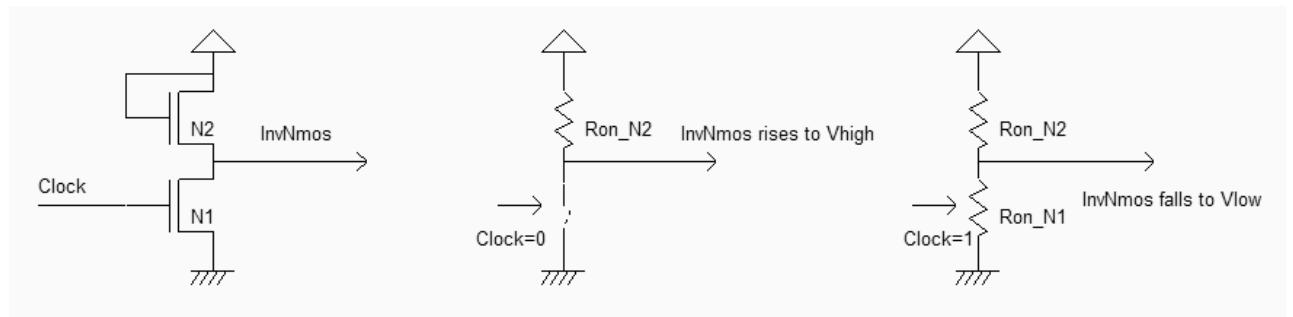


Figure 4-62: An inverter only made with nMOS devices (InvNmos.SCH)

The simulation waveforms given in figure 4-63 are quite unusual as the low state (*Clock*=1) of the output leads to a stand-by current which had not appeared until now in CMOS circuits. This DC power waste is a major drawback for this kind of design. Furthermore, the logic level 0 corresponds to 0.3V while the logic level 1 corresponds to 0.8V. The switching is slow, specifically from 0 to 1, due to a weak nMOS device. However, no pMOS device is required, which simplifies both the design and the process. The device N2 is sized with a large length and a small width to increase the resistance  $R_{on\_N2}$ , to lower the output voltage when *clock*=1. All n-MOS inverters were used before CMOS technology was made available.

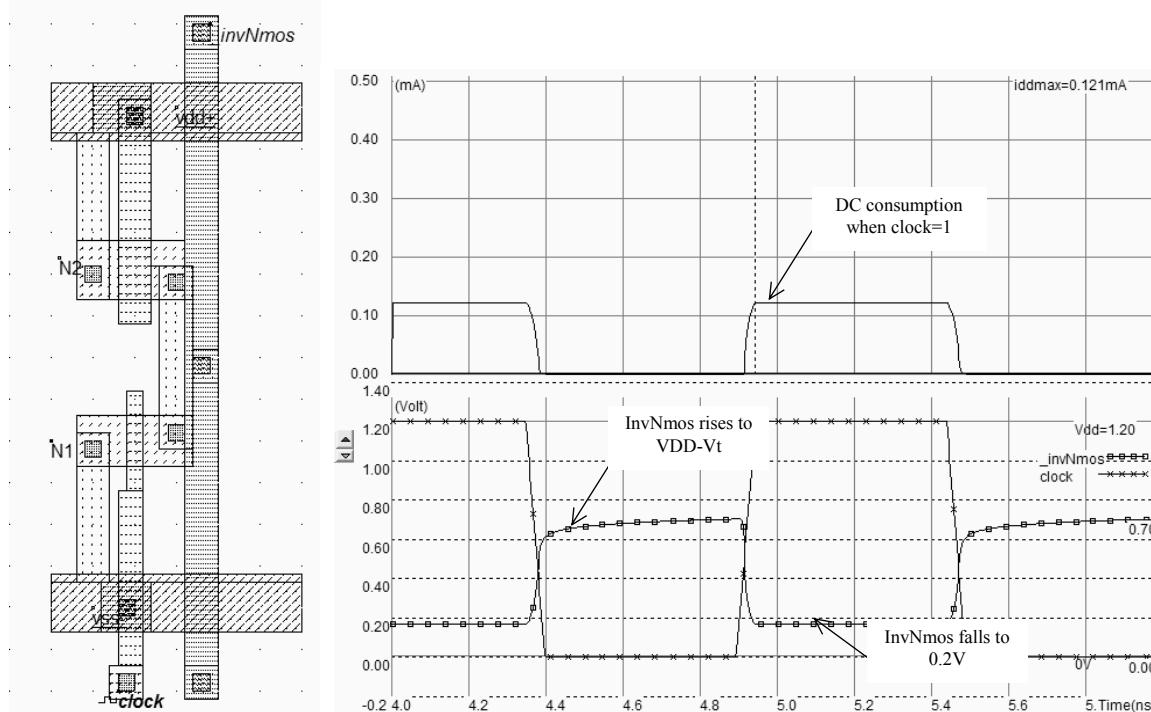
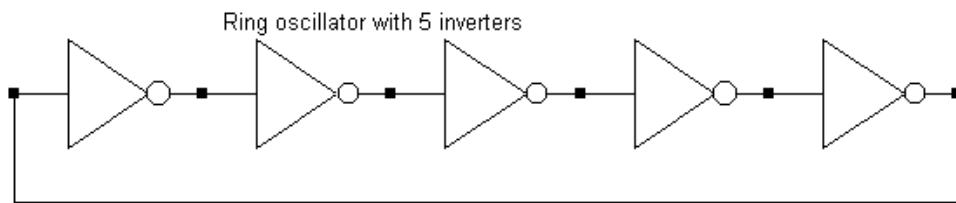


Figure 4-63: All n-MOS inverter(INVNMOS.MSK)

## 11. Ring Oscillator

The ring oscillator made from 5 inverters has the property of oscillating naturally. We observe in the circuit of figure 4-64 the oscillating outputs and measure their corresponding frequency.



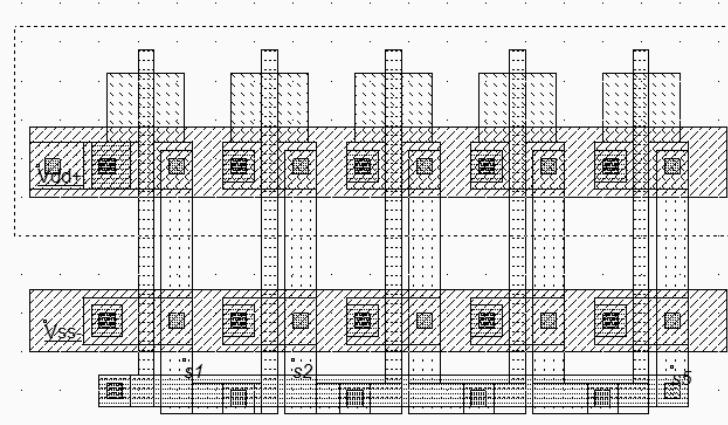


Figure 4-64: Schematic diagram and layout of the ring oscillator used for simulation (INV5.MSK)

The ring oscillator circuit can be simulated easily at layout level with Microwind using various technologies. The time-domain waveform of the output is reported in figure 4-65 for 0.8, 0.12 $\mu$ m and 70nm technologies. Although the supply voltage (VDD) has been reduced (VDD is 5V in 0.8 $\mu$ m, 1.2V in 0.12 $\mu$ m, and 0.7V in 70nm), the gain in frequency improvement is significant.

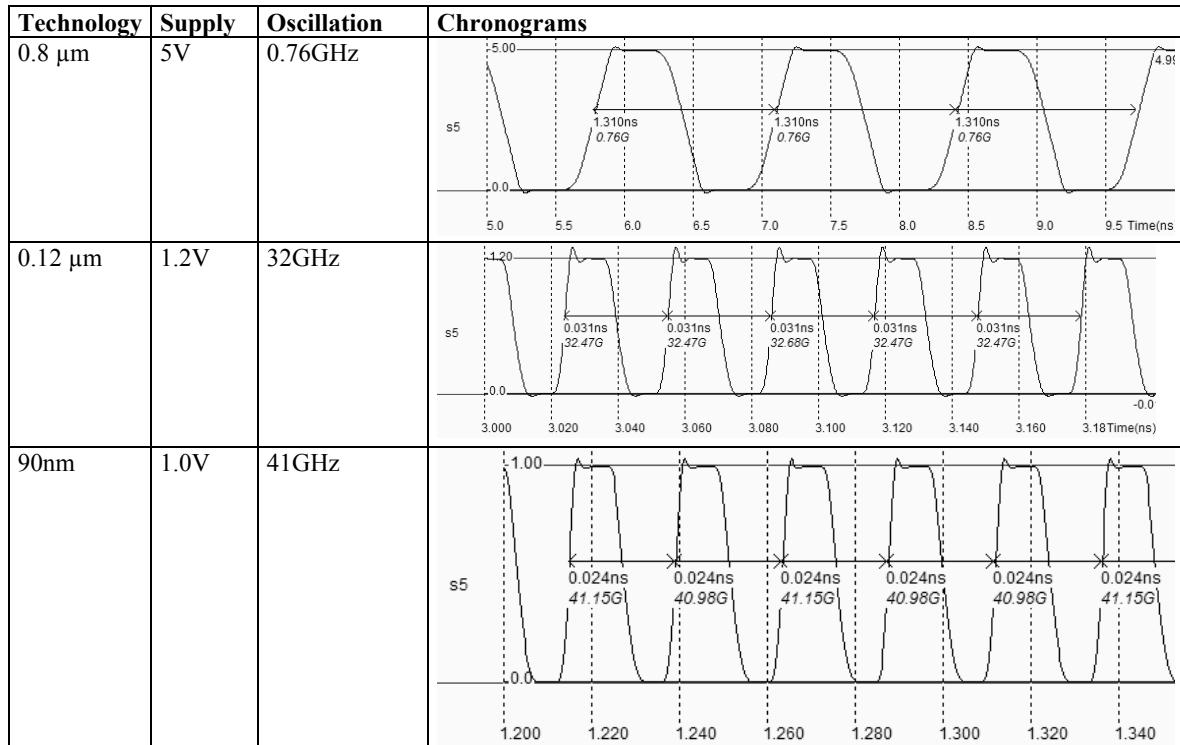


Fig. 4-65: Oscillation frequency improvement with the technology scale down (Inv5.MSK)

By default the software is configured with 0.12 $\mu$ m technology. Use the command **File → Select Foundry** to change the configuring technology. For example, select **cmos08.RUL** which corresponds to the CMOS 0.8 $\mu$ m technology, or the file **cmos90nm.RUL** which configures Microwind to the CMOS 90nm technology. When you run again the simulation, you may observe the change of VDD and the significant change in oscillating frequency.

## High Speed vs. Low leakage

Let us consider the ring oscillator with an enable circuit, where one inverter has been replaced by a NAND gate to enable or disable oscillation (Inv5Enable.MSK). The schematic diagram is shown in figure 4-66, as well as its layout implementation. We analyze the switching performances in high speed and low leakage mode, by changing the properties of the option layer which surrounds all devices.

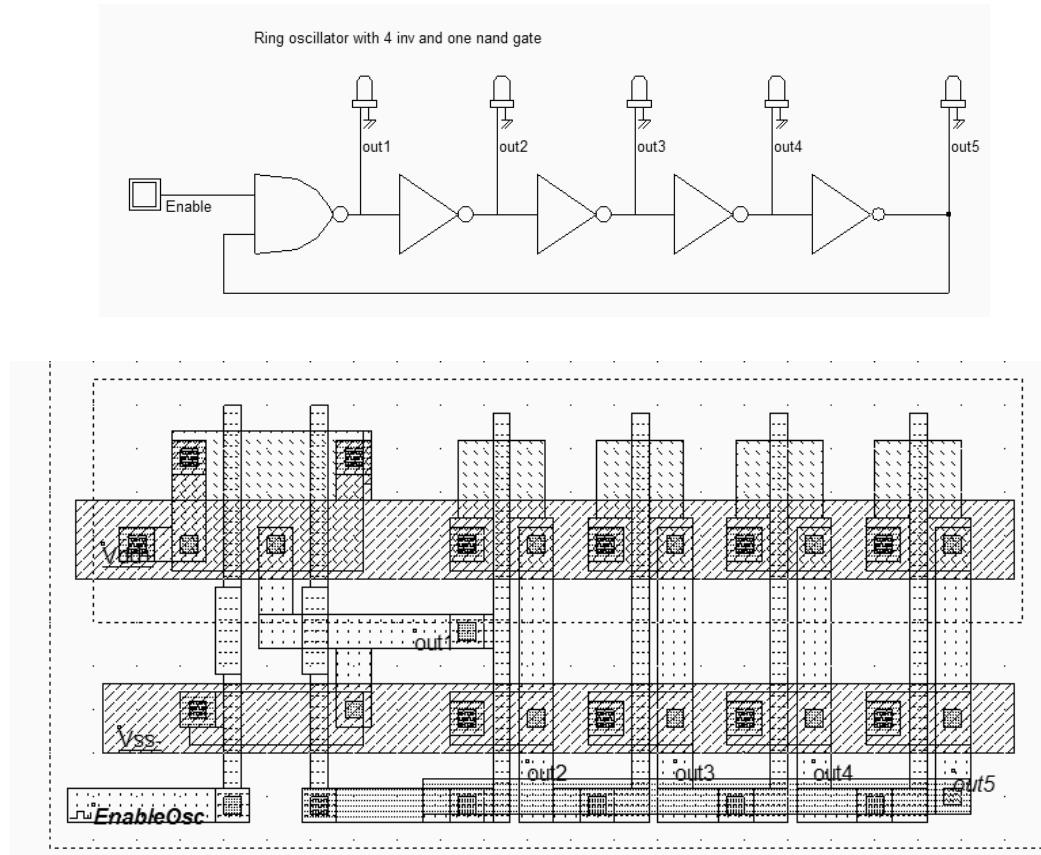


Fig. 4-66 The schematic diagram and layout of the ring oscillator used to compare the analog performances in high speed and low leakage mode (INV5Enable.MSK)

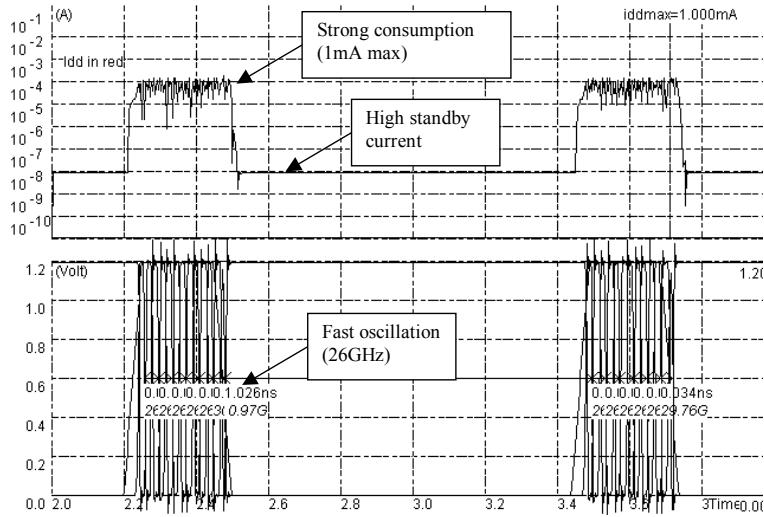


Fig. 4-67: Simulation of the ring oscillator in high speed mode, using BSIM4 model. The oscillating frequency is fast but the standby current is high (Inv5Enable.MSK)

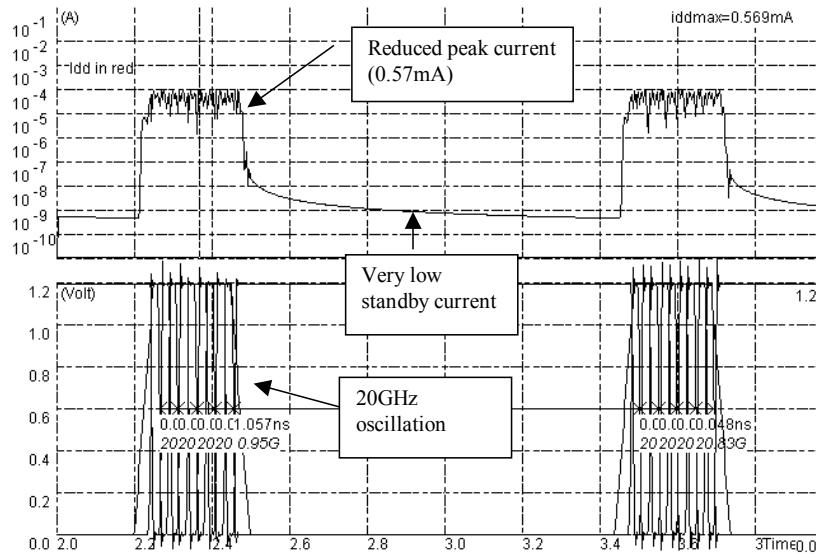


Fig. 4-68: Simulation of the ring oscillator in low voltage mode, using BSIM4 model. The oscillating frequency is slower but the standby current is very low (Inv5Enable.MSK)

Parameter	Low leakage mode	High Speed mode
Imax	0.6 mA	1.0mA
I standby	<1nA	>10nA
Oscillating frequency	20GHz	26GHz

Table 4-4: Comparative performances of the ring oscillator (Inv5Enable.MSK)

The option layer which surrounds the oscillator is set to high speed mode by a double click inside that box. In high speed mode, the circuit works fast (26GHz) but consumes a lot of power (1mA) when on, and a significant standby

current when off ( $10\text{nA}$ ), as shown in the simulation of the voltage and current given figure 4-66. Notice the tick in front of "Scale I in log" to display the current in logarithmic scale.

In contrast, the low leakage MOS features slower oscillation ( 20GHz in figure 4-67, that is approximately a 25% speed reduction) , but with 40% less current when ON, and more than one decade less standby current when off ( $1\text{nA}$ ). In summary, low leakage MOS devices should be used whenever possible. High speed MOS should be used only when speed is critical, such as communication bus, critical path, etc.. The analog performances are summarized in table 4-4.

## Temperature effects

The main consequence of temperature increase is a decreasing mobility of the electrons and holes of the MOS channel, leading to slower transient performances. Thus, the propagation delay due to the logic gate is increased, as illustrated in figure 4-68 which concerns the switching characteristics of the 5-inverter ring oscillator. In Microwind2, you can get access to temperature using the command **Simulate → Simulation Parameters**. The temperature is given in  $^{\circ}\text{C}$ .

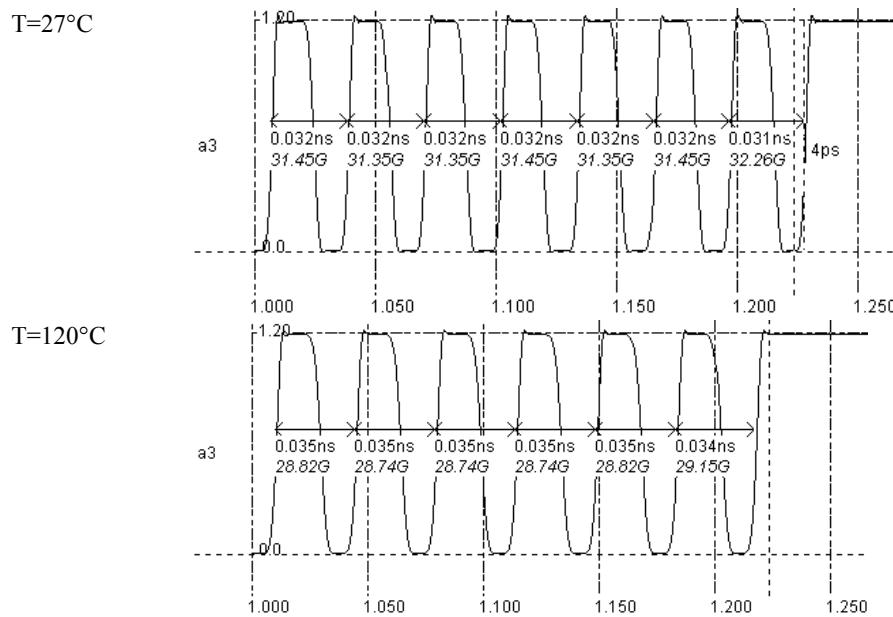


Fig. 4-69. Propagation delay increases with temperature (Inv5Enable.MSK)

In the simulation of figure 4-69, we used the BSIM4 model for a temperature set to  $25^{\circ}\text{C}$  and  $120^{\circ}\text{C}$ . We can observe a 10% decrease of switching speed, which finds its origin in the mobility degradation, which is computed by the following formulation.

$$U_0 = U_{0(T=27)} \left( \frac{T + 273}{300} \right)^{-1.8} \quad (\text{eq. 4-4})$$

We may conduct the parametric analysis of the temperature influence on the oscillating frequency, in order to obtain the results reported in figure 4-70. It may be seen that the frequency variation from  $-100^{\circ}\text{C}$  to  $+100^{\circ}\text{C}$  is kept below

15%. The reason for this reduced dependence is that the mobility reduction is compensated by the threshold voltage decrease, also strongly dependent on the temperature, which tends to limit the overall effects of temperature variations.

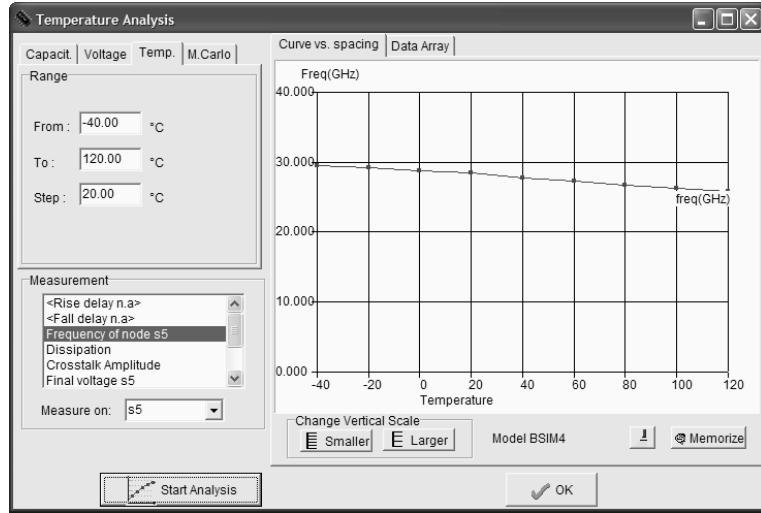


Fig. 4-70: Performances of the ring oscillator versus temperature increase (Inv5Enable.MSK)

## A 2.5GHz ring oscillator

The previous ring oscillator operated around 30GHz, which is of no practical use. In contrast, the 2.5GHz frequency is widely used for a variety of wireless network applications. In this paragraph, we investigate several possibilities to slow the oscillator frequency down to 2.5GHz. One immediate idea consists in designing a ring oscillator with more inverter stages (Around 70). This is a power consuming and silicon area consuming approach. A more attractive solution consists in the reduction of the MOS current capabilities. Then a new question rises: how should we proceed, as we may increase the channel length, decrease the channel width, or add parasitic capacitance in each switching node (Figure 4-70)?

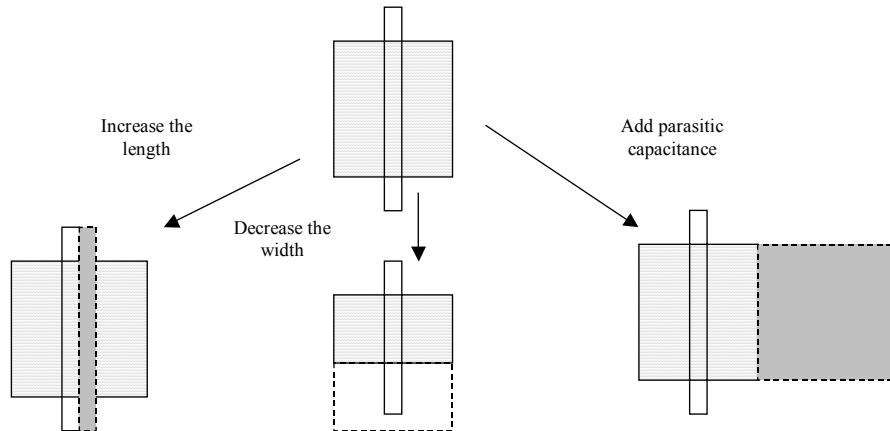


Figure 4-71: Reducing the current of the MOS device may be performed by increasing the length or decreasing the width

One solution is proposed in figure 4-72. It combines the channel length increase, the width decrease to its minimum value, and the enlarging of drain areas whenever possible to increase the parasitic junction surface, and consequently its parasitic capacitance. All these layout modification have a sufficient impact to reduce the oscillating frequency to around 2.5GHz. Notice that this frequency is very sensitive to process parameters, temperature, and supply voltage variation.

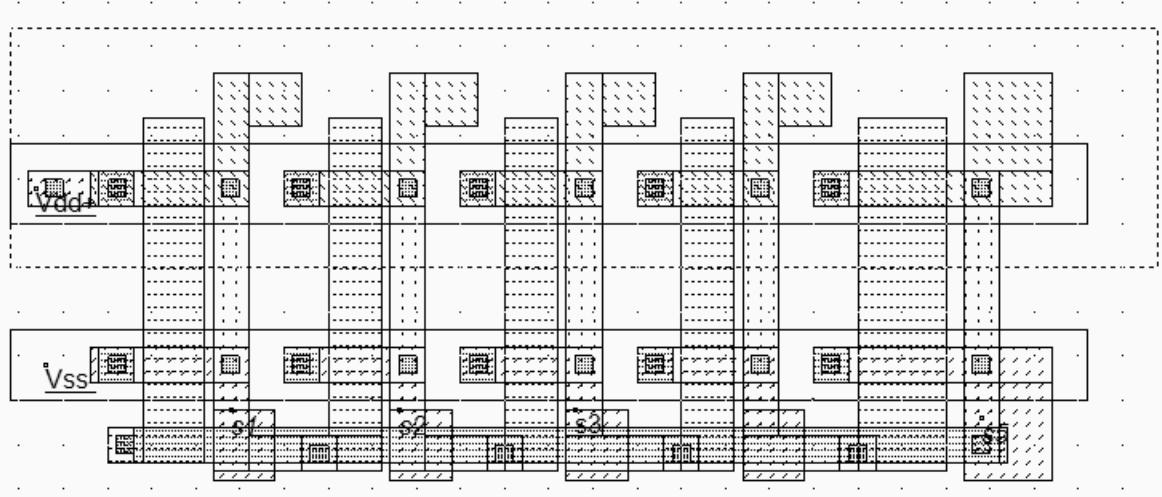


Figure 4-72: The 5-inverter oscillator tuned to 2.5GHz (Inv2,5GHz.MSK)

## 12. Latch-up effect

The latch-up effect is a parasitic shortcut between VDD and VSS that can lead to the destruction of the integrated circuits. The origin of latch-up is the activation of a parasitic N/P/N/P device (Also called thyristor) appearing in the vertical cross-section of the nMOS and pMOS structures as reported in figure 4-73.

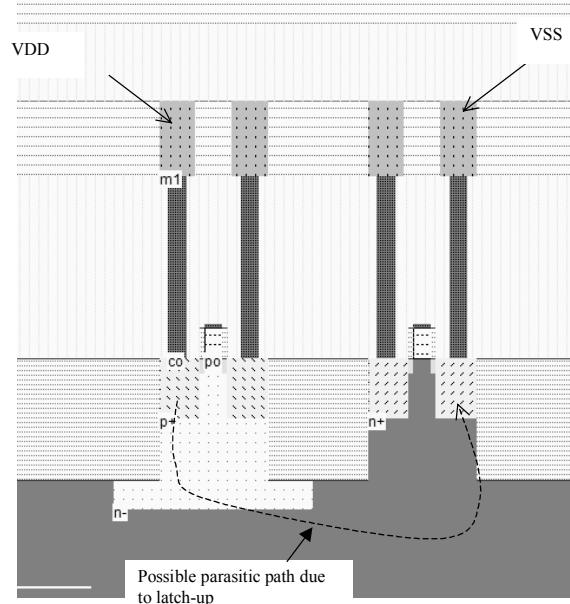


Fig. 4-73. Origin of latch-up

## Limiting the latch-up effect

The latch-up effect is almost eliminated if the substrate is locally polarized to ground, and the n-well is locally polarized to VDD (Figure 4-74). In the upper layout (Figure 4-74a), the situation is extremely dangerous as the n-well region is floating. If the n-well potential drops around  $VDD/2$  and the local substrate voltage rises to  $VDD/2$ , the latch-up phenomenon is initiated. Most layout tools alert the designer in the case of floating n-well regions. The good approach consists in inserting a polarization diode N+/N-well and stuck it to the highest possible potential, typically VDD.

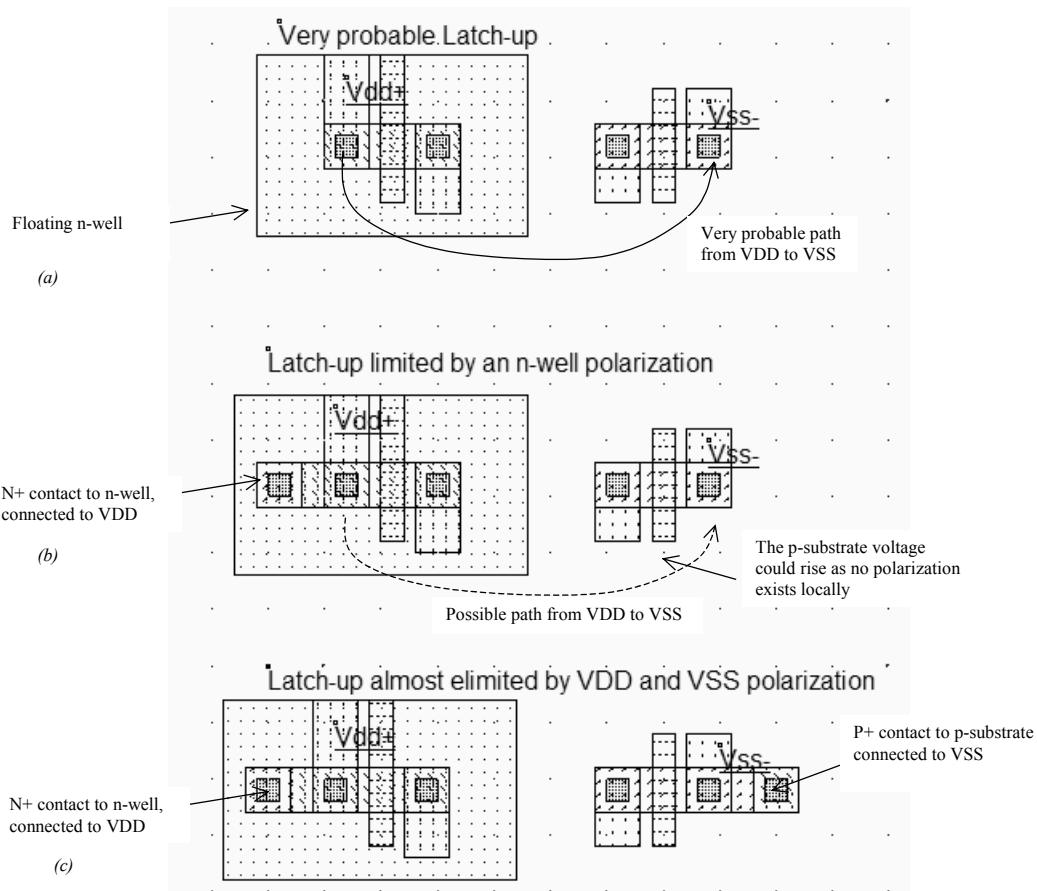


Fig. 4-74 Limiting the latch-up effect by polarization diodes

Many designers consider that there exists an « automatic » polarization of the substrate to ground and forget to add a local P+/P-substrate contact to ground, near the nMOS device (Figure 4-74b). This might be a dangerous assumption which can cause latch-up: in  $0.12\mu m$  technologies, several manufacturers use a highly resistive p-doped substrate. In that case, the electrical link between the physical ground (Back of the IC) and the local nMOS area is equivalent to a resistor of several Kohm. Consequently, what is supposed to be a good 0V reference is a very weak 0V, that can easily fluctuate and turn on the N/P/N/P device, which may lead to latch-up and possible destruction. This is why it is highly recommended to add also a P+/P-substrate polarization to ground, which protects the logic cell from latch-up (Figure 4-74c).

## 13. CONCLUSION

This chapter has described the CMOS inverter, from a logic and analog point of view. The mobility difference between electrons and holes has been counterbalanced at layout level to obtain symmetrical static and dynamic characteristics. The effect of MOS model and temperature on the simulation results have also been investigated. The 3-state inverter, all n-MOS inverter and ring-oscillator circuits have been designed and simulated. Finally, we have presented the basic polarization techniques to avoid the parasitic latchup effect.

## REFERENCES

- [Weste] N. Weste, K. Eshraghian "Principles of CMOS VLSI design", Addison Wesley, ISBN 0-201-53376-6, 1993
- [Baker] R.J. Baker, H. W. Li, D.E. Boyce "CMOS circuit design, layout and simulation", IEEE Press, ISBN 0-7803-3416-7, 1998

## EXERCISES

4.1 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.

*Answer: The most symmetrical behavior is obtained for  $W_p \sim 2.W_n$*

4.2 We consider the two inverters of figure 4-75. We define  $t_{PLH1}$  as the delay from a low to high value of the output *inv1* (Figure 4-75). We define  $t_{PLH2}$  as the delay from a low to high value of the output *inv2*. Find the relation between the propagation delays  $t_{PLH1}$  and  $t_{PLH2}$ .

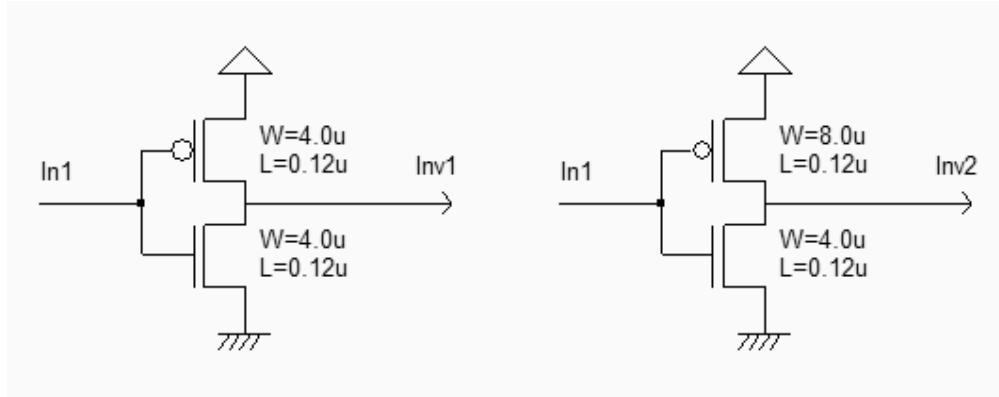


Figure 4-75: Compared performances of two inverters with different sizing (ch42.sch)

*Answer:  $t_{PLH2} = 2/3 * t_{PLH1}$*

4.3 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  $RON$  and  $ROFF$  of each transistor and calculate  $t_{PHL}$  and  $t_{PLH}$ . Compare their value to the simulated results.

*Answer:*  $RON=$ ,  $ROFF=$ ,  $t_{PHL}=$ ,  $t_{PLH}=$  <Sonia>

4.4 Configure Microwind in 90nm and design a ring oscillator using an odd number of inverters (For example 11).

Based on the analog simulation using BSIM4, find the inverter propagation delay ( $t_p$ ) and the oscillator frequency ( $f_{osc}$ ). Extract  $t_{PHL}$  and  $t_{PLH}$  and deduce the theoretical oscillator frequency  $f_{osc\_theory}$ . Compare it to the simulation ( $f_{osc}$ ). Playing with the available MOS options in 90nm technology, analyze the variation of  $f_{osc}$ .

*Answer:*  $tp=$ ,  $fosc=$ ,  $fosc\_theory=$ ,  $fosc\_lowleakage=$ ,  $fosc\_highspeed$ ,  $fosc\_ultrahigh=$  <Sonia>

4.5 Using Microwind, design 2 inverters: INV1 ( $L_{\min}$ ,  $WN_{\min}$ ,  $WP_{\min}$ ), INV2 ( $L_{\min}$ ,  $4 \times WN_{\min}$ ,  $4 \times WP_{\min}$ ). What is the input capacitance of INV1? Simulate the 4 following configurations using the same input clock (clock switching:  $tr=tf=10\text{ps}$ ) and extract the switching delays. Would the switching delay be different with a larger  $tr$  and  $tf$ ?

- INV1 alone
- INV1 connected to INV1
- INV1 connected to INV2
- INV1 connected to a 10fF load

*Answer:*  $Cin\_inv1=$ ,  $td1=$ ,  $td2=$ ,  $td3=$ ,  $td4=$ , when  $tr>xxxps$ , the delay starts to increase with  $tr$  <Sonia>

4.6 Using Microwind in 0.12 $\mu\text{m}$ , draw an inverter ( $L=0.12\mu\text{m}$ ,  $W_{\text{PMOS}}=0.6\mu\text{m}$ ,  $W_{\text{NMOS}}=0.3\mu\text{m}$ ).

- Simulate the PMOS characteristics to find its  $I_{ON}$  current.
- Use the time domain simulation in mode **voltage and current** to compare  $I_{CCmax}$  to  $I_{ON}$ . Extract the power dissipation.
- Add a capacitor on the inverter output. For the two following values (1fF, 100fF) compare  $I_{CCmax}$  and the power dissipation.
- Design a new inverter ( $L=0.12\mu\text{m}$ ,  $W_{\text{PMOS}}=6\mu\text{m}$ ,  $W_{\text{NMOS}}=3\mu\text{m}$ ) with a 100fF load. What is the minimum number of diffusion contacts necessary to avoid current overstress in the inverter?

*Answer:*  $Ion=$ ,  $Iccmax=$ ,  $pow=$ ,  $iccmaxIf=$ ,  $iccmax100f=$ ,  $contacts=<\text{Sonia}>$

4.7 Analyze the variation of frequency versus the technological parameter variation, for the 5-inverter ring oscillator (Inv5.MSK). Use the command **Analysis → Parametric analysis** for this study.

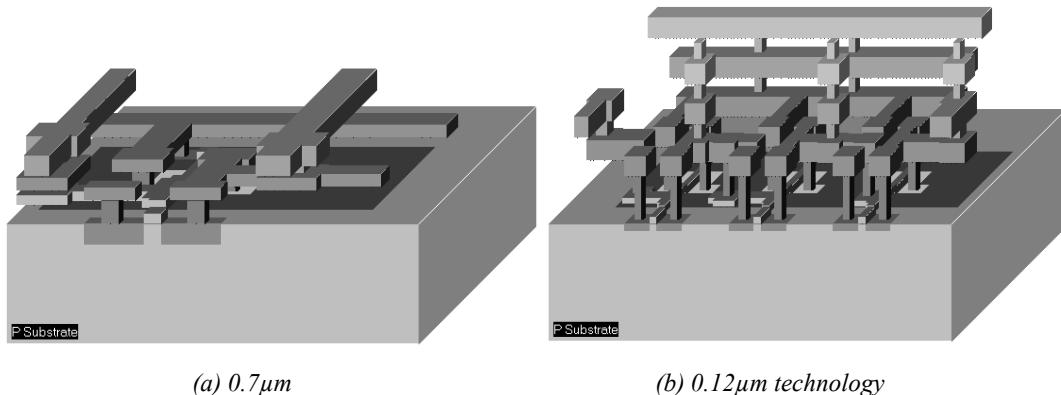
*Answer:* The variation is around

# 5

## Interconnects

### 1. Introduction

The role of interconnects in integrated circuit performances has considerably increased with the technology scale down. Figure 5-1 shows the evolution of the aspect of the integrated circuit. In  $0.12\mu\text{m}$ , 6 to 8 metal layers are available.



*Figure 5-1: Evolution of interconnect between  $0.7\mu\text{m}$  technology and  $0.12\mu\text{m}$  technology (Inv3.MSK)*

### 2. Metal Layers

In the previous chapter, we designed the CMOS inverter using two layers of metal. However, 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 5-2. 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.

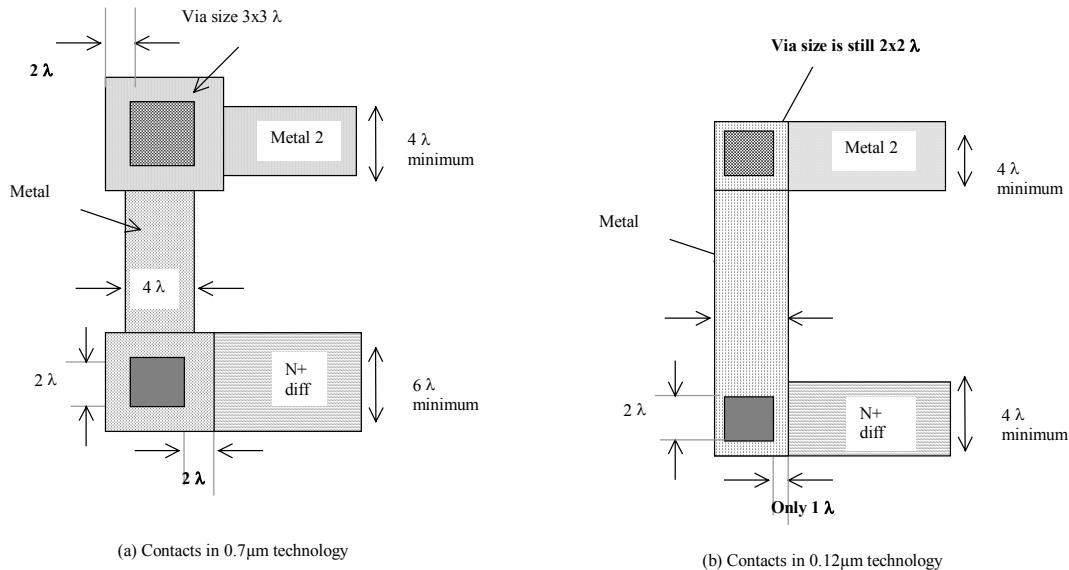


Figure 5-2: Contacts in 0.7 $\mu\text{m}$  technology require more area than in 0.12 $\mu\text{m}$  technology

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 5-3a) as contacts must be drawn in a separate location. In deep-submicron technology (Starting 0.35 $\mu\text{m}$  and below), stacked contacts are allowed (Figure 5-3b).

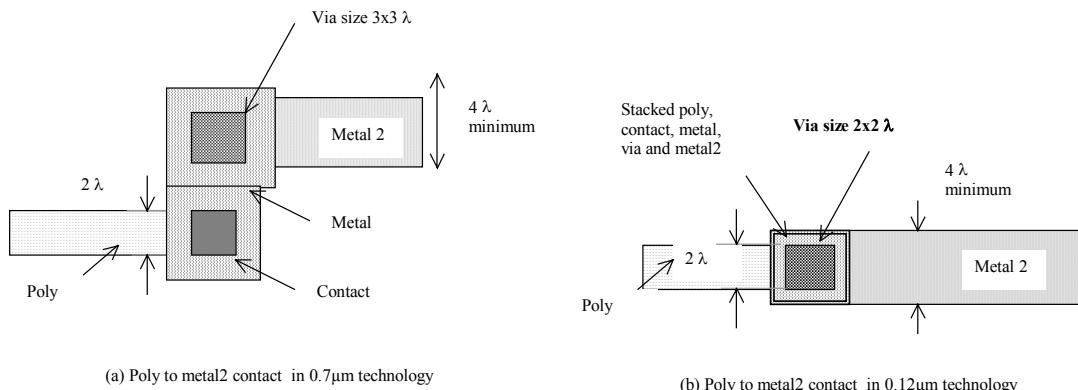
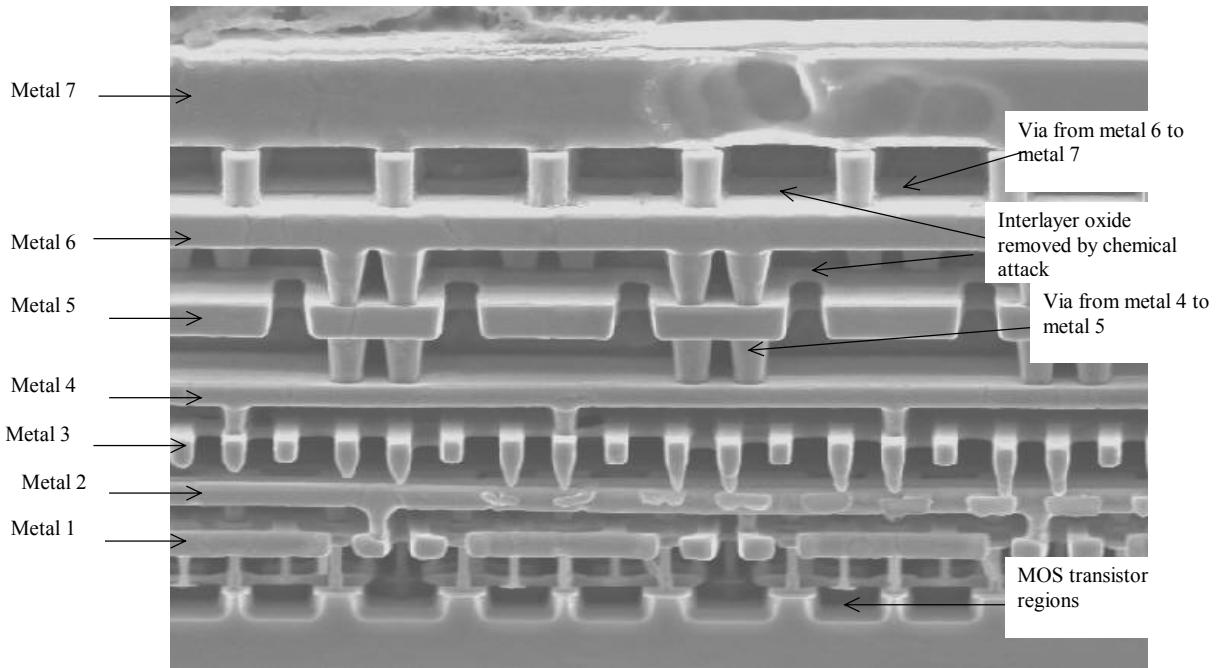


Figure 5-3: Stacked vias are allowed in 0.12 $\mu\text{m}$  technology, which saves a significant amount of silicon area compared to 0.7 $\mu\text{m}$  design style.

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 0.12 $\mu\text{m}$ ). Each layer is embedded into a silicon oxide ( $\text{SiO}_2$ ) which isolates layers from each other. A cross-section of a 0.12 $\mu\text{m}$  CMOS technology is shown in figure 5-4.



*Figure 5-4: Cross-section of a  $0.12\mu\text{m}$  technology (Courtesy Fujitsu)*

### 3. Contact & Vias

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..

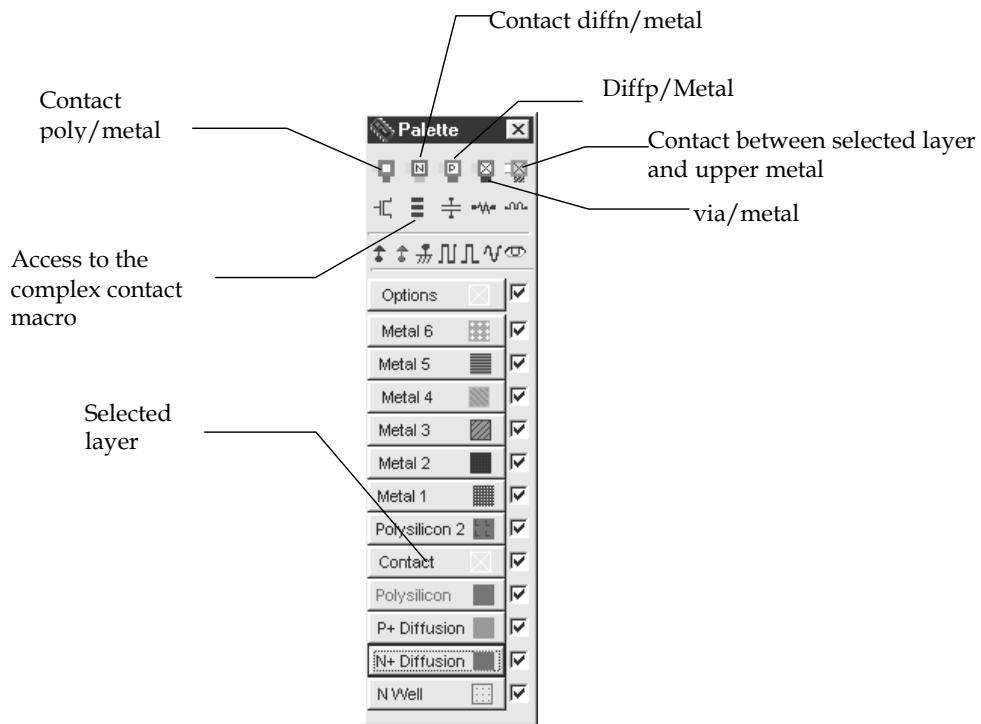


Figure 5-5: Access to basic contact macros

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 5-6. As an example, you may instanciate 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.

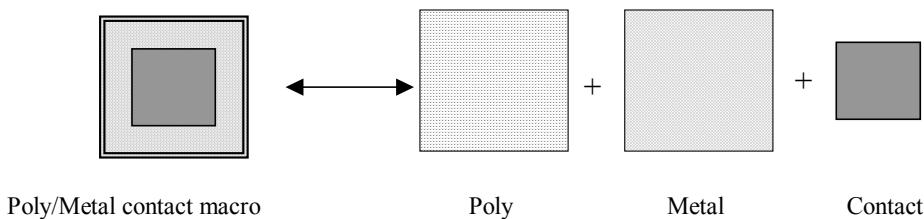


Figure 5-6: Access to basic contact macros

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.

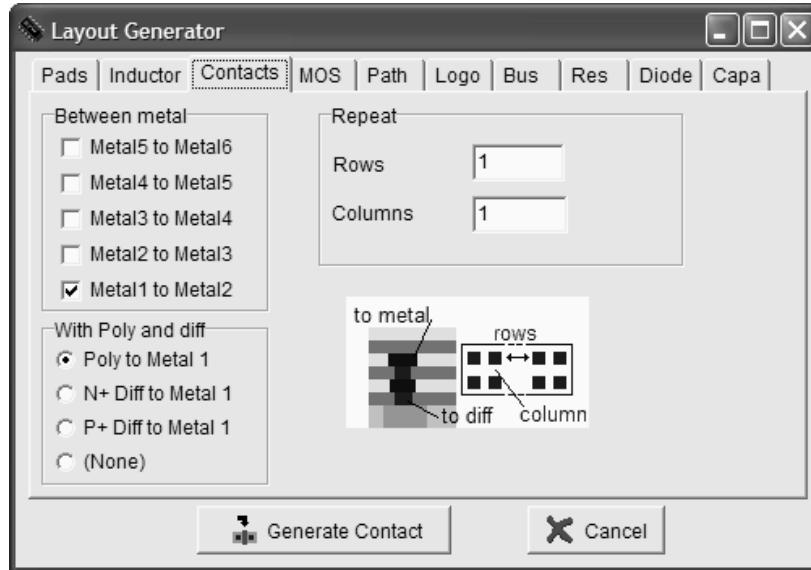


Figure 5-7: Access to complex stacked contact generator

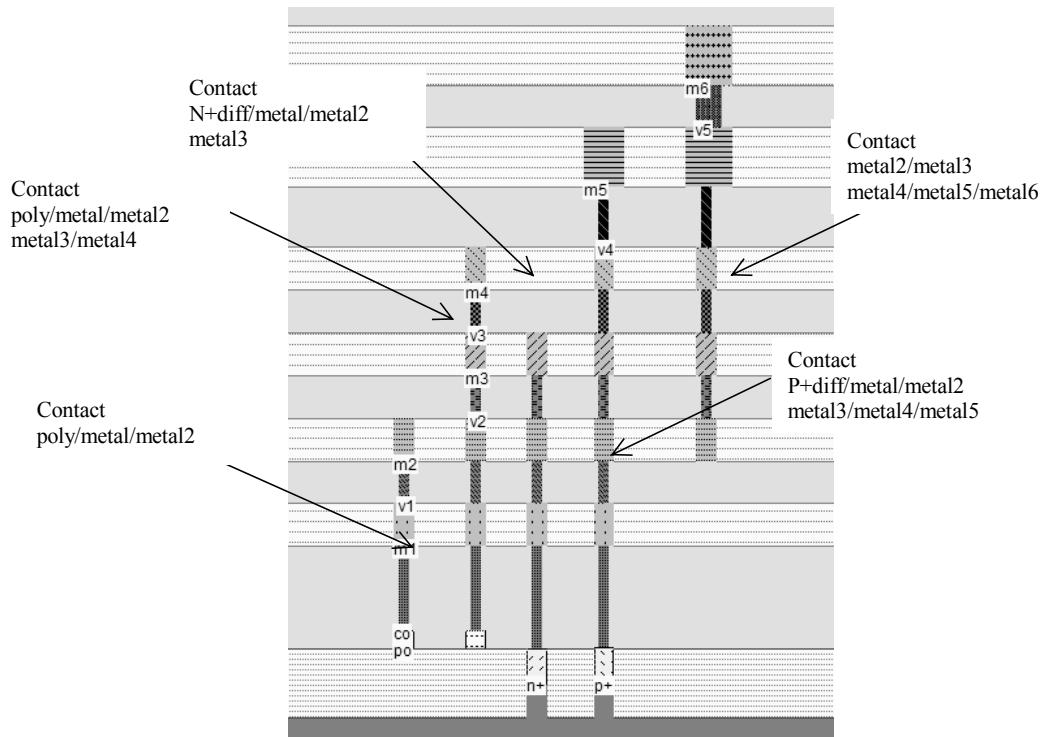
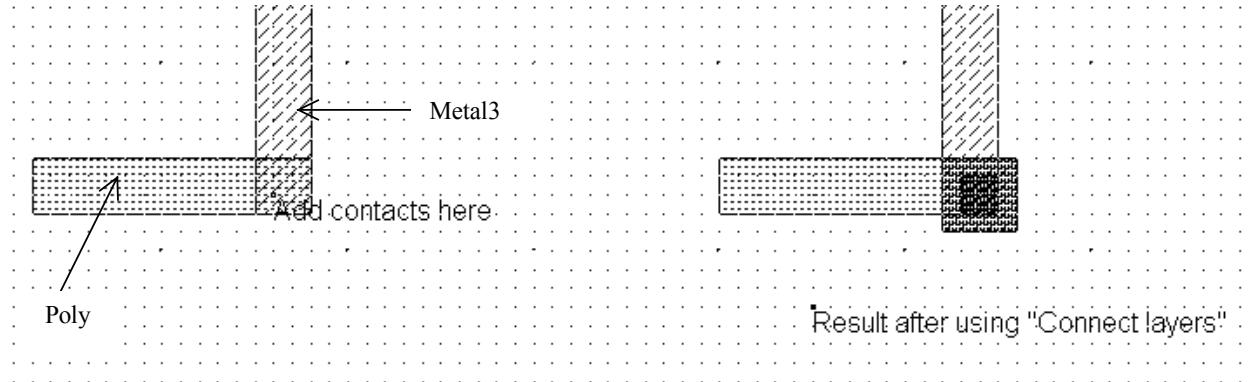


Figure 5-8: Examples of layer connection using the complex contact command from Microwind (Contacts.MSK)

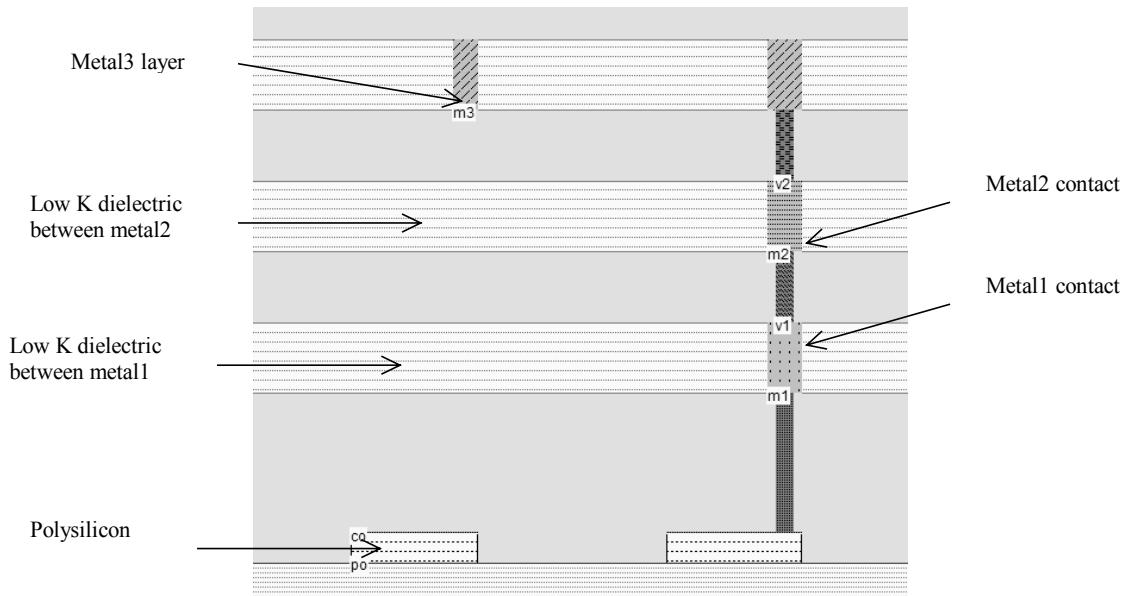
### DIRECT LAYER CONNECTION

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.



*Figure 5-9: The command "Connect layers" inserts the appropriate stacked contacts to build the connection between the desired layers (ConnectLayers.MSK)*



*Figure 5-10: 2-D cross-section of the layout before and after connecting poly and metal3 layers(ConnectLayers.MSK)*

## 4. Design rules

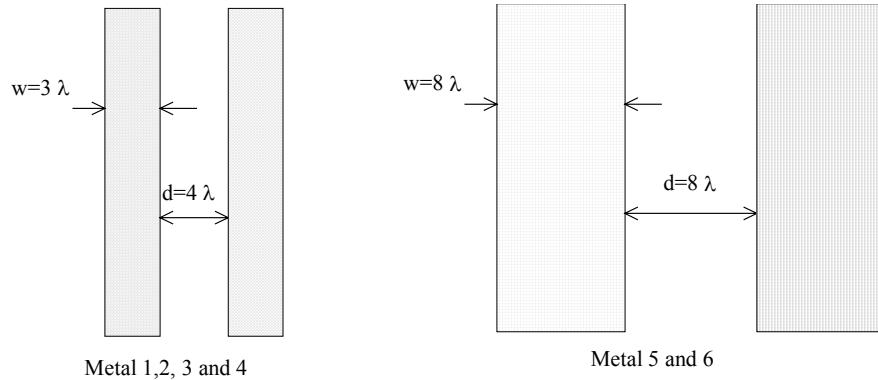
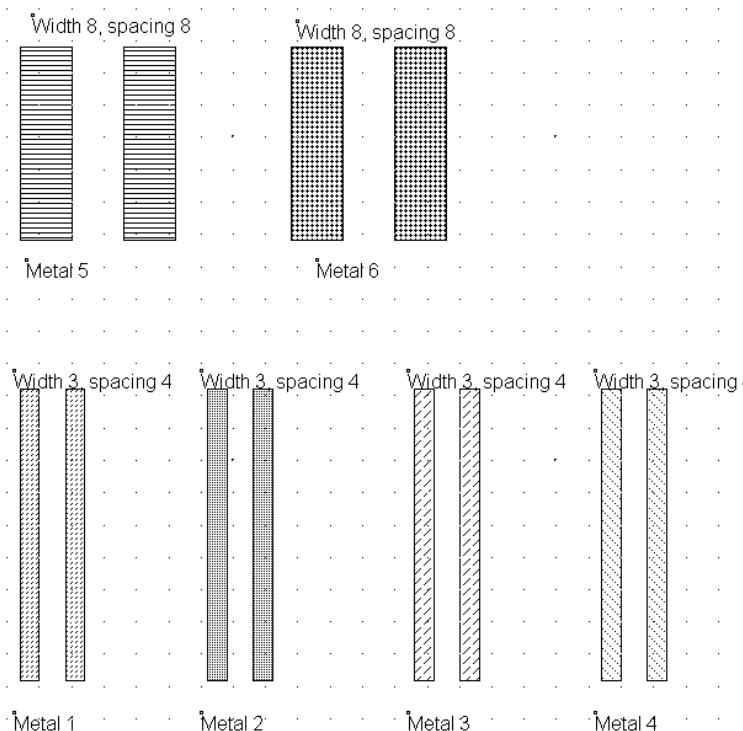


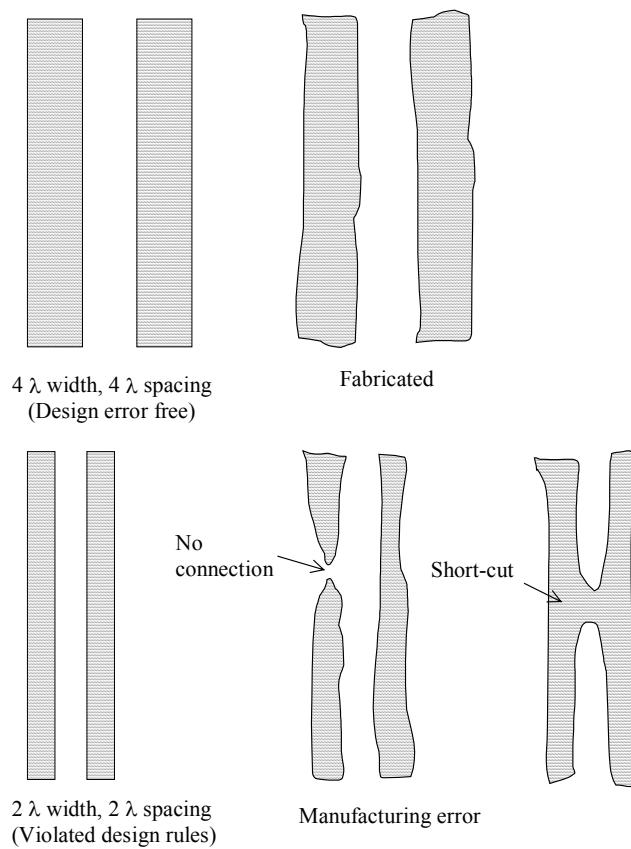
Figure 5-11: Minimum width  $w$  and distance  $d$  between metal layers

In  $0.12\mu\text{m}$  technology, the metal layers 1, 2, 3 and 4 have almost identical characteristics. Concerning the design rules, the minimum size  $w$  of the interconnect is  $3 \lambda$ . The minimum spacing is  $4 \lambda$  (Figure 5-11). In Microwind, each interconnect layer is drawn with a different color and pattern. Examples of minimum width and distance interconnects are reported in figure 5-12.



*Figure 5-12: Metal layers, associated rules and patterns as appearing in the Microwind editor (DesignRulesMetal.MSK)*

These minimum width and spacing are critical dimensions. They define the limit below which the probability of manufacturing error rises to an unacceptable level. If we draw metal 1 lines with 2 lambda width and 2 lambda spacing, interconnect interruptions or short-cuts may appear, as illustrated in figure 5-13. Prior to fabrication, the design rules must be checked to ensure that the whole circuit complies with the width and spacing rules, to avoid unwanted interruptions or bridges in the final integrated circuit. The Microwind command to get access to the design rule checker is **Analysis → Design Rule Checker**. Still, there exists a significant probability of manufacturing error even if the circuit complies to all design rules. A wafer of 500 integrated circuits has a typical yield of 70% in mature technologies, which means that 30% of the circuits have a fabrication error and must be rejected. The yield may drop to a percentage as low as 20%, for example in the case of state-of-the art technologies with all process constraints pushed to their limits, and very large silicon dies.



*Figure 5-13: The manufacturing of interconnects which violate the minimum width and distance may result in interruptions or short-cuts, which may have catastrophic consequences on the behavior of the integrated circuit.*

The practical design width for metal interconnects is usually a little higher than the minimum value. In Microwind, the routing interconnects are drawn in 4 lambda width. The pitch is the usual distance that separates two different interconnects. In 0.7 $\mu$ m, due to severe constraints in the contact size, the pitch has been fixed to 10 lambda. In deep-submicron technology, improvements in contact sizing may reduce that pitch to 8 lambda. In 0.12 $\mu$ m technology, this routing pitch is equivalent to 0.48 $\mu$ m.

Technology	Filename	Upper metal			Lower metal		
		Width	Thickness	Distance	Width	Thickness	Distance
0.8µm	cmos08.RUL	1.2	0.70	1.20	3.20	0.70	6.00
0.6µm	cmos06.RUL	0.9	0.70	0.90	2.40	0.70	4.50
0.35	cmos035.RUL	0.6	0.70	0.60	1.60	0.70	3.00
0.25	cmos025.RUL	0.38	0.60	0.50	1.00	0.70	1.88
0.18	cmos018.RUL	0.30	0.50	0.40	0.80	1.00	0.80
0.12	cmos012.RUL	0.18	0.40	0.24	0.48	0.80	0.48
90nm	cmos90n.RUL	0.15	0.35	0.20	0.40	1.60	0.40
70nm	cmos70n.RUL	0.10	0.30	0.14	0.28	1.00	0.52
50nm	cmos50n.RUL	0.08	0.25	0.10	0.20	0.50	0.38

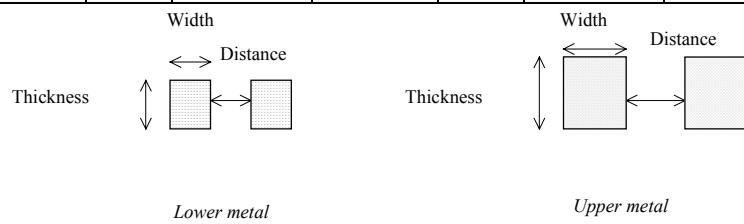


Table 5-2: Conductor parameters vs. technology

Integrated circuit manufacturers usually specify this routing pitch in their non-confidential technological descriptions, as one of the commercial arguments for designing compact (and behind this, low cost) integrated circuits. Common industrial pitch in 0.12µm CMOS process is around 0.4µm. The design styles in 0.7µm and 0.12µm are illustrated in figure 5-14.

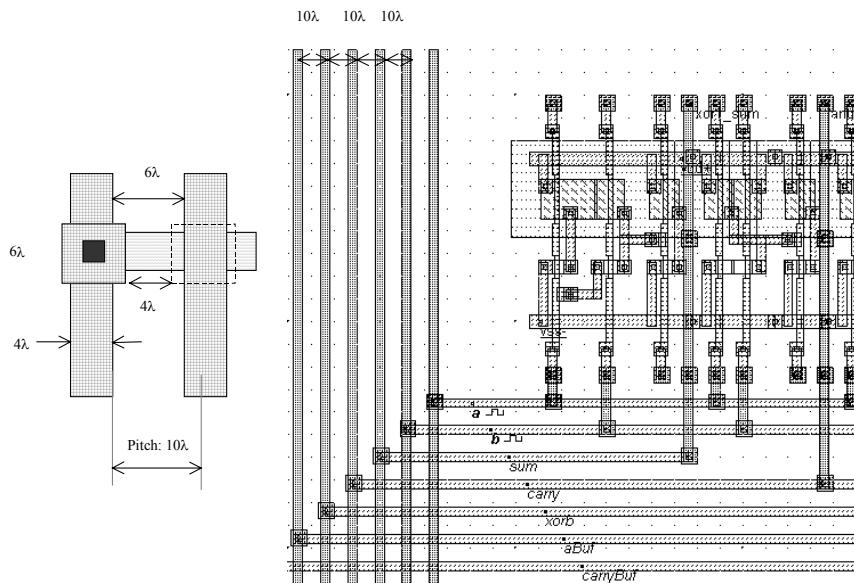


Figure 5-14: Illustration of the routing pitch in 0.7µm, set to 10 lambda due to the large size of the contacts

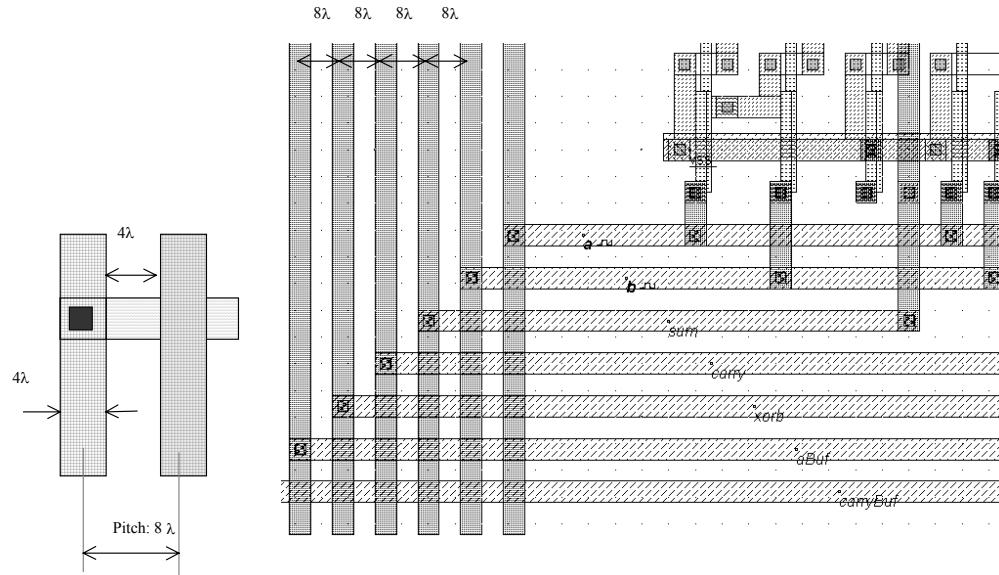


Figure 5-15: Illustration of the routing pitch in  $0.12\mu m$ , set to  $8\lambda$  thanks to the reduced size of the contact

## 5. Capacitance associated with interconnects

Interconnect lines exhibit the property of capacitance, as they are able to store charges in the metal interface with oxide. The capacitance effect is not simple to describe and to modelize. This is due to the fact that interconnects are routed very close to each other, as shown in the example in figure 5-16. The capacitance effects are represented by a set of capacitors which link interconnects electrically.

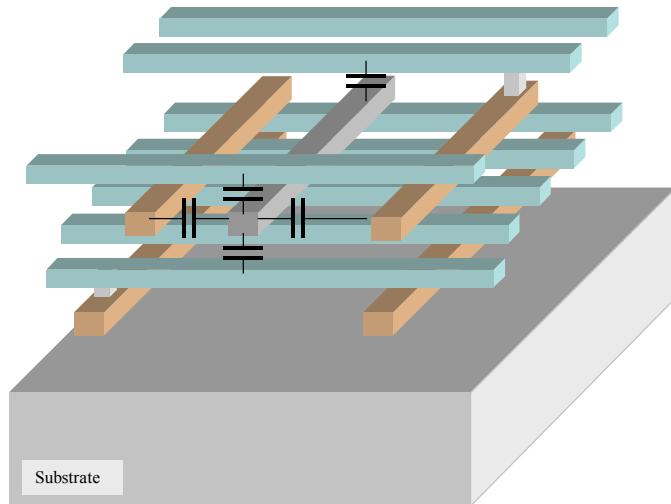


Figure 5-16: One interconnect is coupled to other conductors in several ways, both lateral and vertical

## LARGE PLATES

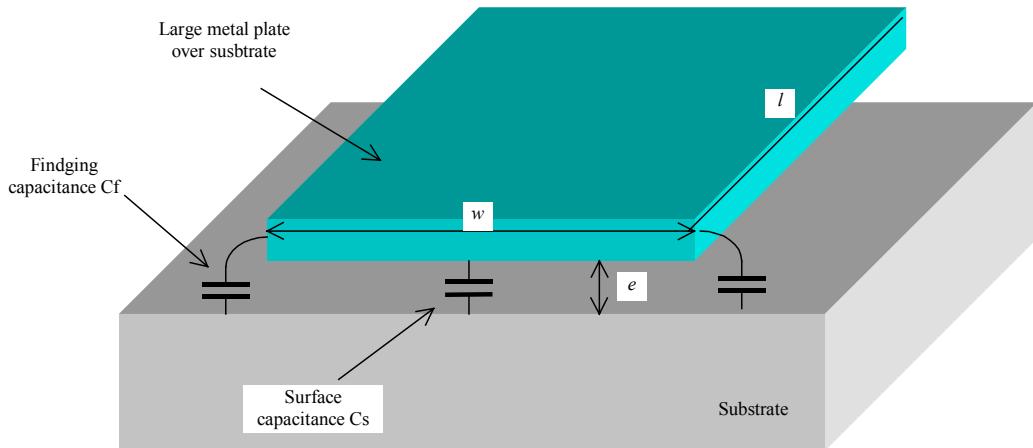


Figure 5-17: Large plate of metal above the substrate

In the case of a large metal area (width  $w$ , length  $l$ ) separated from the substrate or other metal areas by an oxide with a thickness  $e$ , the formulation 5-1 is quite accurate. We neglect the fringing capacitance  $C_f$  in that case. Large plates of metal are used in pads (See chapter 15 for input/output pad description) and supply lines.

$$C_s = \epsilon_0 \epsilon_r \frac{w l}{e} \quad (5-1)$$

$\epsilon_0 = 8.85 \text{ e}^{-12}$  Farad/m

$\epsilon_r = 3.9$  for  $\text{SiO}_2$

$w$  = conductor width (m)

$l$  = conductor length (m)

$e$  = dielectric thickness (m)

## CONDUCTOR ABOVE A PLANE

Several formulations have been proposed [Sakurai][Delorme] to compute the capacitance of a conductor when the width is comparable to the oxide thickness, which is the case with the large majority of conductors used to transport signals. We give the formulation of the total capacitance which consists in the sum of  $C_s$  and twice the fringing capacitance  $C_f$ , from [Delorme].

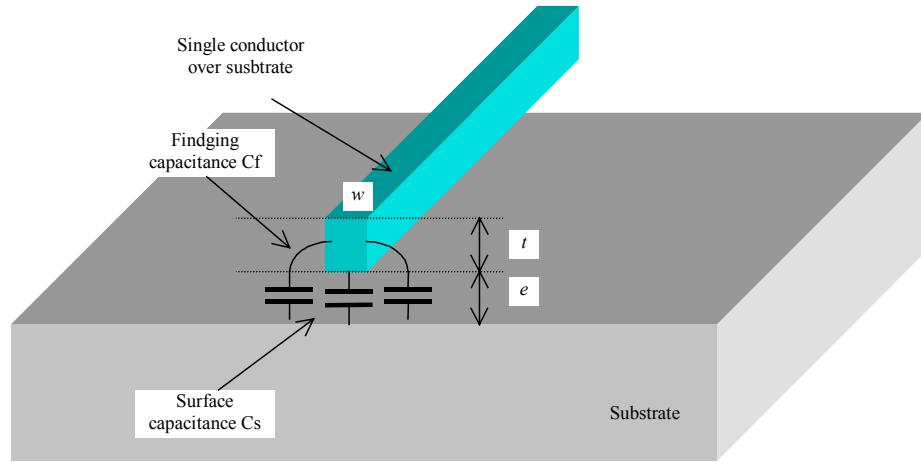


Figure 5-18: One conductor above a ground plane

$$C = C_s + 2 \cdot C_f = \epsilon_0 \epsilon_r (1,13 \cdot \frac{w}{e} + 1,44 \cdot (\frac{w}{e})^{0,11} + 1,46 \cdot (\frac{t}{e})^{0,42}) \quad (5-2)$$

C = total capacitance per meter (Farad/m)

Cs = surface capacitance (Farad/meter)

Cf = fringing capacitance (Farad/m)

$\epsilon_0 = 8.85 \text{ e}^{-12}$  Farad/m

$\epsilon_r = 3.9$  for SiO<sub>2</sub>

w = conductor width (m)

t = conductor thickness (m)

e = dielectric thickness (m)

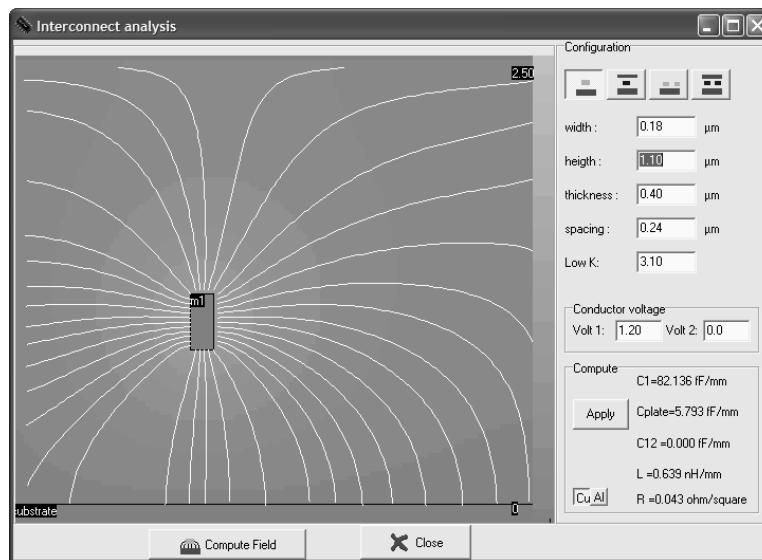


Figure 5-19: 2D simulation of the field lines between the conductor and the ground

The command **Analysis → Interconnect Analysis** with FEM gives some interesting information about the electric coupling between the conductor and the ground (Figure 5-19). The palette of colors indicate the potential in the oxide region, in volt. The field lines are the white wires which link the conductor to the ground. Some field lines are directly coupled to the substrate ground, some other field lines go to the free space. In  $0.12\mu\text{m}$  technology, the metal conductor is  $1.2\mu\text{m}$  above the ground plane, which corresponds to the simulation reported in figure 5-19. Its minimum width is  $0.18\mu\text{m}$  (Equivalent to  $3 \lambda$ ), its thickness is  $0.4\mu\text{m}$ . These physical dimensions are listed in the parameter menu situated on the right side of the window. The formulations given in (5-1) produce an underestimated value for  $C_{plate}$ , equal to  $6\text{fF/mm}$ . The formulations proposed in (5-2) give a more reliable result for  $C_I$ , equal to  $82\text{fF/mm}$ .

### Two Conductors above a ground plane

When a conductor is routed close to another conductor, a crosstalk capacitance, defined as  $C_{12}$  is created between the two conductors (Figure 5-20). In  $0.12\mu\text{m}$  technology, a specific dielectric with a low permittivity (This parameter, called LowK, is approximately 3 instead of 4) is used to fill the gaps between interconnects. This is an efficient technique to reduce the crosstalk capacitance while keeping the upper and lower capacitance almost unchanged. Consequently, the oxide stack alternates between high K and low K materials, as shown in the cross-section. Low K dielectrics were introduced with  $0.18\mu\text{m}$  technology. Air gaps are the ultimate low K materials, with a lowest possible  $K=1$ . Intensive research is being conducted on the enclosure of air gaps in between coupled connectors, and could become a standard in future technologies.

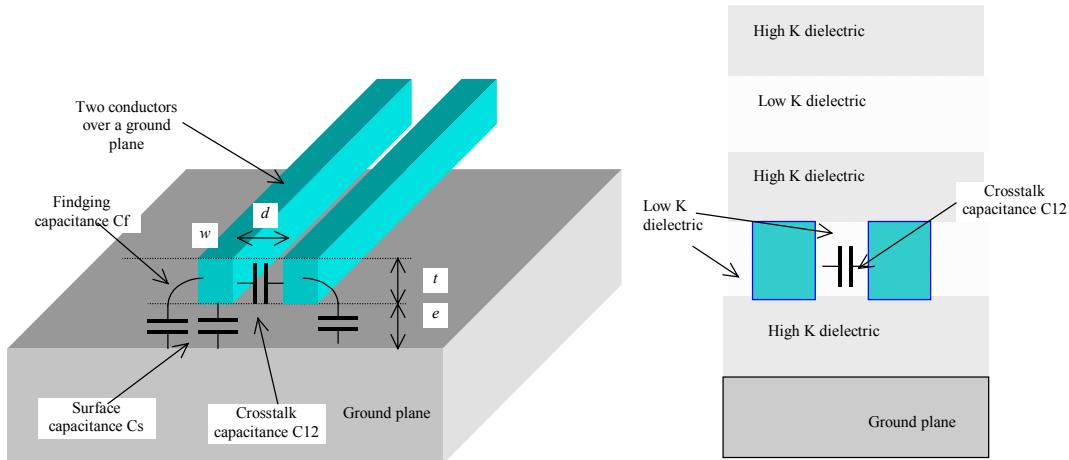


Figure 5-20: Two conductors above a ground plane

$$C = C_s + C_f = \epsilon_0 \epsilon_r (1.10 \frac{w}{e} + 0.79 (\frac{w}{e})^{0.1} + 0.46 (\frac{t}{e})^{0.17} (1 - 0.87 e^{\frac{-d}{e}})) \quad (5-3)$$

$$C_{12} = \epsilon_0 \epsilon_{rlowK} \left( \frac{t}{d} + 1.2 \left( \frac{d}{e} \right)^{0.1} \cdot \left( \frac{d}{e} + 1.15 \right)^{-2.22} + 0.253 \ln(1 + 7.17 \frac{w}{d}) \cdot \left( \frac{d}{e} + 0.54 \right)^{-0.64} \right) \quad (5-4)$$

$C$  = conductor capacitance to ground per meter (Farad/m)

$C_s$  = surface capacitance (Farad/meter)

$C_f$  = fringing capacitance (Farad/m)

$C_{12}$  = crosstalk capacitance (Farad/m)

$\epsilon_0 = 8.85 \text{ e}^{-12}$  Farad/m

$\epsilon_r = 3.9$  for  $\text{SiO}_2$

$\epsilon_{\text{lowK}}$  = permittivity of low dielectric material (around 3.0 in 0.12 $\mu\text{m}$ )

w= conductor width (m)

t= conductor thickness (m)

e = dielectric thickness (m)

d = conductor distance (m)

In the default 0.12 $\mu\text{m}$  technology, two coupled interconnects with the minimum width and distance routed with the first level of metallization have a cross-section shown in figure 5-21. The benefits of the low permittivity dielectric appear clearly in the value of the coupling capacitance  $C_{12}$  (75fF/mm), which is comparable to the ground capacitance  $C_1$  (51fF/mm). If we change the parameter *Low K* to the silicon dioxide permittivity equal to 4, the coupling capacitance becomes much higher than the ground capacitance. More about the crosstalk effect, including simulations and measurements, is given at the end of this chapter.

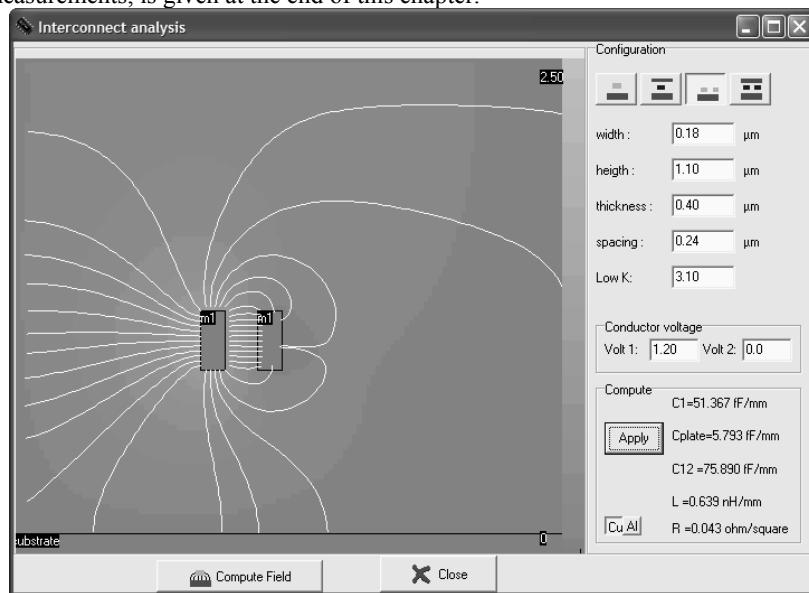


Figure 5-20: 2D simulation of the field lines between two conductors and the ground

### Real case conductors

In practice, the accurate extraction of the capacitance of each node is based on a 2 dimensional partitioning of the layout, and the 3D computing of capacitance for each elementary configuration. Even a simple interconnect configuration, such as the one shown in figure 5-21, is equivalent to a large set of 3D configurations, each of them requiring the use of a 3D static field solver.

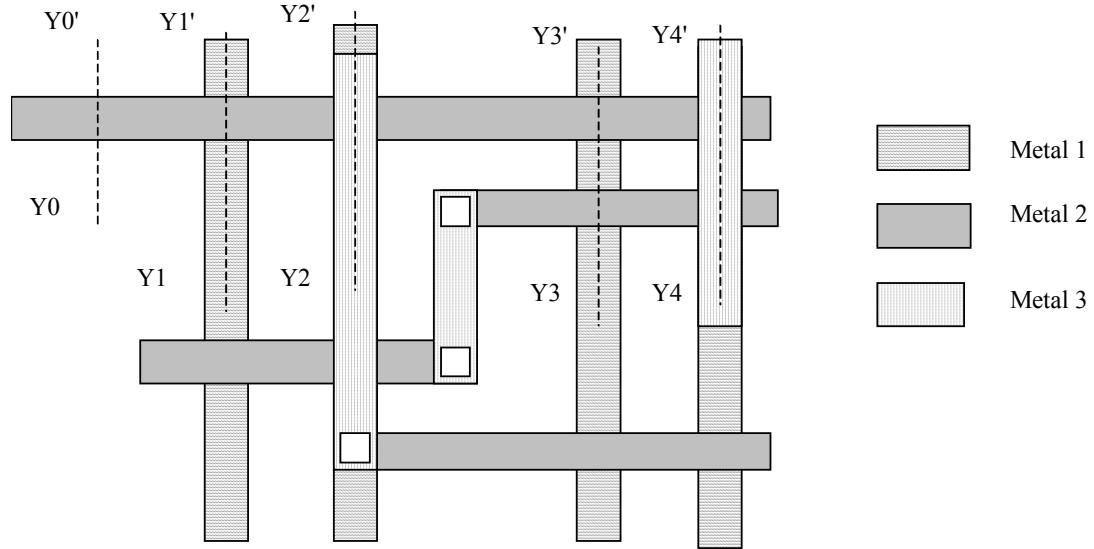


Figure 5-21: Example of interconnects routed in metal 1, 2 and 3

Examples of standard configurations that may be found in the layout of figure 5-21 are shown in figure 5-22, with numerical values corresponding to  $0.12\mu\text{m}$  technology, metal wire width 4 lambda ( $0.24\mu\text{m}$ ), and wire spacing 4 lambda ( $0.24\mu\text{m}$ ).

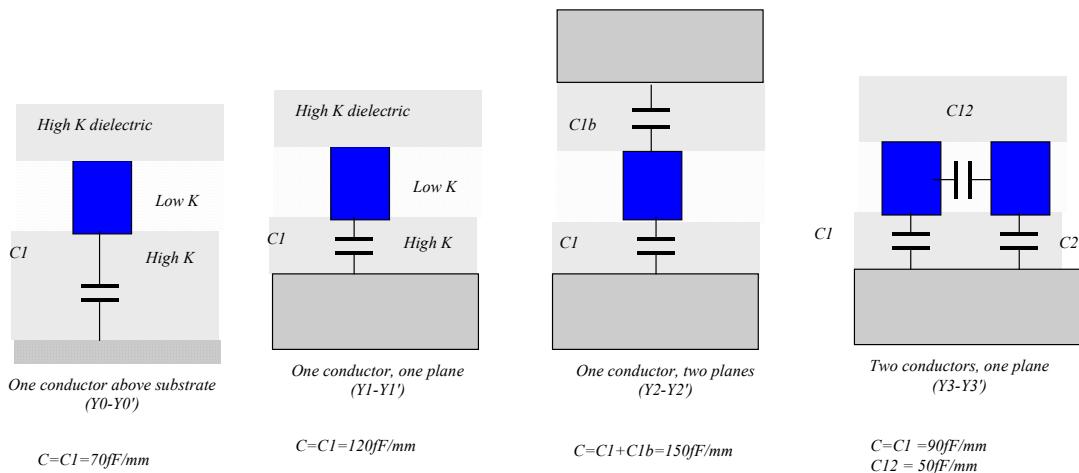


Figure 5-22: Basic configurations illustrating the cross-sections of the interconnect network ( $0.12\mu\text{m}$ )

The extraction of the layout is conducted according to the flow described in figure 5-23. The MOS devices and interconnects are extracted separately. To compute the crosstalk capacitance, the interconnect network is parsed into elementary structures, each one having a fixed stack of layers. An iterative procedure permits to locate vertical and horizontal elementary crosstalk contributions and to compute the sum which appears in the SPICE netlist and the electrical node properties.

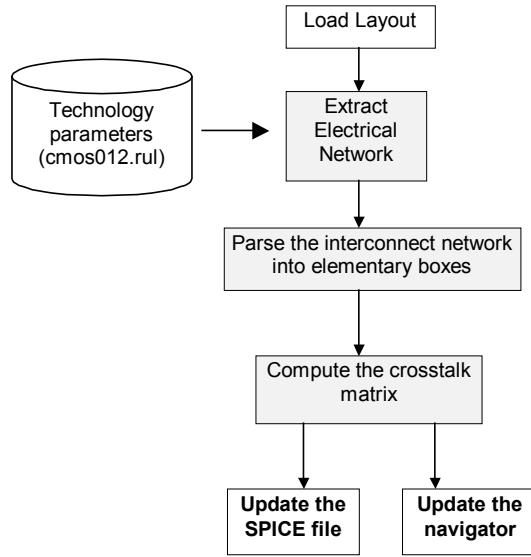


Figure 5-23: The crosstalk capacitance extraction steps in Microwind

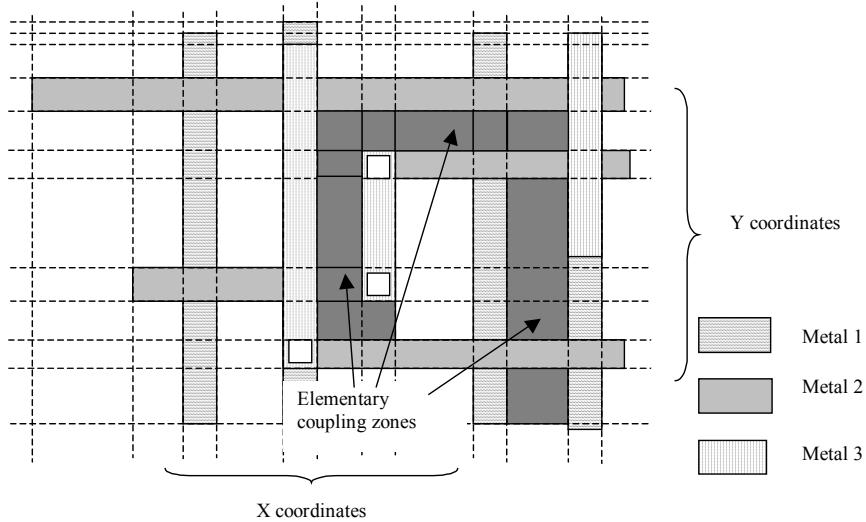


Figure 5-24: Identification of coupling zones in the test case presented in figure 5-21

Figure 5-24 shows where the elementary coupling zones have been detected. Only lateral coupling between identical layers, close enough to generate a significant crosstalk contribution are considered. The other couplings are neglected in this extraction phase. In the test case proposed in figure 5-21, couplings occur in metal1, metal2 and metal3, as seen in figure 5-24. Each contribution is very small (less than 1 femto-Farad). However, the sum of elementary coupling contribution may result in hundreds of femto farad, which may be comparable to the ground capacitance, and thus create significant crosstalk coupling effects.

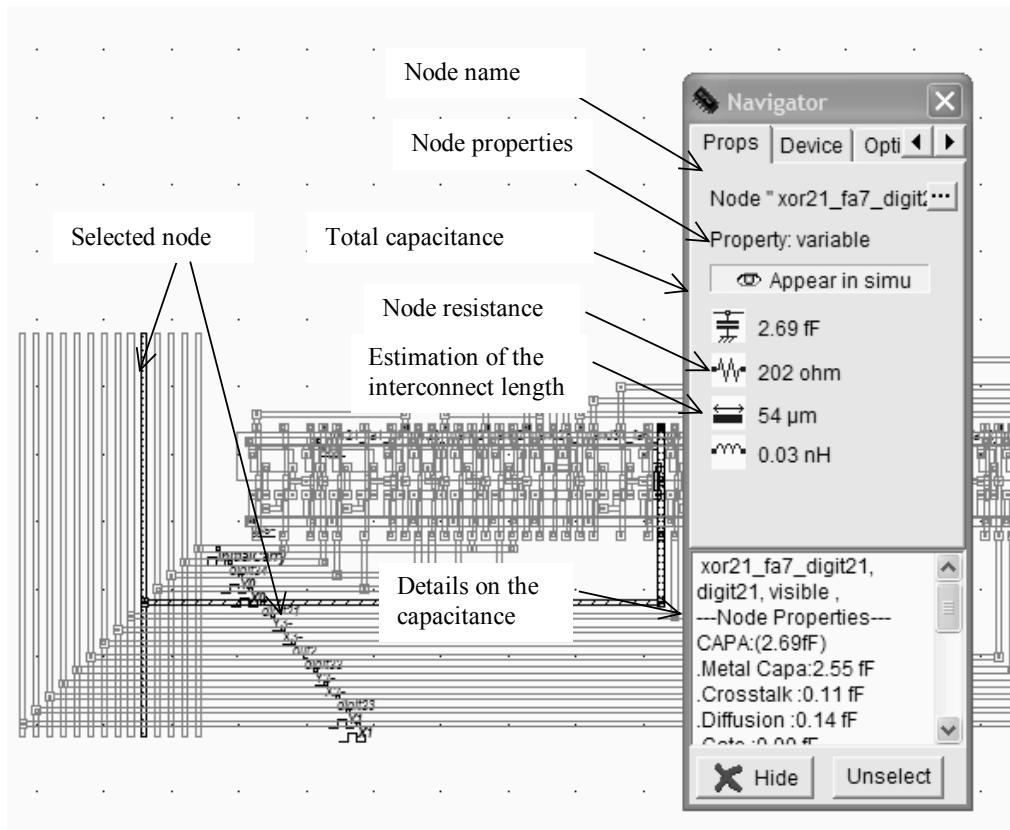


Figure 5-25: Extraction of a real-case interconnect capacitance using the command "View Electrical Node"



The command **View → View Electrical Node** or the above icon above launches the extraction of ground and crosstalk capacitance for each electrical net of the layout. The ground capacitance of the node and the crosstalk capacitance are detailed in the navigator window, as illustrated in figure 5-25. Notice that the global capacitance is split into metal, crosstalk, gate and diffusion capacitance. The selected net has weak crosstalk coupling as compared to ground coupling.

## 6. Resistance associated with interconnects

The resistivity of interconnect materials used in CMOS integrated circuits is listed in table 5-3. Conductors have very low resistivity, while semiconductor materials such as highly doped silicon have a moderate resistivity. In contrast, the intrinsic silicon resistivity is very high.

Symbol	Description	Used for	Resistivity at 25 °C
$\rho_{cu}$	Copper resistivity	Signal transport	$1.72 \cdot 10^{-6} \Omega \cdot \text{cm}$
$\rho_{al}$	Aluminum resistivity	Signal transport	$2.77 \cdot 10^{-6} \Omega \cdot \text{cm}$

$\rho_{Ag}$	Gold resistivity	Bonding between chip and package	$2.20 \cdot 10^{-6} \Omega.cm$
$\rho_{tungsten}$	Tungsten resistivity	Contacts	$5.30 \cdot 10^{-6} \Omega.cm$
$\rho_{Ndiff}$	Highly doped silicon resistivity	N+ diffusions	$0.25 \Omega.cm$
$\rho_{Nwell}$	Lightly doped silicon resistivity	N well	$50 \Omega.cm$
$\rho_{si}$	Intrinsic silicon resistivity	Substrate	$2.5 \cdot 10^5 \Omega.cm$

Table 5-3: Resistivity of several materials used in CMOS circuits

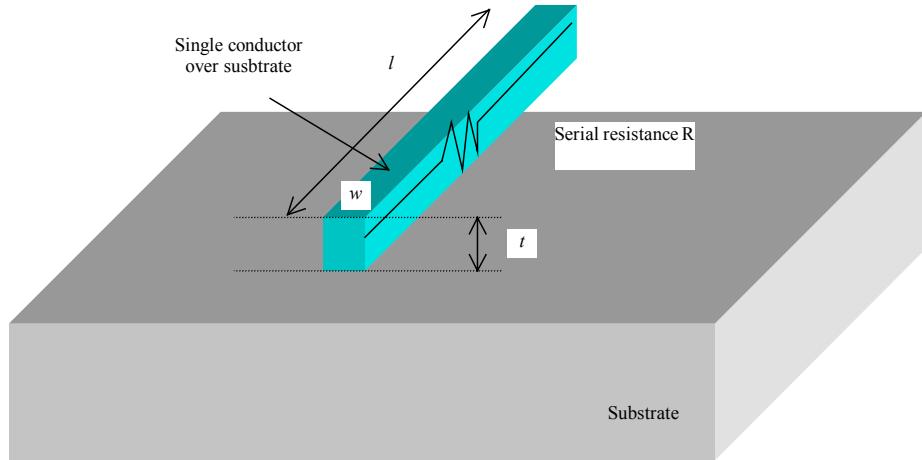


Figure 5-26: Resistance of a conductor

If a conductor with a resistivity has length  $l$ , width  $w$ , and thickness  $t$ , then its serial resistance  $R$  (Figure 5-26) can be computed using the following formula:

$$R = \rho \frac{l}{w \cdot t} \quad (\text{Equ. 5-5})$$

where

$R$ =serial resistance (ohm)

$\rho$ =resistivity (ohm.m)

$w$ = conductor width (m)

$t$ = conductor thickness (m)

$l$  = conductor length (m)

$d$  = conductor distance (m)

### Resistance per square

When designing interconnects, a very useful metric is the "resistance per square". We assume that the width is equal to the length, that is:

$$R_{square} = \rho \frac{w}{w \cdot t} = \frac{\rho}{t} \quad (\text{Equ. 5-6})$$

The interconnect material has long been aluminum as it was an easy-to-process material. Unfortunately, the resistivity of this material is quite high. Recently, copper has replaced aluminum for the manufacturing of interconnects, with a significant gain in terms of resistance, as the intrinsic resistivity of copper is almost twice lower than that of aluminum (See table 5-3). Tungsten is used in several CMOS technologies to fabricate contact plugs. Its ability to fill narrow and deep holes compensates its high resistance. Gold is only used to connect the final chip to its packaging, as described in chapter 15.

For a copper interconnect ( $1.72 \cdot 10^{-6} \Omega \cdot \text{cm}$ ) and a  $0.4\mu\text{m}$  thickness, the resistance is around  $0.043\Omega/\text{square}$ . The measured square resistance is higher than this theoretical value as the conductor is not homogenous. The titanium barriers located on both sides of the conductor have an important resistance which reduces the effective section of the conductor. In the CMOS  $0.12\mu\text{m}$  process files, a value of  $50\text{m}\Omega$  is used.

The square resistance is used to estimate rapidly the equivalent resistance of an interconnect by splitting its layout into elementary squares. The sum of square is then multiplied by  $R_{\text{square}}$  in order to evaluate the global interconnect resistance. We illustrate this concept in the layout shown in figure 5-27. We assume a resistance per square  $R_{\text{square}}$  of  $50\text{m}\Omega$ . The resistance from A to B can be approximated by 10 squares, that is a resistance of  $0.5\Omega$ .

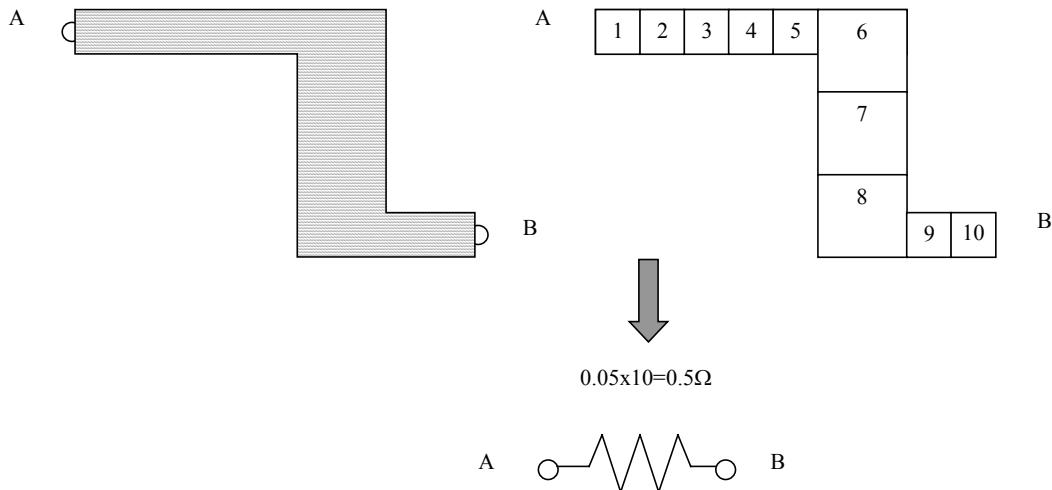


Figure 5-27: applying the concept of resistance per square to a portion of interconnect

The conductor resistivity is usually considered as a constant value. However, it depends on temperature in a complex manner [Hastings]. A usual approximation consists in considering the linear temperature coefficient of resistivity (TCR) expressed in parts per million per degree (ppm/ $^{\circ}\text{C}$ ). The copper and aluminum materials have similar behaviors, with a TCR around 4000 ppm/ $^{\circ}\text{C}$ . The resistance at a temperature T is given by the following equation:

$$R_T = R_{T_0} [1 + 10^{-6} TCR(T - T_0)] \quad (\text{equ 5-7})$$

where

$RT$ =serial resistance at temperature T(ohm)

$RT_0$ =serial resistance at reference temperature  $T_0$  (ohm)

TCR= temperature coefficient of resistivity (ppm/ $^{\circ}$ C)

T=temperature ( $^{\circ}$ C)

T0=reference temperature (Usually 25 $^{\circ}$ C)

Considering a typical 4 lambda wide metal interconnect, we observe that its cross-section is dramatically reduced with the scale down, due to a decrease of both the lateral and vertical dimensions of the elementary conductors. The elementary resistance  $R_{square}$  is increased at each technology generation, as shown in figure 5-29. The introduction of copper in high performance CMOS process has lowered almost by 50% the square resistance, but significantly increased the process complexity and overall fabrication cost.

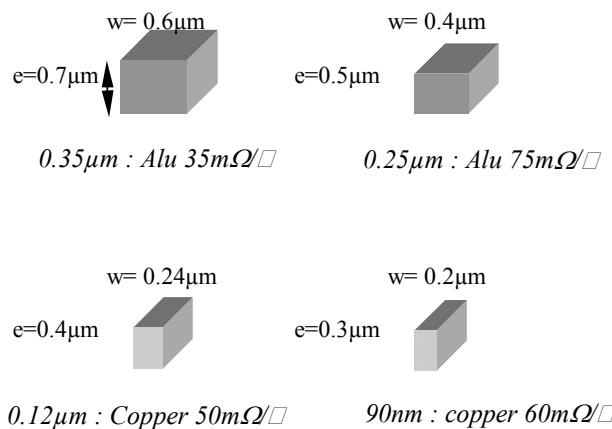


Figure 5-28: Evolution of interconnect resistance with the technology scale down [ITRS]

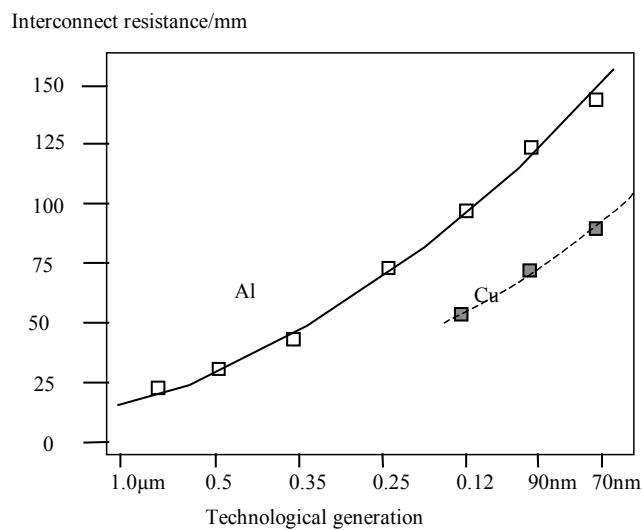


Figure 5-29: Evolution of interconnect resistance with the technology scale down

### Via Resistance

Each contact and via has a significant resistance. Typical values for these resistance are given in table 5-4 for three technologies:  $0.7\mu\text{m}$ ,  $0.12\mu\text{m}$  and  $90\text{nm}$ .

Technology	$0.7\mu\text{m}$	$0.12\mu\text{m}$	$90\text{nm}$
Contact resistance	$0.5 \Omega$	$15 \Omega$	$20 \Omega$
Via	$0.3 \Omega$	$4 \Omega$	$8 \Omega$
Upper via	-	$1 \Omega$	$3 \Omega$

Table 5-4: typical resistance of contacts and vias

Globally, the contact resistance and via increase with the technology scale down, due to the continuous reduction of the contact plug section, resulting in a reduced path for current. The contact resistance from active regions to the first metal layer is very important due to the thick oxide (Around  $1.0\mu\text{m}$ ) that separates the active MOS device altitude from the metal 1 altitude. A thick oxide is necessary to insert several optional materials such as double gate MOS devices for EEPROM memories, or large storage capacitance for DRAM memories. The upper via resistance is quite small as the size of that via is very large compared to the lower via.

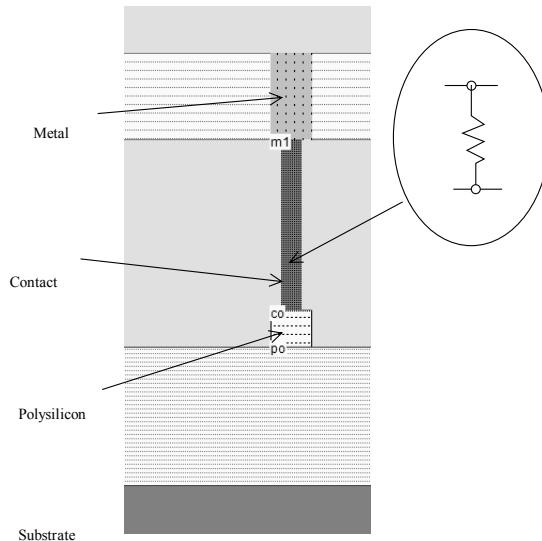


Figure 5-30: The contact is equivalent to a resistance (Contacts.MSK)

## 7. Signal Transport

The signal transport from one logic cell to another logic cell uses metal interconnects. Depending on the distance between the emitting cell and the receiving cell, the interconnect may be considered as a simple parasitic capacitance or a combination of capacitance and resistance. By default, Microwind only considers the parasitic capacitance. This assumption is valid for short to medium length interconnects. In  $0.12\mu\text{m}$ , interconnects with a length up to  $1000\mu\text{m}$  can be considered as a pure capacitance load  $C_l$  (Figure 5-31). For interconnects larger than  $1000\mu\text{m}$ , the serial resistance  $R_l$  should be included into the model. The usual way consists in splitting the capacitance  $C_l$  into two equivalent capacitances and place the serial resistance in between.

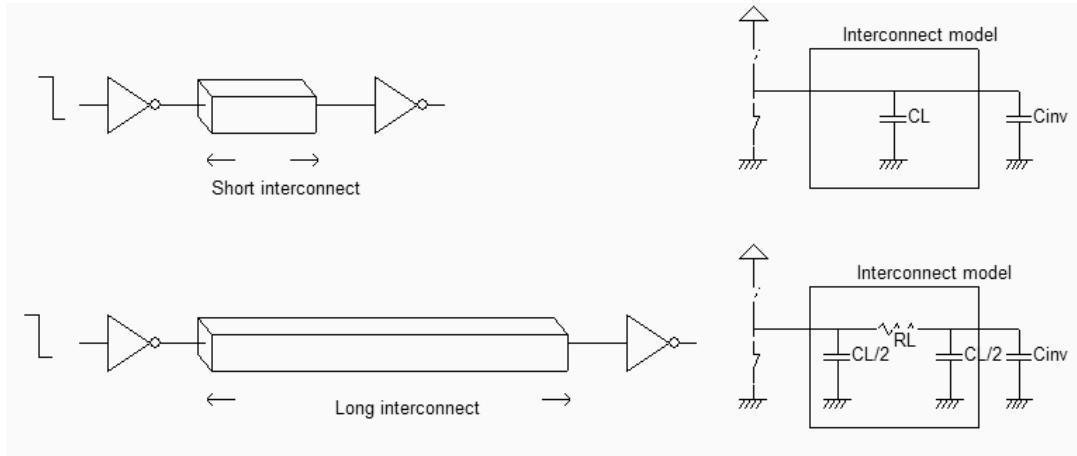


Figure 5-31: C versus RC model for interconnects (RcModels.SCH)

### Simulation of the RC effect

The RC delay within interconnects can be simulated using Microwind2 as follows. Consider the circuit shown in figure 5-32. It represents a buffer (Left lower corner) which drives a long interconnect, and then a loading inverter. From a layout point of view, the interconnect is usually straight, but for simplicity's sake, we use here a serpentine to emulate the RC effect without the need of a very large silicon area. The 4mm interconnect is obtained by connecting small portions of metal lines. The layout reported in figure 5-33 is based on 40 bars of metal4, with a length of 100 $\mu$ m. The serial resistance  $RL$  is equal to 820 ohm, and the capacitance  $CL$  is around 130fF, divided into two parts, shared on each side of the virtual resistance.

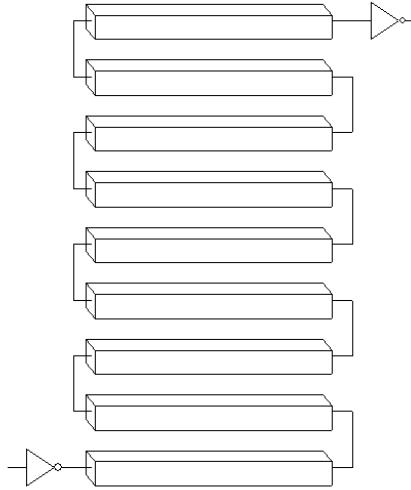


Figure 5-32: Emulation of the RC effect in a 4mm interconnect using serpentine (RcEffect.SCH)

Although both the capacitance and resistance of each node are extracted, the simulator considers by default only the capacitance and ignores the resistance. We add a “virtual” resistance using the resistance icon situated in the palette (Figure 5-33). This resistance is placed directly in the interconnect, as detailed in the layout of figure 5-34, to force

Microwind to handle the equivalent resistance of the interconnect. We assign to the resistor the value of the metal interconnect resistance, appearing as  $R(\text{metal})$ . This value is updated during the electrical network extraction phase.

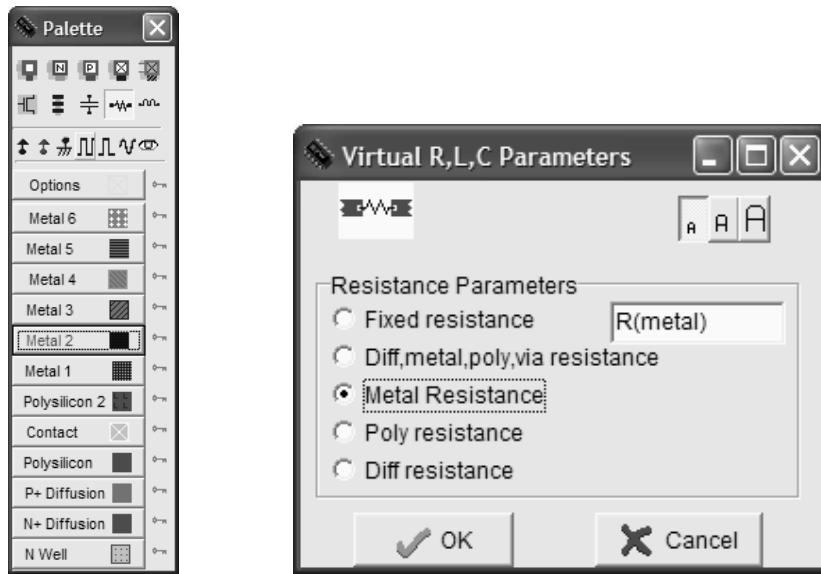


Figure 5-33: Inserting a virtual resistance to take into account the serial resistance of the interconnect  
(RcModels.SCH)

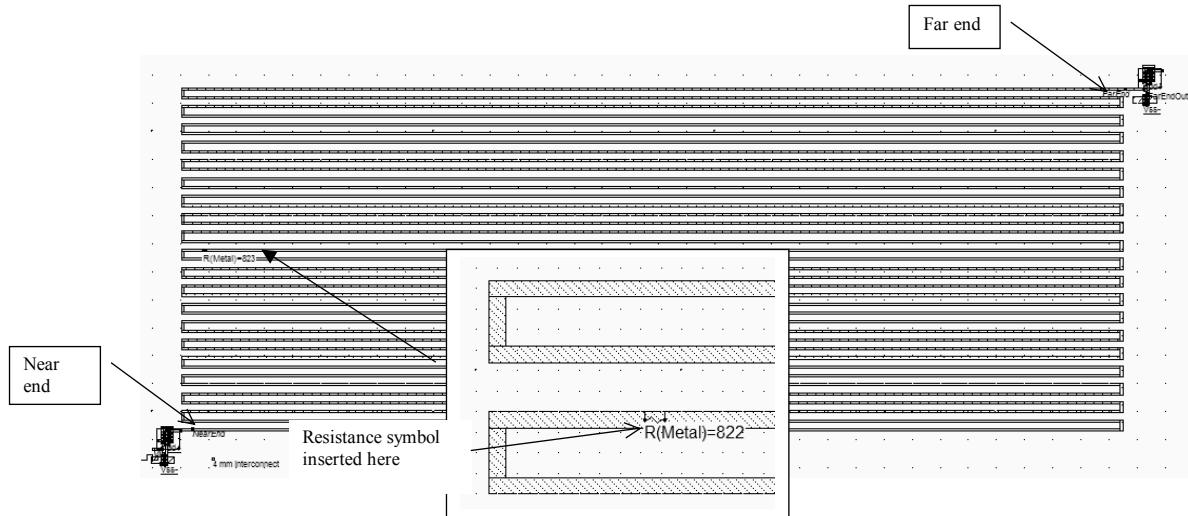


Figure 5-34 : adding a virtual resistance within the layout to simulate the resistance (RcEffect.MSK)

The RC effect of the interconnect appears very clearly in the analog simulation (figure 5-35). The initial phase runs from time 0.0 to time 1.0ns, which is not significant. At time  $t=1.0\text{ns}$ , the input shifts from a high to a low state. The near end of the line switches within approximately 200ps. The far end of the line reaches  $\text{VDD}/2$  after twice this delay. The same delay effects are observed at the fall edge of the clock ( $t=1.5\text{ns}$ ).

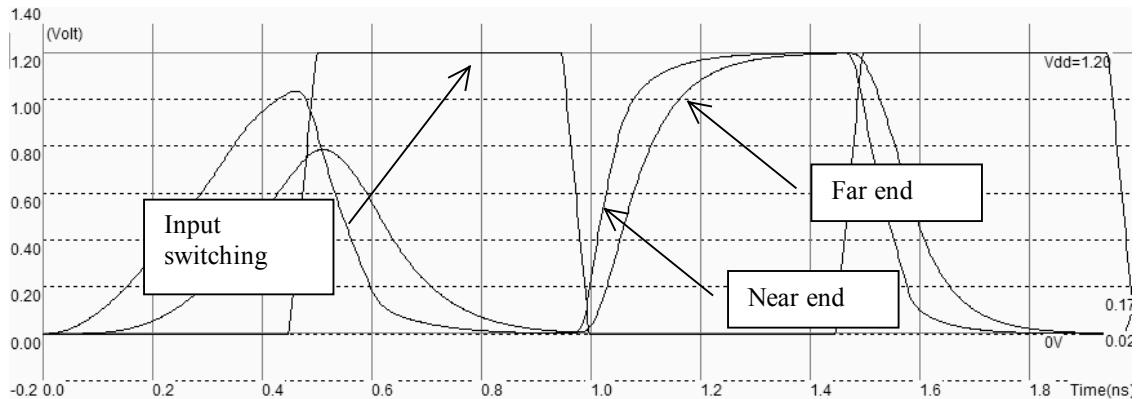
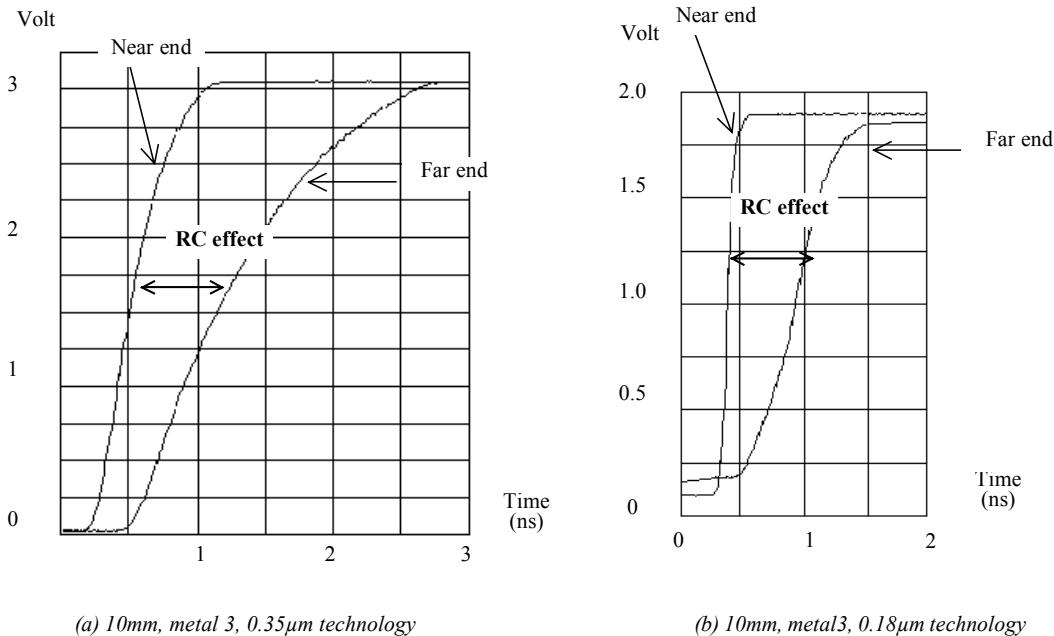


Figure 5-35 : RC delay simulation (RCEffect.MSK)

Figure 5-36: Measurement of RC effect in a very long interconnect (10mm) in metal 3, experimented in  $0.35\mu\text{m}$  and  $0.18\mu\text{m}$  technologies [Bendhia]

The RC effect can be measured thanks to an on-chip oscilloscope approach [Bendhia] at the near end and far end of metal interconnects. In the left part of figure 5-36, the measurement concerns a 10mm interconnect routed in metal3, fabricated in a  $0.35\mu\text{m}$  five metal layer process from ST-Microelectronics [ST]. The observed waveforms confirm the important impact of the serial resistance, estimated in this particular case at around 300 ohm. The waveform is similar in  $0.18\mu\text{m}$  technology, 6 metal layers, but the near end of the line is prompt to switch within 100ps, while, the far end of the line needs more than 500ps to pass the  $VDD/2$  limit. Notice the supply voltage difference between these two technologies.

### Limit between C and RC models

In  $0.12\mu\text{m}$  technology, the interconnect length limit above which the resistance should not be neglected is around 1mm. This limit can be illustrated by implementing one configuration with the capacitance model, and another configuration with the RC model. To implement a long interconnect, Microwind offers a bus generation command, with an interface reported in figure 5-37. This menu is accessible by the command **Edit → Generate → Metal bus**. Rather than drawing a straight line with a  $300$ ,  $800$  or  $2000\mu\text{m}$  length, we split the metal wire into several portions. This makes the layout more compact, and is equivalent to the straight line. In the example given in figure 5-37, we split the  $2\text{mm}$  line into  $10$  portions of interconnects with a length of  $200\mu\text{m}$  each.

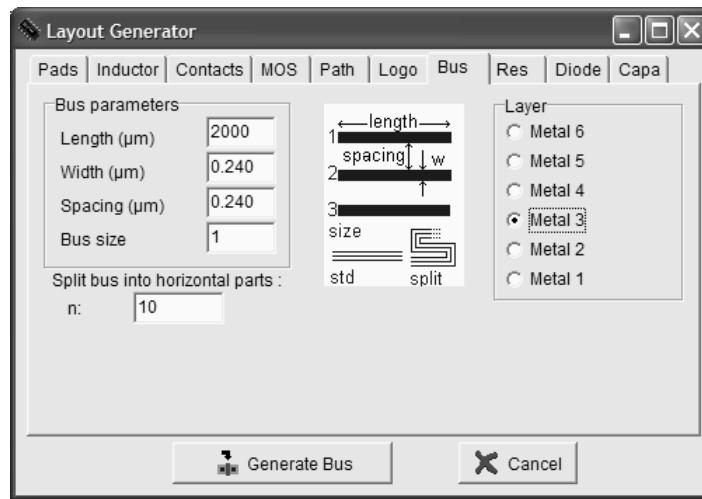


Figure 5-37: Generating a long metal interconnect using the bus generator command

In this study, three interconnect lengths are investigated:  $300\mu\text{m}$ ,  $800\mu\text{m}$  and  $2\text{mm}$ . Each configuration is implemented with and without the resistance to compare the simulation with the C model alone or the RC model.

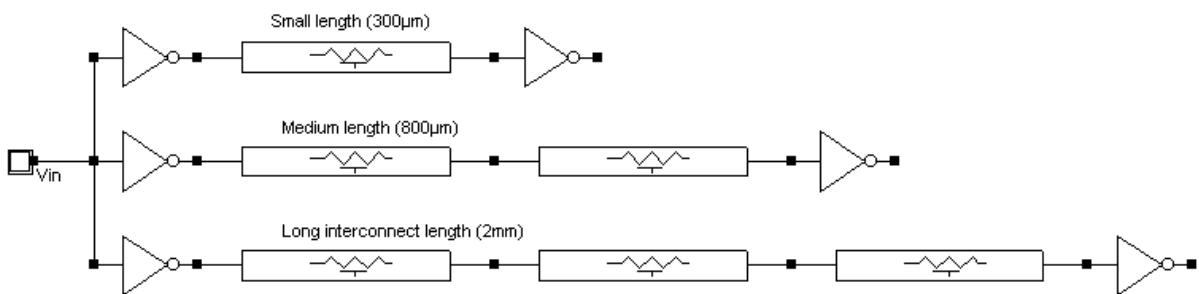


Figure 5-38: Simulation of 3 interconnect configurations to investigate the impact of C/RC models (RcModels.SCH)

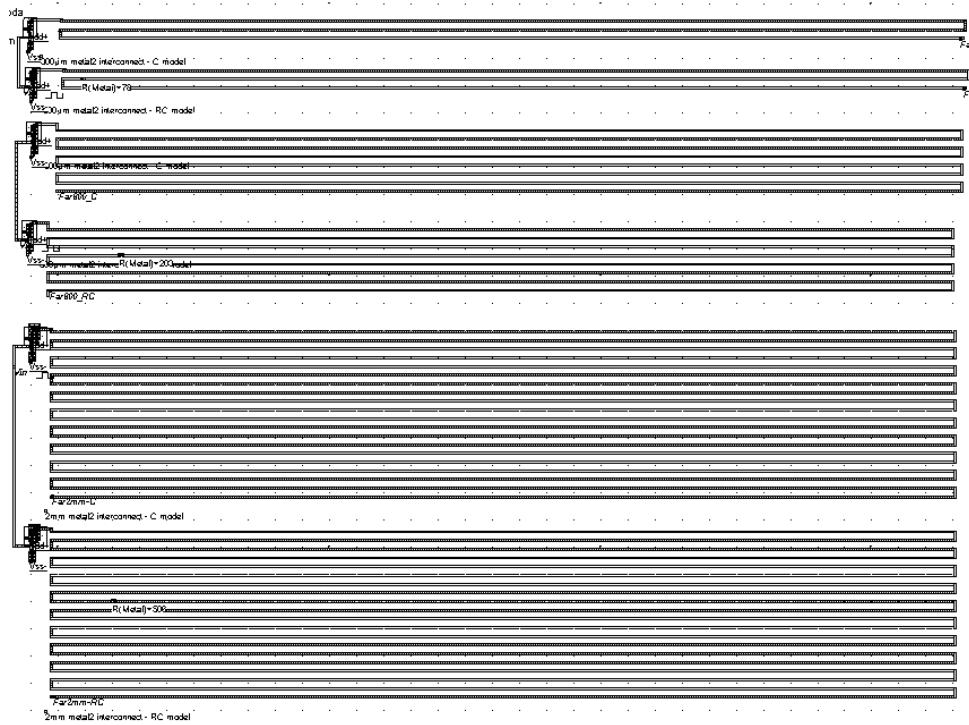


Figure 5-39: Layout of the 3 interconnect configurations (300 $\mu$ m, 800 $\mu$ m and 2mm) to investigate the impact of C/RC models (RCModel.MSK)

Each wire is implemented twice: one version is used to simulate the capacitor model, the other one includes a small resistance fixed approximately half way between the near and far end of the interconnect, to force the simulator to take into account the serial resistance of the interconnect.

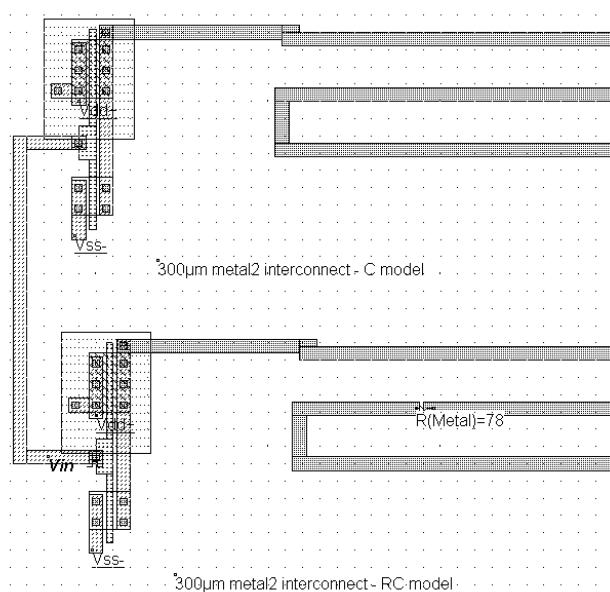


Figure 5-40: Portion of the layout showing the 2 versions of the 300 $\mu$ m configuration, with and without resistance (RCModel.MSK)

In order to handle the resistance effect of the interconnect, we place a virtual resistance approximately in the middle of the metal path. In the menu, we choose **Metal resistance**, which indicates that the extracted metal resistance should be used as the interconnect resistance. In the layout, the text appearing in the R symbol will be  $R(\text{metal})$ . The corresponding value appears after extraction or simulation. When changing the technology, the value of the resistance will be updated according to the new sheet resistance parameters.

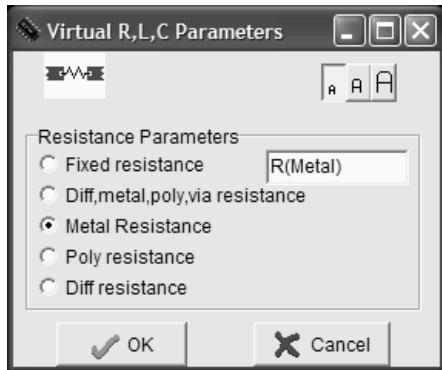
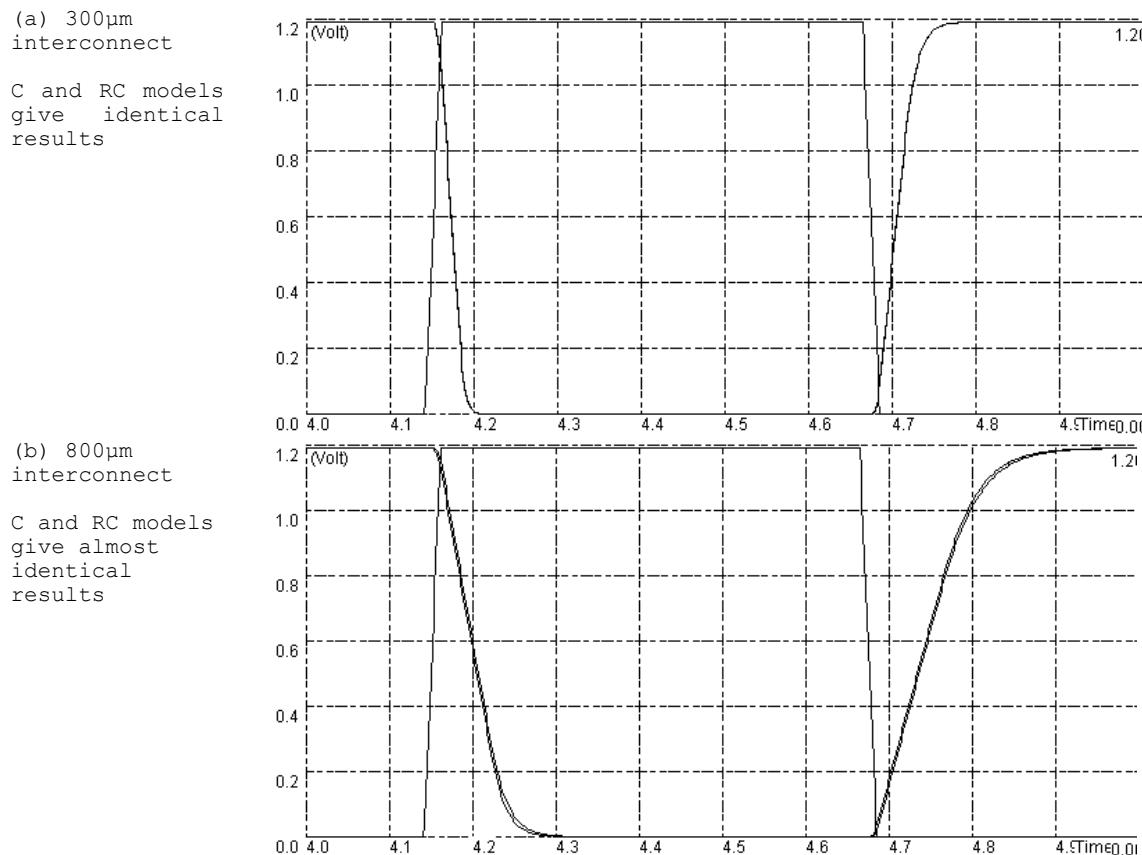


Figure 5-41: Configuring the virtual resistance to handle the metal serial resistance effect in simulation  
(RCModel.MSK)



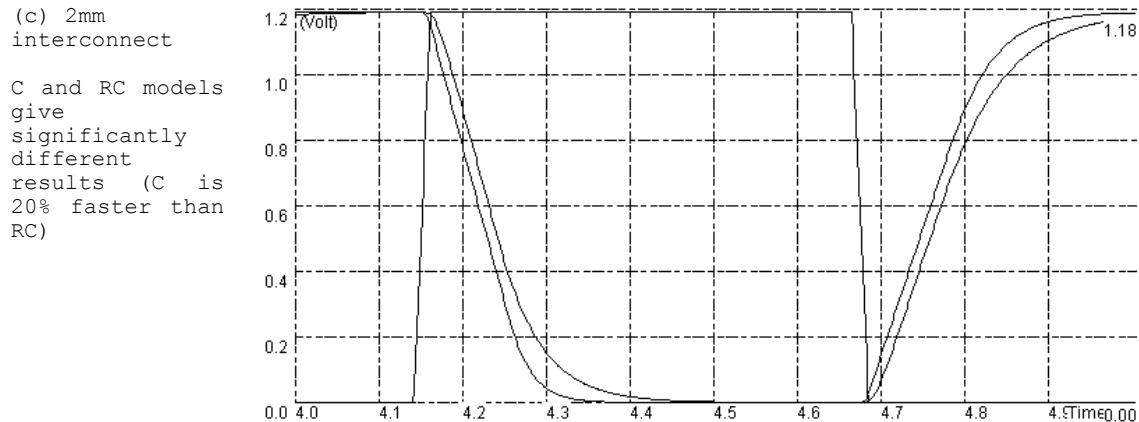


Figure 5-42: Comparative simulation of the C and RC model in signal propagation (RCModel.MSK)

The analog simulation of the signal transport with the 300 $\mu$ m, 800 $\mu$ m and 2mm interconnects are given in figure 5-42. The simulated propagation with C and RC models gives no visible difference below 1mm. Above 1mm, the C model gives optimistic prediction of the delay compared to the RC model. When we plot the delay vs. interconnect length (Figure 5-43), we may see that the RC model is preferable for interconnects with a length greater than 1mm, while the C model is sufficient below 1mm.

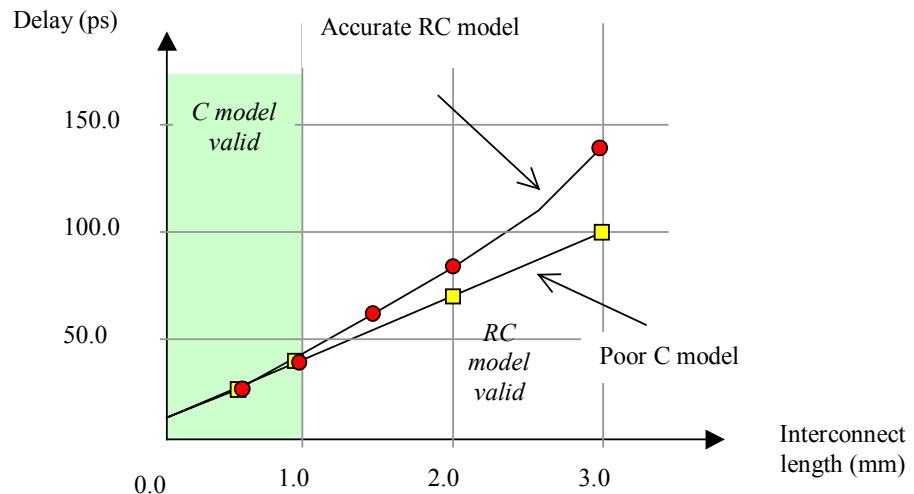


Figure 5-43: Below 1mm, the C model is valid. Above 1mm, the RC model should be considered in 0.12 $\mu$ m CMOS technology

## 8. Improved signal transport

The basic layout techniques to reduce the signal transport within interconnects are detailed in this paragraph. Two approaches are considered: improving the drive of the switching inverters, and inserting repeaters. We shall also discuss the crosstalk effect and its possible reduction through technology and design improvements.

### Increased Current Drive

The simplest approach to reduce the gate delay consists in connecting MOS devices in parallel. The equivalent width of the resulting MOS device is the sum of each elementary gate width. Both nMOS and pMOS devices are designed using parallel elementary devices. Most cell libraries include so-called  $x1$ ,  $x2$ ,  $x4$ ,  $x8$  inverters. The  $x1$  inverter has the minimum size, and is targeted for low speed, low power operations. The  $x2$  inverter uses two devices  $x1$  inverters, in parallel. The resulting circuit is an inverter with twice the current capabilities. The output capacitance may be charged and discharged twice as fast as with the basic inverter (Figure 5-44), because the  $R_{on}$  resistance of the MOS device is divided by two. The price to pay is a higher power consumption. The equivalent  $R_{on}$  resistance of the  $x4$  inverter is divided by four.

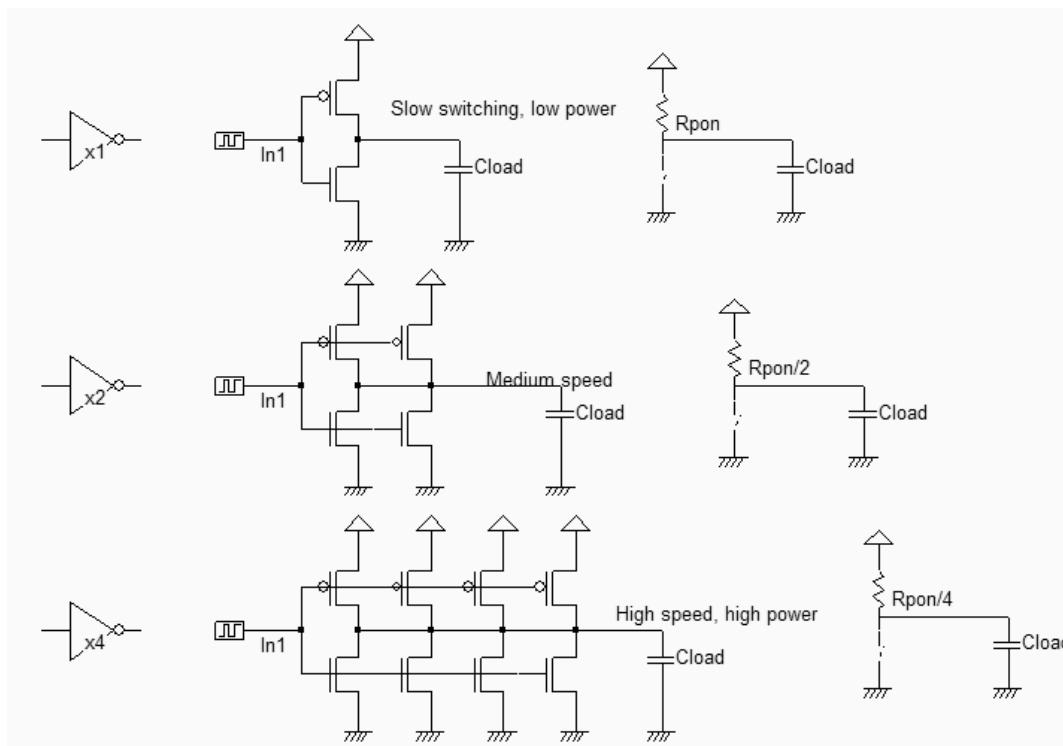


Figure 5-44: The  $x1$ ,  $x2$  and  $x4$  inverters (Invx124.SCH)

We may use the parametric analyzer included in Microwind to investigate the delay increase with the capacitance load on the output node, for the  $x1$ ,  $x2$  and  $x4$  inverters. In a first approximation, the capacitance increase is similar to the interconnect length increase. Remember that below 1mm, the interconnect is basically a parasitic capacitance, with a value of  $80fF/mm$  approximately in  $0.12\mu m$ . What the tool does during the parametric analysis is to modify the output node capacitance step by step according to the desired range of study, to perform the simulation, and to plot the desired delay information.

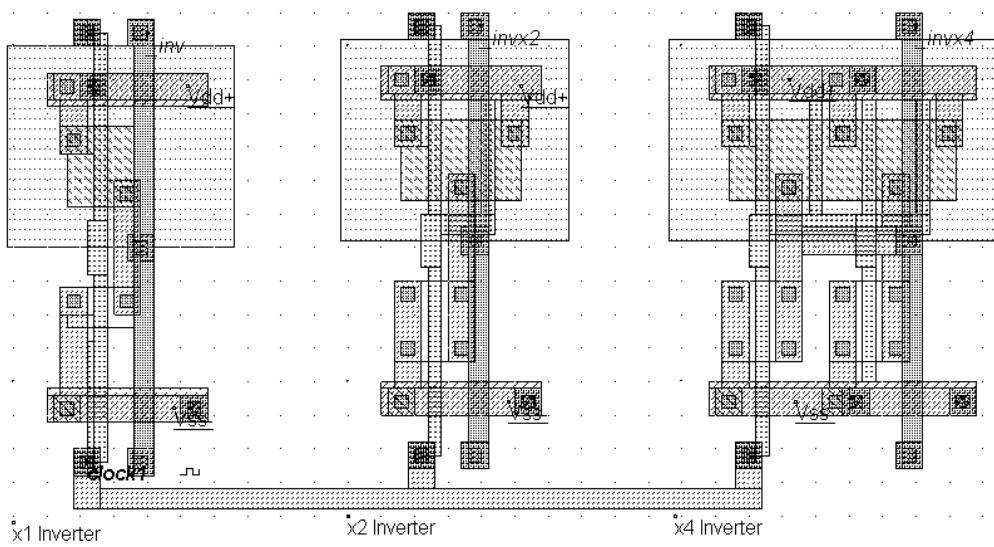


Figure 5-45: Three sizes of inverters used to investigate the delay vs. capacitance load (Invx124.MSK)

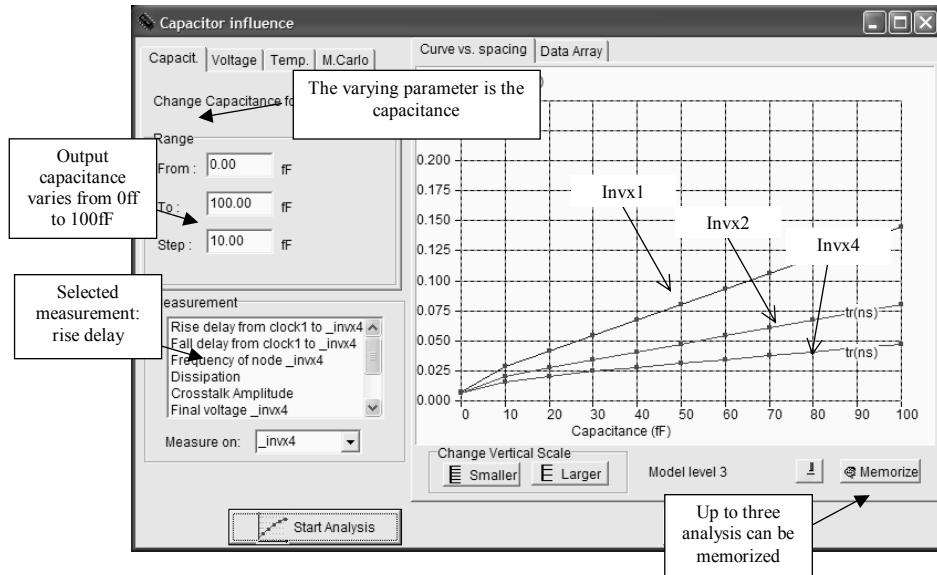


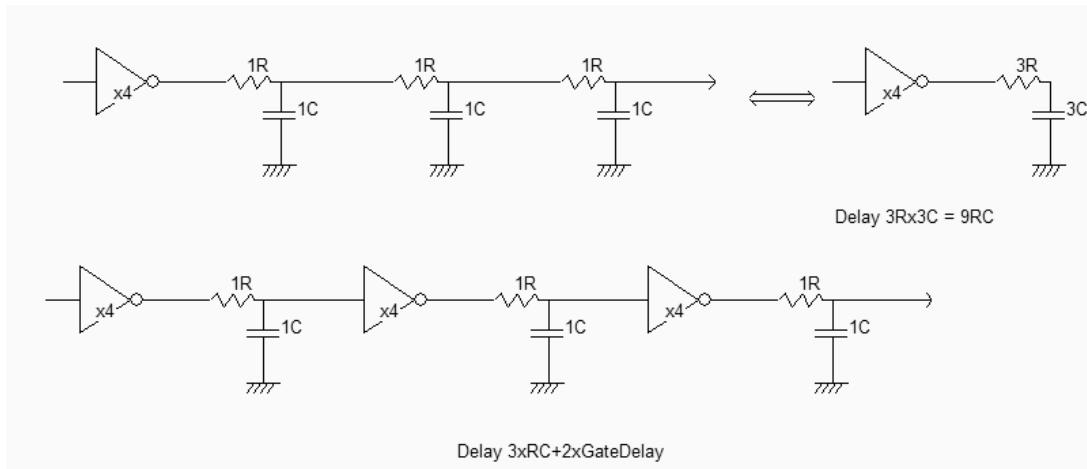
Figure 5-46: Three sizes of inverters used to investigate the delay vs. capacitance load (Invx124.MSK)

From the simulations of figure 5-46, it can be seen that a standard inverter delay (x1) increases rapidly with the capacitance. The inverter with a double drive (x2) has a switching delay divided by 2, the inverter with a quadruple drive (x4) has a switching delay divided by 4. As interconnects may be assimilated to capacitance, driving long interconnects requires large buffers. High drive buffers keep the propagation delay short, at the price of a proportionally higher current consumption. The clock signals, bus, ports and long wires with severe time constraints use such high drive circuits.

A fixed ratio is maintained between p-channel MOS width and n-Channel MOS width to balance the rise and fall time. It is much easier and safer to design the logic cells with similar rise and fall time performances, otherwise the timing analyzer would have to consider rise and fall time cases separately.

## 9. Repeaters for Improved signal transport

Long distance routing means a huge loading due to a series of RC delays, as shown in figure 5-47. A long line may be considered as a series of RC element where R is the serial resistance and C the ground capacitance. For example, one RC cell represents one millimeter of interconnect. If a very long interconnect is implemented between an emitter and a receiver inverter, the delay is increased according to  $n^2$ , where n is the number of RC cells, as given in equation 5-6.



*Fig. 5-47 : The propagation delay of a long line with one inverter can be longer than with three inverters  
(RcLines.SCH)*

In the case of a long line driven by a single inverter, the propagation delay on a line modelized by RC cells is given by:

$$t_{dly} = t_{gate} + nR.nC = t_{gate} + n^2 RC \quad (\text{Equ. 5-6})$$

The propagation delay on a long line is not linearly dependent on the number of cells  $n$ , but proportionally dependent on the square of  $n$ . A good alternative is to use repeaters, by splitting the line into several pieces. Why can this solution be a better one in terms of delay? Because the gate delay is quite small compared to the RC delay. If two repeaters are inserted, the delay becomes:

$$t_{dly} = 3t_{gate} + 3RC \quad (\text{Equ. 5-7})$$

Consequently, if the gate delay is much smaller than the RC delay, repeaters improve the switching speed performances, at the price of a higher power consumption. In the case of very long interconnects (Several mm), it is interesting to place repeaters on the path of the interconnects to limit the slowing down effect of the interconnect resistance.

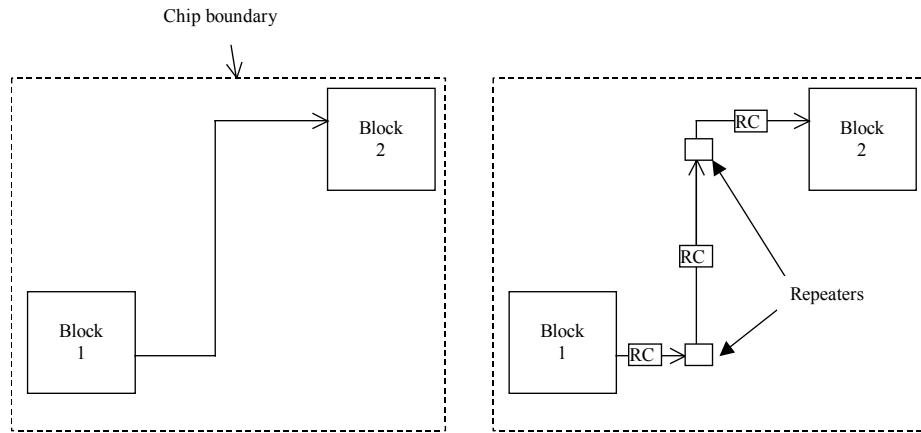


Fig. 5-48 : Inserting repeaters in long lines

The example given in figure 5-48 corresponds to the propagation of a signal from block 1 to block 2, situated at opposite corners of the integrated circuit. The associated model is a set of three RC elements. Each portion of interconnect is several millimeters long.

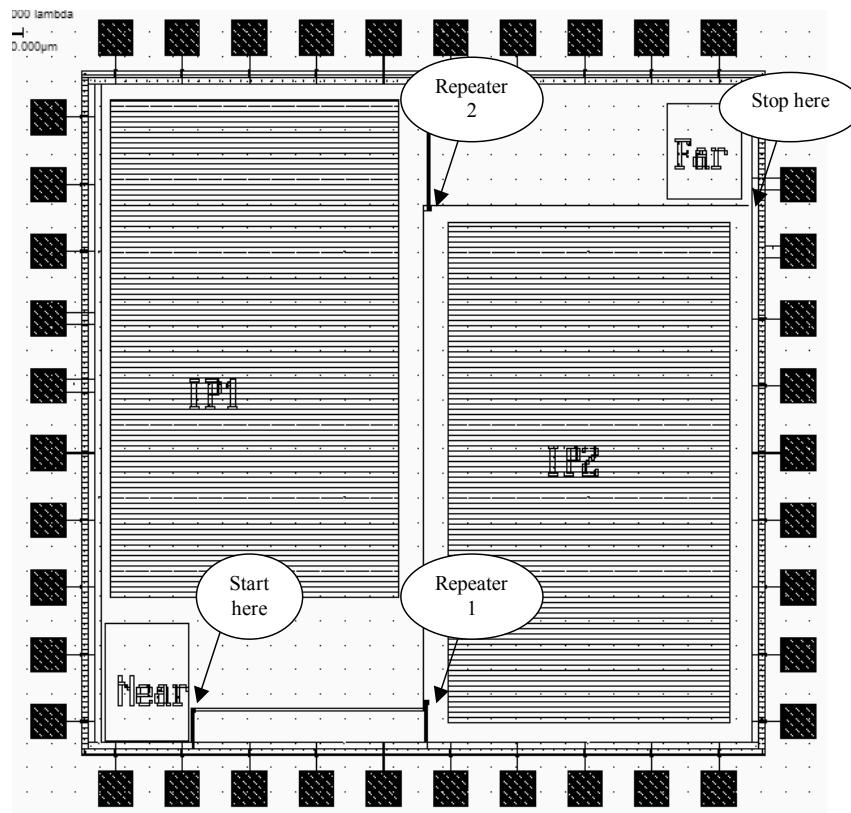


Fig. 5-49 : Propagation with and without repeaters (Repeater.MSK)

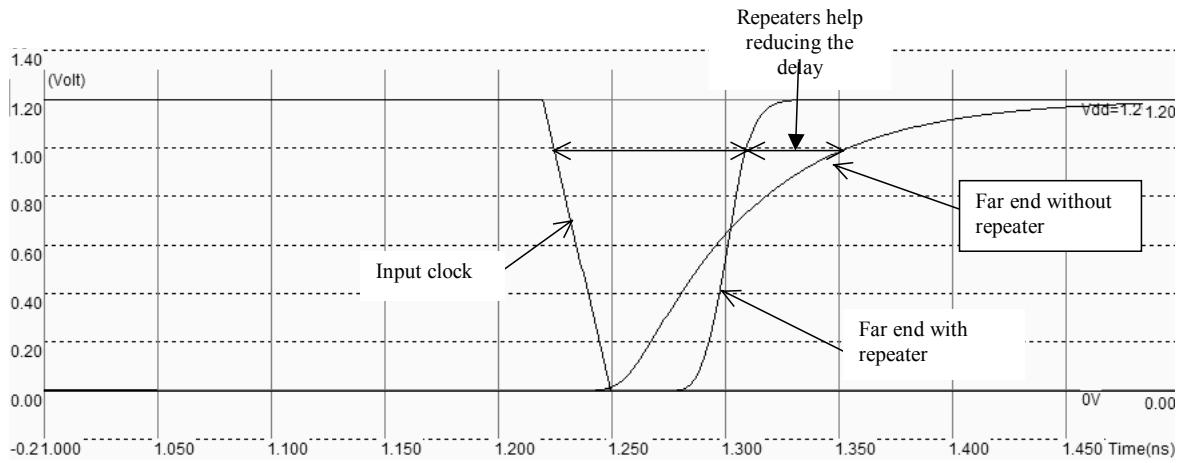


Fig. 5-50 Inserting repeaters in long lines (Repeater.MSK)

## 10. Crosstalk effects in interconnects

### Coupling Increased with scale down

The crosstalk coupling represents the parasitic transient voltage induced by a switching interconnect on a neighbor interconnect. The disturbance may be high enough to create a temporary erroneous state on an interconnect which is supposed to be constant. Over the past recent years, the crosstalk effect has been the focus of active research, in terms of modeling and technology. The main reason for this interest is illustrated in Figure 5-51. The aspect of interconnects has dramatically changed with the technology scale down.

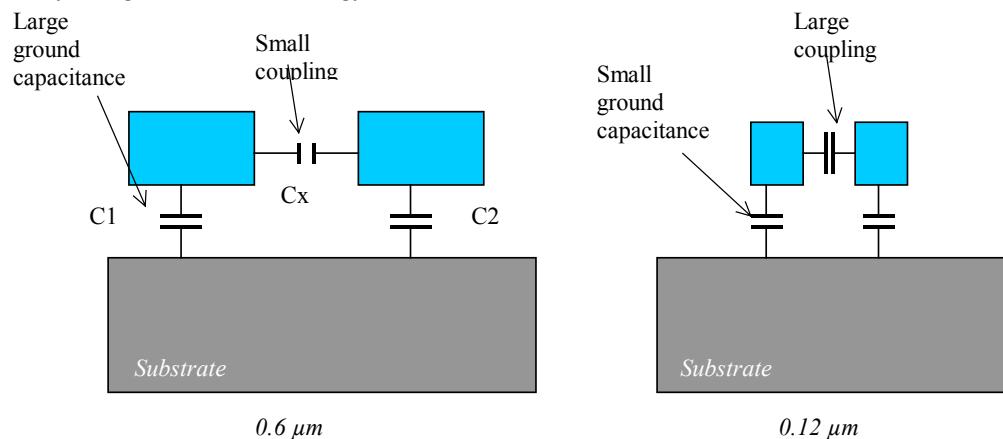
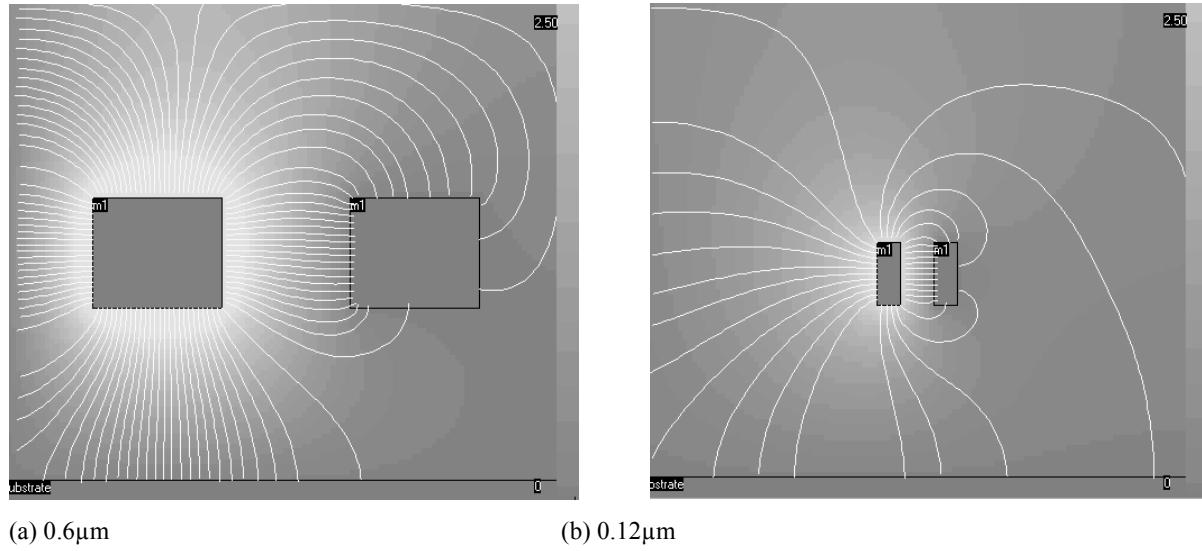


Fig. 5-51 The scale down tends to increase lateral coupling and decrease vertical coupling



*Fig. 5-52 The coupling between adjacent interconnects in 0.6 $\mu\text{m}$  and 0.12 $\mu\text{m}$  technology showing a very strong coupling increase with the scale down*

Using the command **Analysis → Interconnect analysis** with FEM, we compare the coupling effects within two conductors, between the 0.6 $\mu\text{m}$  and the 0.12 $\mu\text{m}$  technologies. The field lines are computed by a clock on **Compute Field**. In figure 5-52-a, the field lines link the left conductor mainly to ground, with about one third of the lines to the right conductor, which creates the coupling effect. In figure 5-53-b, the number of field lines have been reduced, because of reduced conductor surfaces. Only 15 field lines couple to ground, which means a very low capacitance to ground. However, the coupling field lines are still numerous and very short, which means a very high coupling

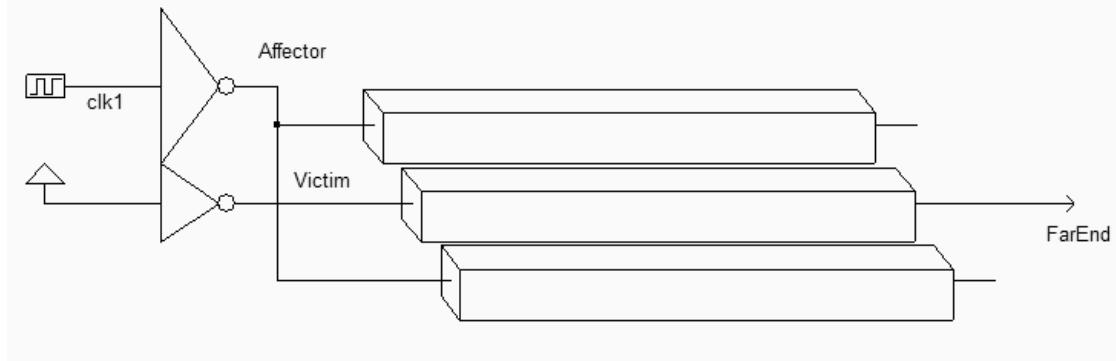
Some details about the size and electrical properties of the interconnects are reported in table 5-5. The metal width and spacing are scaled according to the lithography improvement, but the interconnect thickness is not reduced with the same trend. Starting at 0.18 $\mu\text{m}$ , copper has been proposed as an alternative to aluminum, for its lower resistivity. To decrease the crosstalk coupling capacitance, low permittivity (Low K) dielectrics have been introduced, with 0.18 $\mu\text{m}$  technology.

Technology	Metal layers	Lower metal width ( $\mu\text{m}$ )	Metal spacing ( $\mu\text{m}$ )	Thickness ( $\mu\text{m}$ )	Low K	Interconnect material	Microwind2 file
1.2 $\mu\text{m}$	2	1.8	2.4	0.8		Al	cmos12.rul
0.7 $\mu\text{m}$	2	1.2	1.6	0.7		Al	cmos07.rul
0.6 $\mu\text{m}$	3	0.75	1.0	0.7		Al	cmos06.rul
0.35 $\mu\text{m}$	5	0.6	0.8	0.7		Al	cmos035.rul
0.25 $\mu\text{m}$	6	0.5	0.6	0.6		Al	cmos025.rul
0.18 $\mu\text{m}$	6	0.3	0.4	0.5	3.1	Al, Cu	cmos018.rul
0.12 $\mu\text{m}$	6-8	0.18	0.24	0.4	2.8	Al, Cu	cmos012.rul
90nm	8-10	0.15	0.2	0.35	2.5	Cu	cmos90n.rul
70nm	8-12	0.1	0.14	0.3	2.0	Cu	cmos70n.rul

*Table 5-5: Evolution of interconnect parameters with the technology scale down*

### Simulation of the crosstalk effect

The simulation of the crosstalk effect is based on two inverters, one considered as the affecting signal, the other as the victim signal. The inverters are connected to long interconnects routed with the minimum distance. The victim is connected to a weak inverter, and surrounded by two aggressor lines connected to a very powerful inverter, to create the maximum crosstalk effect (Figure 5-53).



*Fig. 5-53 The coupling configuration used to simulate the crosstalk effect (Crosstalk.SCH)*

The serpentine shown in the layout of figure 5-54 corresponds to approximately 1mm of interconnect. Virtual resistance symbols are added in the middle of the interconnect to handle the RC effect in the simulation. Notice the unbalanced inverter size to create the worst case conditions for parasitic coupling.

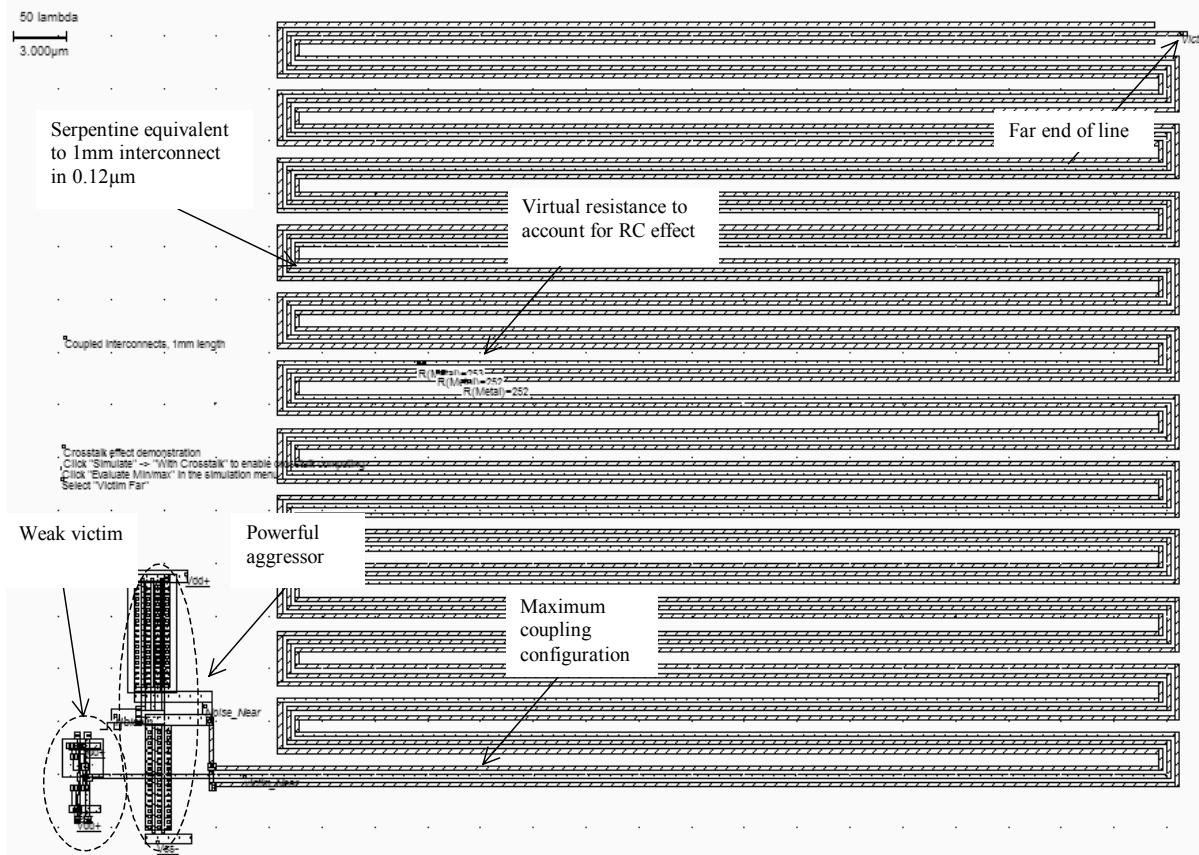


Fig. 5-54 Implementation of strongly coupled lines in worst case configuration (CrossTalk.MSK)

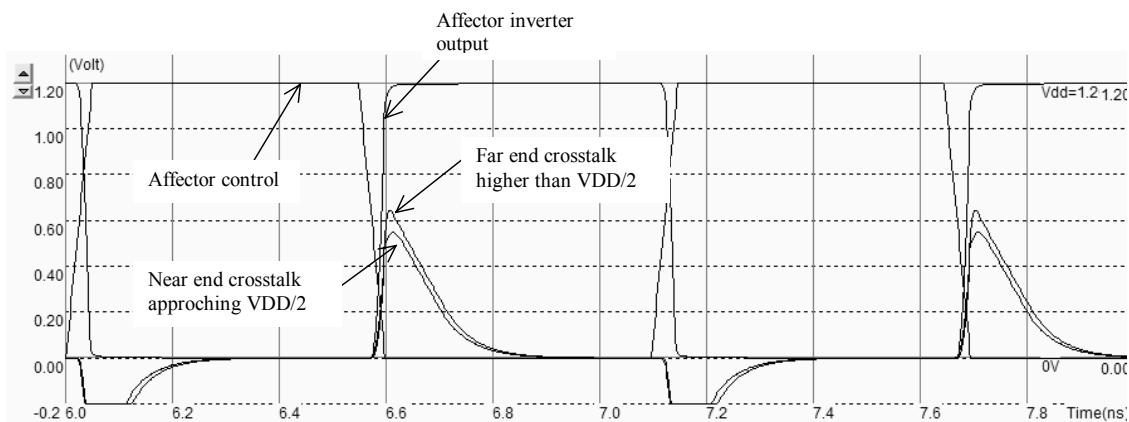


Fig. 5-54 Simulation of the crosstalk coupling in a 1mm interconnect (CrossTalk.MSK)

When the aggressor lines are switching, the coupling is strong enough to increase the voltage at the far end of the victim line, higher than the switching threshold of logic gates (Which is around  $VDD/2$ ), which may provoke a permanent logic fault (Figure 5-54). The noise is quite impressive. Remember that the line is only 1mm long, which is very common in circuit design. However, the situation where the 1mm interconnect is driven by a very low drive inverter is not usual. Nevertheless, crosstalk is very dangerous and almost uncontrollable when dealing with millions of

interconnects, as may be found in high complexity designs. One solution to avoid crosstalk is to avoid routing long interconnects. The critical routing length is the limit above which a crosstalk fault may occur. This metric has recently been introduced in design guidelines. In  $0.12\mu\text{m}$  CMOS technology, the critical routing length is 1mm. It means that interconnects longer than 1mm could suffer from crosstalk noise in worst case conditions. In the case of very long routing, repeaters should be used.

### Low K Dielectrics

When investigating the maximum crosstalk amplitude versus the technology for a given interconnect length, we observe a severe increase of the coupling effect, as a direct consequence of lithography improvements. In ultra-deep submicron technologies (Lithography lower than  $0.18\mu\text{m}$ ) the permittivity of the lateral oxide that fills the spacing between adjacent interconnects is reduced (Low K dielectric with a permittivity of around 3.0), while the oxide that separates vertical layers is kept with a high permittivity (Around 4.0 for  $\text{SiO}_2$ ).

The main effect is the decrease of lateral coupling effects. The introduction of low K materials may reduce the coupling effect up to a certain limit. Several CMOS compatible materials exist, which must be compatible with the CMOS process. Low K oxides are usually called SiOLK (For Silicon Oxide Low K<Gloss>). The SiOLK permittivity should ideally be 1, that is corresponding to an air gap between interconnects, still in a research phase. In practice, for  $0.12\mu\text{m}$  technology, SiOLK  $\epsilon_r$  is around 3.0.

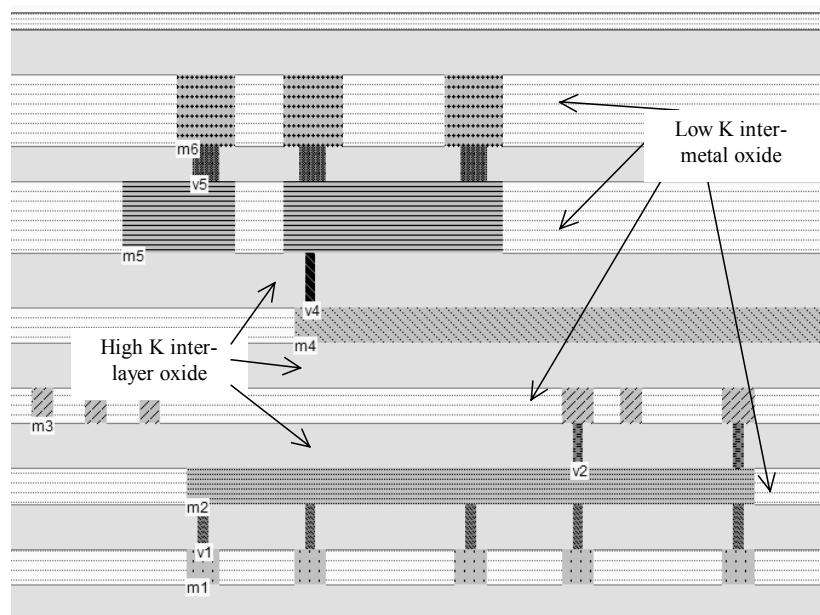


Fig. 5-55: Low dielectric permittivity between lateral metal interconnects reduces the crosstalk effect (Metals.MSK)

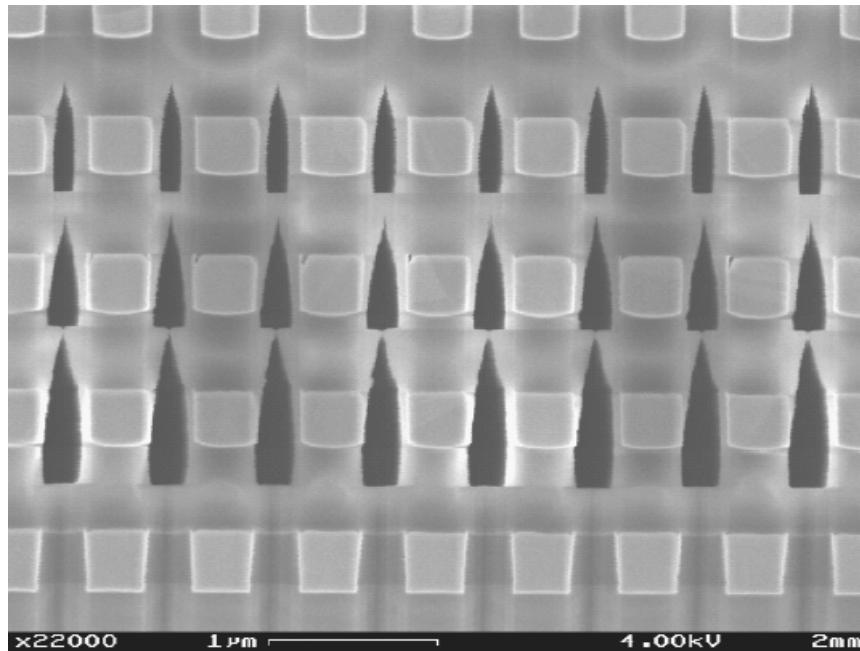


Fig. 5-56: Air-gap between interconnects to cut by half the crosstalk coupling (Courtesy ST-Microelectronics)

## 11. Antenna Effect

During the fabrication of interconnects, charges may accumulate and endanger the MOS gate by forcing current through the gate oxide [Hastings]. This effect, called antenna effect, appears on long metal interconnects connected to small gate oxide areas, without any path to diffusion. The antenna effect is particularly important in deep submicron technology: without any possible discharge path, the interconnect accumulates sufficient charges to rise its potential to several volts, positive or negative depending on the nature of the chemical process step. Usually, plasma etching charges the interconnect with electrons, corresponding to a negative charge with respect to the substrate ground voltage.

### Antenna rule

With  $0.35\mu\text{m}$  process, specific antenna design rules have been introduced. If we consider an interconnect with a length  $Li$ , a width  $Wi$ , its surface  $Ai$  is  $WixLi$ . If this interconnect is connected to a MOS device with a length  $L$  and width  $W$ , corresponding to a channel surface  $A$ , the following rule should be verified:

$$Ai \leq R_{\text{antenna}} \cdot A \quad (\text{Equ. 5-10})$$

where

$A = WxL = \text{MOS channel surface (m}^2\text{)}$

$$A_i = W_i \times L_i = \text{interconnect surface (m}^2\text{)}$$

$R_{\text{Antenna}} = \text{antenna ratio (around 100)}$

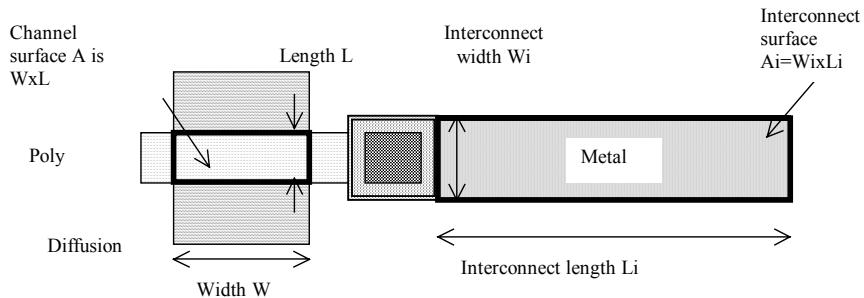


Figure 5-56: The antenna rules related to the surface of the interconnect with respect to the surface of the gate

### Design example

An example of valid and invalid interconnect design is given in figure 5-57. The upper layout complies with the antenna rules as the interconnect surface is less than 100 times the gate surface. In contrast, the design (b) is dangerous as the interconnect surface is more than 100 times larger than the gate surface, which has been designed very small. The antenna rules are not yet verified by the design rule checker of Microwind.

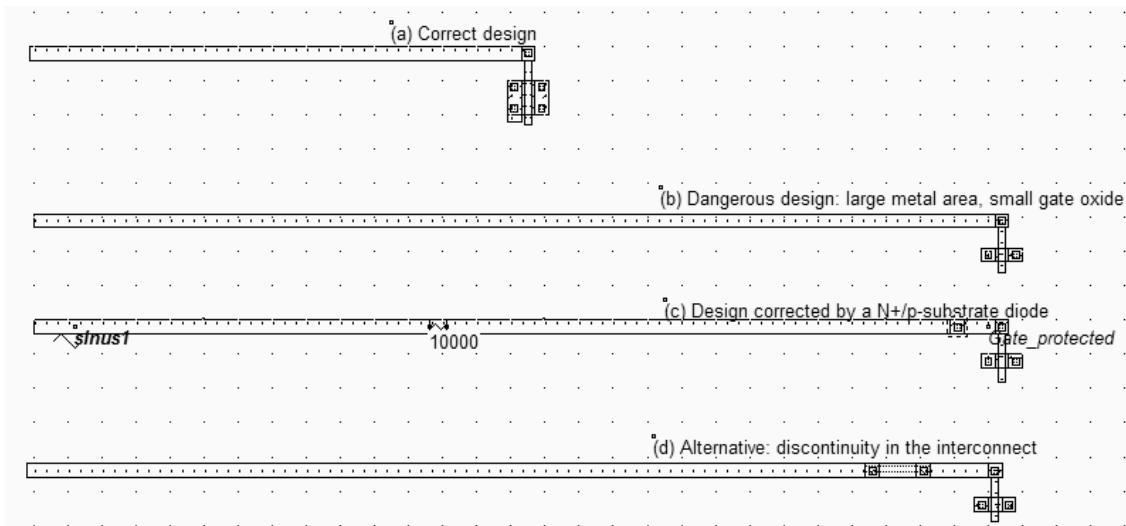


Figure 5-57 Illustration of the antenna design rule (AntennaRules.MSK)

One solution consists in creating a discontinuity by using a bridge with a higher metal, so that when the lower metal is etched, the main part of the interconnect will not be connected to the gate (Figure 5-57-d). An alternative way to avoid the antenna effect is to build a discharge path to evacuate the parasitic charges accumulated during fabrication. A

simple diode is an efficient solution (Figure 5-58). Its parasitic capacitance is quite small, and the diode has no important electrical effect on the nominal signal transport.

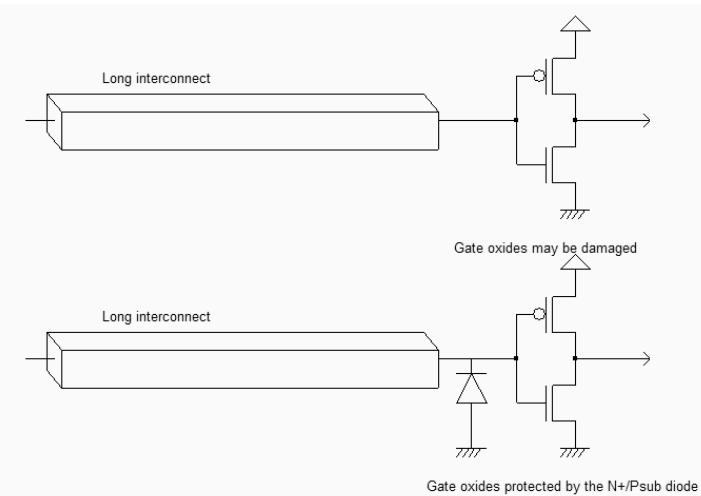


Figure 5-58 Inserting a diode to discharge the interconnect during plasma etching (AntennaRules.SCH)

## Simulation

The N+/P-substrate diode inserted near the gate (Figure 5-57-c) turns on when the interconnect voltage is negative, or higher than the reverse Zener voltage, as seen in the simulation reported in figure 5-59. Notice that the BSIM4 model has been used to handle the Zener effect. The Y axis has been changed to -2 to 24V, thanks to the Y scale cursors situated at the left upper part of the voltage chronograms. By default, Microwind does not extract diodes. This is why an option layer surrounds the N+ area near the gate, with a tick in front of **Extract Diode inside box**.

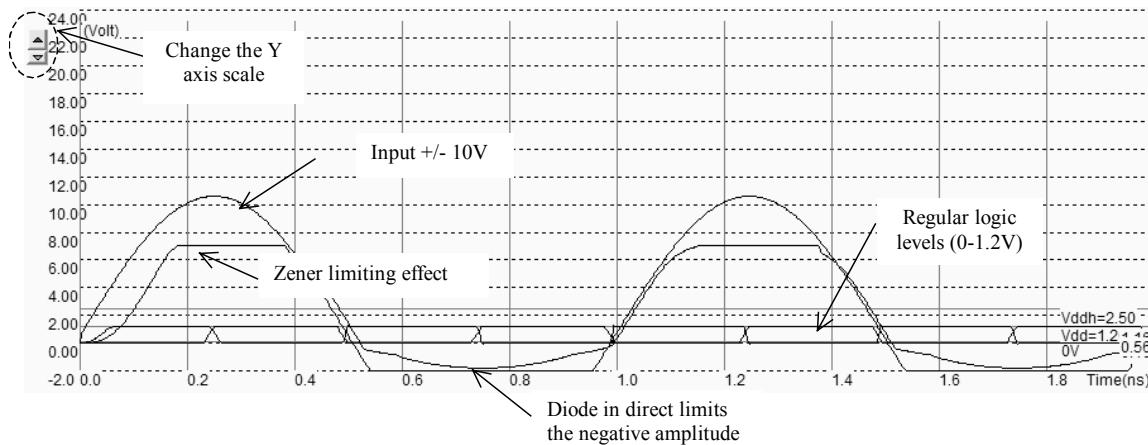


Figure 5-59 The diode clamps the negative charges, and limits the positive amplitudes (AntennaRules.MSK)

The serial resistance used for simulation is very high (10Kohm). Removing that resistance would completely hide the clamping effect of the diode. In reality, the charging of the interconnect is not equivalent to a perfect voltage source, as

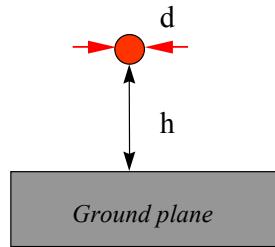
Microwind does with the sinusoidal property. The high serial resistance accounts for the weak charging process, which can be counterbalanced by a small discharge diode.

## 12. Inductance

The inductance effect is not significant in signal transport because of the high serial resistance of interconnects. This is why we have not paid a lot of attention to the inductance value and its possible consequence on the delay estimation or crosstalk amplitude. A lot of research has been dedicated in the recent years to the extraction and handling of inductance. The inductor is described in details in chapter 13, as a stand-alone passive component for application in radio-frequency circuits. In that case, the inductance is no more a parasitic effect but a voluntary effect. We give here a brief evaluation and illustration of the parasitic inductance effect in deep submicron interconnects.

### Parasitic inductance formulation

The wire inductance formulation is based on the estimation of a cylinder for which a very simple formulation exists [Lee]. A well known rule of thumb consists in approximating the serial inductance to 1nH/mm, which is close to reality in the case of bonding wires. The wire has a cylindrical shape and is situated far from the ground plane.



$$L = \frac{\mu_0}{2\pi} \ln(4 \frac{h}{d}) \quad (\text{Equ. 5-7})$$

with

$\mu_0 = 1.27e^{-6}$  H/m for most materials (Al, Cu, Si, SiO<sub>2</sub> and Si<sub>3</sub>N<sub>4</sub>)

$d$  = wire diameter (m)

$h$  = height of the wire vs. ground (m)

In the case of metal interconnects, the formulation 5-7 is adapted with an approximation of the interconnect diameter, based on the conductor width and thickness. The serial parasitic inductance of the conductor appears in the navigator menu, after extraction, together with the capacitance and resistance (Figure 5-60). A metal interconnect exhibits an inductance of around 0.5nH/mm. Notice that the lineic inductance value is also provided in the **interconnect analysis** window, accessible from the **Analysis** menu.

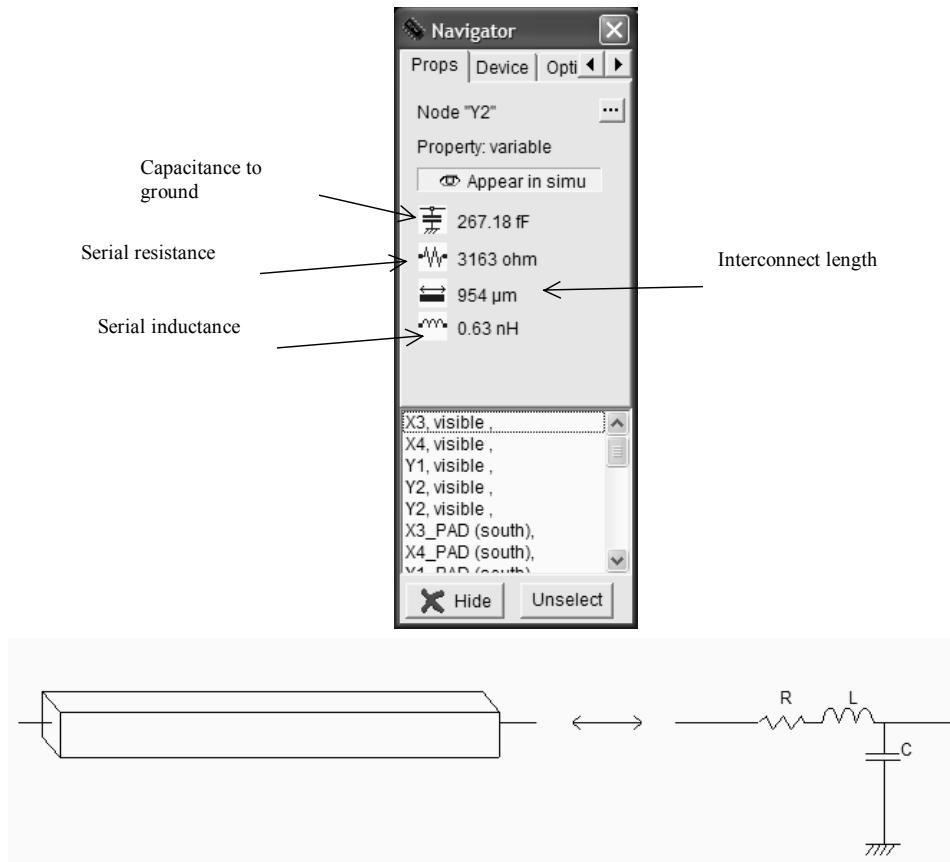


Figure 5-60 An evaluation of the parasitic serial inductance is given in the navigator menu (Rlcg.SCH)

### Simulation with/without inductance

Let us compare the signal propagation of a logic signal within a 500 $\mu$ m interconnect with and without the serial inductance. In Microwind, the inductance must be added through a virtual inductor symbol that may be found in the palette. The inductor is placed at the beginning of the line. Handling the simulation of the inductance is not simple in Microwind, which has been optimized for RC networks. Several problems rise at simulation: the initial ringing is very high, which is not realistic at all. This numerical instability is due to the initialization of the inductance, which disturbs the circuit in the early nanoseconds. Secondly, the simulation step by default is not small enough to ensure a correct simulation. In most cases, keeping the default time step of 0.3pS will create the oscillations of all floating nodes with several volts of amplitude. The only solution consists in reducing the simulation time step, at least to 0.03ps or even less. After some nanoseconds, the simulation becomes stable, as displayed in figure 5-61. We see no significant difference between the RC and RLC models.

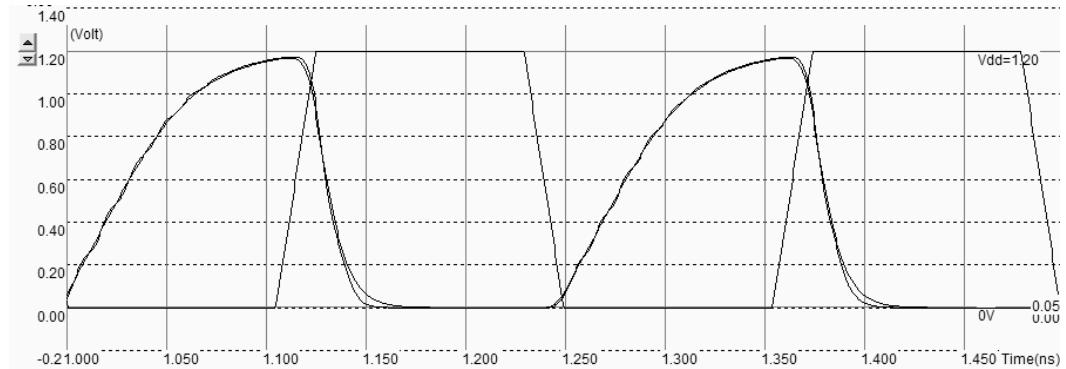


Figure 5-61 Simulation of the signal propagation within a  $500\mu\text{m}$  interconnect with using the RC and RLC models  
(InductanceInv.MSK)

To observe the inductance effect, we change the configuration and now drive the same RLC line with strong buffers, as shown in figure 5-62. Notice the ringing effect due to the combination of inductance and capacitance at the far end of the interconnect (Figure 5-63). In reality, the resistance, capacitance and inductance are distributed along the wire, which tend to limit the amplitude of the ringing effect, and fastens the damping of the oscillation to a stable logic value.

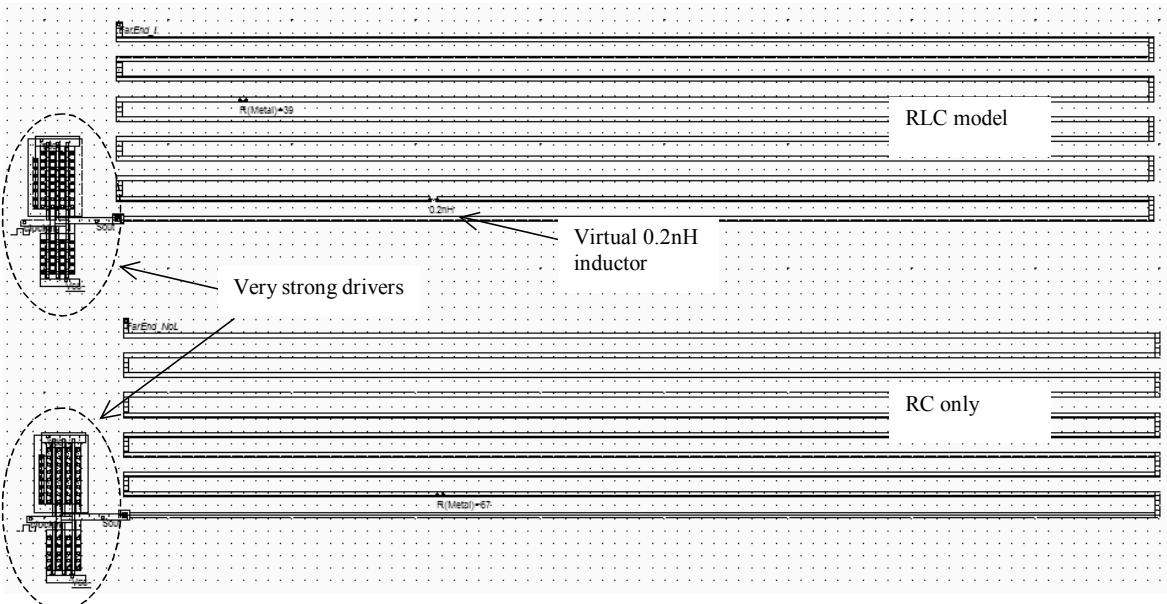


Figure 5-62 Layout of the interconnect with and without serial parasitic inductance (Inductance.MSK)

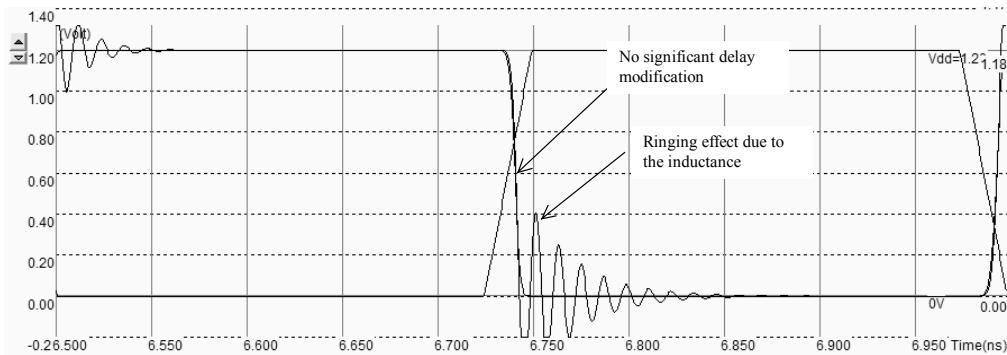


Figure 5-63 Simulation of the signal propagation with and without inductance. This result was obtained with a simulation time step decreased to 0.02ps (inductance.MSK)

Although the ringing effect is very important in this new simulation (Figure 5-63), the switching delay is not altered in a significant way. Consequently, the inductance effect may be neglected in a first order approximation in the analysis of signal propagation. This assumption is valid if the buffer strength is not too large, and the interconnect not too short.

## 13. Conclusion

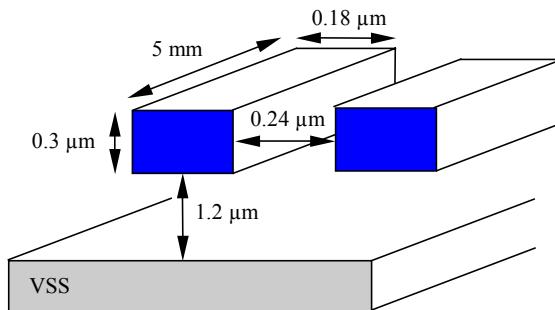
In this chapter, we have described some layout techniques for designing metal interconnects between devices. We have given some information regarding the design rules and the electrical parameters for metal interconnects, such as the resistance, capacitance, and later the inductance. The signal propagation has been analyzed from a point of view of RC delay, technology scale down, and parasitic crosstalk effect. We have also made a rapid investigation of the role of the parasitic inductance in the signal transport.

## REFERENCES

- [Weste] N. Weste, K. Eshraghian "Principles of CMOS VLSI design", Addison Wesley, ISBN 0-201-53376-6, 1993
- [Baker] R.J. Baker, H. W. Li, D.E. Boyce "CMOS circuit design, layout and simulation", IEEE Press, ISBN 0-7803-3416-7, 1998
- [Hastings] Alan Hastings "The Art of Analog Layout", Prentice Hall, ISBN 0-13-087061-7
- [Sakurai] Sakurai T. « Closed-form expressions for interconnection delay, coupling and crosstalk in VLSIs », IEEE Transactions on Electron Devices, vol 40, n°1, pp 118-124, January 1993.
- [Delorme] Delorme N., Belleville M., Chilo J. « Inductance and capacitance analytic formulas for VLSI interconnects » Electronic letters, vol 32, n°11, pp 996-997, May 1996.
- [Lee] Thomas H. Lee "The design of CMOS radio Frequency Integrated Circuis", Cambridge University Press, isbn 0-521-63922-0
- [Bendhia] Sonia Delmas-Bendhia, Fabrice Caignet, Etienne Sicard "A new method for measuring signal integrity in CMOS Ics" Paper in Microelectronics International, Volume 17, N°1, January 2000, pp 17-21.
- [ST] [www.st.com](http://www.st.com)
- [Smith] ASIC (Elmore delay)

**EXERCISES**

5.1 We consider two coupled lines in metal1, with a 5mm length, in  $0.12\mu\text{m}$  technology (Figure 5-64). What is the equivalent model of the line, in a first order approximation? Calculate the total serial resistance  $R$ , ground capacitance  $C_{\text{sub}}$  and crosstalk capacitance  $C_{12}$ , using the command **Analysis → interconnect analysis with FEM** in Microwind. What is the best technological option for interconnect/oxide material as far as signal integrity is concerned (We suppose LowK=2)?

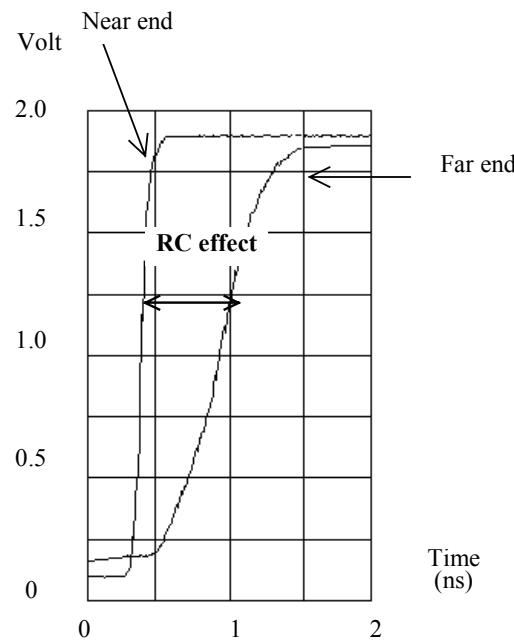


Case	$R$	$C_{12}$	$C_{\text{sub}}$
N°1: Al, SiO <sub>2</sub>			
N°2: Al, Low K			
N°3: Cu, SiO <sub>2</sub>			
N°4: Cu, Low K			

Figure 5-64: Two coupled lines in metal1, 10mm length

Answer: The model is based on coupled RC lines. The best technological option is <Sonia>

5.2 An experimental measurement of the RC effect has been realized in  $0.18\mu\text{m}$  technology (Figure 5-65). Details on the real case configuration are also provided. Create the corresponding layout for this configuration and compare the simulation with measurements.



Technology	Cmos 0.18μm
RUL file	Cmos018.RUL
Interconnect length	10mm
Interconnect width	4 lambda (0.4μm)
Metal layer	Metal 5
NMOS buffer size	W=32μm, L=0.18μm
PMOS buffer size	W=54μm, L=0.18μm

Figure 5-65: The parameters of the RC propagation test-case in 0.18μm technology

Answer: See Appendix F

# 6

## Basic Gates

The basic logic gates are described in this chapter. The principles for building combinational logic circuits are developed. Details on layout implementation are also provided.

### 1. Introduction

Table 6-1 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=~in;
AND		Out=a&b;
NAND		Out=~(a.b);
OR		Out=(a b);
NOR		Out=~(a b);
XOR		Out=a^b;
XNOR		Out=~(a^b);

Table 6-1. The list of basic gates

### 2. Combinational logic

The construction of logic gates is based on MOS devices connected in series and in parallel. If two n-channel MOS switches are connected in series (Figure 6-1), the resulting switch connects ports C1 and C2 if both gates A and B are set to '1'. This yields an AND operator, represented by the symbol '&' in equation 6-1, where C12 is the logical variable which represents the connection between C1 and C2.

$$C_{12} = A \& B \quad (6-1)$$

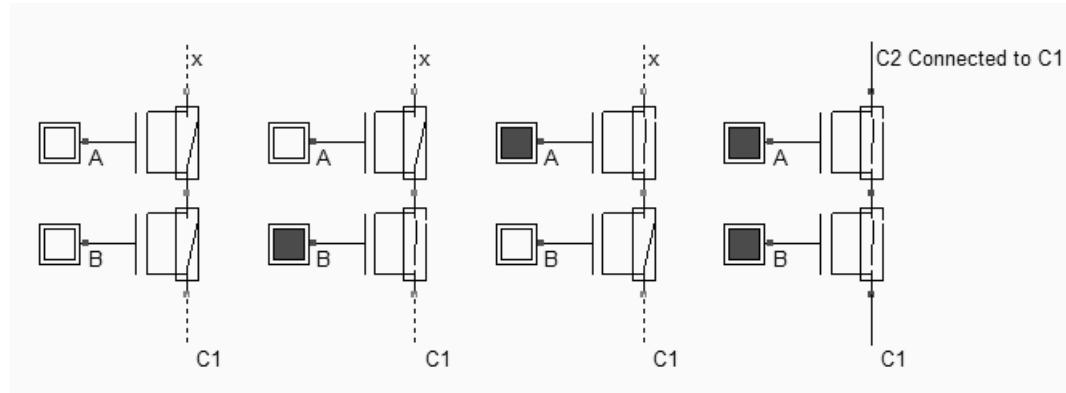


Fig. 6-1. Connecting N-channel devices in series creates a path between C1 and C2 when A and B are set to '1'  
(BaseCmos.SCH)

When two nMOS switches are connected in parallel (Figure 6-2), the resulting switch is ON if either gates A and B are set to '1'. This yields an OR operator (described as '|' in equation 6-2).

$$C_{12} = A | B \quad (\text{Equ. 6-2})$$

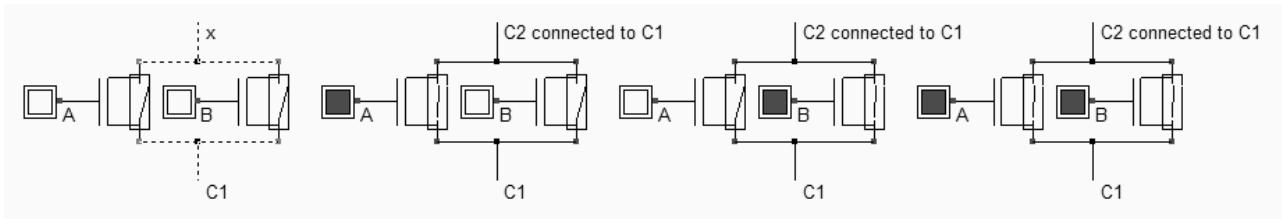


Fig. 6-2. Connecting N-channel devices in parallel creates a path between C1 and C2 when either A or B are set to '1'  
(BaseCmos.SCH)

Considering p-channel devices, we observe that two pMOS switches connected in series (Figure 6-3) behave as an AND between negative logic values: the resulting switch is ON if both gates A and B are set to '0'. The corresponding Boolean operator is as follows.

$$C_{12} = \overline{A} \& \overline{B} \quad (\text{Equ. 6-3})$$

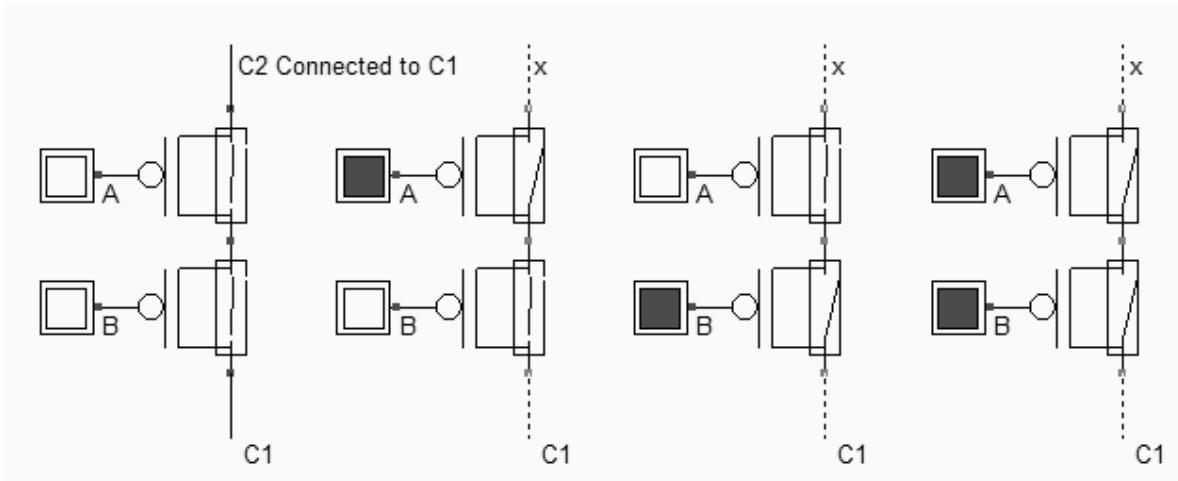


Fig. 6-4. Connecting P-channel devices in series creates a path between C1 and C2 when A and B are set to '0'  
(BaseCmos.SCH)

When two pMOS switches are connected in parallel (Figure 6-5), the resulting switch is ON if either gates A and B are set to '0'. The Boolean operator is described by equation 6-4.

$$C_{12} = \overline{A} \mid \overline{B} \quad (\text{Equ. 6-4})$$

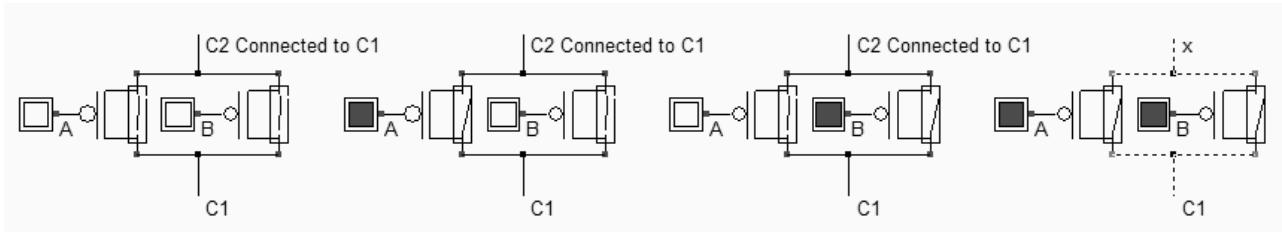


Fig. 6-5. Connecting P-channel devices in parallel creates a path between C1 and C2 when either A or B are set to '0'  
(BaseCmos.SCH)

### 3. CMOS logic gate concept

The structure of a CMOS logic gate is based on complementary networks of n-channel and p-channel MOS circuits. Remember that the pMOS switch is good at passing logic signal '1', while nMOS switches are good at passing logic signal '0'. The operation of the gate has two main configurations:

- the nMOS switch network is closed, the output s=0 (figure 6-6 left)
- the pMOS switch network is closed, the output s=1 (figure 6-6 right)

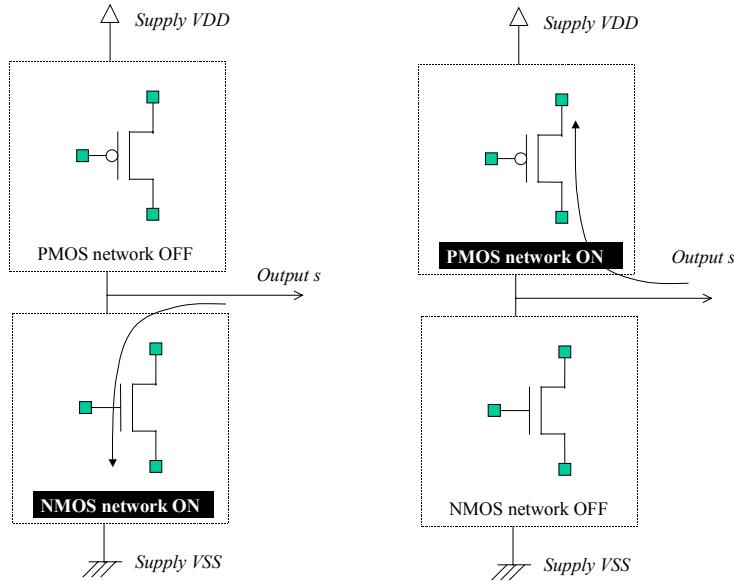


Fig. 6-6. General structure of a CMOS basic gate

Using complementary pairs of nMOS and pMOS devices, either the lower nMOS network is active, which ties the output to the ground, either the upper pMOS network is active, which ties the output to VDD. In conventional CMOS basic gates, there should exist no combination when both nMOS and pMOS networks are ON. If this case happened, a resistive path would be created between VDD and VSS supply rails. The situation where neither nMOS and pMOS networks are OFF should also be avoided, because the output would be undetermined. These illegal situations are illustrated in figure 6-7.

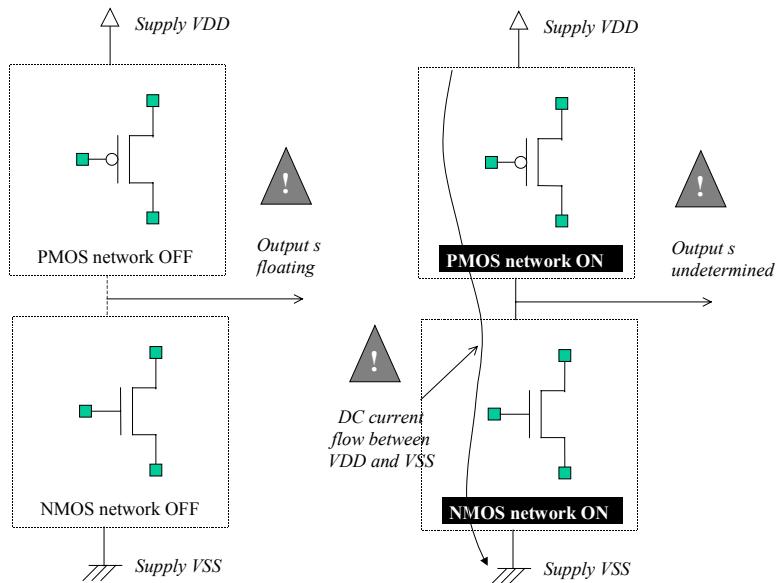


Fig. 6-7. Network configurations to be avoided: both OFF (left) and both ON (right).

## 4. The Nand Gate

**Truth-table**

The truth-table and logic symbol of the NAND gate with 2 inputs are shown below. The truth tables use 0 for logic level zero (usually 0V), 1 for logic level 1, (also called VDD, equal to 1.2V in 0.12µm technology) and x for unknown value. Some books [Uyemura] also include the z state, that is the high impedance value. We shall make no difference between the unknown and high impedance state in this book.

A B	Out
0 0	1
0 1	1
1 0	1
1 1	0
x 0	1
x 1	x
0 x	1
1 x	x

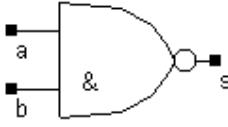


Figure 6-8. The truth table and symbol of the NAND gate

In DSCH2, the NAND gate is part of the symbol palette, which appears at initialization on the right side of the main window. Select the NAND symbol in the palette, keep the mouse pressed and drag the shape to the editing window at the desired location. Also add two buttons and one lamp as shown in figure 6-9. Add interconnects if necessary to link the button and lamps to the cell pins. The icon **Add a Line** is available for this purpose. Notice that the right click with the mouse enables the editing of an interconnect too. You may verify the logic behavior of the cell by a click on **Simulate → Start Simulation** or the icon **Run Simulation**. The logic level '0' corresponds to a white color, or to dot lines, and the logic level '1' is drawn in black, or with solid lines.

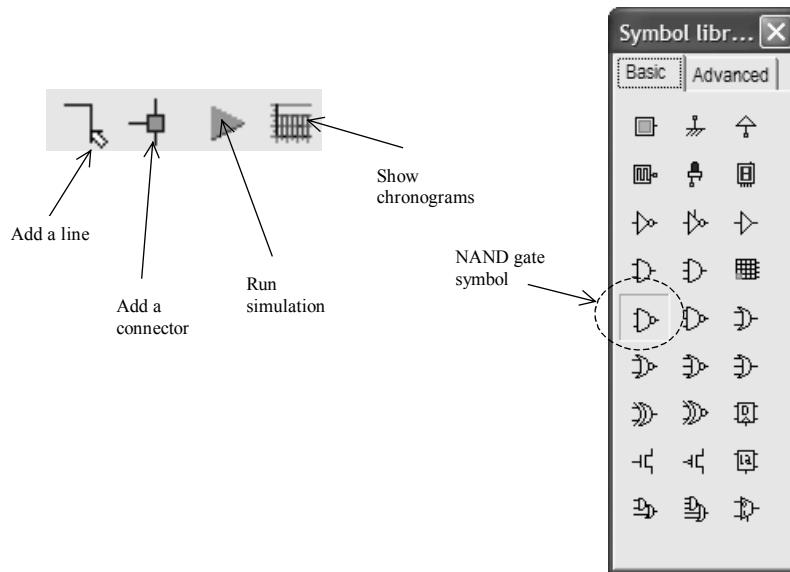


Figure 6-9. Editing commands for simulating the NAND gate

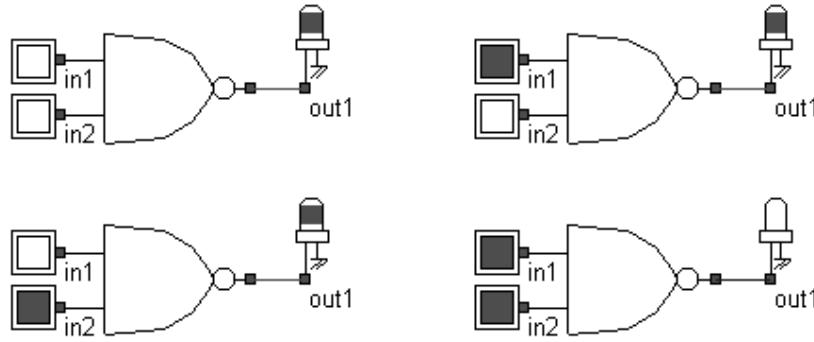


Figure 6-10. The logic simulation of the NAND gate verifies the truth table (NandTruthTable.SCH)

### Logic Design of the CMOS 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 CMOS NAND cell is reported below. The nMOS devices in series tie the output to the ground for one single combination A=1 and B=1. For the three other combinations, the nMOS path is cut, but at 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 let “0” pass, the pMOS let “1” pass.

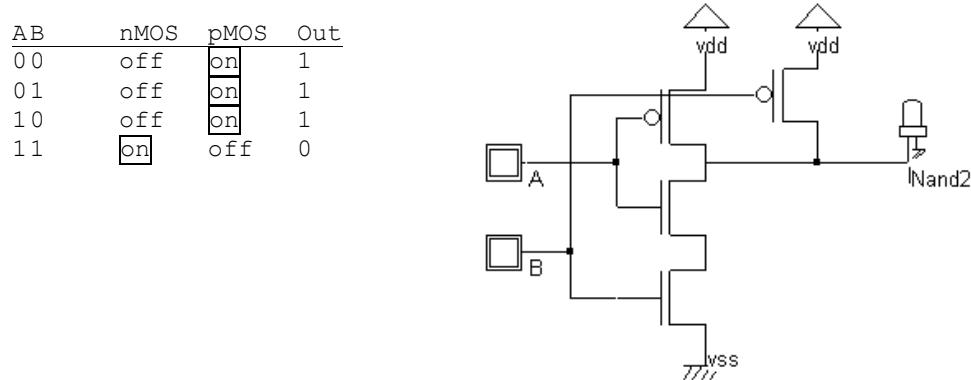


Figure 6-11. Schematic diagram of the CMOS NAND gate (NandCmos.SCH)

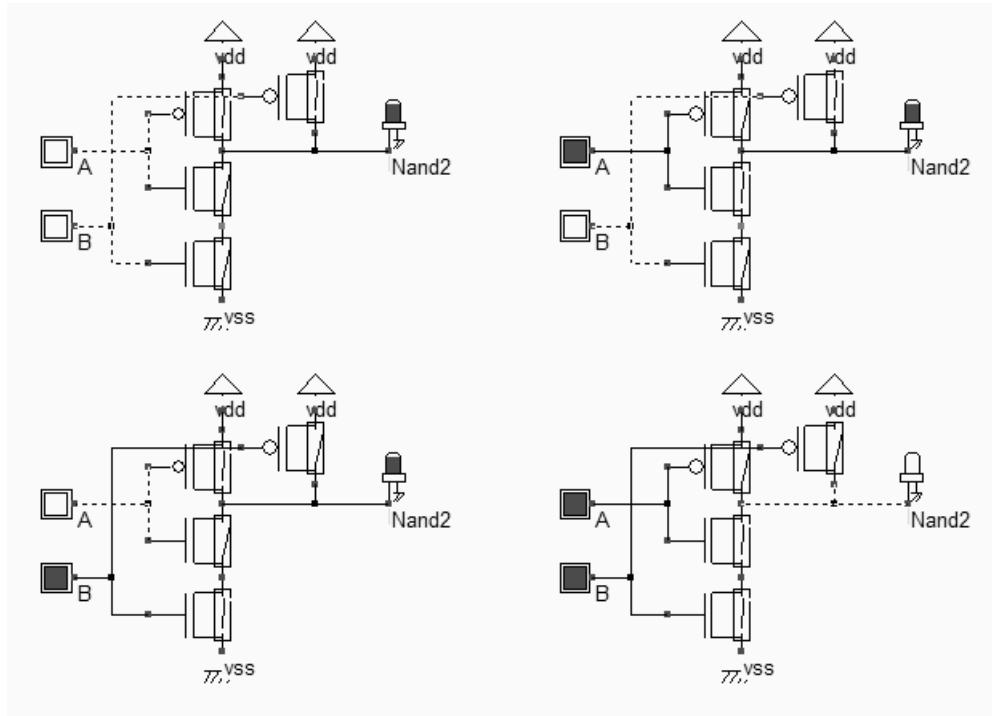


Figure 6-12. The logic simulation of the NAND gate (Nand2Cmos.SCH)

Furthermore, the circuit of figure 6-12 eliminates the static power consumption when A and B are steady by ensuring that the situation "PMOS ON", "NMOS ON" never happens.

### Automatic Generation of the NAND layout

Microwind features a built-in cell compiler that can generate the NAND gate automatically. In Microwind2, click on **Compile→Compile One Line**. Either select the line corresponding to the 2-input NAND description (Figure 6-13) or type the logical expression that describes the link between one output and two inputs. The ' $\sim$ ' operator represents the NOT operator, the ' $\&$ ' symbol represents the AND operator.

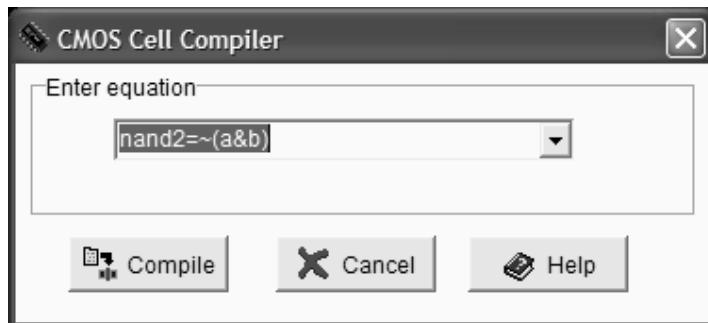


Figure 6-13. The CMOS cell compiler is used to generate a NAND gate

When you click “Compile”, the layout of the NAND gate appears in the screen, as drawn in figure 6-14. The compiler has fixed the position of the VDD power supply in the upper part of the layout, near the p-channel MOS devices. The compiler has also fixed the ground VSS in the lower part of the window, near the n-channel MOS devices. The texts *A*, *B*, are placed in the layout, on the gates, and the text *nand2* is fixed on the output, at the upper location, near the contact. The layout generation is driven by the default design rules, corresponding in this case to a CMOS 0.12 $\mu$ m technology. Depending on the design rules and the technology generation the layout may look a little different. The implantation of the four devices is detailed in the schematic diagram at the right side of figure 6-14. The MOS devices have been arranged in such a way that the diffusion regions are merged, both for the two nMOS in series and the two pMOS in parallel. Sharing a common diffusion always leads to more compact designs, which saves silicon area and minimizes parasitic capacitance.

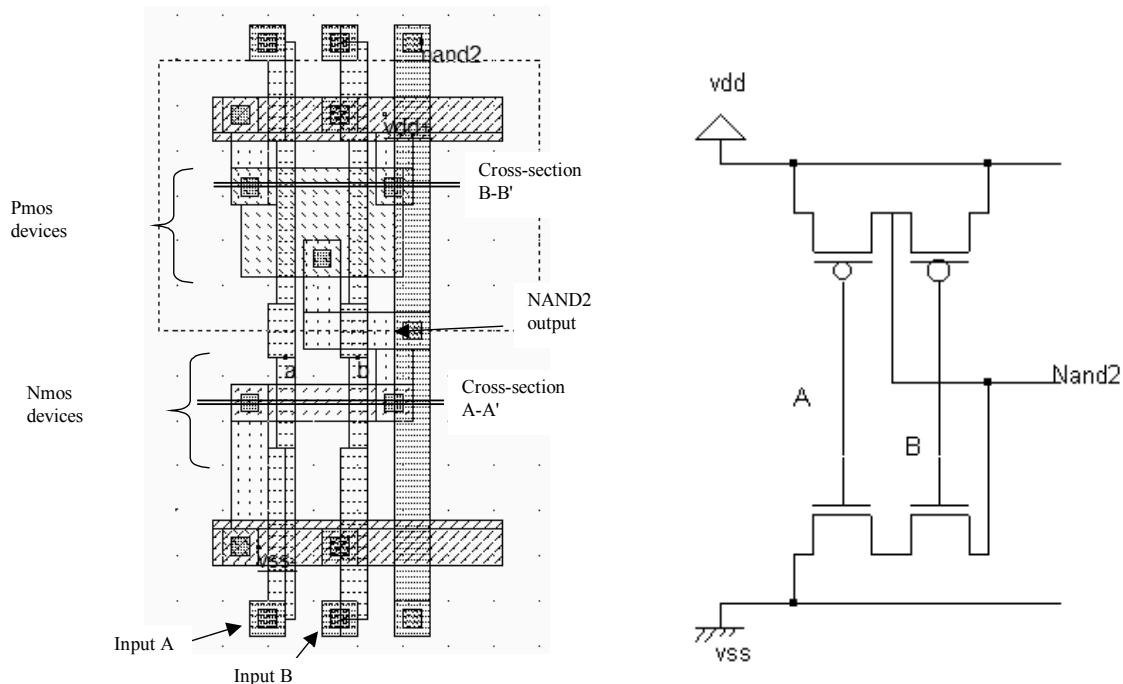


Figure 6-14. The layout of the NAND gate generated by the CMOS cell compiler (Nand2.MSK)

### Inside the Nand gate



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 a horizontal line in the layout in the middle of the nMOS channels, at location A-A' shown in figure 6-14. The figure below appears. The path from vss to the output *nand2* goes through two transistors connected in series, one controlled by *A*, the other controlled by *B*. In Figure 6-15, the output **nand2** may be tied to VDD either through the pMOS device controlled by *A* or through the pMOS device controlled by *B*. Notice the n-well under the pMOS devices, polarized to VDD.

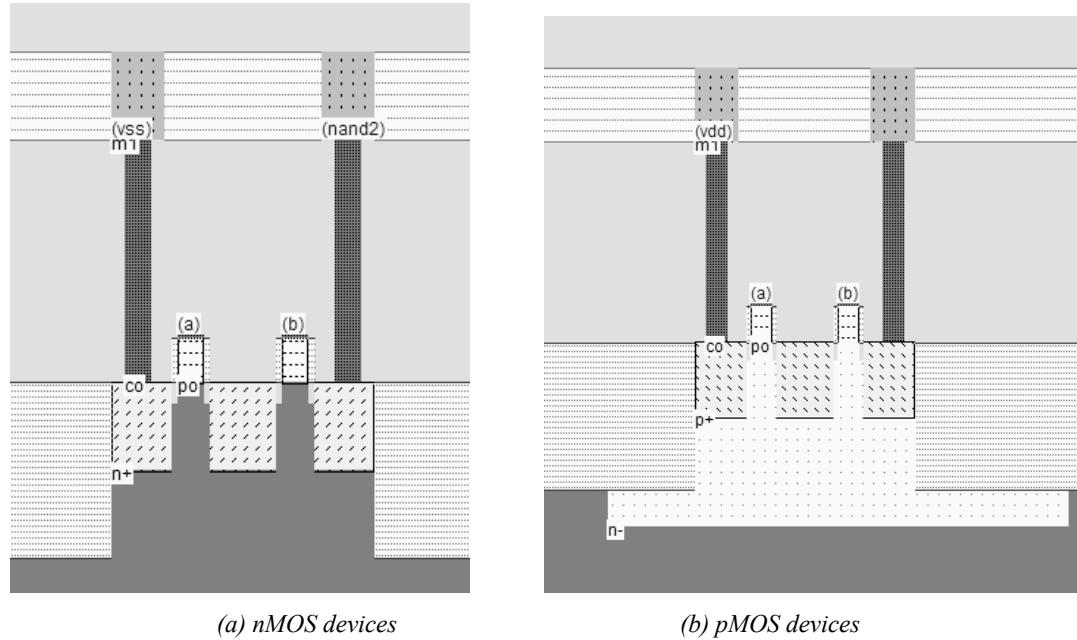
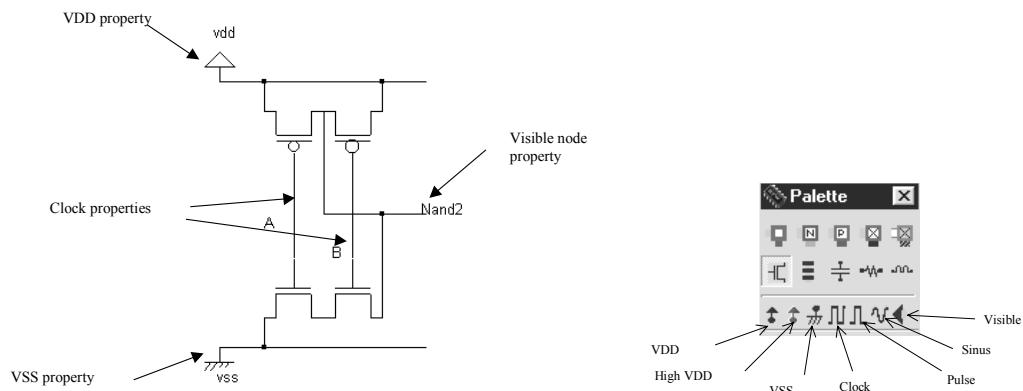


Fig. 6-15. The nMOS devices in series and the pMOS devices in parallel (Nand2.MSK)

## Adding Simulation properties

The simulation icons add properties to the nodes. Properties are applied to the electric nodes of the circuit in order to serve as simulation guides. The list of properties required to perform the analog simulation of the NAND gate are shown in figure 6-16. First of all, the NAND gate should be supplied by a 0V (VSS) and 1.2V (VDD). The VDD and VSS properties are already placed in the layout by the CMOS cell compiler.



*Fig. 6-16. Adding simulation properties to simulate the NAND gate (Nand2.MSK)*

Secondly, clocks should be assigned to the input gates  $A$  and  $B$ . The clock icon is the fourth icon from the right in the palette menu. Simply activate the clock icon, and click in the letter  $a$  in the layout window. The following screen appears (Figure 6-17).

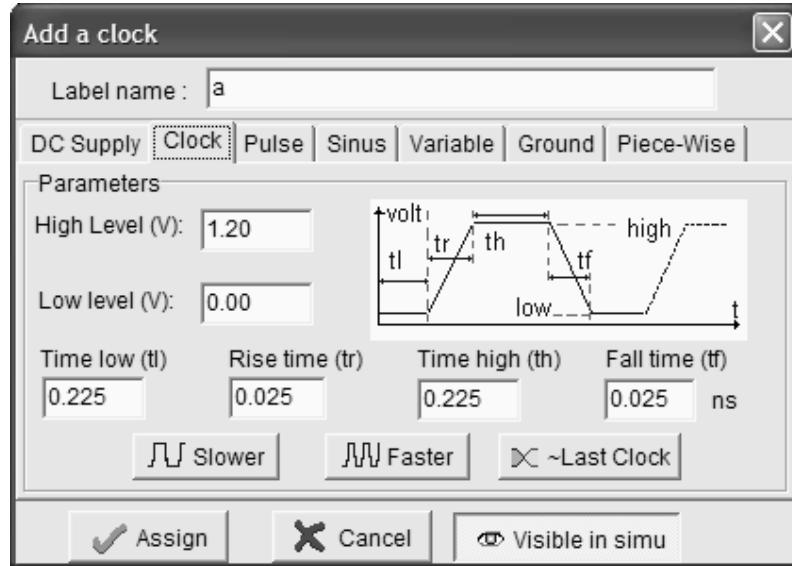


Fig. 6-17 Clock property added to node A (Nand2.MSK)

The parameters of the clock are divided as follows : time at low level (*tl*), rise time(*tr*), time at high level (*th*) and fall time(*tf*). All values are expressed in nanosecond (ns). Clock **Assign** to assign a clock to label *a*. Click again the clock icon, and this time click on label *b* in the layout. As you ask for a second clock, the period is automatically multiplied by two.

- ◆ You may alter level 0 and level 1 by entering a new value with the keyboard.
- ◆ To generate a clock which works in opposite phase, click **~Last Clock**.
- ◆ Use **Slower** to multiply the clock period by two.
- ◆ Use **Faster** to divide the clock period by two.

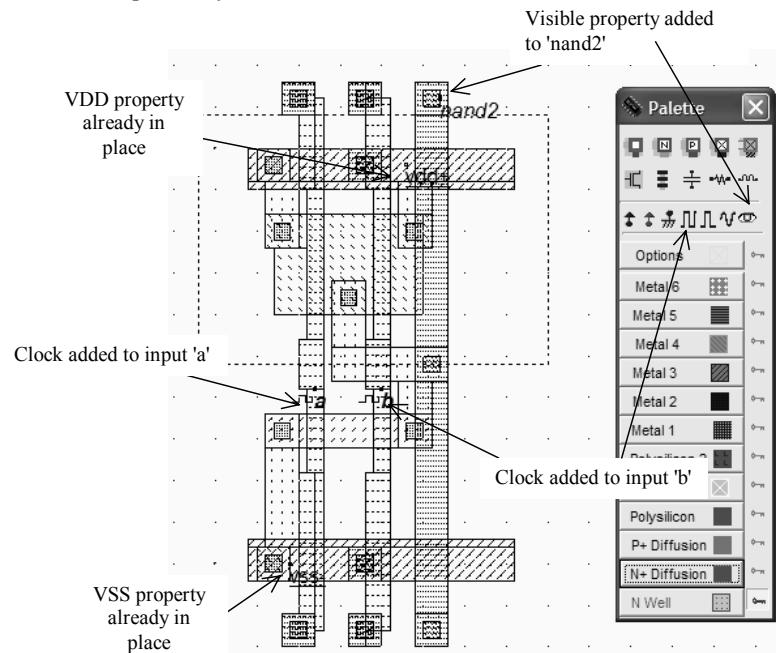


Fig. 6-18 Simulation properties added to the NAND gate (Nand2.MSK)

Finally, click on the “eye” in the palette, and click on the text *nand2* in the layout to make the chronograms of the node appear. Initially, all nodes are invisible. However, the nodes with clocks, impulse and sinus properties are subsequently made visible.

### Electrical structure of the Nand Gate



The icon above is useful to get an insight of the electrical node structure of the layout. Select the icon and simply click inside the layout at the desired location. Information corresponding to the parasitic capacitance, resistance, as well as simulation properties are also displayed in a separate window, called navigator. In figure 6-19, the ground node and supply node are illustrated. In the Navigator window, the electrical properties of the node are displayed.

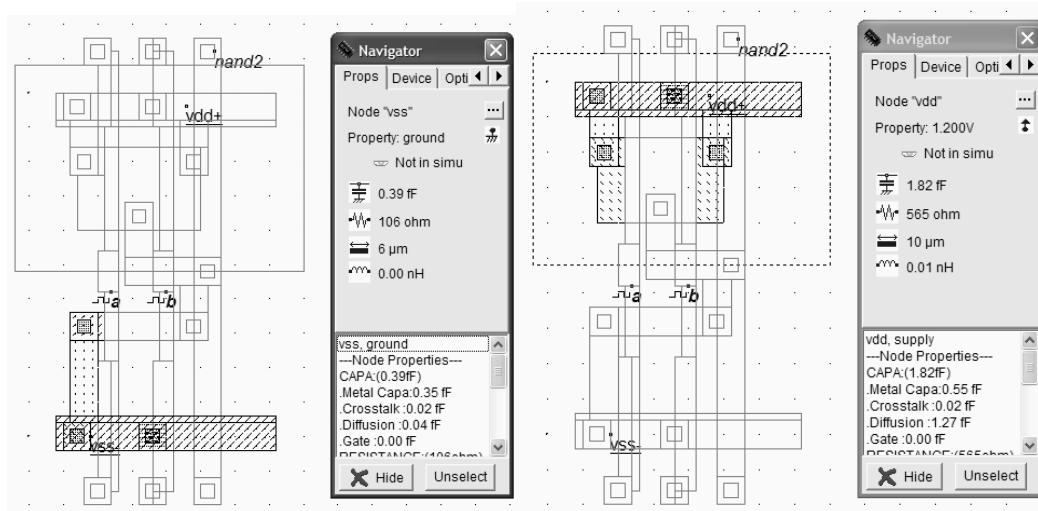


Fig. 6-19. The VSS node and VDD node with their associated properties (Nand2.MSK)

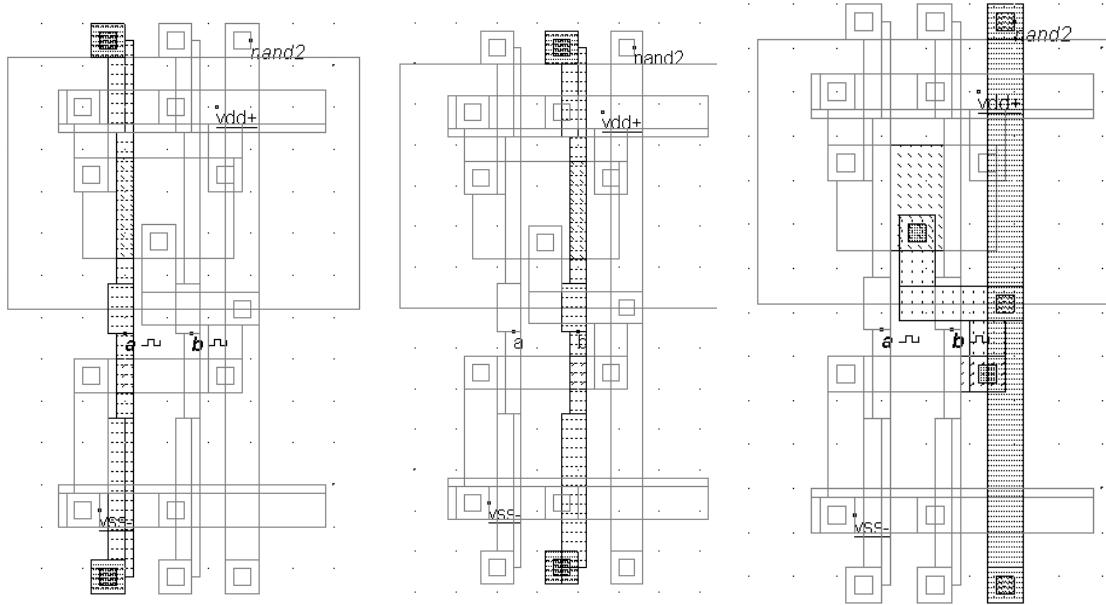


Fig. 6-20. The structure of the input nodes A,B and the output node nand2 (Nand2.MSK)

The electrical connection of the inputs a,b and the output nand2 is revealed in figure 6-20. The inputs correspond to the nMOS gates, to a small portion of polysilicon between the pMOS and nMOS areas, and to the pMOS gates. The contacts situated on the lower and upper parts of the cell serve as a simple connection point to upper metal layers. The output node is quite complex. It consists of a drain area in the n-channel MOS region, connected to the central part of the pMOS diffusion area. The output is also connected to routing contacts up and down, thanks to a vertical metal bar in metal2.

### Analog Simulation of the NAND gate

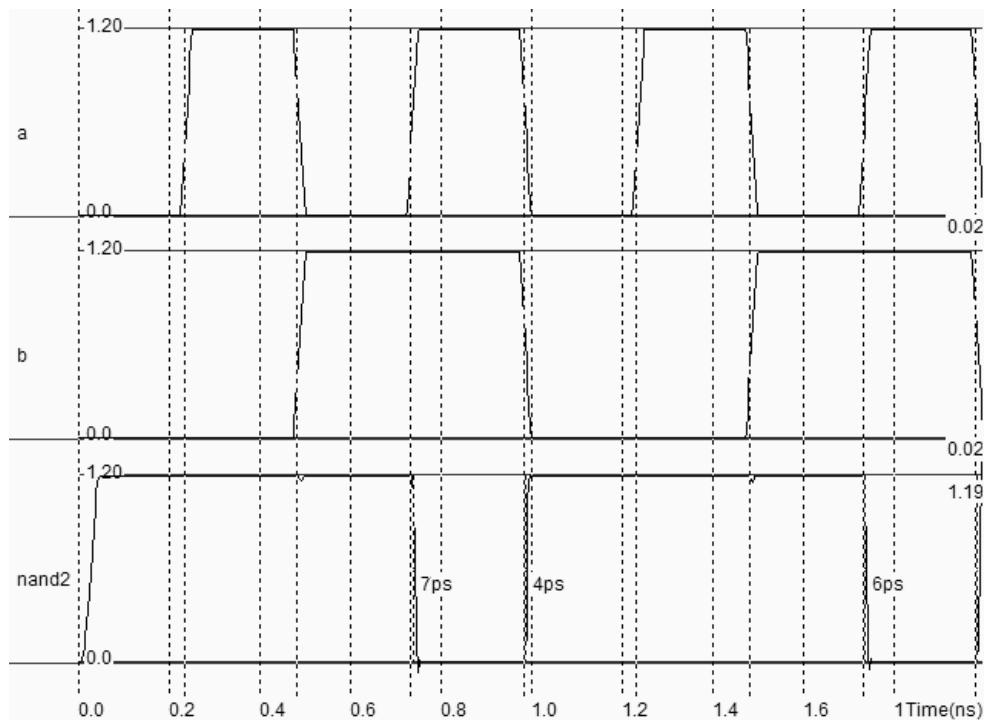


Figure 6-21. Simulation of the NAND gate (NAND2.MSK)

The simulation of figure 6-21 is obtained by the command **Simulate → Run simulation**, or the above icon. We verify that *nand2* is equal to 0 when both *a*=1 and *b*=1, according to the truth-table, otherwise the output is at 1. The rise time and fall time are computed according to the following scenario. The simulator starts computing the delay when the selected signal, chosen here as *a*, crosses VDD/2. The simulator stops when the output *nand2* also crosses VDD/2 (Figure 6-22). We should not emphasize too much the extremely small switching delay (7ps for the fall edge, 4ps for the rise edge). There are two reasons for such a high speed: one is the delay computation mode, based on VDD/2 which is always optimistic compared to a 10%-90% delay evaluation, the second is the absence of any output load, which gives a best-case estimation of the switching delay.

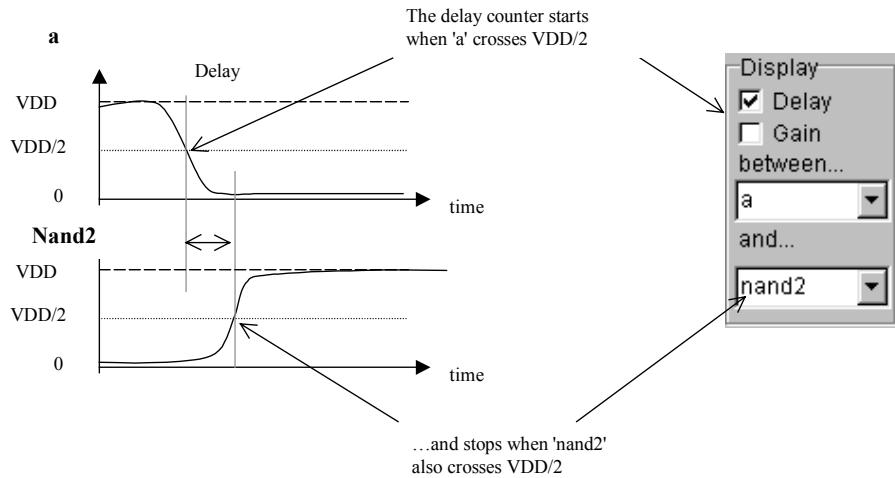


Figure 6-22. Delay computation in the simulation menu

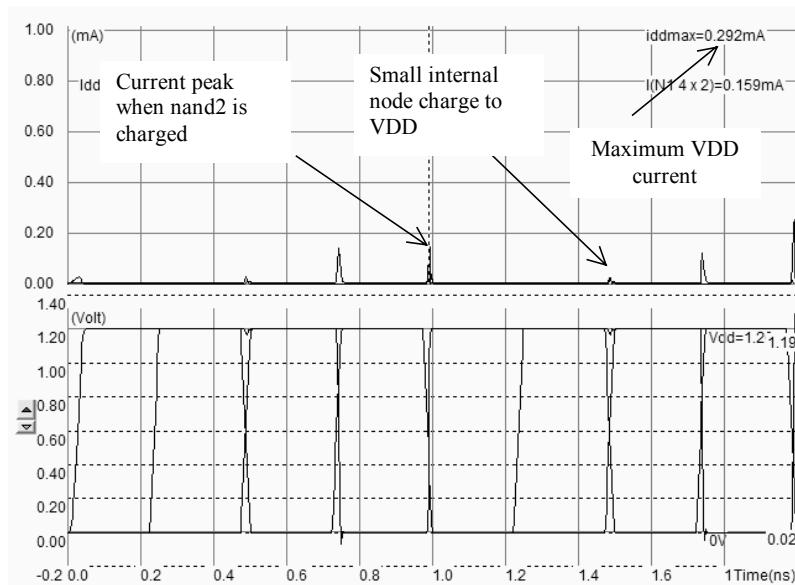


Figure 6-23. Current consumption on VDD supply line vs. time (Nand2.MSK)

Let us consider now another simulation mode, Voltage and Currents, accessible through the main menu via the command **Simulate → Run Simulation → Current, Voltage vs. Time**. The simulator displays all voltages in the lower window, and a selection of currents in the upper window (The IDD current, which is the sum of currents flowing from the supply VDD, and the current of one selected MOS device). The ISS currents can also be displayed. A transient current peak appears on IDD when the output node *nand2* is charged, as shown in figure 6-23. The current consumption is important only during a very short period corresponding to the charge of the output node.

Without any switching activity, the current is very small and cannot be seen accurately in linear scale. When looking at the same diagram in logarithmic scale (Assert **Scale I in log** in the simulator parameter window), we observe that the NAND gate consumes around 1nA of standby current. The default simulation model is Model 3, which does not

account accurately for leakage currents. The BSIM4 model, accessible by the command **Simulate → Using Model → BSIM4**, configures the simulation with BSIM4 model parameters.

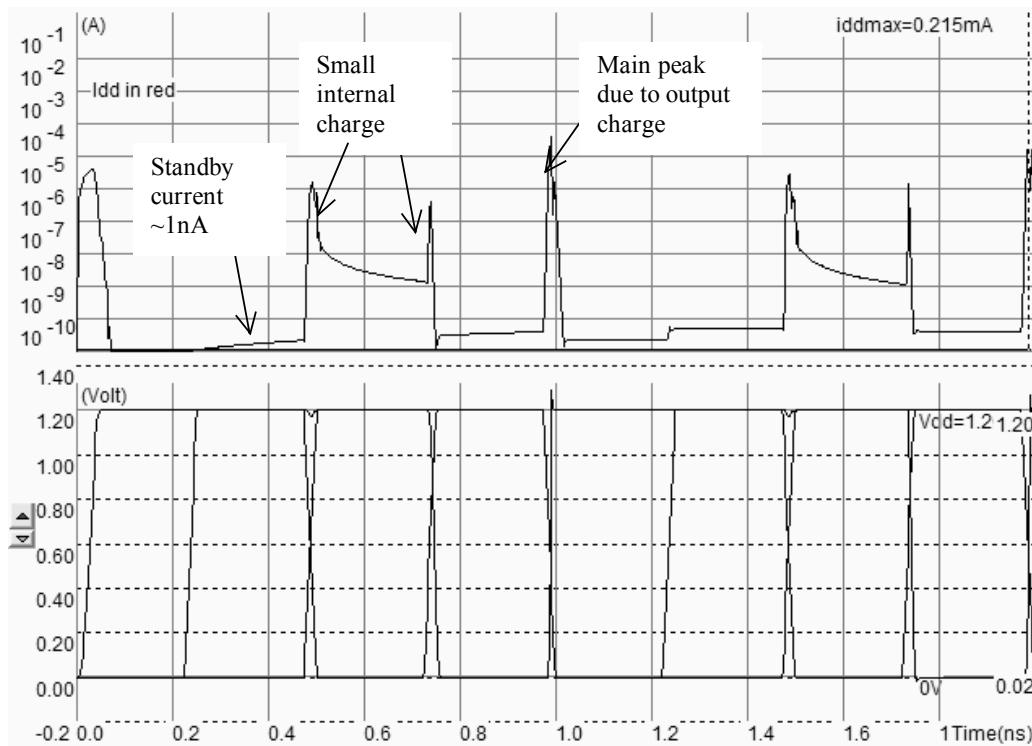


Figure 6-24. Current consumption in log scale showing the stand-by current (Nand2.MSK)

At time 0.5ns, the internal node situated in the n-channel MOS area between gates *a* and gate *b* is charged with a peak of current around  $1\mu\text{A}$ . The area consists of a small n+ diffusion which is shown in figure 6-25. This diffusion region creates a N+/P-substrate junction, polarized in invert, which may be considered as a parasitic capacitance. This capacitance is charged and discharged under certain conditions. For example, at time 0.75ns, the short-circuit current resulting from a temporary situation where both n-channel MOS and p-channel MOS devices are ON, produces a current consumption also around  $1\mu\text{A}$ .

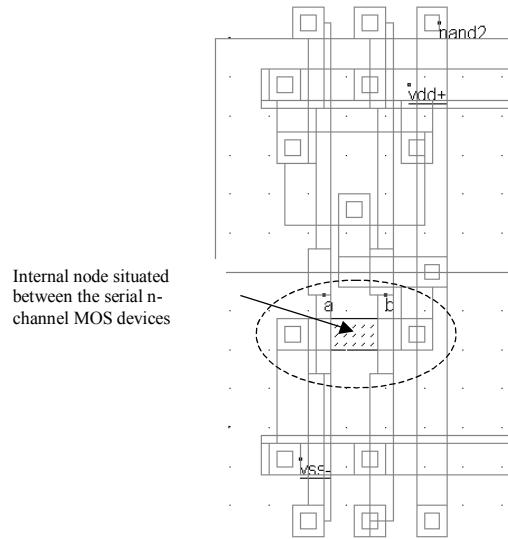


Figure 6-25. At time 0.5ns, an internal node is charged and induces a  $1\mu A$  current on VDD supply line (Nand2.MSK)

### MOS sizing

In the time-domain chronograms of figure 6-21, we observed a 7ps fall time and a 4ps rise time. This is due to the non-symmetrical structure of the NAND gate regarding low-to-high and high-to-low output switching. A fall edge of *nand2* is provoked by a discharge current  $i_{VSS}$  through the two n-channel MOS in series, meaning an equivalent of  $2 \times R_N$ , where  $R_N$  is the nMOS resistance when the channel is ON. In the case where both *a* and *b* are set to 0, the rise edge of *nand2* is provoked by the charge current  $i_{VDD}$  through the two n-channel MOS in parallel. In this case, the equivalent resistance of the path is  $R_P/2$ . Consequently, there is a switching speed difference due to the asymmetry inherent to the NAND gate structure. This asymmetry can be improved by resizing the nMOS or pMOS devices.

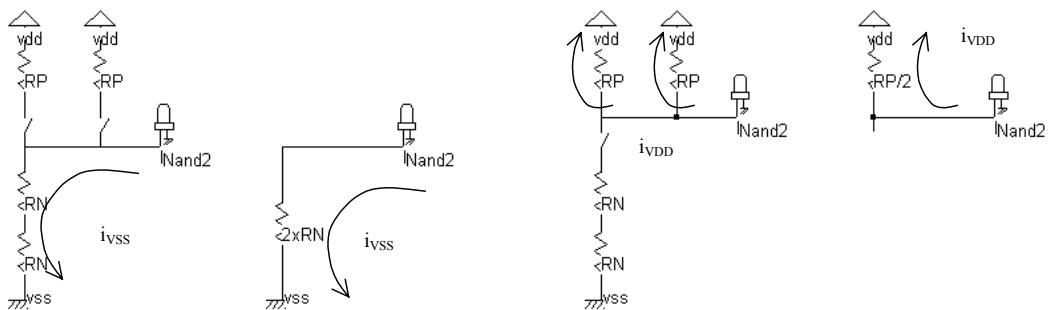


Figure 6-26. The unsymmetrical charge and discharge paths for the currents  $i_{VDD}$  and  $i_{VSS}$  lead to a faster rise time of the output.

### Optimization of the Nand Surface

The main advantage in joining the MOS devices diffusion whenever possible, rather than implementing all devices separately, is the silicon surface reduction, and consequently cost savings. A second advantage is the speed improvement. Joint diffusions lead to smaller areas, meaning lower parasitic capacitance, and thus shorter charge/discharge delays. The origin of the parasitic capacitance is mainly the N+/P-substrate junction capacitance due as the diode is polarized the other way round (P at low voltage VSS, N at higher voltage). Furthermore, the direct link between diffusions leaves space for metal routing.

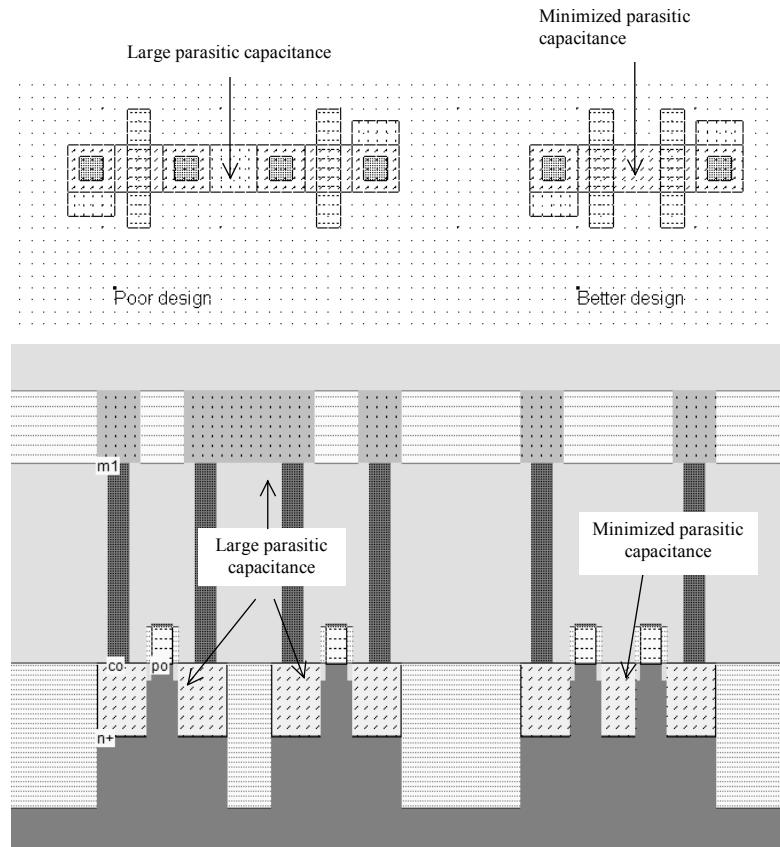


Fig. 6-27: Joined diffusions lead to compact designs and speed improvements (NandComp.MSK)

### Optimum pMOS placement

There are two solutions in implementing the pMOS devices, according to the schematic diagram of figure 6-28. One solution consists in placing the pMOS with two connections to VDD (Left circuit), the second one with two connections to the output (Right circuit). Both solutions work fine. However, from the simulation of both structures, it can be seen that the left structure with minimum diffusion and metal connected to the output switches a little faster than the right structure which includes two diffusion areas.

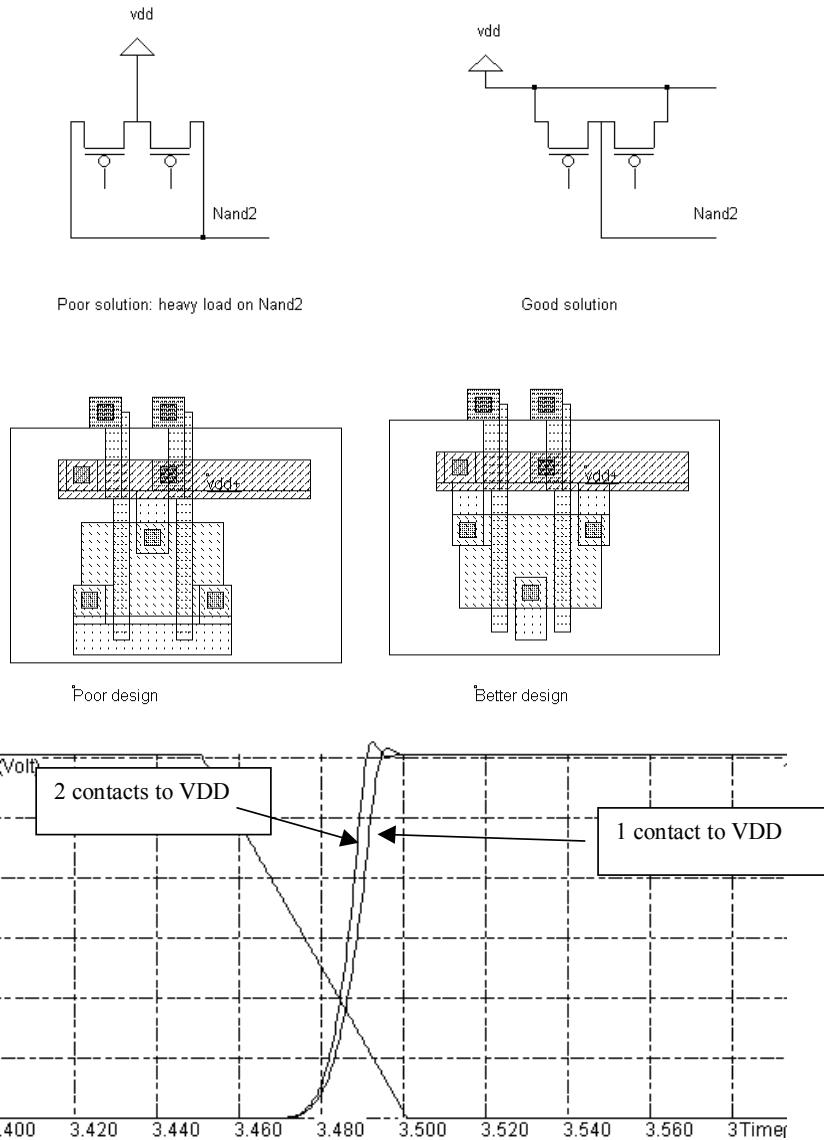


Fig. 6-28: A minimum path for the output is preferred for an optimum speed (NandComp.MSK)

### Sub-micron vs. Deep sub-micron technology

In 0.8 $\mu$ m technology, the design rules concerning the design of a link between polysilicon and metal2 layers lead to the complicate and area consuming layout displayed on the left side of figure 6-29. The cross-section of this poly/metal2 contact is shown in figure 6-30. Notice that the via plug between metal1 and metal2 is slightly larger (7x7 lambda) than the contact from polysilicon to metal (6x6 lambda). In 0.12 $\mu$ m technology, the contacts have the same small dimensions (4x4 lambda), and can be stacked on top of each other. Consequently, the routing can be more dense and the cell design can be compacted.

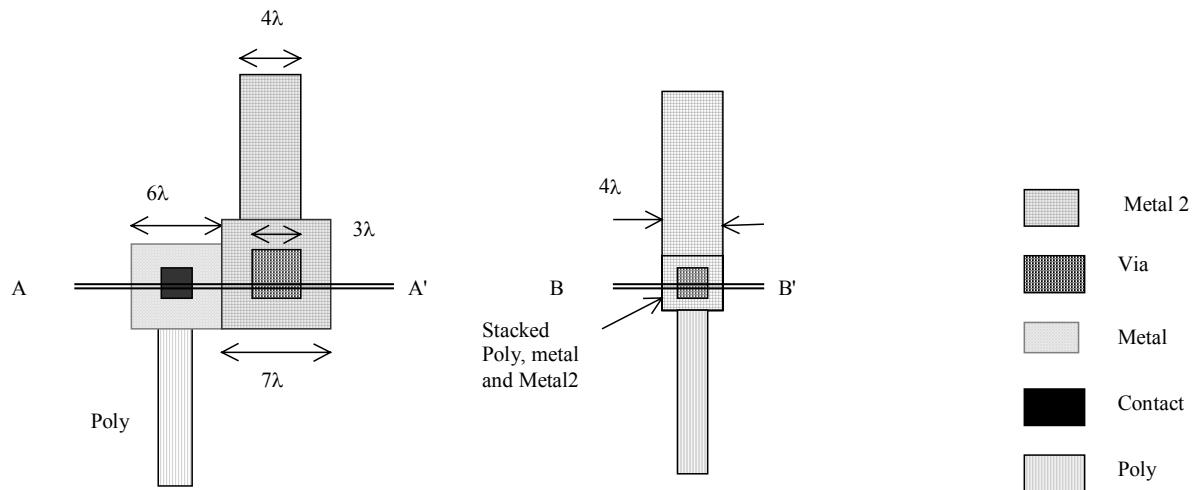


Fig. 6-29 : Sub-micron via (left) and deep sub-micron stacked vias (right)

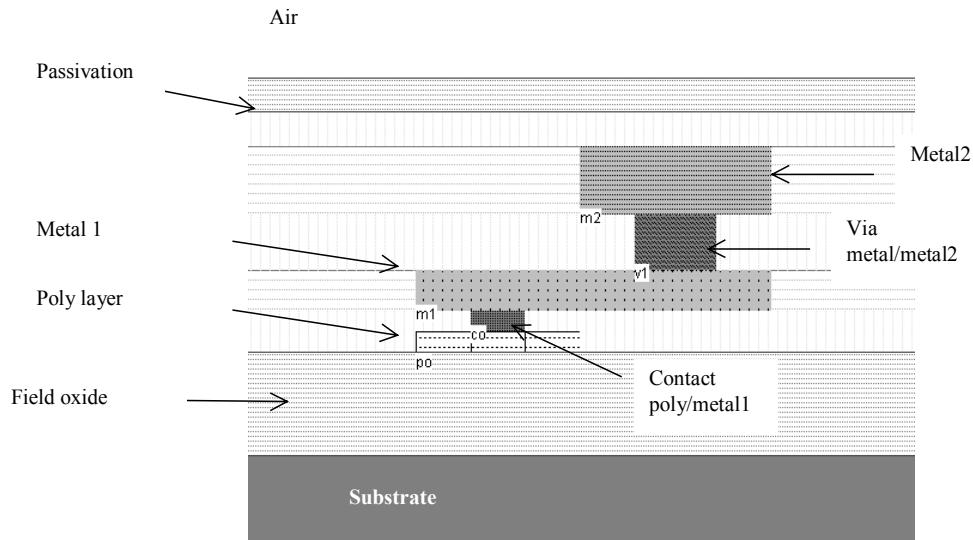


Fig. 6-30 : 2D cross-section of the poly/metal12 contact in  $0.8\mu\text{m}$  (A-A' of figure 6-29)

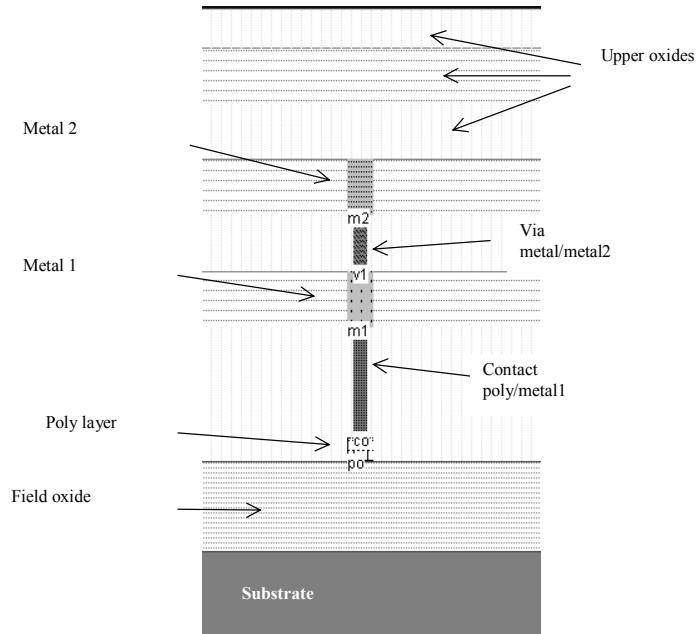


Fig. 6-31 : 2D cross-section of the poly/metal12 contact in  $0.12\mu\text{m}$  (B-B'' of figure 6-29)

The benefits of deep sub-micron technology for a complete logic gate are illustrated in figure 6-32. On the left side, the NAND gate compiled with the  $0.8\mu\text{m}$  technology parameters (cmos08.rul) is drawn. On the right side, the same NAND gate compiled with the  $0.12\mu\text{m}$  technology parameters (cmos012.rul) is reported. Although both layout designs are in lambda scale, the gain in surface is obvious.

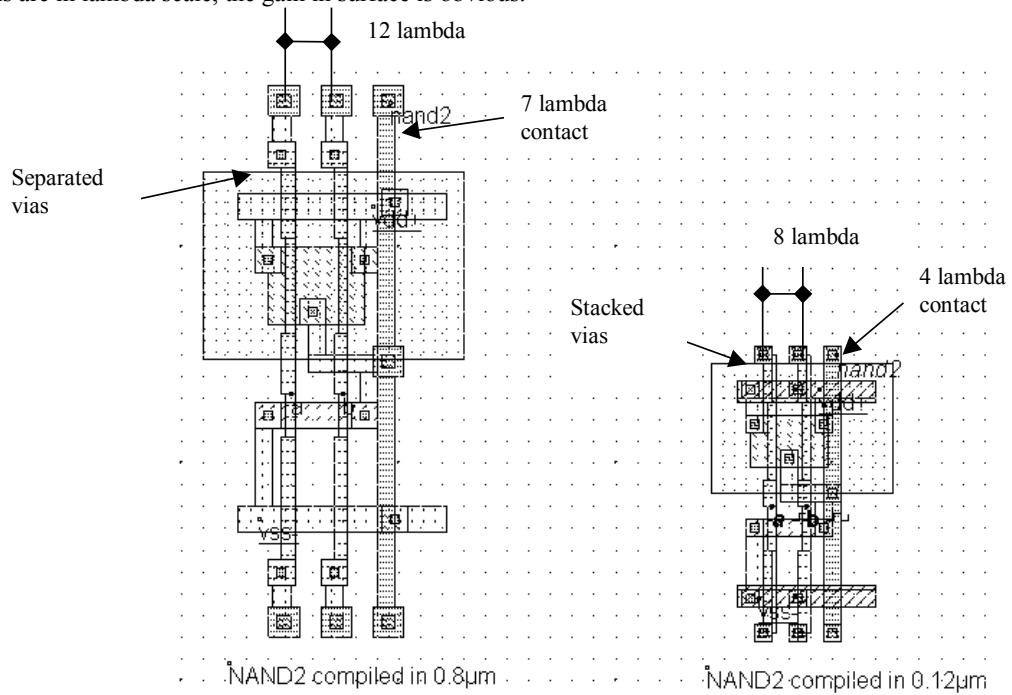


Fig. 6-32: silicon surface improvement in deep-submicron technology (NandCompo.MSK)

The major reasons for this improvement are:

- a smaller routing pitch: 12 lambda in  $0.8\mu m$ , 10 lambda in  $0.35\mu m$ , 8 lambda in  $0.12\mu m$  thanks to more aggressive rules, specifically for the size of contacts.
- the possibility to stack via: 2 routing pitches are required in  $0.8\mu m$  to connect polysilicon to metal2, one single pitch is required starting  $0.35\mu m$  technology.

### 3-input NAND Gate

The schematic diagram of the n-input NAND gate is derived from the architecture of the NAND2 gate, with n-channel MOS devices in series and p-channel MOS devices in parallel. Using the built-in cell compiler, you can generate the 3 or 4 input NAND gate. The command is **Compile→Compile One Line**. Enter the text `nand3=~(a&b&c)` and click **Compile**.

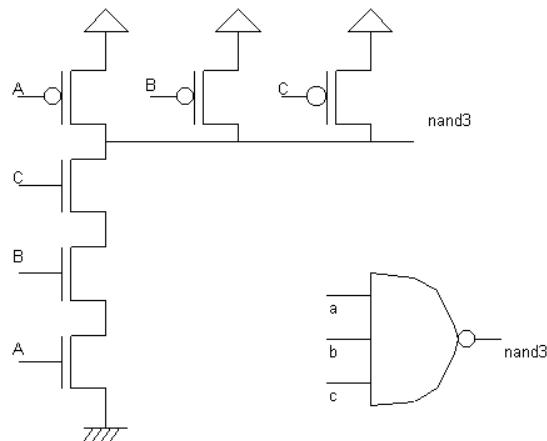


Fig. 6-33: Symbol and implementation of the 3-input NAND gate (nand3Cmos.SCH)

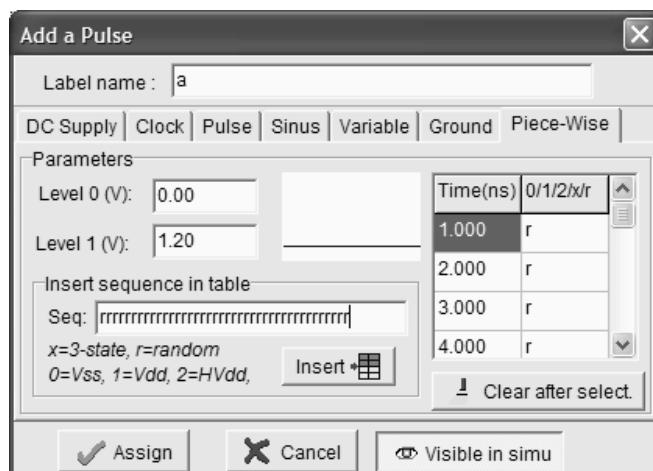


Fig. 6-34: Editing the pulse properties to generate a random logic signal (Nand3Rand.MSK)

A convenient way to observe the switching performances of the NAND3 gate is to stimulate the inputs using random patterns and observe the simulation results in **eye diagram** mode. The way to proceed consists in changing clock

properties into pulse properties, and to assign the value 'r' in place of '1' or '0', which is understood as a random logic value. The procedure is as follows: enter an "rrrrrrrrrrrrrr" sequence and click **insert** to transfer these values into the table. Then click **Assign**. The result is a series of random logic values each 1ns, as shown in the window reported in figure 6-35. The inputs (Figure 6-35) consist of a random series of logic values. Click **Reset** and a new and uncorrelated series of samples will be generated. The output change may be seen in this particular example at time 3ns.

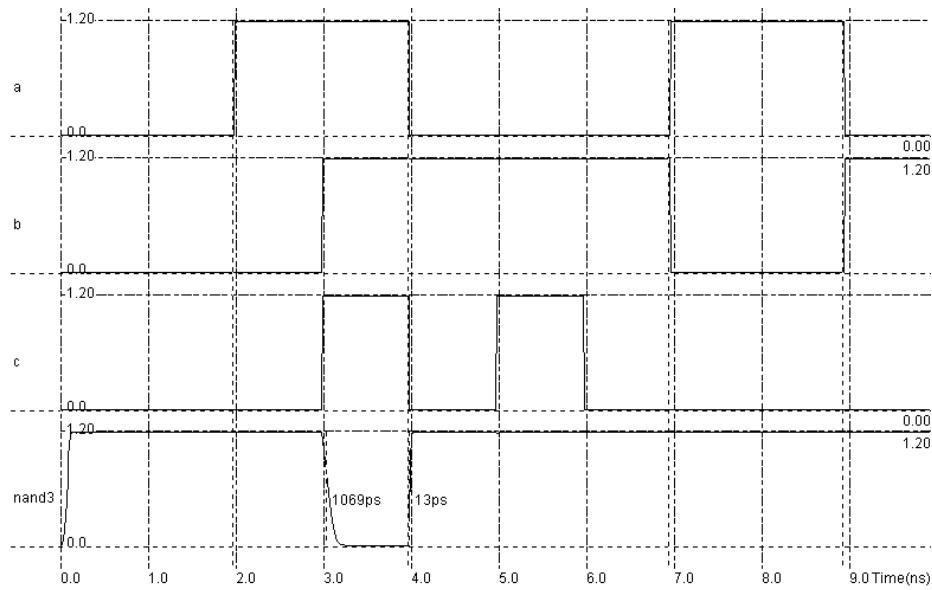


Fig. 6 35: Random stimulation of the 3-input NAND gate (nand3Rand.MSK)

### Eye Diagram Simulation

The eye diagram is created by a press on the "eye diagram" in the choice bar situated in the lower menu of the simulation window. Each time the input data changes, the simulation chronograms of the output are replaced in the left corner of the simulation window. This makes it possible to compare the switching delays. What we observe is a constant fall delay when discharging a 10fF load. Depending on the value of A,B and C, the rise delay may change significantly, as one, two or three p-channel MOS devices may be put in parallel to charge the 10fF load. Notice that the case A,B,C "on" simultaneously is a very rare event that has not been observed in the figure 6-36.

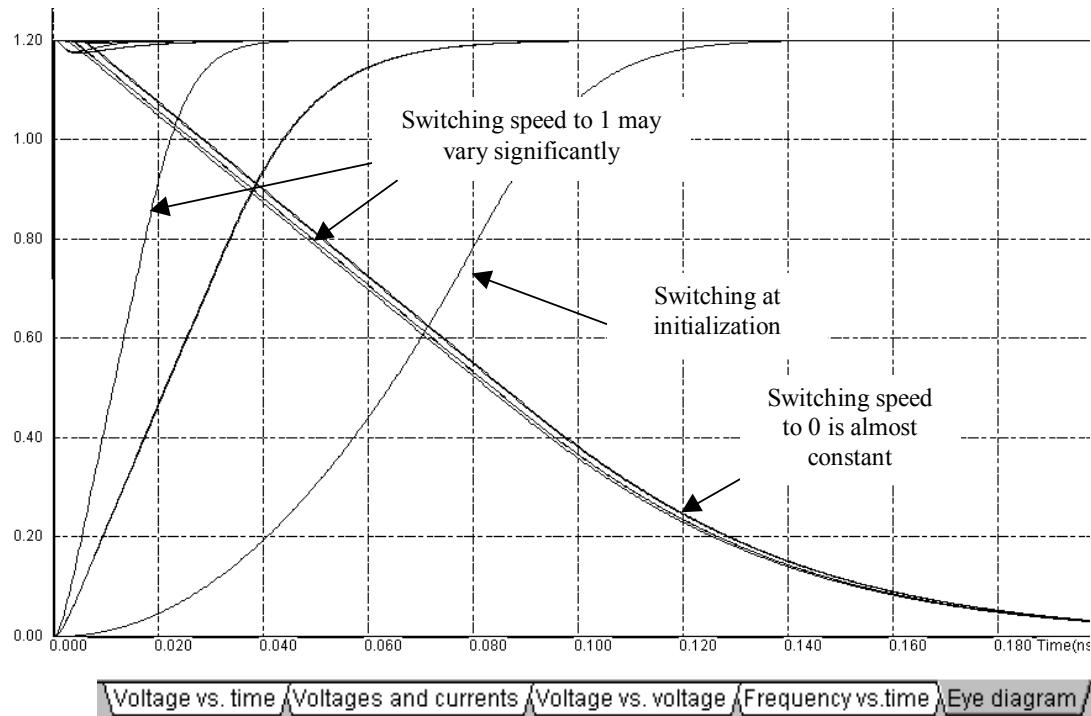


Fig. 6- 36: Eye diagram of the random simulation showing significant variations of the rise delay in the 3-input NAND gate (nand3Rand.SCH)

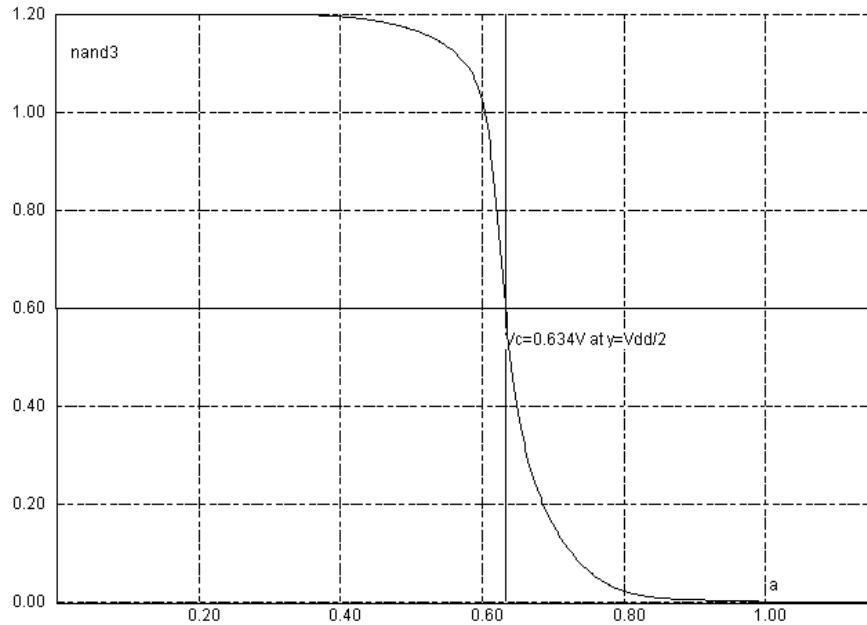


Fig. 6- 37: Static characteristics of the 3-input NAND gate (Nand3.MSK)

The switching characteristics of the 3-input **nand** gate may be improved by reducing the width of the p-mos devices and increasing the width of n-mos devices. However, the commutation point of the cell is not significantly different from

VDD/2. As seen in figure 6-37, the transfer characteristics simulated using BSIM4 between input a and the output *nand3* exhibit a commutation point  $V_c$  near 0.63V which is close to VDD/2.

## 5. The AND gate

The truth-table of the AND gate is reported in figure 6-38. In CMOS design, the AND gate is the sum of a NAND gate and an inverter. More generally, the negative gates (NAND, NOR, INV) are simpler to implement in CMOS technology, than the non-negative gates (AND, OR, Buffer). In the logic simulation at switch level (Figure 6-39), the NAND output serves as the input of the inverter output stage, and verifies the truth-table.

A	B	AND2
0	0	0
0	1	0
1	0	0
1	1	1
X	0	0
X	1	X
0	X	0
1	X	X

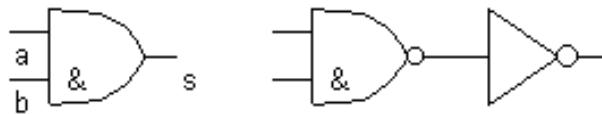


Fig. 6- 38 Truth-table of the AND gate

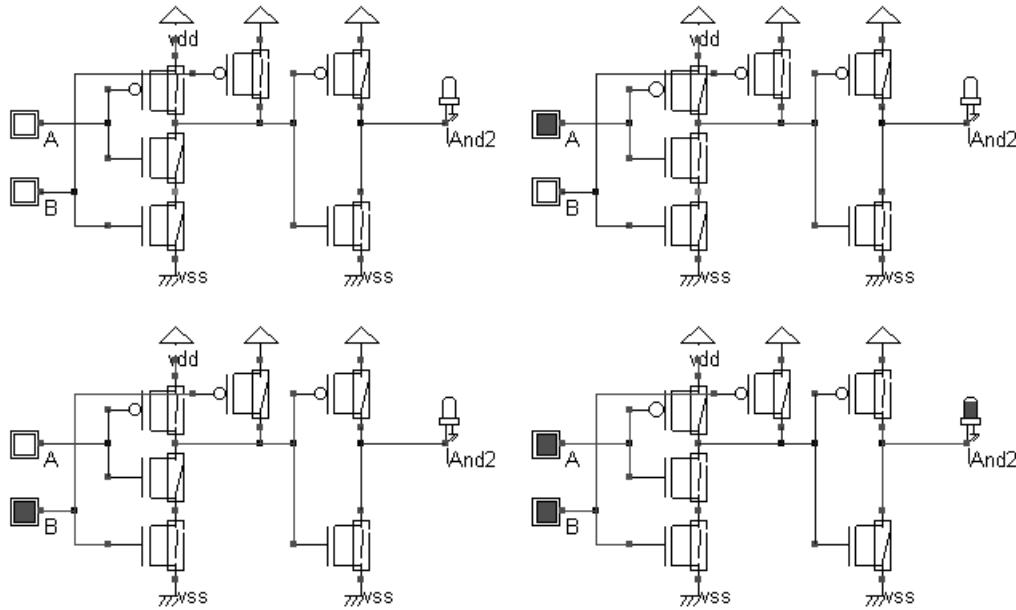


Fig. 6- 39 The switching details of the 2-input AND gate (And2Cmos.SCH)

The layout of the *and* cell may be compiled using the command **Compile → Compile One Line**. Notice that a more compact layout of the AND cell may be found by joining the diffusions of the NAND and the Inverter cells. This leads to several design rule errors that must be corrected by moving contacts, decreasing polysilicon width, etc... The final arrangement (Figure 6-40 right) saves one horizontal routing pitch, that is 8 lambda.

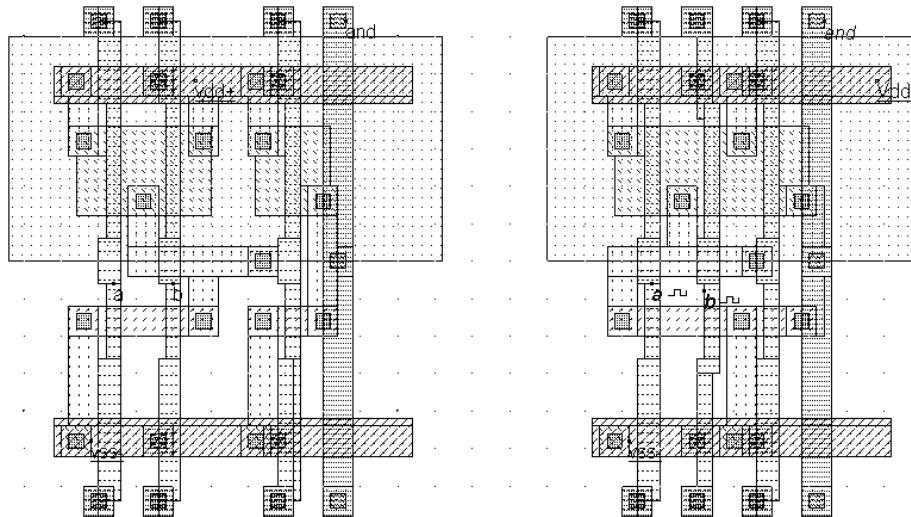


Fig. 6-40: Compiled layout of the AND gate (left) and manual arrangement (right) to achieve a more compact cell (And2.MSK)

The simulation described in figure 6-41 concerns the 2-input AND gate with a 10fF load on the output *and*. We also observe the internal node named *nand2*. We confirm that the *and2* output reacts with a certain delay, due to a cascade of two logic cells. However, the major cause of delay is the 10fF load on the output.

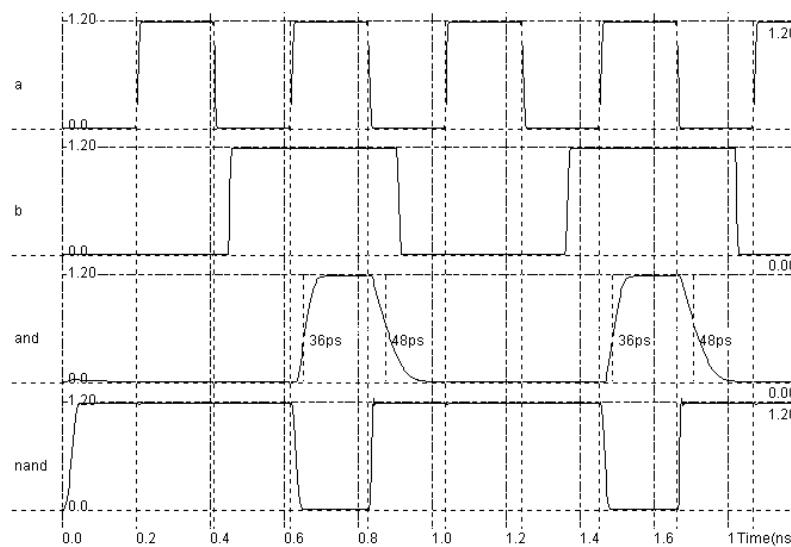


Fig. 6-41: Simulation of the AND gate with a 10fF load (And2.MSK)

## 6. The NOR Gate

AB	Out
0 0	1
0 1	0
1 0	0
1 1	0
x 0	x
x 1	0
0 x	x
1 x	0

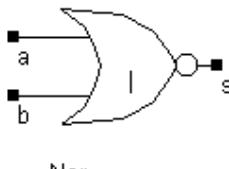


Figure 6-42. The truth table and symbol of the NOR gate

In CMOS design, the NOR gate consists of two nMOS in parallel connected to two pMOS in series. The schematic diagram of the CMOS NOR cell is reported below. The nMOS in parallel ties the output to the ground if either A or B are at 1. When both A and B are at 0, the nMOS path is cut, but the two pMOS devices in series tie the output to the supply VDD.

AB	nMOS	pMOS	Out
0 0	off	on	1
0 1	on	off	0
1 0	on	off	0
1 1	on	off	0

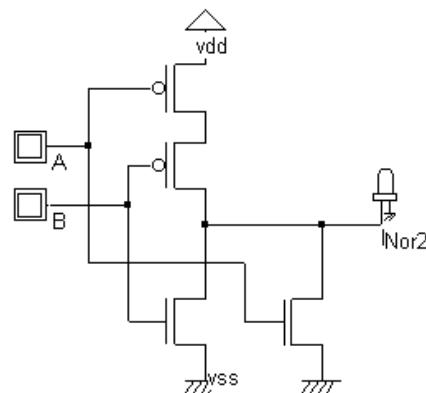


Figure 6-43. Schematic diagram of the CMOS NOR gate (NorCmos.SCH)

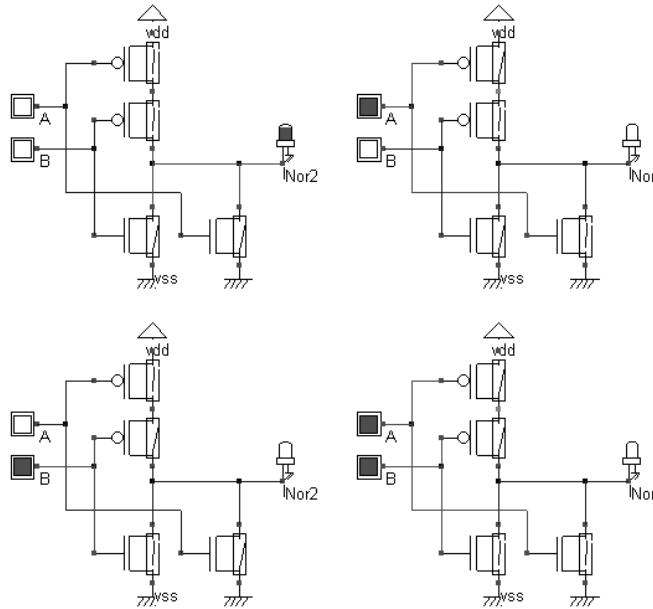


Figure 6-44. The logic simulation of the NOR gate verifies the truth table (NorCmos.SCH)

Using the cell compiler we generate the NOR gate from its VERILOG description by entering the equation  $\text{Nor2}=\sim(a \mid b)$ . The ' $\sim$ ' operator represents the NOT operator, the ' $\mid$ ' symbol represents the OR operator. The compiled layout of the NOR gate appears in the screen, as drawn in figure 6-45. The compiler has fixed the position of the VDD power supply in the upper part of the layout, near the p-channel MOS devices. The compiler has also fixed the ground VSS in the lower part of the window, near the n-channel MOS devices. The texts *A*, *B*, and *out* have been fixed to the layout as for the NAND gate.

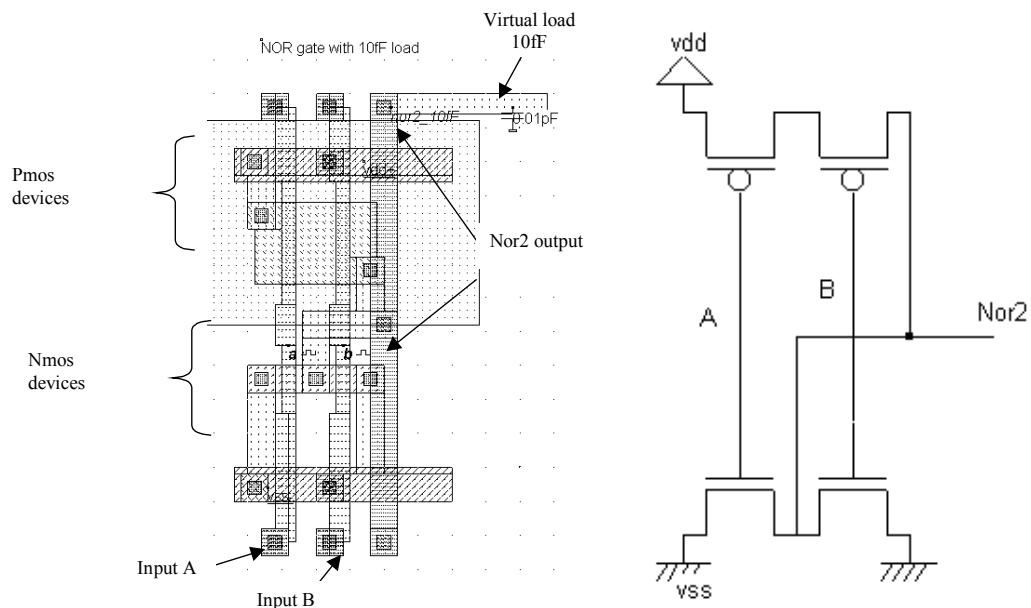


Figure 6-45. The layout of the NOR gate generated by the CMOS cell compiler (Nor2.MSK)

The simulation of figure 6-46 is obtained by the command **Simulate → Run simulation**. We verify that **nor2** output signal is equal to 0 when either  $a=1$  or  $b=1$ , according to the truth-table. The rise time and fall time are shown for a nor2 gate without any load connected to its output or with a virtual 10fF load. We see that the 10fF slows down the switching speed considerably. Furthermore, the rise time is significantly larger than the fall time, due to p-channel MOS in series.

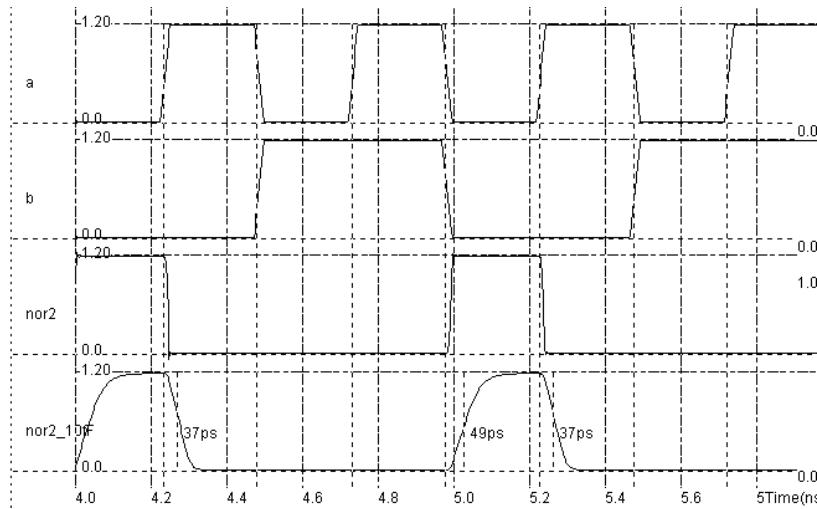
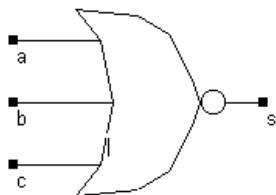


Figure 6-46. Simulation of the NOR gate with and without loading capacitance (Nor2.MSK)

### The NOR3 Gate



The 3-input NOR gate circuit may be deduced from the 2-input NOR gate design, but a problem rises. The pMOS devices in series lead to very poor output node charge to VDD, and consequently poor rise time performances, as shown below. Meanwhile, the nMOS devices in parallel provoke a very efficient path for discharge to ground, meaning a short fall time. This non-symmetrical behavior can be tolerated up to a certain limit. This is why the n-input NOR gates (with  $n=3,4..\right)$  are rarely used. NAND gate based designs are preferred because of the pMOS in parallel which naturally compensate the poor hole mobility of their channel, producing symmetrical switching characteristics.

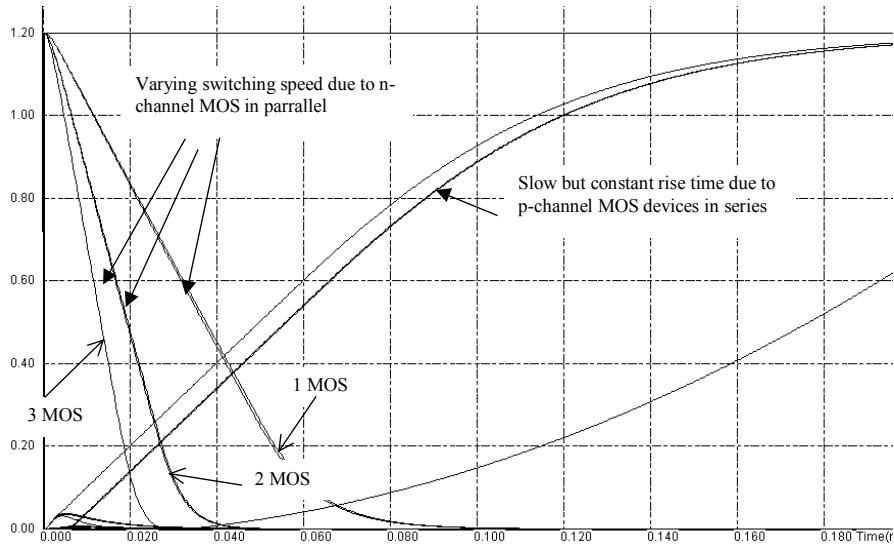


Fig. 6-47 : the non-symmetrical behavior of the NOR3 gate (Nor3.MSK)

## 7. The OR Gate

AB	Out
0 0	0
0 1	1
1 0	1
1 1	1
x 0	x
x 1	1
0 x	x
1 x	1



OR2 Gate

Figure 6-48. The truth table and symbol of the OR gate

Just like the AND gate, the OR gate is the sum of a NOR gate and an inverter. The implementation of the OR2 gate in CMOS layout requires 6 transistors. An arrangement may be found to obtain continuous diffusions on n-MOS regions and pMOS regions, as illustrated in figure 6-49.

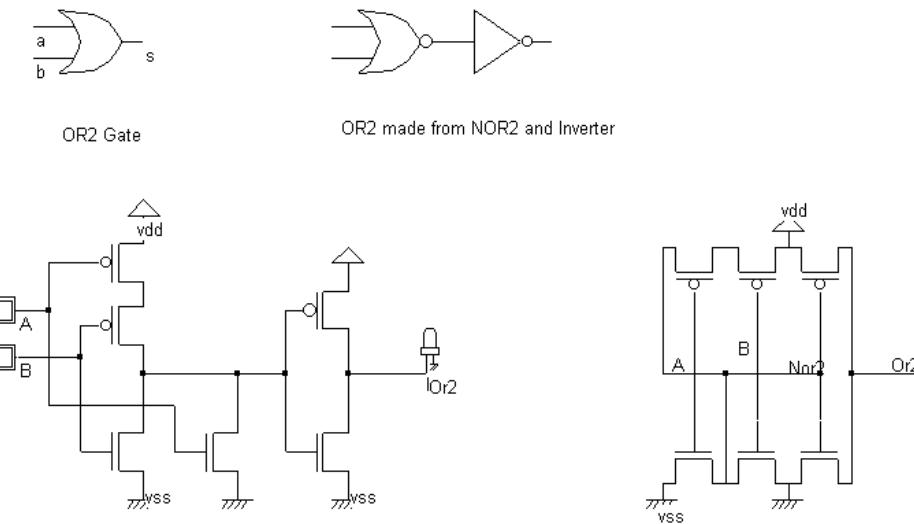


Fig. 6-49 : The schematic diagram of the OR gate and its CMOS structure (Or2.SCH)

The layout generated by the VERILOG compiler (`or2=a | b`) is shown in figure 6-50. A more compact version of the OR2 gate is also provided (Figure 6-51 right) where the supply pins VSS and VDD are shared by the NOR2 and Inverter cells.

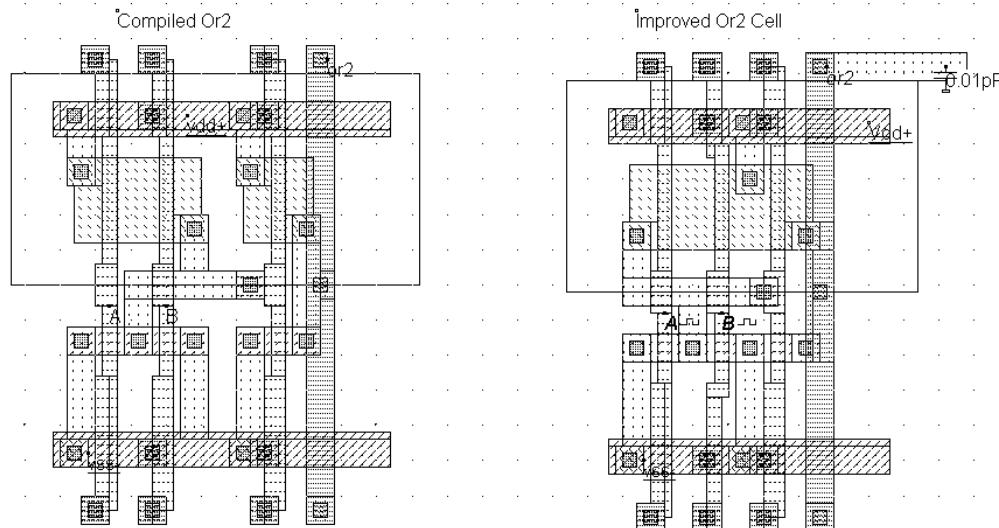


Fig. 6-50 : Layout of the compiled OR2 cell (left) and a more area-efficient manual arrangement (right) (OR2.MSK)

## 8. The XOR Gate

A	B	XOR2
0	0	0
0	1	1
1	0	1
1	1	0
X	0	X

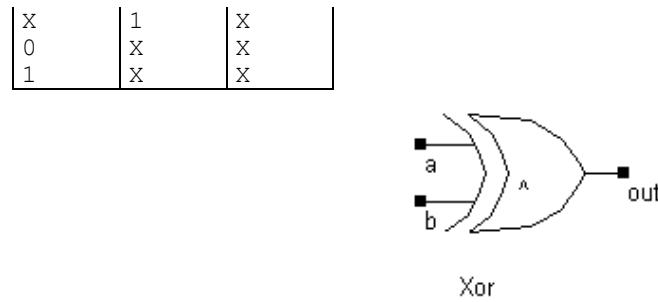


Fig. 6-51 : Truth table and symbol of the XOR gate

The truth-table and the usual symbol of the CMOS XOR gate are shown in Figure 6-51. There exist many possibilities to implement the XOR function into CMOS, which are presented in the following paragraphs.

### A poor design

The least efficient design, but the most forward, consists in building the XOR logic circuit from its Boolean equation given by equation 6-4. The direct translation into primitives leads to a very complicated circuit shown in figure 6-52. The circuit requires two inverters, two AND gates and one OR gate, that is a total of 22 devices. Furthermore, the XOR gate has a critical path based on 6 stages of CMOS gates, which slows down the XOR switching response considerably.

$$\text{xor} = \overline{AB} + \overline{BA} \quad (\text{Equ. 6-4})$$

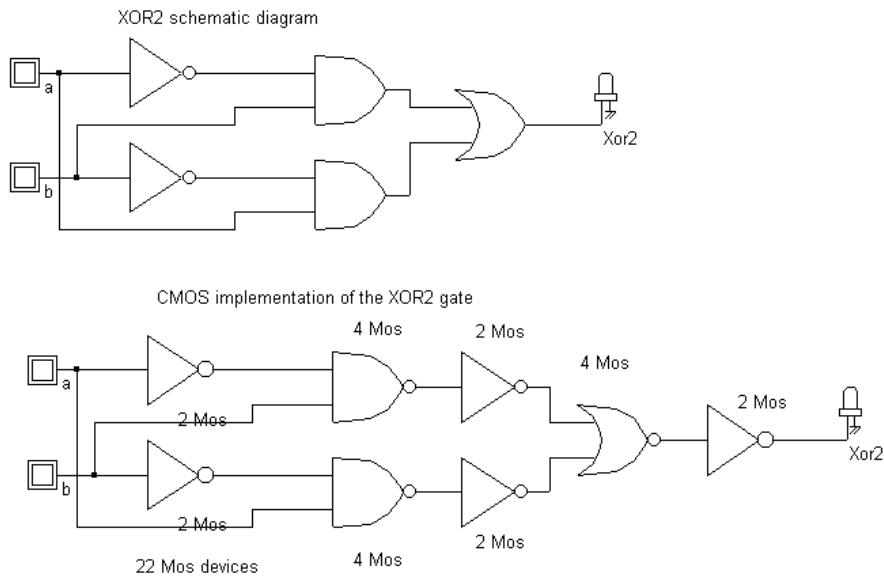
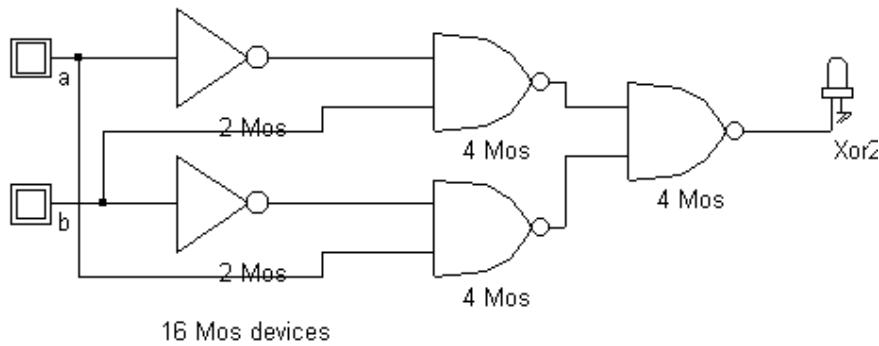


Fig. 6-52. The direct translation of the XOR function into CMOS primitives leads to a 22-transistor cell with 5 delay stages (Xor2.SCH)

Keeping in mind that negative logic is preferred in CMOS, we can re-arrange the expression of the **xor** cell in a less complex circuit, by replacing the AND/OR stage by NAND gates, following the equation 6-5. The total number of transistors is decreased to 16.

$$\overline{\overline{AB}} = \overline{BA} = \overline{AB} \cdot \overline{BA}$$

(Eqn. 6-5)



*Fig. 6-53. An XOR function based on NAND gates leading to 16 transistors and 3 delay stages (Xor2\_16.SCH)*

### Compiling a Schematic Diagram

An alternative to the manual design consists in describing the logic circuit of the XOR gate in DSCH, and then in compiling the schematic diagram into layout using MICROWIND. The design flow is detailed in figure 6-54. The XOR circuit is created according to the schematic diagram of figure 6-53. The circuit is saved under the name **Xor2\_16.SCH**, for example. Next, we create the VERILOG description corresponding to the circuit, using the command **File → Make Verilog File**. The text file **Xor2\_16.TXT**, which includes the VERILOG description, serves as the input of MICROWIND, to drive the automatic compilation of the circuit into layout. In MICROWIND, the command is **Compile → Make Verilog File**.

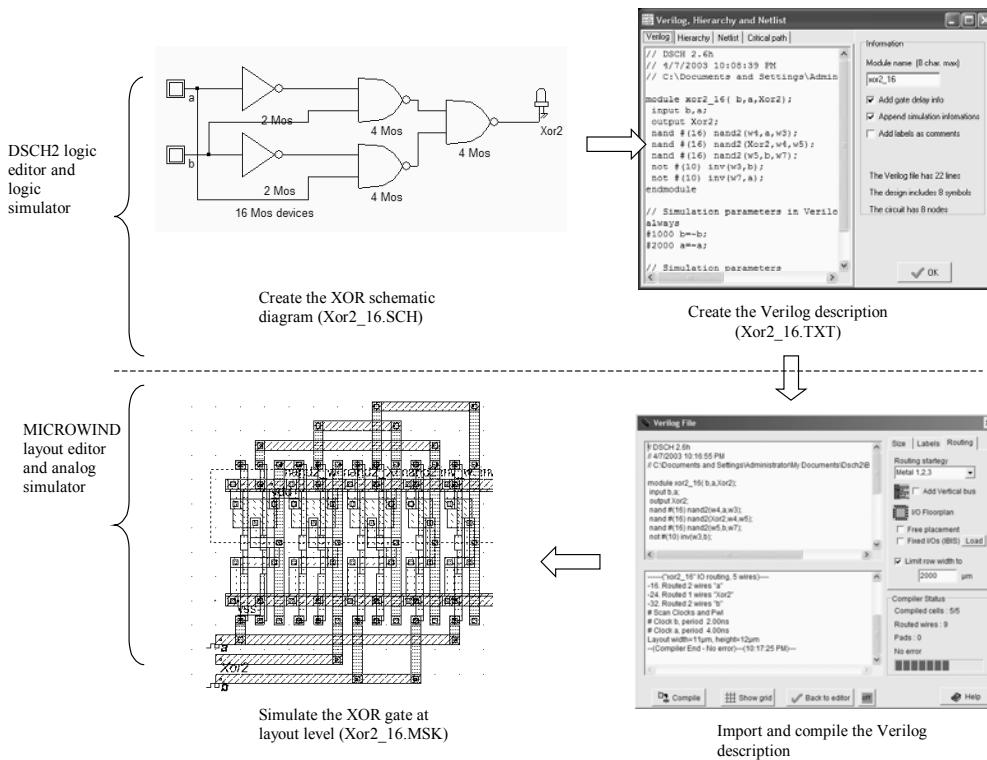


Figure 6-54: Compiling the XOR circuit from its logic description, through the VERILOG format (Xor2\_16.MSK)

The result is a layout dynamically created by MICROWIND which corresponds to the initial circuit defined at logic level. The simulation works fine, as shown in figure 6-55. Notice that no simulation property is required as it is inherited from the logic circuit simulation. The VERILOG text file not only includes the structural description of the circuit but also the list of inputs and the associated simulation parameters. Still, the layout is quite large, and the switching performances suffer from the three delay stages.

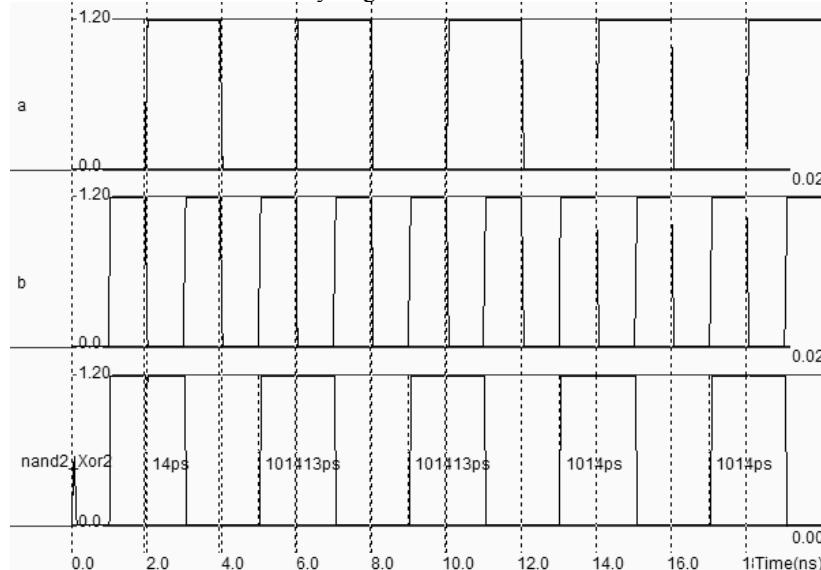


Figure 6-55: Simulation of the 3-stage XOR circuit (Xor2\_16.MSK)

### A better Design

An interesting design is shown in figure 6-55. The XOR function is built using AND/OR inverted logic (AOI logic <gloss>). The function created by the n-channel MOS network is equivalent to  $(A|\sim B) \& (\sim A|B)$ . The p-channel MOS network gives the function where all AND functions are transformed into OR, and vice-versa. In other words, the pMOS network realizes the function  $(A \& \sim B) | (\sim A \& B)$ .

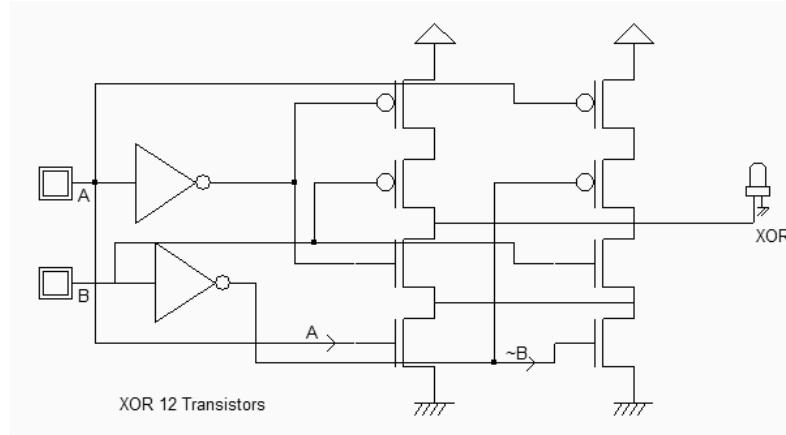


Figure 6-55: A 12-transistor XOR gate (XorAoi.SCH)

### An efficient design

The most efficient solution consists of 2 inverters and 2 pass transistors, that is only 6 MOS. The cell is not a pure CMOS cell as two MOS devices are used as transmission-gates. The truth table of the XOR can be read as follow: IF  $B=0$ ,  $OUT=A$ , IF  $B=1$ ,  $OUT = \sim A$ . The principle of the circuit presented below is to enable the  $A$  signal to flow to node  $N1$  if  $B=1$  and to enable the  $\sim A$  signal to flow to node  $N1$  if  $B=0$ . The node  $OUT$  inverts  $N1$ , so that we cover the truth-table of the XOR operator. Notice that the nMOS and pMOS devices situated in the middle of the gate serve as pass transistors (Figure 6-56).

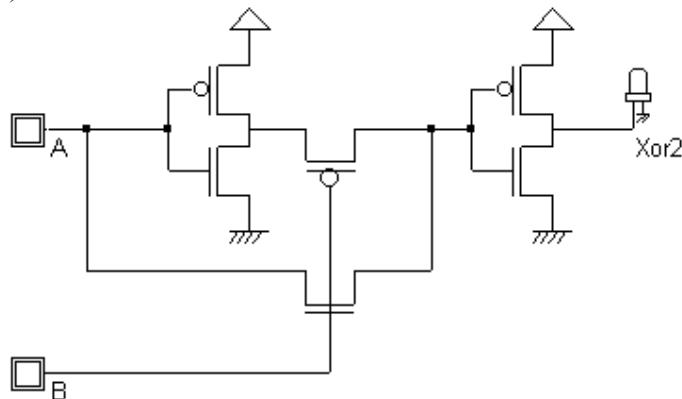


Fig. 6-56. The schematic diagram of the XOR gate with only 6 MOS devices (Xor2.SCH)

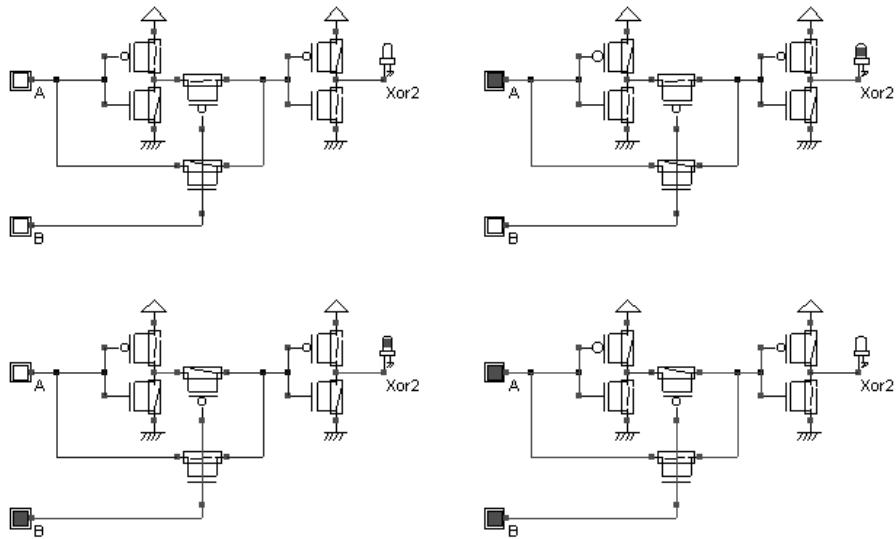


Fig. 6-57. Simulation of the XOR gate (Xor2Cmos.SCH)

The Verilog description of the XOR gate in Microwind is `xor=a^b`. The layout compiler produces an XOR cell layout as reported in Figure 6-58.

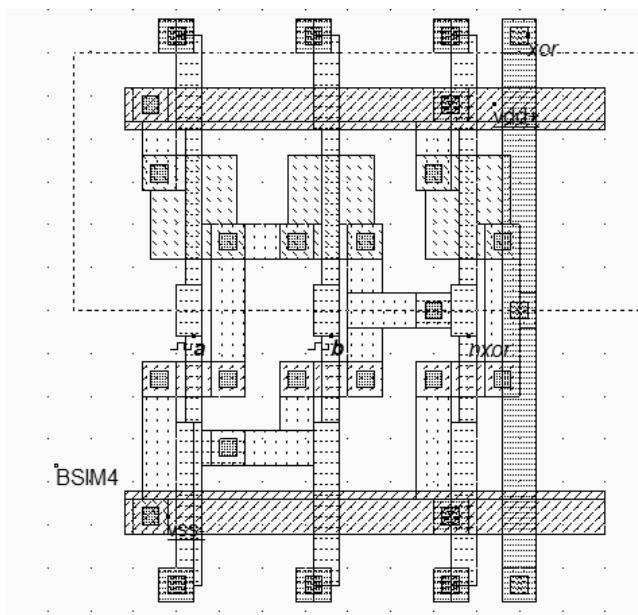


Fig. 6-58. Layout of the XOR gate. The BSIM4 label configures the simulator with BSIM4 model (XOR2.MSK).

When you add a visible property to the intermediate node `xnor` which serves as an input of the output stage inverter, see how the signal is altered by  $V_{tn}$  (when the nMOS is ON) and  $V_{tp}$  (when the pMOS is ON).

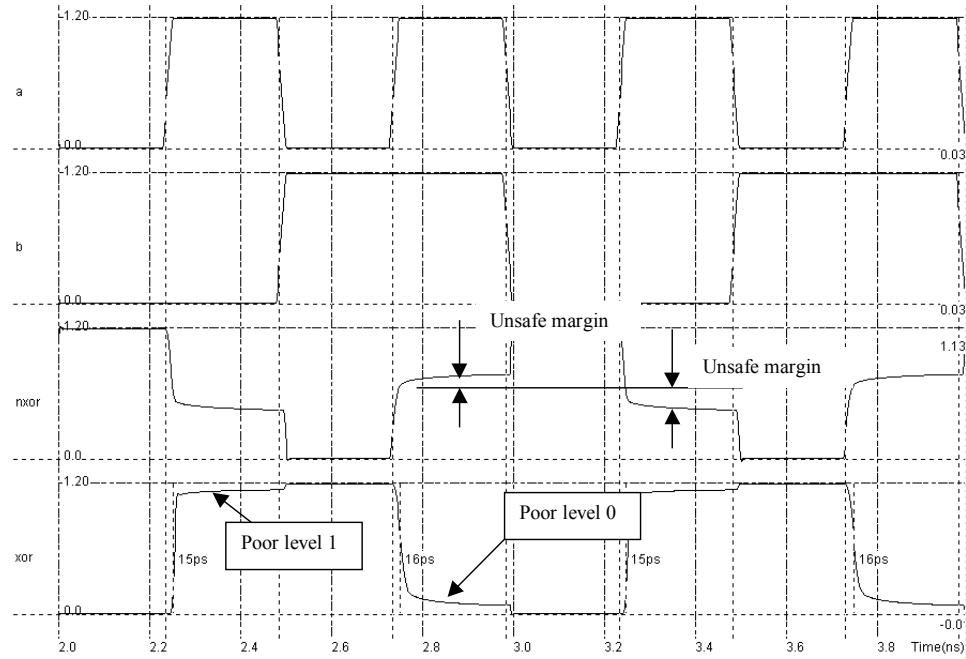


Fig. 6-59 Simulation of the XOR gate using BSIM4 model (XOR2.MSK).

The inverter regenerates the signal, but fails to produce clean 0 and 1 levels. This is because the MOS devices used as pass transistor reduces the voltage amplitude, resulting in dangerous voltage levels, close to the switching point of the logic. The alternative is to abandon the single pass-gate and use a complete pair of n-channel MOS and p-channel MOS devices, as shown in figure 6-60.

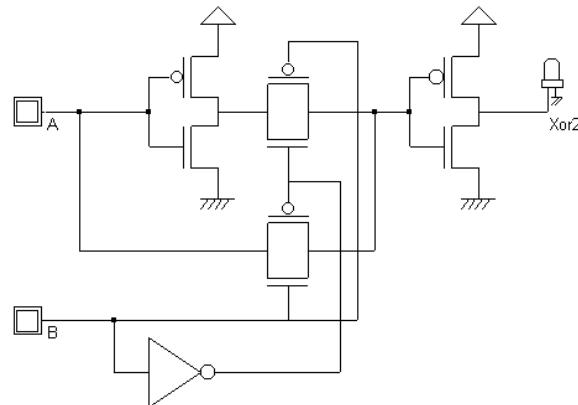
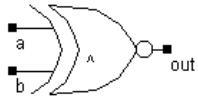


Fig. 6-60: Schematic diagram of the XOR gate with complete transmission gates (XOR2Full.MSK).

### About the XNOR gate



Xnor

The XNOR gate symbol is shown above. The XNOR circuit is usually an exact copy of the XOR gate, except that the role of the B and  $\sim B$  signals are opposite in the transmission-gate structures. Removing the last inverter is a poor alternative as the output signal is no more amplified. Adding a supplementary inverter would increase the propagation delay of one stage.

## 9. Complex Gates

### Principles

The complex gate design technique applies for any combination of operators AND and OR. The AND operation is represented in logical equations by the symbol "&". The OR operation is represented by "|" (Table 6-3). The complex gate technique described below produces compact cells with higher performances, in terms of spacing and speed, than conventional logic circuits.

Bit operation	Verilog symbol used in complex gates
Not	$\sim$
And	&
Or	

Table 6-3: Logic operator used to compile complex gates

To illustrate the concept of complex gates, let us take the example of the following Boolean expression (Equation 6-6). Its truth-table is reported in table 6-4.

$$F = A \& (B | C) \quad (\text{Equ. 6-6})$$

A	B	C	$A \& (B   C)$	F
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Table 6-4: Truth-table of the function  $F = A | (B \& C)$ 

A schematic diagram corresponding in a straightforward manner to its equation and truth-table is reported below. The circuit is built using a 2-input OR and a 2-input AND cell, that is 12 transistors and four delay stages.

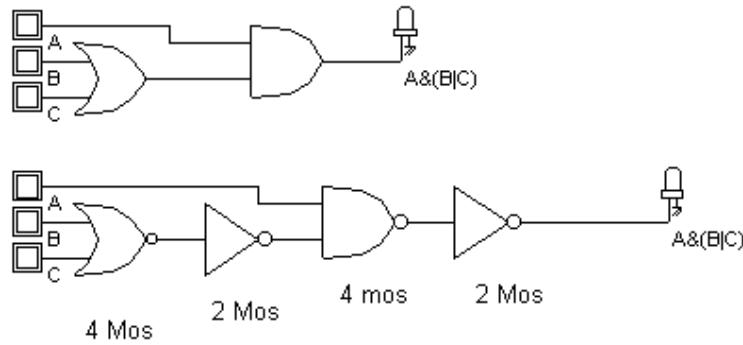


Fig. 6-61: The conventional schematic diagram of the function  $F$  (ComplexGate.SCH)

A much more compact design exists in this case (Figure 6-62), 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.

$$F_n = A \text{ series} (B \text{ parallel} C) \quad (\text{Eqn. 6-7})$$

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

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

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

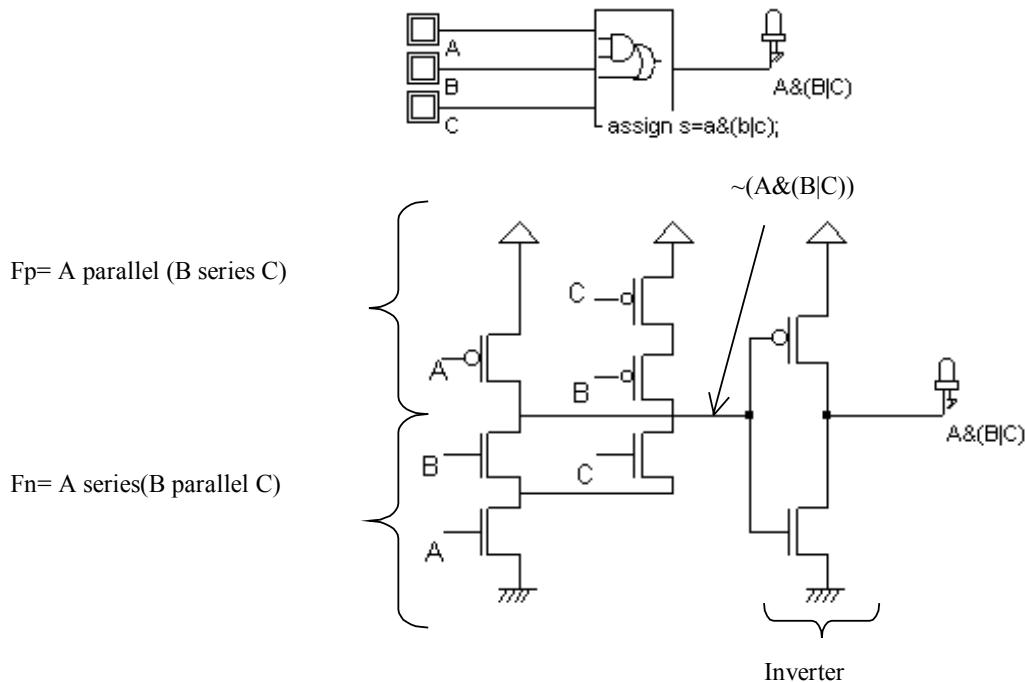


Fig.6- 62: The complex gate implementation of the function  $F = A \& (B | C)$  (ComplexGate.SCH)

### Complex Gates in Dsch

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 6-63. At a double click inside the symbol, the menu shown in figure 6-64 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 ( $\sim$  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.

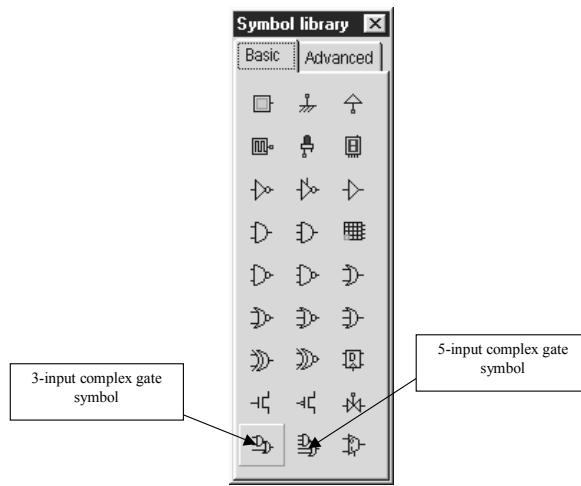


Fig. 6-63. Specific symbols proposed for complex gate description at logic level

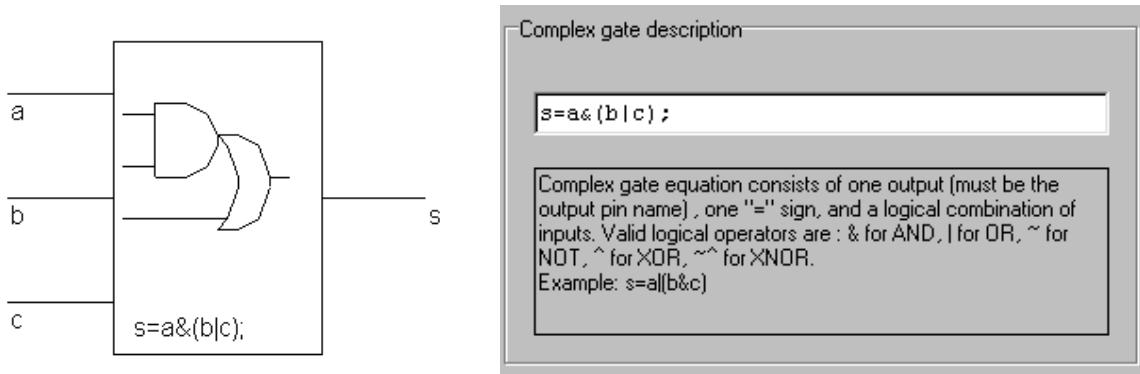


Fig. 6-64. The complex gate symbol and its logic description

### Complex Gates in Microwind

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 equation 6-9, the first parameter *AndOr* is the output name. The sign '=' is obligatory, and follows the output name. The ' $\sim$ ' 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 and '|' is the OR operator.

$$\text{Andor} = A \& (B | C)$$

(Equ. 6-9)

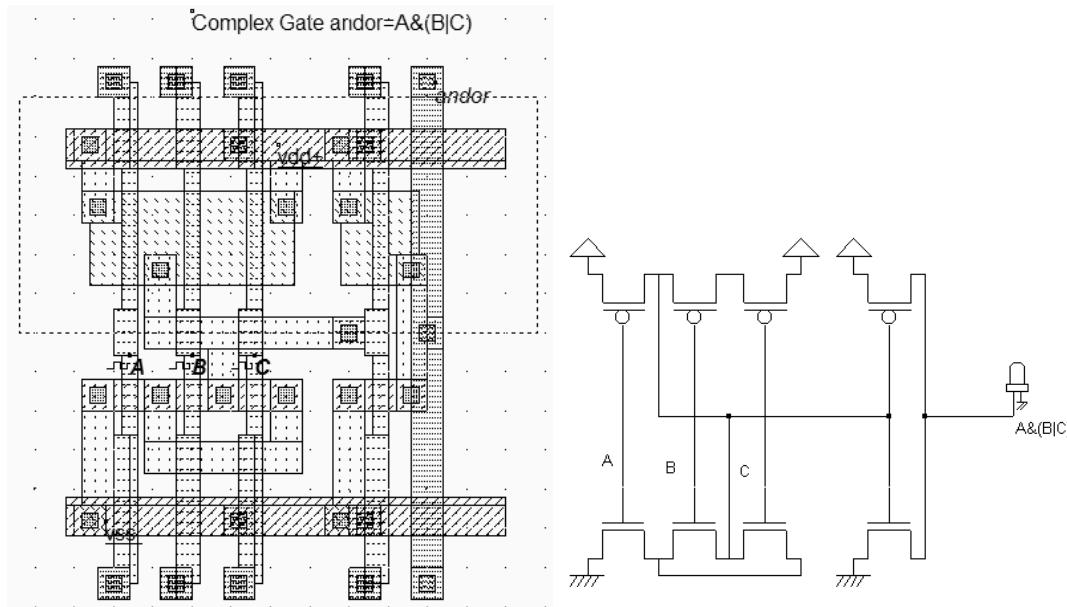


Fig. 6-65. Compiled complex gate  $\text{andor}=A\&(B|C)$  (ComplexGate.MSK)

The MOS arrangement proposed by Microwind consists of a function  $\sim A \& (B|C)$  and one inverter. Notice that the cell could be rearranged to avoid the diffusion gap by a horizontal flip of the left structure and the sharing of VDD and VSS contacts.

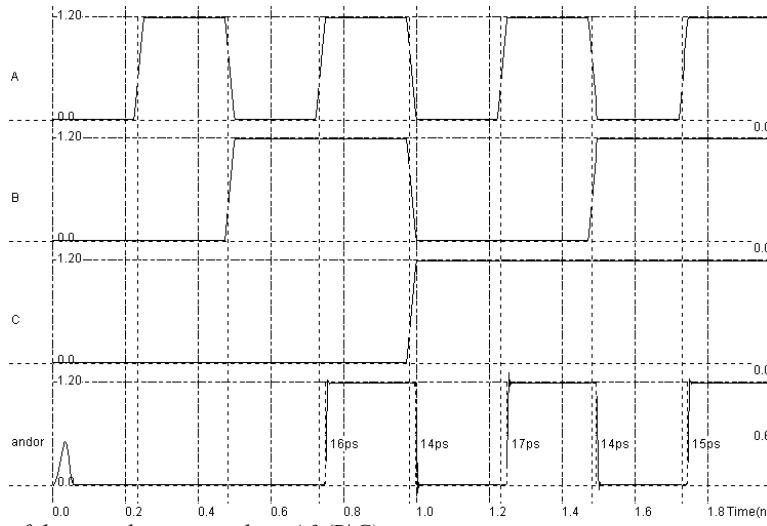


Fig. 6-66. Simulation of the complex gate  $\text{andor}=A\&(B|C)$

## 10. Multiplexor

Generally speaking, a multiplexor is used to transmit 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. A behavioral description of the multiplexor is the case statement:

```
Case (sel)
  0 : f=in0;
  1 : f=in1
endcase
```

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

Table 6-8: the multiplexor truth-table

The usual symbol for the multiplexor is given in figure 6-67. It consists of the two multiplexed inputs *in0* and *in1* on the left side, the command *sel* at the bottom of the symbol, and the output *f* on the right.

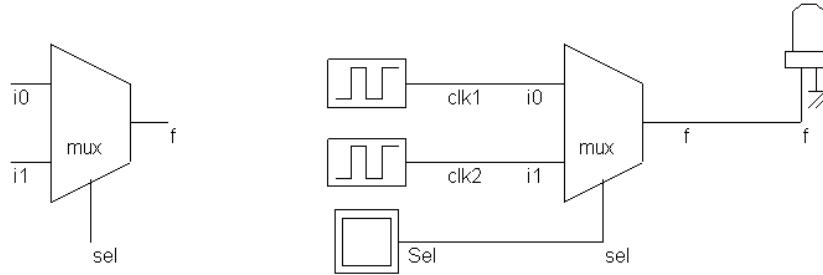


Fig. 6-67. The multiplexor circuit (MUX.SCH)

The simulation of the multiplexor proposed in figure 6-68 uses two different clocks *clk1* and *clk2*. When *sel*=0, *f* copies *clk1*. When *sel*=1, *f* copies *clk2*.

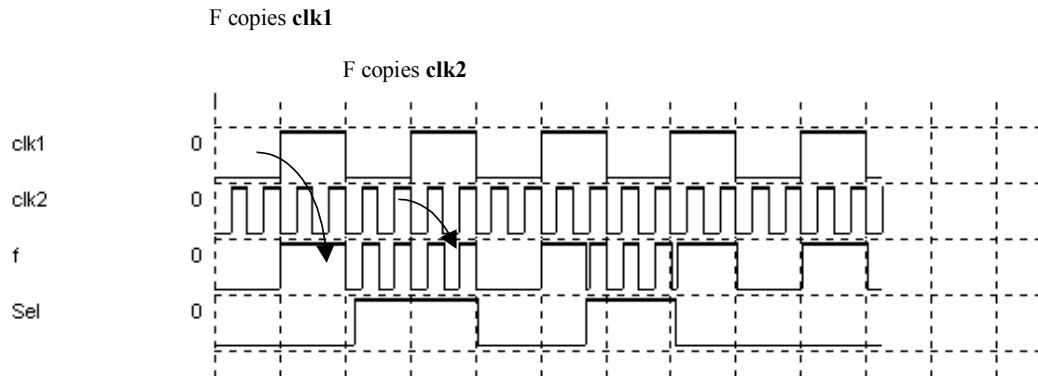


Fig. 6-68. Logic simulation of the multiplexor circuit (MUX.SCH)

### Design of the Multiplexor

The most simple schematic diagram of the 2-input multiplexor is based on two MOS devices. The main problem of using single pass devices is the degradation of voltage levels. In  $0.12\mu\text{m}$ , the default MOS devices ("low leakage") have a high threshold voltage, meaning that  $f$  does not reach clean high and low voltages in many cases. Furthermore, the output is not buffered, so  $f$  cannot drive significant loads. Moreover, inputs  $i0$  and  $i1$  are connected to the diffusions of n-channel and p-channel transistors. Depending on the value of  $sel$ , the load may vary, so the cell delay may vary accordingly. This circuit, appearing as structure (1) in figure 6-69 is rarely found in CMOS design as the signal  $f$  is weak and very sensitive to noise.

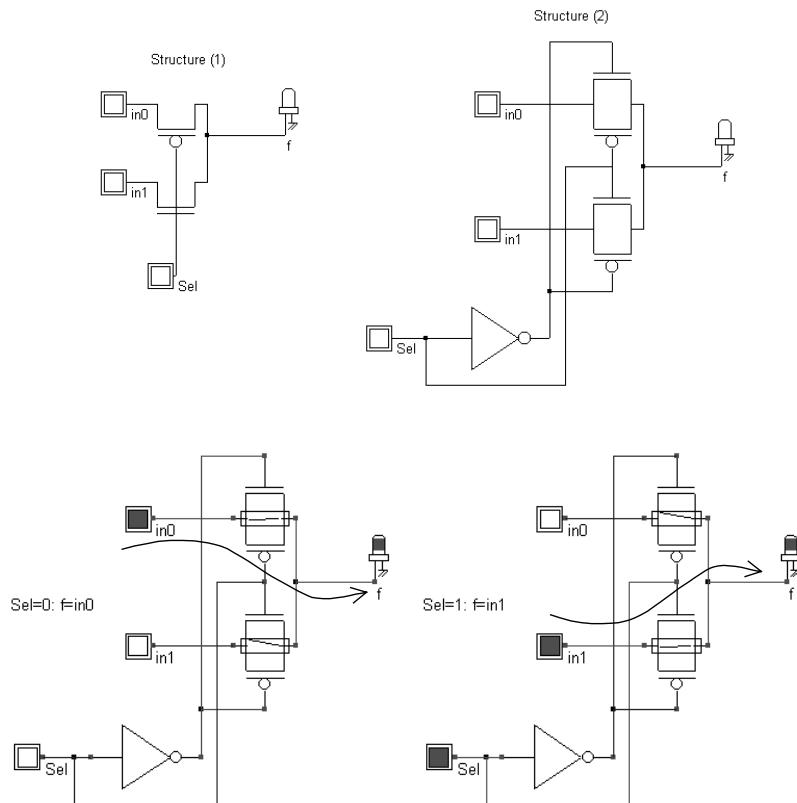


Fig. 6-69. Two-transistor and 6-transistor implementation of the multiplexor (MUX.SCH)

A better circuit, proposed in figure 6-69 as structure (2) is commonly used for low power operations. The threshold voltage degradation is eliminated by the use of transmission gates. However, the signal  $f$  is not amplified. Two implementations of the multiplexor are proposed in figure 6-70. The layout on the left has been generated automatically by placing one inverter and four single MOS devices one after the other. A manual arrangement of the MOS devices (Figure 6-70 right) leads to a much more compact circuit. When the output is loaded by  $10fF$ , the circuit behaves quite well in terms of delay. As predicted, no parasitic threshold effect may be seen.

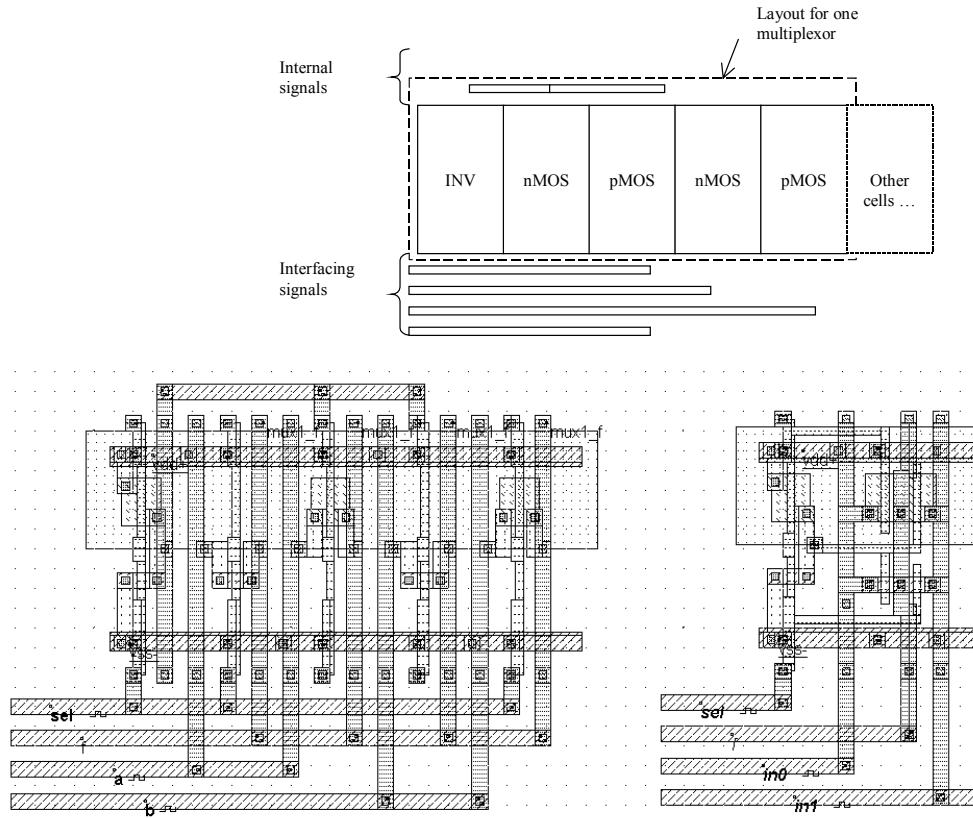


Fig. 6-70. compiled and manual implementation of the 6-transistor multiplexor (MUX.MSK)

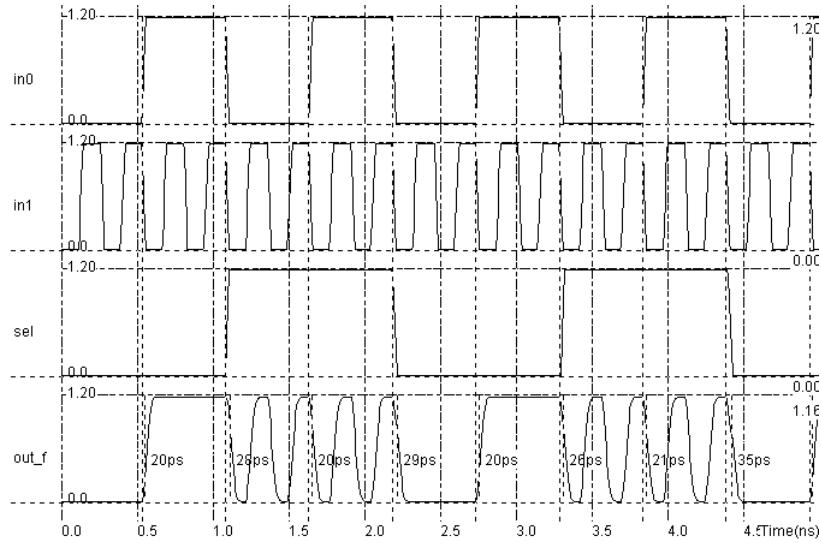


Fig. 6-71. Simulation of the 6-transistor multiplexor with a 10fF load (MUX.MSK)

Still, inputs i0 and i1 are connected to the diffusions of n-channel and p-channel pass transistors, which may lead to varying switching delays. Adding an output buffer and providing isolation from input signals to drain/sources lead to the multiplexor structure (3), presented in figure 6-72. This design is safe, easy to modelize at logic level, but requires many transistors and consumes a lot of power. When a multiplexor with large strength is required, the output inverter

can be modified by enlarging the output buffer width. The structure (4) also implements the multiplexor function, using complex gates for the combined OR/AND function, instead of transmission gates.

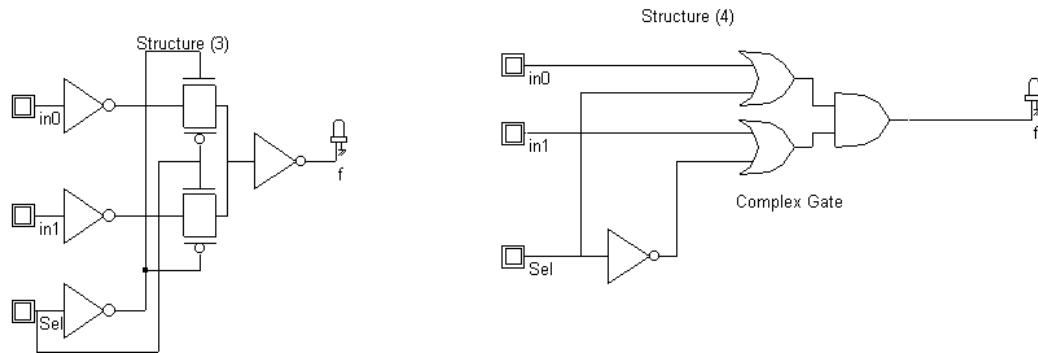


Fig. 6-72. Multiplexor implemented with transmission-gates or with complex gates (MUX.SCH)

### n to 1 Multiplexer

The n-to-1 multiplexor performs the selection of a particular input line among n input lines. The selection is controlled by a set of lines which represent a number *sel*. Normally, there are  $2^m$  input lines and m selection lines whose bit combinations determine which input is selected. Figure 6-73 shows one possible implementation of the 8-to-1 multiplexor, based on an elementary multiplexor as described in paragraph 6-69. A behavioral description of the n-to-1 multiplexor is given below:

```
Case (sel)
  0 : f=in0;
  1 : f=in1;
  2 : f=in2;
  3 : f=in3;
  4 : f=in4;
  5 : f=in5;
  6 : f=in6;
  7 : f=in7;
endcase
```

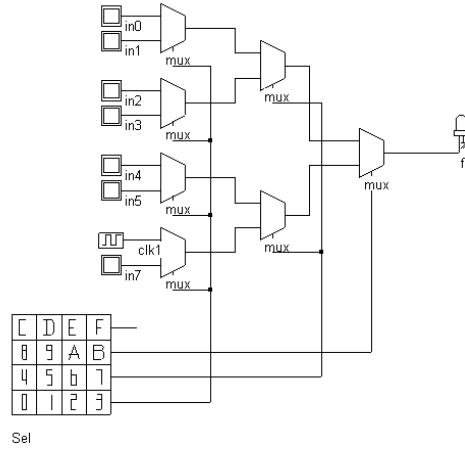
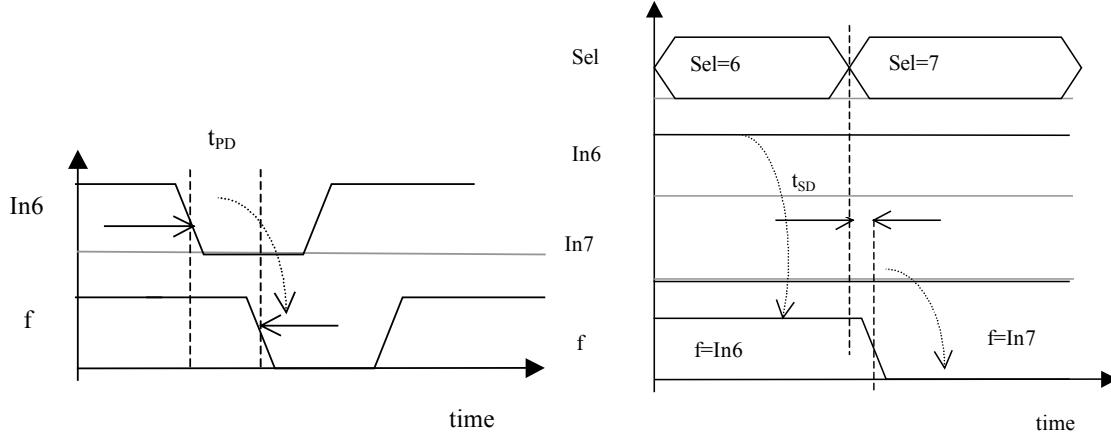


Fig. 6-73 8-to-1 multiplexing based on elementary multiplexor cells(Mux8to1.sch)

### Timing Analysis

One important characteristic of the multiplexor cell is the propagation delay between the change of the input and the corresponding change of the output. The propagation delay is usually named  $t_{PD}$ . Another important delay, named  $t_{SD}$  is measured between the change of the selection and the effective setup of the corresponding channel. These delays are illustrated in figure 6-74.

Fig. 6-74. Definition of the propagation delay  $t_{PD}$  and setup delay  $t_{SD}$ 

The main drawback of the multiplexor design proposed in figure 6-73 is the use of local inverters at each elementary multiplexor gate, that lead to important power consumption and set-up delay. Figure 6-75 shows the direct transmission gate implementation of the 8-to-1 multiplexor. The result is simpler than for the multiplexor implementation, it works faster and requires fewer devices.

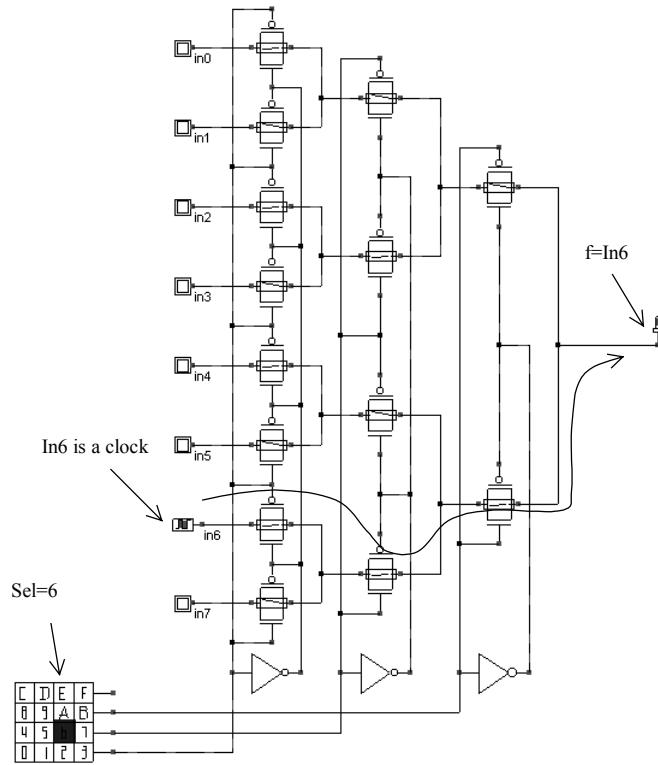


Fig. 6-75. 8-to-1 Multiplexor based on transmission gates (Mux8to1Tgate.SCH)

We have implemented the 8-to-1-transmission gate multiplexor by compiling its Verilog description, as shown in figure 6-77. The layout properties have been changed to consider only the selection of  $In_0$  and  $In_6$  alternatively.  $In_0$  is assigned a fast clock while  $In_6$  is assigned a slow clock. The simulation scenario consists first in multiplexing first the input  $In_0$  to the output  $f$  (Named *pmos\_f* in the layout), with  $Sel1=0$ ,  $Sel2=0$ , and then in multiplexing the input  $In_1$  with  $Sel1=1$ ,  $Sel2=1$ , at time 2.0ns in figure 6-78. The output copies successively  $In_0$  and  $In_1$ , as expected. The eye diagram shows an homogeneous switching delay performance.

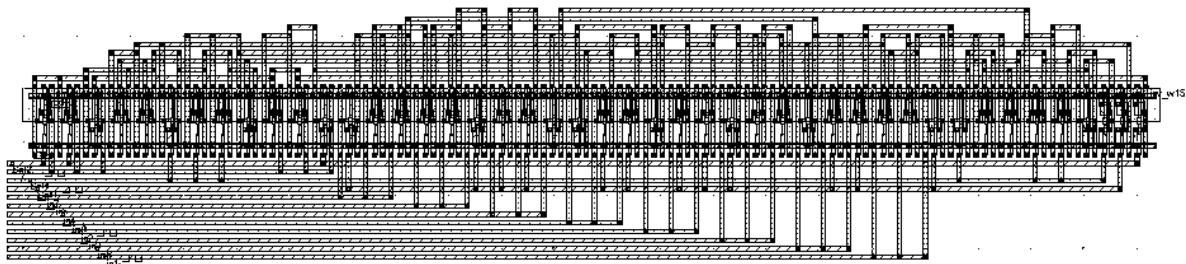


Fig. 6-77. Automatic generation of the 8-to-1 Multiplexor based on transmission gates (Mux8to1Tgate.MSK)

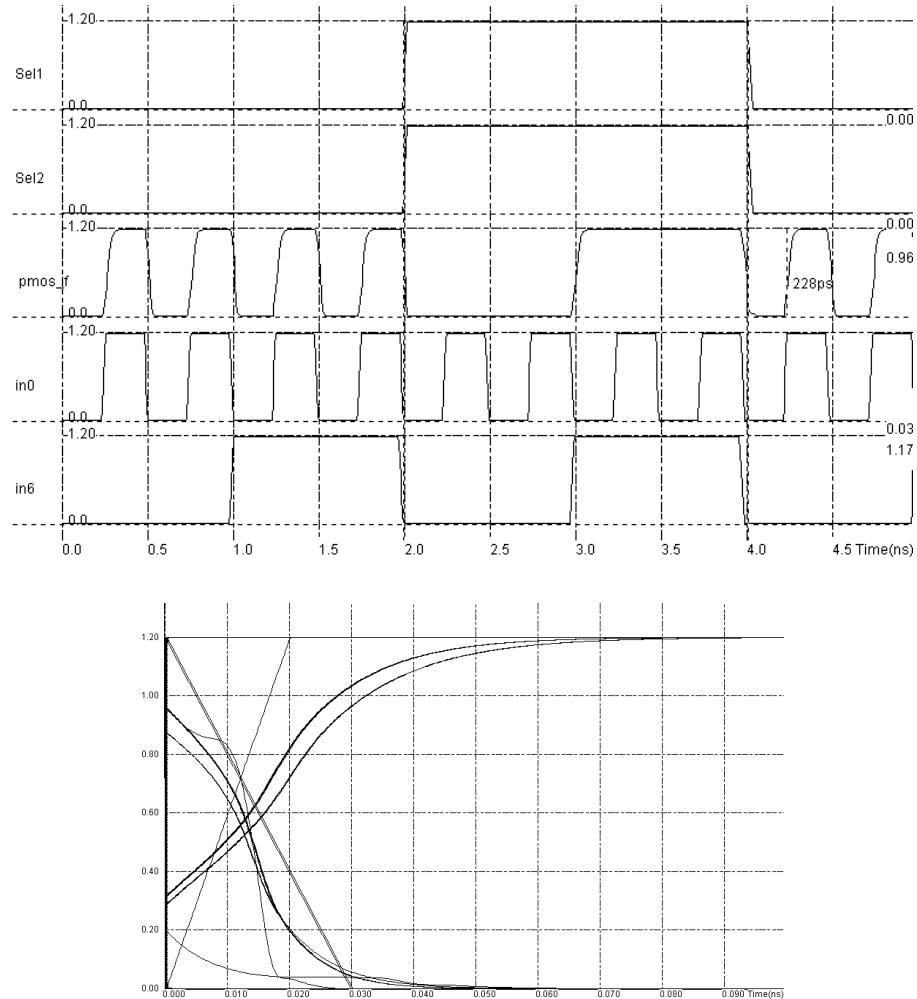


Fig. 6-78 Analog simulation and eye diagram of the 8-to-1 multiplexor (Mux8to1Tgate.MSK)

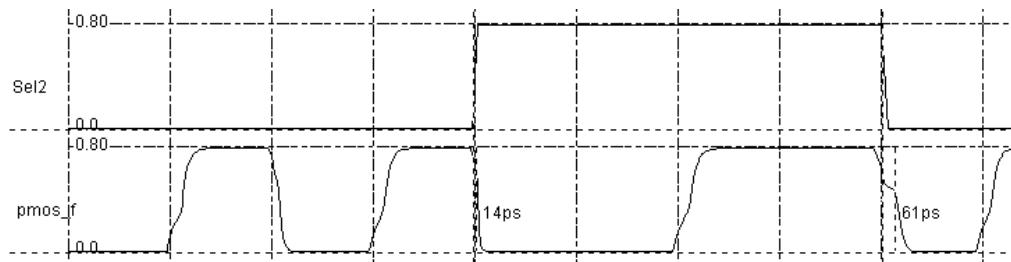


Fig. 6-79 Low voltage operation of the 8-to-1 multiplexor (Mux8to1Tgate.MSK)

If we try to perform the same operation with a voltage supply significantly lower than VDD (Here 0.8V, that is 66% of VDD), the circuit based on transmission gates still operates in a satisfactory way, although significant delays are observed (Figure 6-79).

### All n-MOS multiplexor

Some authors have proposed multiplexor designs only based on n-channel MOS devices, as reported in figure 6-80. Although this architecture is easy to implement in a regular way, the waveform is degraded by the parasitic threshold effect of n-channel MOS devices when passing high signals. Therefore, the output should later be refreshed thanks to a buffer. Although the gain in silicon area is evident, no improvement in switching speed should be expected, as the cascaded threshold loss at each pass transistor leads to very slow rise times.

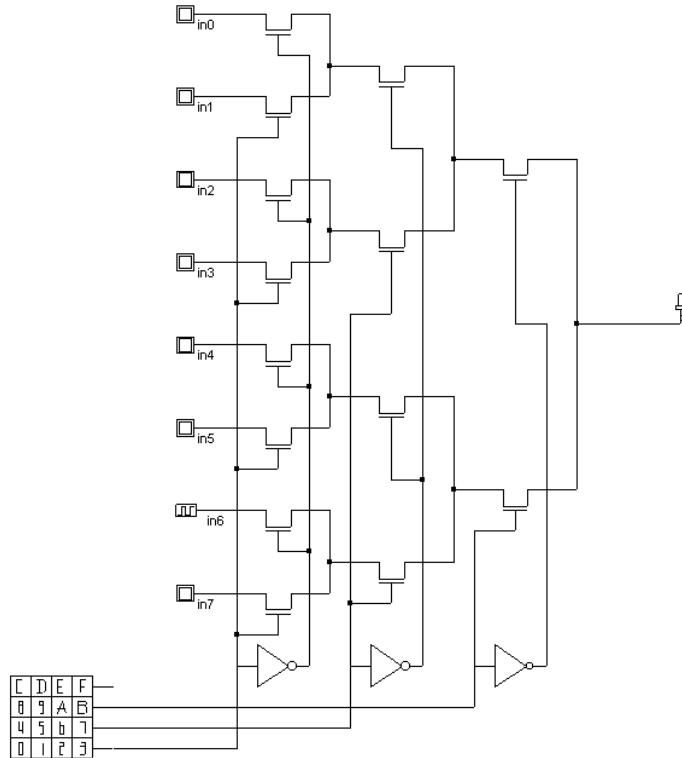


Fig. 6-80. 8-to-1 Multiplexor based on n-channel MOS (*Mux8to1Nmos.SCH*)

We have implemented the 8-to-1 n-channel MOS multiplexor by compiling its Verilog description, as shown in figure 6-81. The layout properties have been changed in a similar way as for transmission gate design. The layout is much more compact as expected. Unfortunately, the switching performances are very poor, which is especially visible in the eye-diagram shown in figure 6-82. When trying to pass the "one", the series of n-channel MOS device degrades the levels considerably. The signal reaches VDD/2 only after 100ps, that is 3 times slower than the transmission gate design.

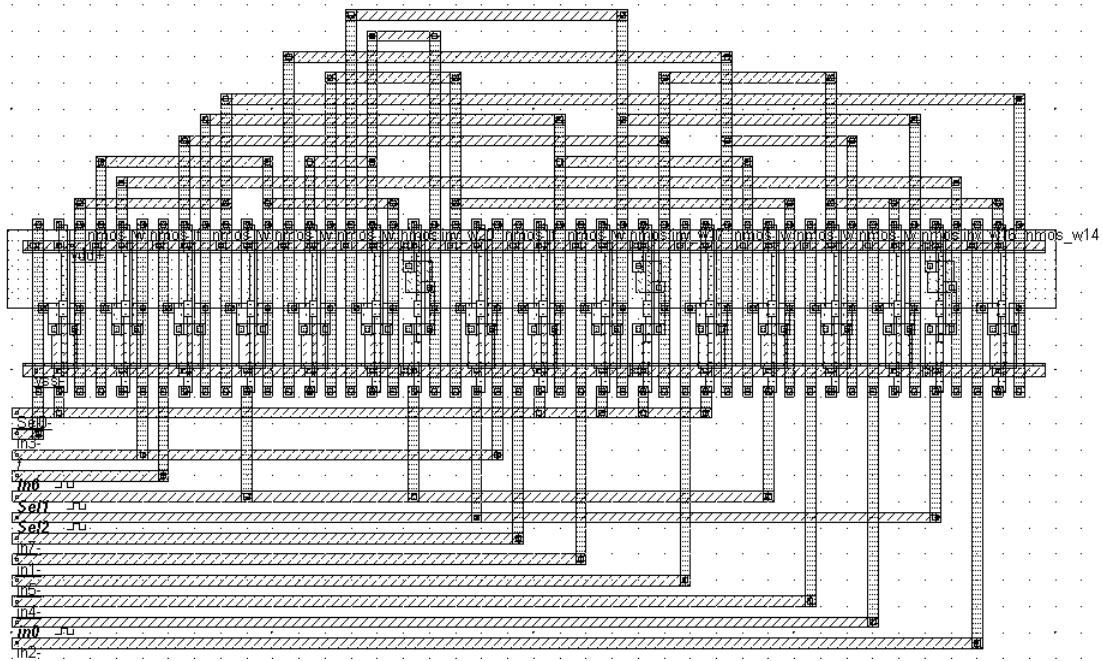


Fig. 6-81. Automatic generation of the 8-to-1 Multiplexor based on n-channel MOS (*Mux8to1Nmos.MSK*)

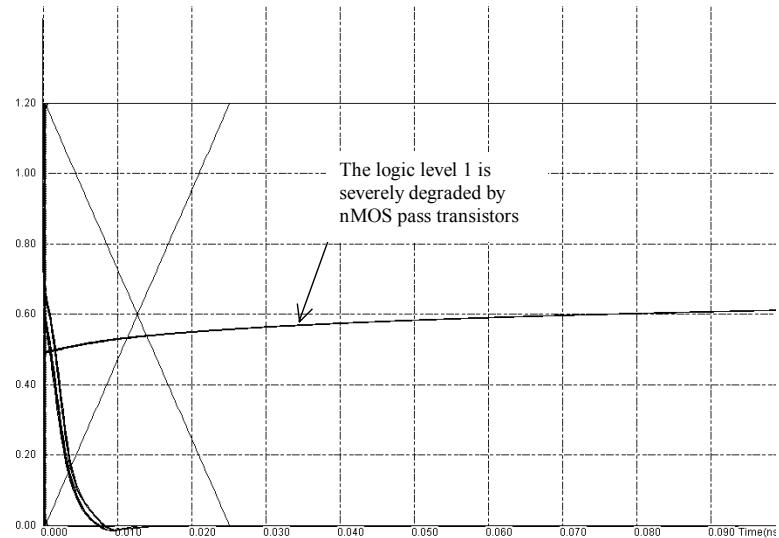


Fig. 6-82. Eye-diagram simulation of the 8-to-1 Multiplexor based on n-channel MOS (Mux8to1Nmos.MSK)

When we try to operate the circuit at low supply voltage (0.8V), the circuit does not work anymore as the final voltage achieved by the output f is as low as 0.3V (Figure 6-83), still below the switching point of most logic gates.

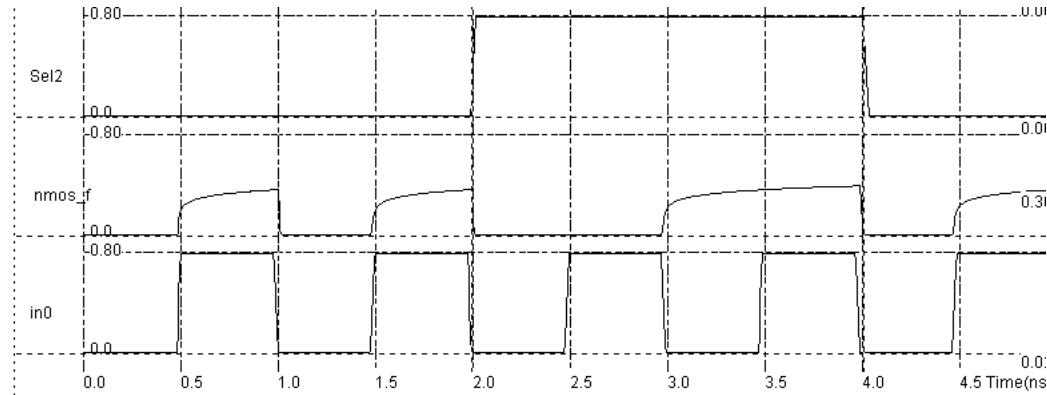


Fig. 6-83. The n-channel multiplexor does not operate at low voltage (Here 0.8V) (Mux8to1Nmos.MSK)

### Demultiplexor

Probably one of the most well-known demultiplexor is the 74LS138 circuit, which is presented in figure 6-84. If the enable circuit is active (G1 must be at 1), all outputs O0..O7 are at 1 except the one issued from the binary decoding of the input A,B and C, according to the statements below.

```

Case (adress)
 0 : O1=0;
 1 : O2=0;
 2 : O3=0;
 3 : O4=0;
 4 : O5=0;
 5 : O5=0;
 6 : O6=0;
 7 : O7=0;
endcase

```

Note that the multiplexor can be implemented using NAND4 cells, to avoid the use of AND logic circuits which require NAND and Inverter circuits. Details on the demultiplexor layout are given in chapter 10 dealing with memory design.

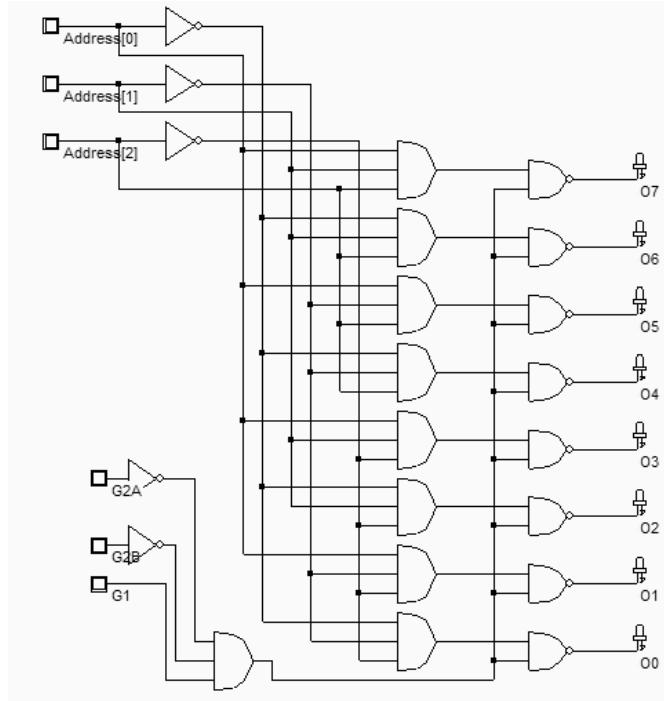


Fig. 6-84. The demultiplexor (74ls138.SCH)

## 11. Shifters

Shifters are very important circuits found in virtually all processor cores. Shifters are able to manipulate data and shift the bits to the right or to the left. Taking the example of an 8-bit input data A, initially set to 0xB3 in hexadecimal (10110011 in binary), the result of shifting A two bits to the right is 0x3c (00101100 binary), as illustrated in figure 6-85. The corresponding symbol is  $\gg$ . Now, the result of shifting A three bits to the left would be 0x98 (10011000 binary). The corresponding symbol is  $\ll$ .

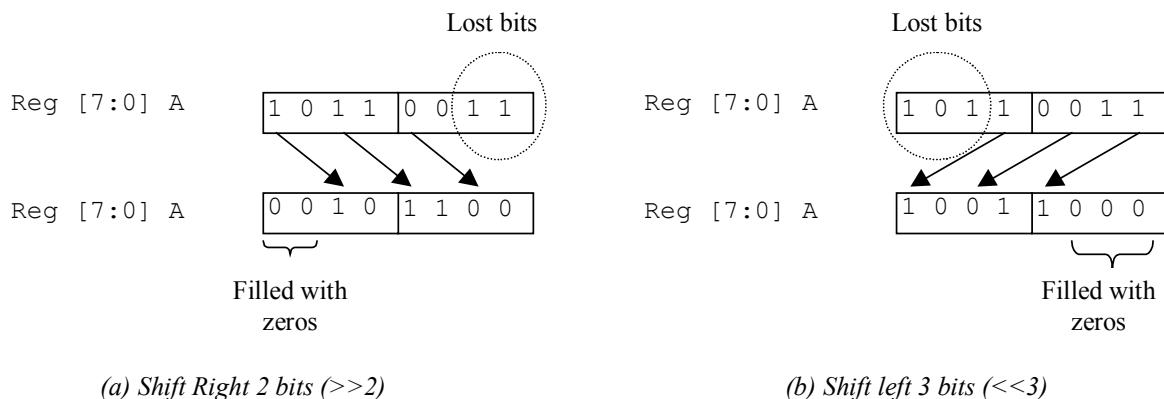
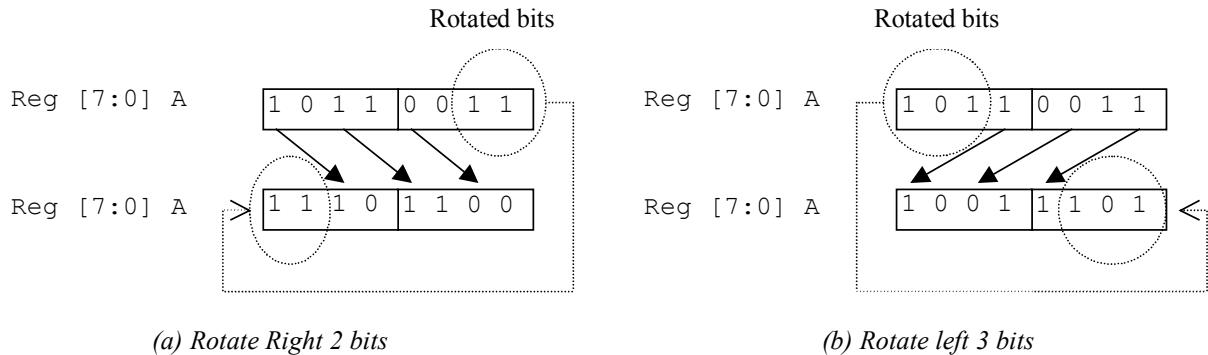


Figure 6-85: Principles for shifting data to the right and to the left

The rotate circuit is based on the shift mechanism, but has the property to re-inject the lost bits in the place left for zeros. An illustration of the rotate structure is given in figure 6-86.



*Figure 6-86: Principles for rotating data to the right and to the left*

There are mainly two ways to implement the shift circuits. One [Weste] is based on multiplexor cells, the other [Uyemura] is based on pass transistors. Pass transistors yield simpler and more regular layout structures. The main drawback is a slower switching and less predictable timing characteristic. The 4-bit shift right circuit is shown in figure 6-87, while the rotate left circuit is reported in figure 6-88. The shifter based on multiplexor has the same structure, except that all single pass transistors are replaced by a pair of nMOS and pMOS in parallel. Each gate control must be inverted, which makes the final circuit significantly more complex than the single MOS shifter.

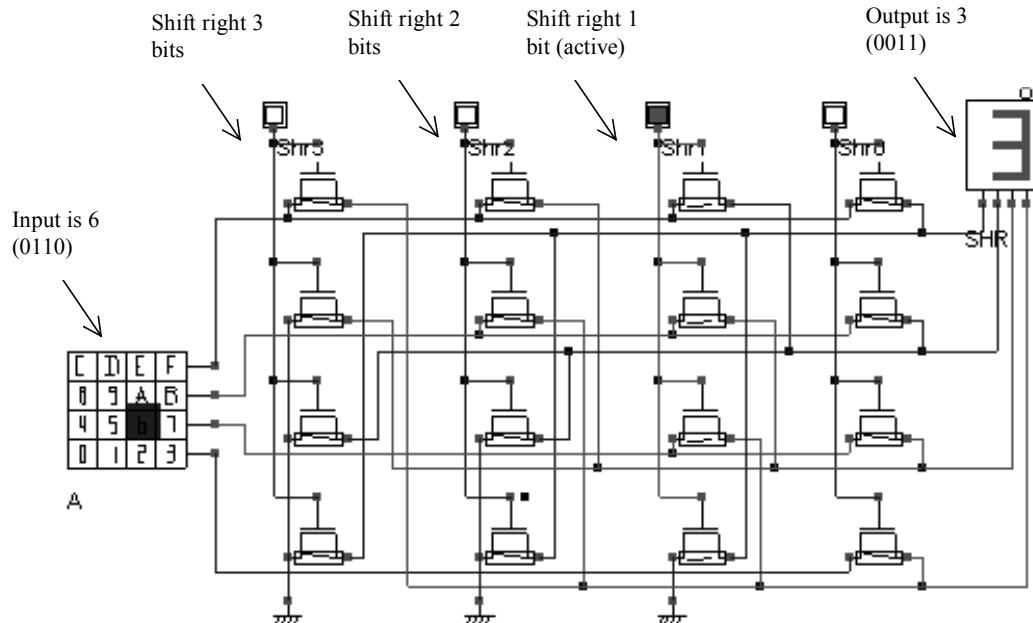


Figure 6-87: Simulation of the shift right circuit (*ShiftRotate4b.SCH*)

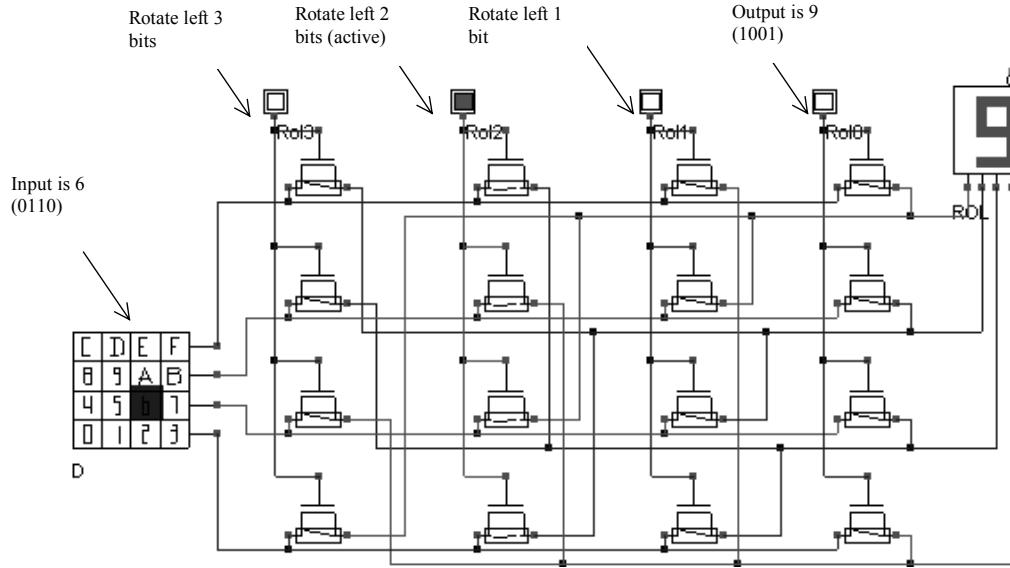


Figure 6-88: Simulation of the rotate left circuit (ShiftRotate4b.SCH)

## 12. Description of basic gates in Verilog

We give in this paragraph a short introduction to Verilog hardware description language (HDL). This language is used by *Microwind* and *Dsch* tools to describe digital logic networks. The description is a text file, which includes a header, a list of primitive keywords for each elementary gate, and node names specifying how the gates are connected together. Let us study how the logic network of Figure 6-89 is translated into a Verilog text.

```

// DSCH 2.5d                                         1
// 14/04/02 16:52:45                               2
// C:\Dsch2\Book on CMOS\Base.sch                  3
//                                                       4
module Base( Enable,A,B,s_BUF,s_NOT,s_NOTIF1,s_AND,s_NAND,      5
s_NOR,s_OR,s_XOR,s_XNOR,s_PMOS,s_NMOS);                      6
  input Enable,A,B;                                              7
  output s_BUF,s_NOT,s_NOTIF1,s_AND,s_NAND,s_NOR,s_OR,s_XOR;    8
  output s_XNOR,s_PMOS,s_NMOS;                                     9
  and #(16) and2(s_AND,A,B);                                       10
  notif1 #(13) notif1(s_NOTIF1,A,Enable);                         11
  not #(10) not(s_NOT,A);                                         12
  buf #(13) buf1(s_BUF,A);                                         13
  nand #(10) nand2(s_NAND,A,B);                                    14
  nor #(10) nor2(s_NOR,B,A);                                      15
  or #(16) or2(s_OR,B,A);                                         16
  xor #(16) xor2(s_XOR,B,A);                                       17
  xnor #(16) xnor2(s_XNOR,B,A);                                    18
  nmos #(10) nmos(s_NMOS,A,Enable); // 1.0u 0.12u                19
  pmos #(10) pmos(s_PMOS,A,Enable); // 2.0u 0.12u                20
endmodule                                                 21

// Simulation parameters                                22
// Enable CLK 10 10                                     23
// A CLK 20 20                                         24
// B CLK 30 30                                         25
//                                                       26

```

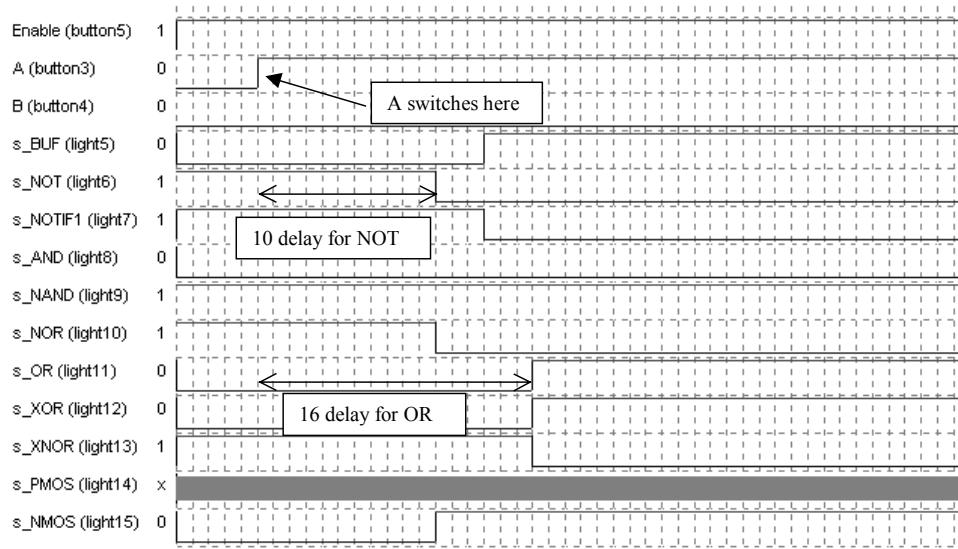
Fig. 6-89. The Verilog description of the circuit BASE.SCH (Base.TXT)

The Verilog description is a text file. Double slash characters are used for comments. The rest of the line is ignored. Consequently, lines 1 to 3 are comments. In line 5, the keyword **module** defines the start of the description, which will end after the keyword **endmodule** (line 21). The name of the module is **base**. Its parameters are the list of input and output signals. From line 7 to line 9, signals are declared as **input**, **output**, or internal **wire**. No internal wire exists in this file. Three input and 11 output signals are listed. Then, from line 10 to 20, the schematic diagram is described as a list of primitives.

Name	Logic symbol	Verilog primitive
INVERTER		Not <name>(out,in);
BUFFER		Buf <name>(out,in);
TRI-STATE INVERTER		Notif1 <name>(out,in,enable);
TRI-STATE BUFFER		Bufif1 <name>(out,in,enable);
AND		And <name>(out,in1,in2,...);
NAND		Nand <name>(out,in1,in2,...);
OR		Or <name>(out,in1,in2,...);
NOR		Nor <name>(out,in1,in2,...);
XOR		Xor <name>(out,in1,in2,...);
XNOR		Xnor <name>(out,in1,in2,...);
NMOS		Nmos <name>(out,source,gate);
PMOS		Pmos <name>(out,source,gate);

Table 6-8. The VERILOG primitives supported by DSCH and MICROWIND

The gate delay is described for the accurate simulation of time delays in large structures. The logic delay through each gate is specified in integer units. One unit corresponds to a single logic simulation cycle. The correspondence between the elementary cycle and the real time varies depending on the technology. In 0.12µm, which is the default directory used by DSCH, the elementary unit is 0.01ns, that is 10ps. Consequently, the AND gate described in line 10 has a switching delay 16x0.01ns=0.160ns, or 160ps. This delay is observed between each active transition of A,B and s\_AND.



*Fig. 6-90. Illustration of the elementary logic delay in the simulation chronograms (Base.SCH)*

In the simulation chronograms, when we zoom strongly on the time scale in order to see each delay step (0.01ns), we observe the discrete aspect of the delay estimation, as shown in figure 6-90. A series of 10 elementary delay is used for the NOT cell, and 16 for the OR cell, according to the Verilog description made in figure 6-89.

The gate delay description is optional. If no delay information has been specified, the logic simulator assigns a default gate delay, which is a basic parameter of the technology. In 0.12 $\mu$ m, the default gate delay is 0.03ns, and each wire connected to the gate adds a supplementary 0.07ns delay. A more detailed description of the Verilog language may be found in [Uyemura].

## 13. Conclusion

In this chapter, the design of basic cells has been reviewed. We have described in details the NAND, AND, OR and NOR gates, and the asymmetrical switching problems of multiple-input NOR circuits. The specific design techniques to design a compact XOR gate has then been reviewed. The techniques for translating AND/OR combinations into an optimized have also been studied. Finally, we have described the structure of multiplexors, shifters, and given some information about the VERILOG format used by DSCH and MICROWIND to exchange structural descriptions of logic circuits.

## REFERENCES

- [Weste] Neil Weste, K. Eshraghian "Principles of CMOS VLSI design", Addison Wesley, ISBN 0-201-53376-6, 1993
- [Baker] R.J. Baker, H. W. Li, D.E. Boyce "CMOS circuit design, layout and simulation", IEEE Press, ISBN 0-7803-3416-7, 1998
- [Uyemura] John.P. Uyemura "Introduction to VLSI Circuits and Systems", Wiley, 2002, ISBN 0-471-12704-3

## Exercises

Exercise 4.1. Give the switching point voltage of the gate shown in figure 6-91. Find its logic function.

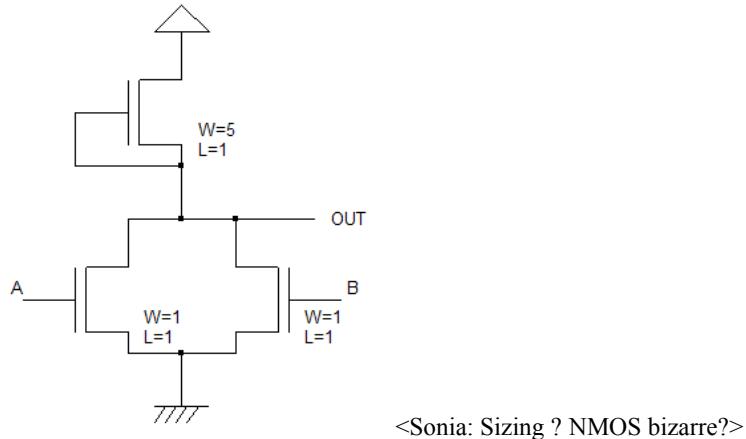


Figure 6-91: A logic gate case study

*Answer: <Sonia>*

Exercise 6.2: Design a CMOS 3-input XOR using CMOS And-OR-Invert logic.

*Answer: See appendix F*

Exercise 6.3: Design the following function using CMOS And-OR-Invert logic and try to find continuous diffusions.

$$F = \sim(A \& B \mid C \& (A \mid B))$$

*Answer: See appendix F*

Exercise 6.4: Using Microwind , compare the switching point voltage of a 3-input NOR gate (minimum size MOS) to the switching point voltage of a 3 inputs NAND gate (minimum size MOS). Which one is closer to the ideal and why?

*Answer: The Nand switching point voltage is closer to VDD/2 because the electron mobility is higher than that of the hole.*

Exercise 6.5: What is the functionality of the circuit shown in figure 6-93 ? Justify with a chronogram using Dsch2.

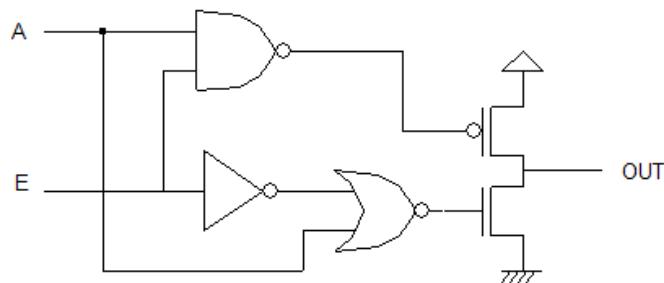


Figure 6-93: logic circuit case study

Exercise 6.6: What is the functionality of the circuit shown in figure 6-94 ? Justify with a chronogram using DSCH2. Implement this circuit into layout. What are the conditions to match the logic behavior?

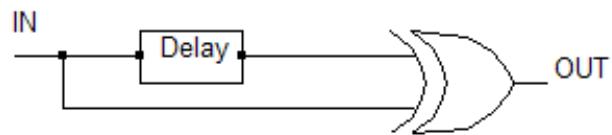


Figure 6-94: A logic gate case study

# 7 Arithmetics

This chapter introduces basic concepts concerning the design of arithmetic gates. Firstly, we illustrate data formats. Secondly, 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.

## 1. Data formats

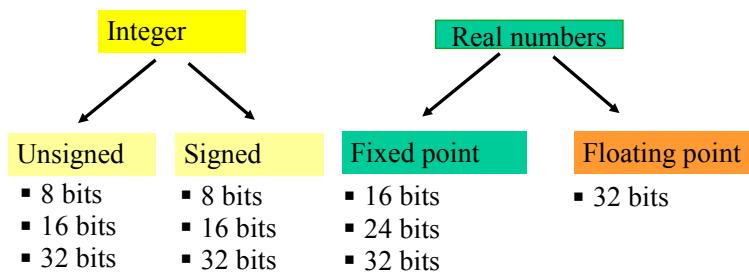


Figure 7-1: Most common data formats used in ASIC designs

The two classes of data formats are the integer and real numbers (Figure 7-1). 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. In particular cases, other formats are used, such as the exotic 24 bit in some optimized applications, such as in application-specific digital signal processors.

### Integer Format

Type	Size (bit)	Usual name	Range
Unsigned integer	8	Byte	0 .. 255
	16	Word	0 .. 65535
	32	Long word	0 .. 4294967295
Signed integer	8	Short integer	-128 .. +127
	16	Integer	-32768 .. + 32767
	32	Long integer	-2147483648 .. + 2147483647

Table 7-1: Size and range of usual integer formats

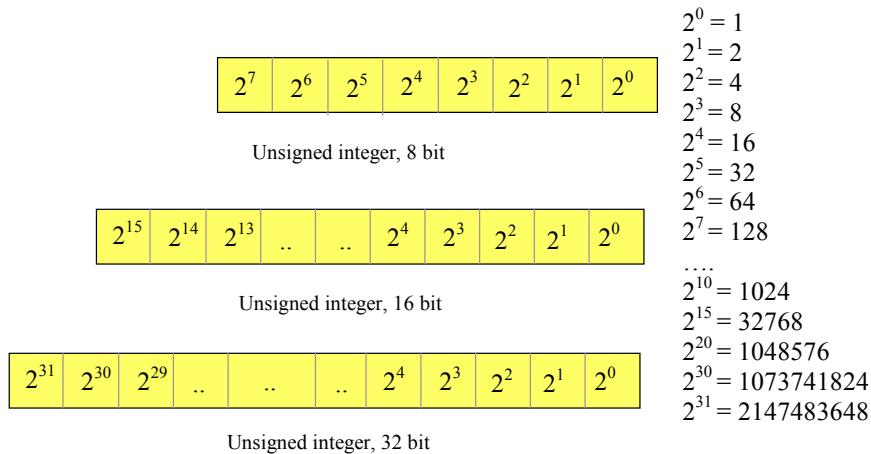


Figure 7-2: Unsigned integer format

A summary of integer formats is reported in table 7-1. The signification of each bit is given in figure 7-2. The unsigned integer format is simply the series of power of 2. As an example, the number 01101011 corresponds to 107, as detailed in equation 7-1.

$$01101011 = 2^6 + 2^5 + 2^3 + 2^1 + 2^0 = 64 + 32 + 8 + 2 + 1 = 107 \quad (\text{Eqn. 7-1})$$

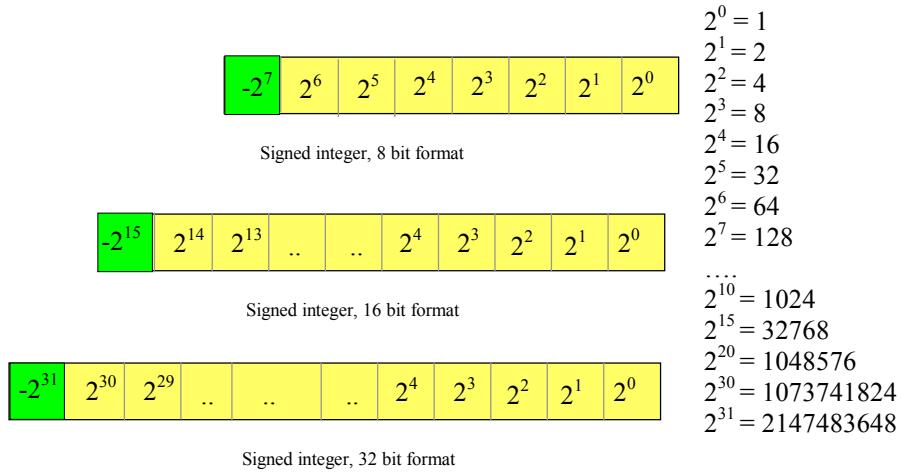


Figure 7-2: Signed integer format

The signed integer format uses the left-most bit for the sign. The coding of the data works as for the unsigned integer, except that the left-most bit accounts for a negative number. In 16 bit format, a 1 in the sign bit equals to -32768. As an example, the 8-bit signed number 11101011 is detailed in equation 7-2. The sign bit appears in the sum as -128, all the other bit remaining positive.

$$11101011 = -2^7 + 2^6 + 2^5 + 2^3 + 2^1 + 2^0 = -128 + 64 + 32 + 8 + 2 + 1 = -21 \quad (\text{Eqn. 7-2})$$

### Real Format

A second important class of numbers is the real format. In digital signal processing, real numbers are often implemented as fixed point numbers. The key idea is to restrict the real numbers within the range [-1.0..+1.0] and to use arithmetic hardware that is compatible with integer hardware. More general real numbers are coded in a 32 bit format. A summary of real formats is reported in table 7-2.

Type	Size (bit)	Usual name	Range
Fixed point	16	Fixed	-1.0..+1.0 (Minimum 0.00003)
	32	Double fixed	-1.0..+1.0 (Minimum 0.0000000046)
Floating point	32	Real	-3.4 <sup>e</sup> 38..3.4 <sup>e</sup> 38 (Minimum 5.8 <sup>e</sup> -39)

Table 7-2: Size and range of usual real formats

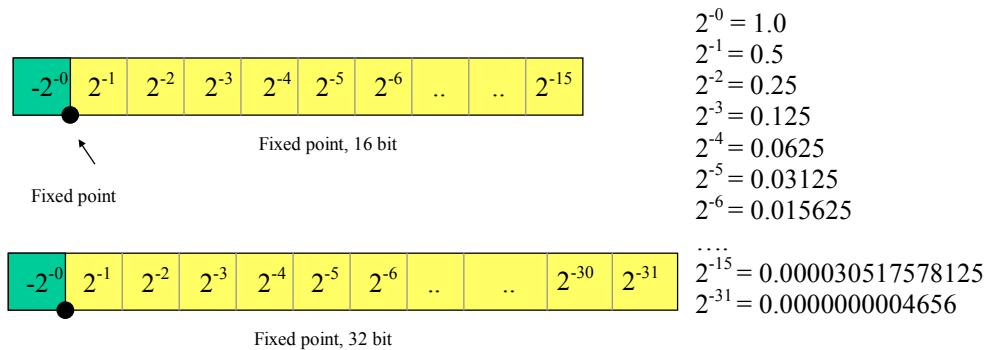


Figure 7-3: Fixed point numbers in 16 and 32 bit format

In the case of fixed point arithmetic, we read bits as fractions in negative power of 2 (Example of equation 7-3). When the left-most bit is set to 1, it accounts for -1.0 in the addition (Equation 7-4). The main limitation of this format is its limited range from -1.0 to 1.0. Its main advantage is a hardware compatibility with integer circuits, leading to low power computing, a particularly attractive feature for embedded electronics. As an example, most digital signal processing of mobile phones work in fixed point arithmetic.

$$01100100 = 2^{-1} + 2^{-2} + 2^{-5} = 0.5 + 0.25 + 0.03125 = 0.78125 \quad (\text{Equ. 7-3})$$

$$11100100 = -2^0 + 2^{-1} + 2^{-2} + 2^{-5} = -1.0 + 0.5 + 0.25 + 0.03125 = -0.21875 \quad (\text{Equ. 7-4})$$

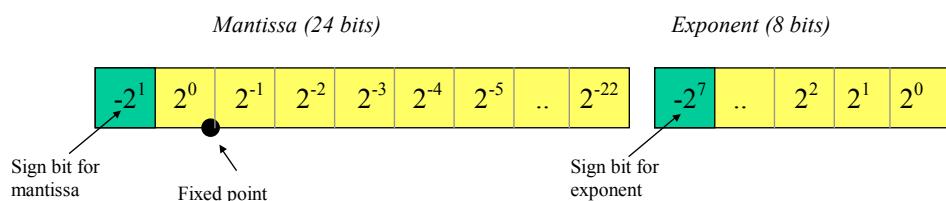


Figure 7-4: Floating point arithmetic format

Finally, floating point data is coded using a mantissa multiplied by an exponent (Figure 7-4). The general formulation of the real number format is as follows:

$$\text{data} = \text{mantissa}.2^{\text{exponent}} \quad (\text{Eq. 7-5})$$

The illustration of this format is given in equation 7-6. Numbers may range from  $-3.4 \times 10^{38}$  to  $3.4 \times 10^{38}$ .

$$\begin{aligned} (0110100\ldots)(0..101) &= (2^0 + 2^{-1} + 2^{-3}) \times (2^2 + 2^0) \\ &= (1.0 + 0.5 + 0.125) \times (4 + 1) \\ &= 1.625 \times 2^5 \\ &= 52.0 \end{aligned} \quad (\text{Eq. 7-6})$$

## 2. The adder circuit

Let's consider two 8-bit unsigned integers A and B. These numbers range from 0 to 255. The arithmetic addition of the two numbers is given in equation 7-7.

$$S = A + B \quad (\text{Eq. 7-7})$$

To obtain the arithmetic addition of these numbers, we build elementary circuits which realize the bit-level arithmetic addition, as described below. Remember that if  $a_0=1$  and  $b_0=1$ , a carry bit is generated in the next column. We need to take into account the carry bit  $c_i$  for each column, by creating a circuit with three logic inputs  $a_i, b_i$  and  $c_{i-1}$  to produce  $s_i$  and  $c_i$ . This circuit is called full adder. The adder in column 0 is a more simple circuit, which realizes the addition of two logic data  $a_0$  and  $b_0$ , which is called the half adder.

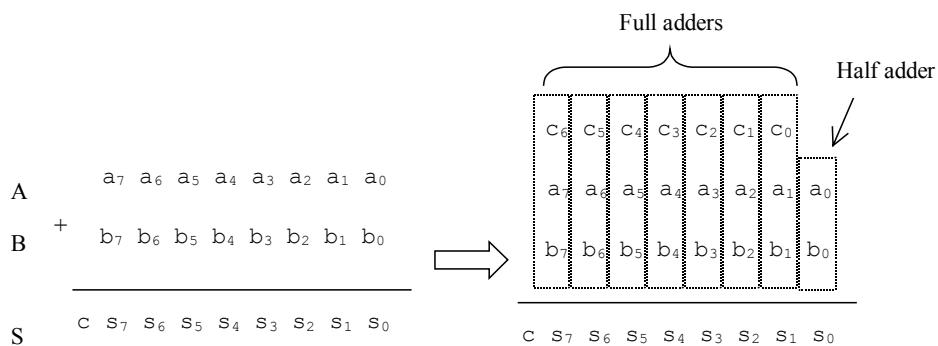


Figure 7-5: 8-bit unsigned integer addition principles

### From Logic Design to Layout

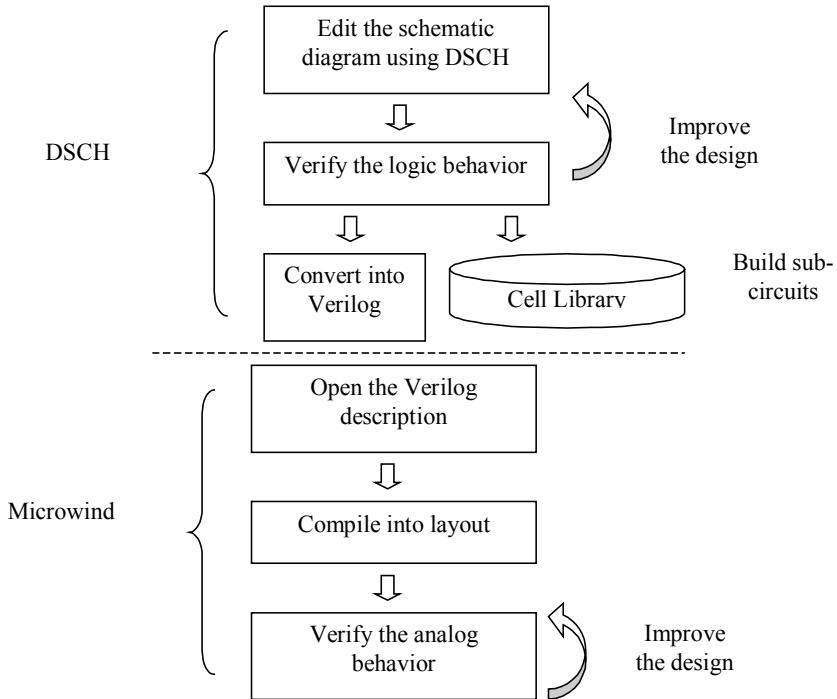


Figure 7-6: Design flow from logic design to layout implementation

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 7-6. The schematic diagram constructed using DSCH is validated first at logic level. At that level, timing analysis is available as well as power consumption estimation. The accuracy of these predictions is quite fair as accurate layout information is still missing. The designer iterates on his circuit until the specified performances are achieved. At that point, it is of key importance to build a sub-circuit and feed a cell library. These sub-circuits may be reused in other designs, if the user provide sufficient information on the performances of the cells.

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. This means that the layout result will significantly change whether we use 0.8 $\mu$ m or 0.12 $\mu$ m 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.

### 3. Adder Cell Design

In this section, we use the design method presented previously to build an 8-bit adder.

#### Half Adder

The Half-Adder gate truth-table and schematic diagram are shown in Figure 7-7. The SUM function is made with an XOR gate, the Carry function is a simple AND gate. When both A and B are at 1 (In black in figure 7-7), the *carry* output is asserted, and the *sum* is at zero. The combination of carry and sum is equal to (10), which is equivalent to 2 in binary format.

A	B	Carry	Sum	Result
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	0	2

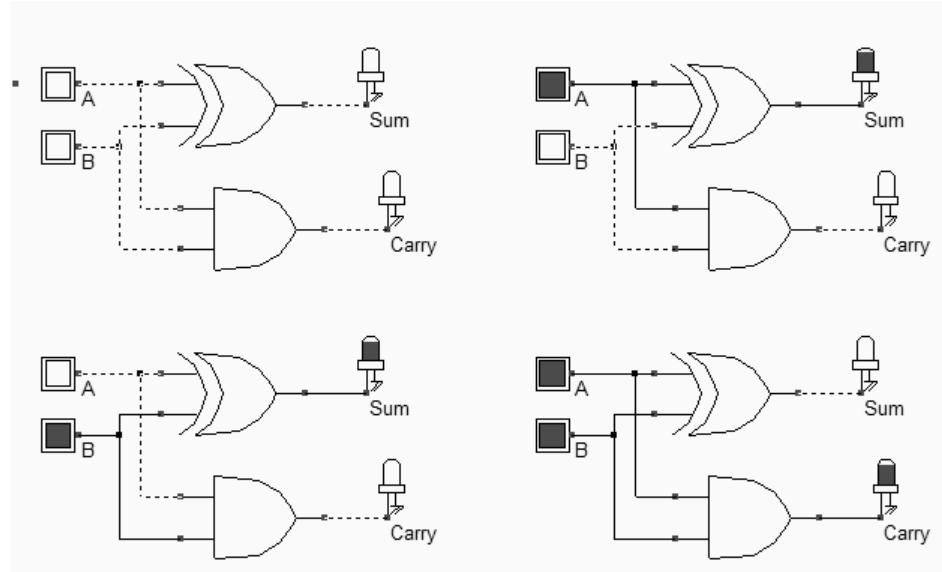


Fig. 7-7. Truth table and schematic diagram of the half-adder gate (halfAdderTest.SCH).

The correct behavior of the adder is proven by the values of the display, which show a result in accordance with the truth table. The command **File → Schema to New Symbol** in DSCH enables to create a sub-circuit. This technique is useful to construct hierarchical designs, by embedding a complete circuit into a single component. The command menu is reported in figure 7-8. By a click on the OK button, a symbol **HalfAdder.SYM** is created. This symbol includes the VERILOG description of the circuit, that is the XOR and AND gate, as seen in the left part of the window. Once created, the symbol may be instantiated in any logic design.

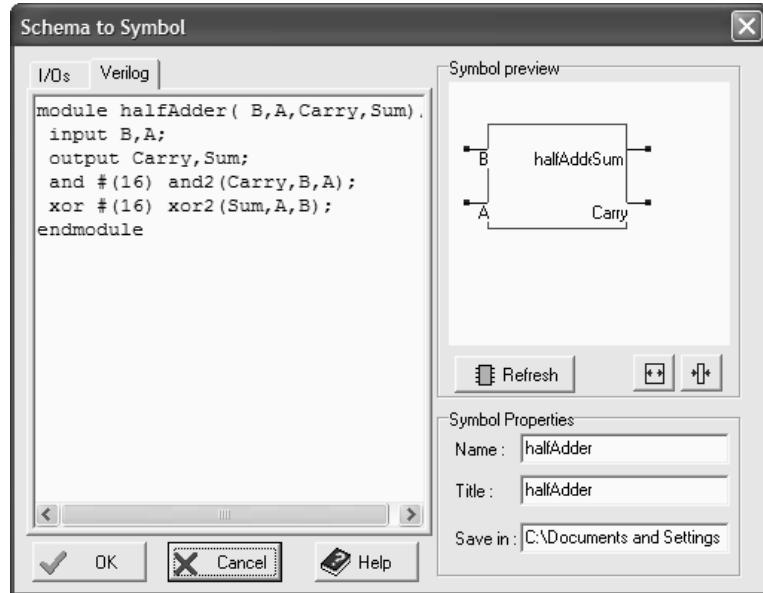


Fig. 7-8. Creating a half-adder symbol (HalfAdder.SYM)

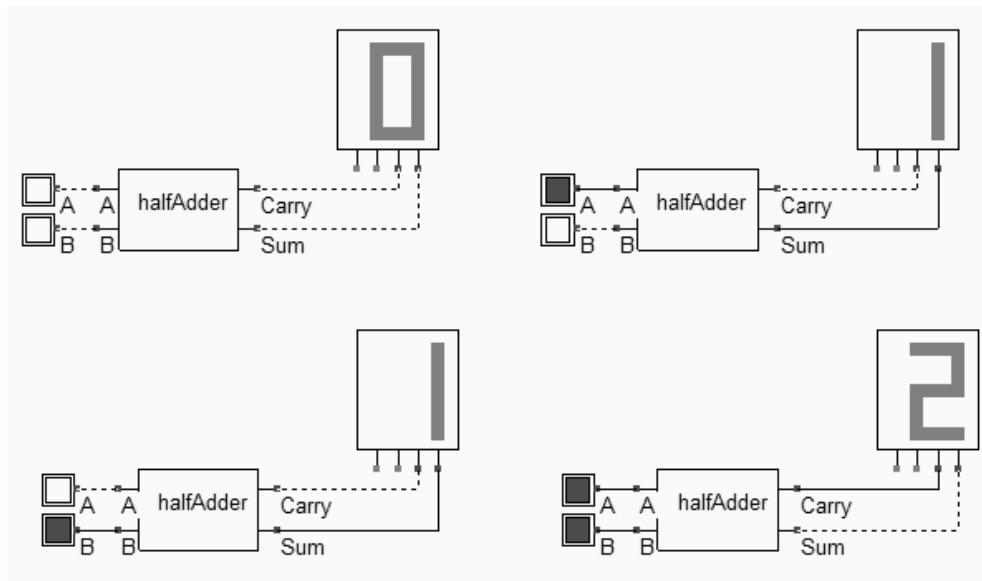


Fig. 7-9. Logic validation of the half-adder symbol (halfAdderTest.SCH).

The logic validation of the half adder is provided in the simulation of figure 7-9. When the symbol was created, a Verilog text was also generated, namely **HalfAdder.TXT**, In Microwind2, the same Verilog text file is used to construct the corresponding layout automatically. Just invoke the command **Compile → Compile Verilog File**, select the corresponding text file and click **Compile**.

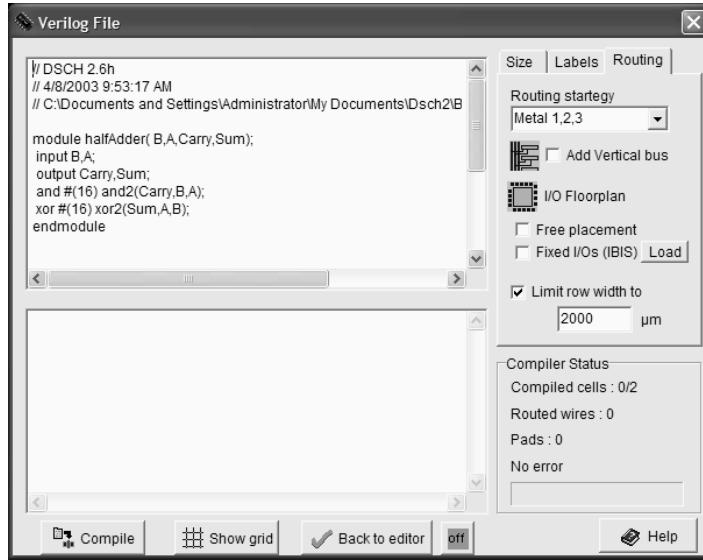


Fig. 7-10 Using the half adder VERILOG description to pilot the layout compiling in MICROWIND (halfAdder.TXT)

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. Click on **Simulate → Start Simulation**. The timing diagrams of figure 7-11 appear where the truth table of the half-adder can be verified.

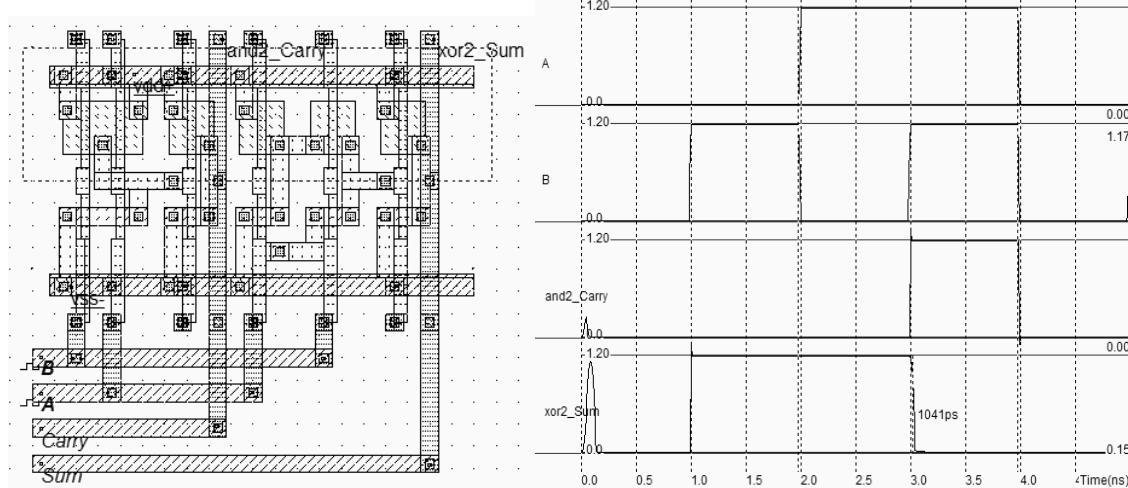


Fig. 7-11. Compiling and simulation of the half-adder gate (Hadd.MSK)

### Full-Adder Gate

The truth table and schematic diagram for the full-adder are shown in Figure 7-12. The most straightforward implementation of the CARRY cell is to use a combination of AND, OR gates. Using negative logic equivalence, the AND,OR combination becomes a series of NAND gates, as suggested by equation 7-8. Concerning the sum, one common approach consists in using a three-input XOR gate. This 3-input XOR is usually constructed from two stages of 2-input XOR.

A	B	C	Carry	Sum	Result
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	1
0	1	1	1	0	2
1	0	0	0	1	1
1	0	1	1	0	2
1	1	0	1	0	2
1	1	1	1	1	3

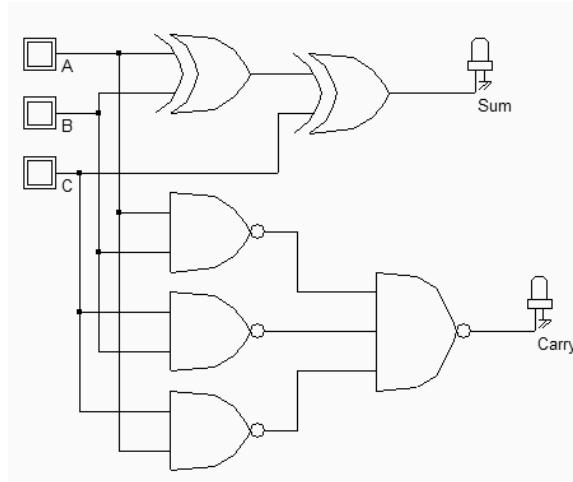


Fig. 7-12. The truth table and schematic diagram of a full-adder(FADD.SCH)

$$\text{sum} = A \wedge B \wedge C$$

$$\text{carry} = (A \& B) | (A \& C) | (B \& C) \quad (\text{Equ 7-8})$$

$$\text{carry} = \overline{(A \& B)} \& \overline{(A \& C)} \& \overline{(B \& C)}$$

### Full-Adder using Complex gate

A more efficient circuit consists in the use of complex gates. We apply this technique for the *carry* cell: rather than assembling NAND gates (Total  $3 \times 4 + 6 = 18$  transistors), we assemble MOS devices in AND/OR combinations. The carry function built using MOS in parallel or in series is presented in figure 7-13. A tiny re-arrangement of the Boolean expression leads to a 12 transistor complex gate (Equation 7-9) rather than a 14 transistor one, if we include the 2 MOS devices of the final stage inverter. The carry circuit shown in figure 7-13 is strictly equivalent to the carry circuit of figure 7-12. We avoid the needs for cascaded NAND gates, and so the number of transistors is lower.

$$\text{carry} = (A \& B) | (C \& (A | B)) \quad (\text{Equ 7-9})$$

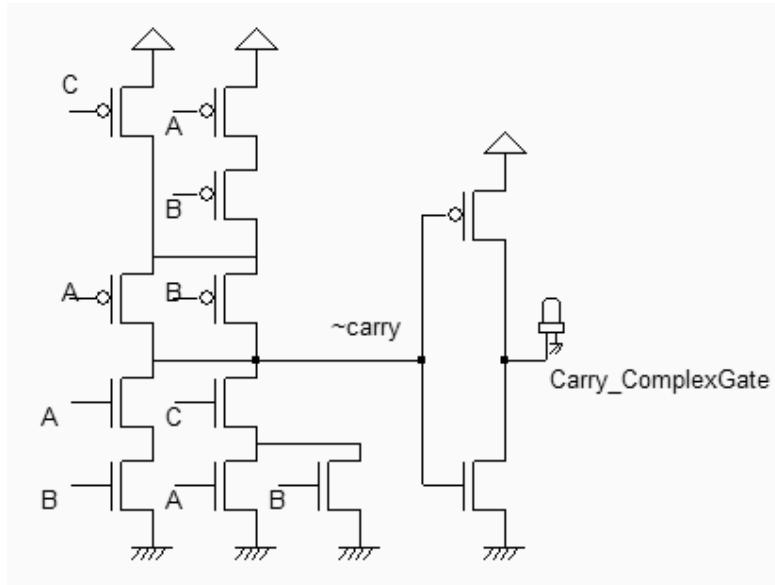


Fig. 7-13. The carry cell based on a complex gate requires less transistors (Fadd.SCH)

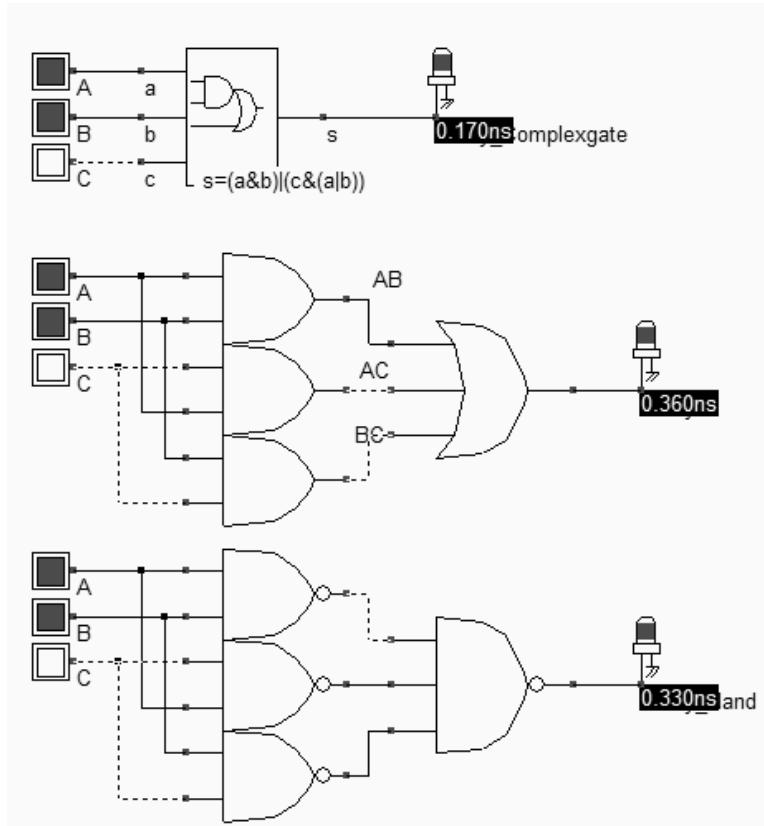


Fig. 7-14. Comparing the switching performances of the carry cells (Carry.SCH)

A switching speed comparison is proposed in figure 7-14: the upper cell is the fastest, as the number of stage is restricted to the complex gate itself, with short internal connections and small diffusion areas. The cell in the middle is the slowest due to the four internal stages (The AND is an AND plus an inverter). The lower circuit is a little faster, but not so. The reason is that DSCH automatically assigns a typical delay (Around 70ps) to each interconnect. At this

stage, DSCH do not make the difference between a short and a long interconnect. Very probably, the interconnect between the NAND2 output and the NAND3 input will be very short and the delay is severely overestimated. But it could happen that the routing is quite large, in that case the supplementary delay is justified.

### Full-Adder using Complex gate

The procedure to generate the symbol of the full-adder from its schematic diagram is the same as for the half adder. When invoking **File → Schema to new symbol**, the screen of figure 7-15 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, loosing the benefits of complex gate approach in terms of cell density and switching speed.

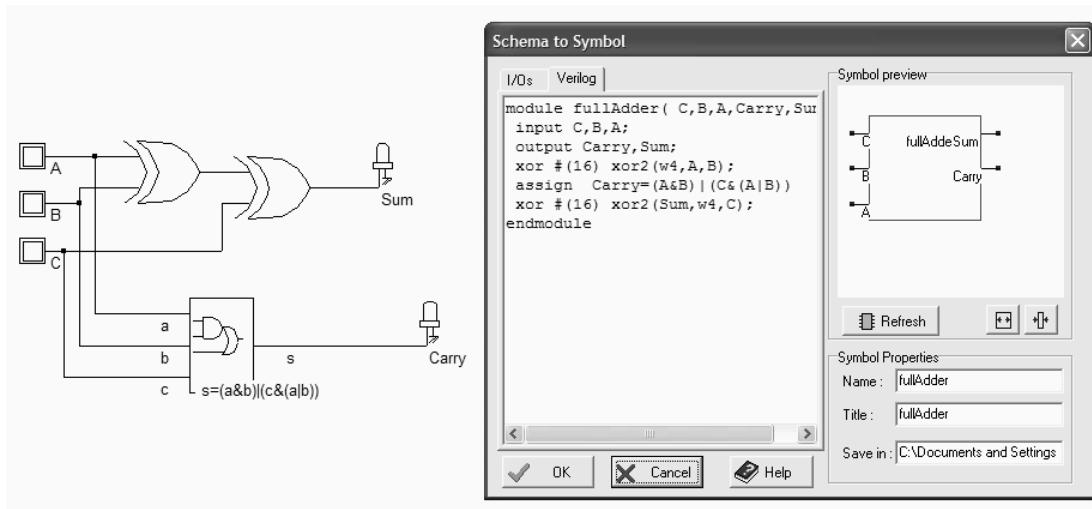


Fig. 7-15 Verilog description of the full adder (*FullAdder.SYM*)

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 7-16 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=70ns, A=B=C=1, the output is 3 after a little delay.

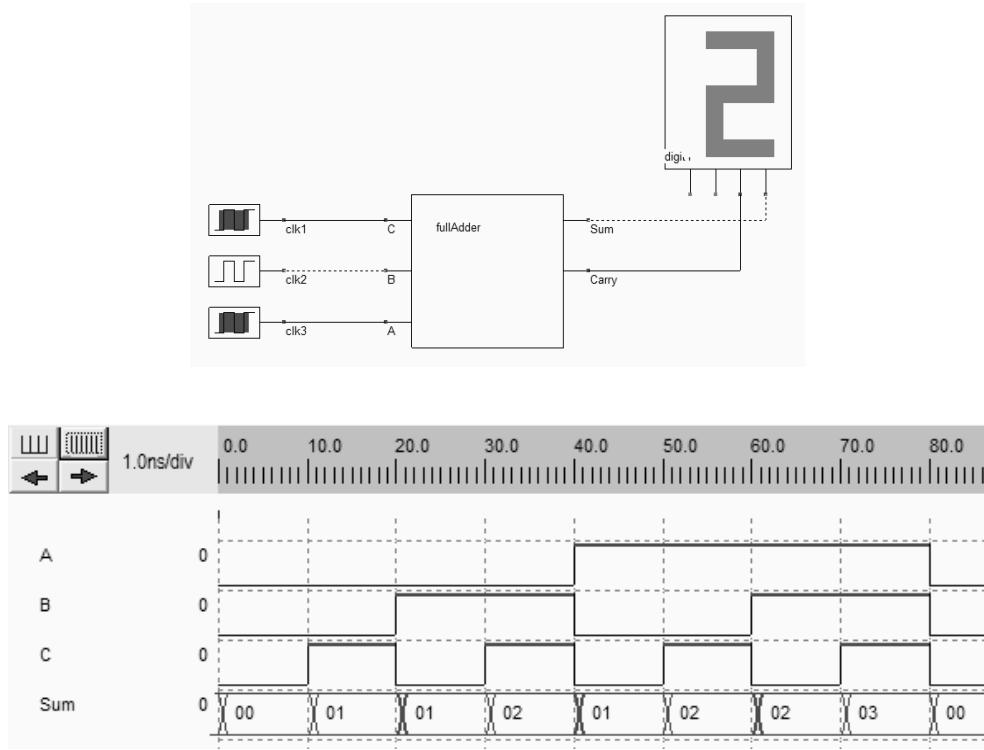


Fig. 7-16. Testing the new Adder symbol using clocks (FaddTest.SCH)

### The Full-Adder Adder

You may create the layout of the full-adder by using the Verilog cell compiler of Microwind. It is recommended to compile the full-adder symbol which includes the complex gate, for an optimum result. Microwind handles complex gate descriptions and has the ability to convert AND/OR logic combinations into a compact layout. The full-adder compiled layout is shown in Figure 7-17. By default, all signals are routed to the left side of the logic cells, for clarity. In real-case designs, the routing creates connections in all directions, depending on the position of signals inputs and outputs.

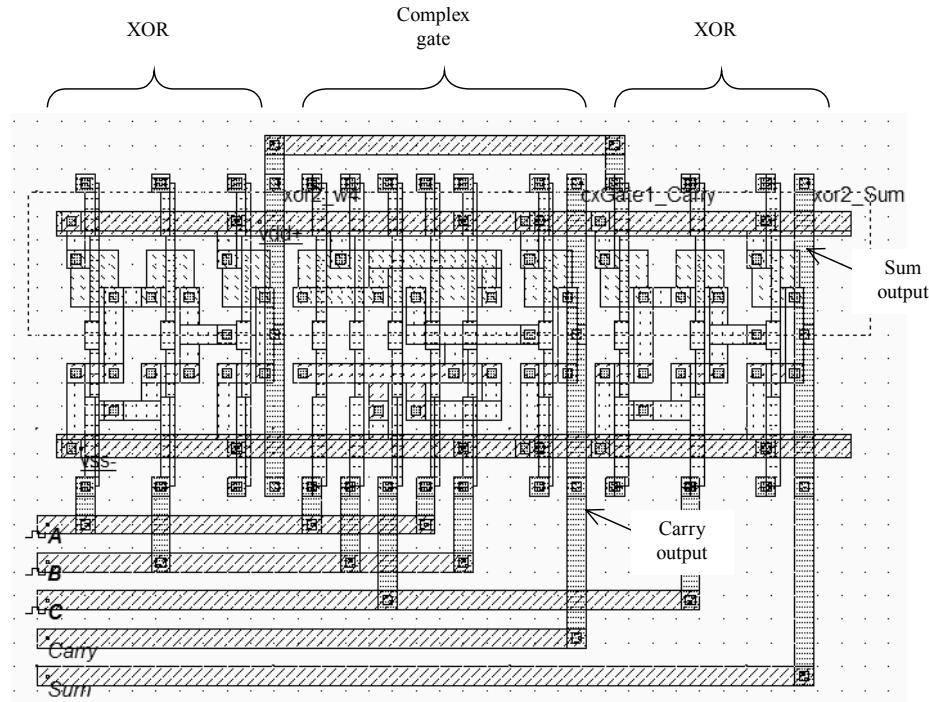


Fig. 7-17 The compiled full-adder (FullAdder.MSK).

The complex gate implementation includes some interesting design techniques to achieve a compact layout. The layout and schematic diagram are detailed in figure 7-18. The cell compiler has organized the contacts to ground and metal bridge in such a way that the n-diffusion and p-diffusion areas are continuous. The MOS arrangement is electrical equivalent to the schematic diagram of figure 7-13. Notice that the diffusion has been stretched to build an internal connection, in the n-MOS area. Diffusion is rarely used as an interconnect due to its very huge parasitic resistance. In that case, however, the role of this diffusion connection is not significant.

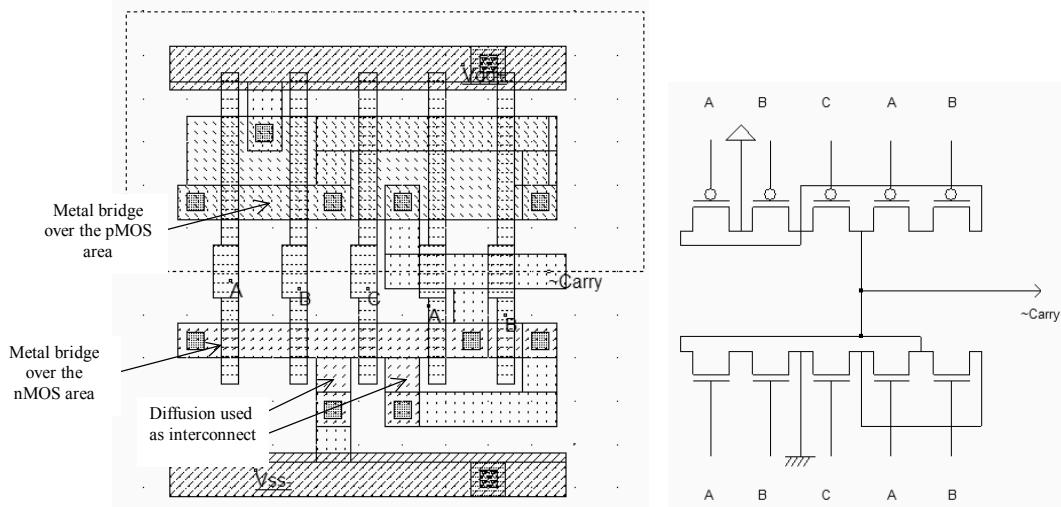


Fig. 7-17 The carry cell layout and associated structure (CarryCell.MSK).

There is no need to add the clock properties in the layout as the clock signals properties have been extracted from the Verilog text, and automatically placed in the compiled layout. The clock description in Verilog format appears after the keyword "Always". Three clocks are declared: A,B and C. The text signifies that for a given time step, the logic signal is inverted. For example clock C switches from 0 to 1 at time 1000, from 1 to 0 at time 2000, etc.. There exist a time scale conversion between DSCH and Microwind. A logic time step of 1000 is transformed into 1.0ns.

```

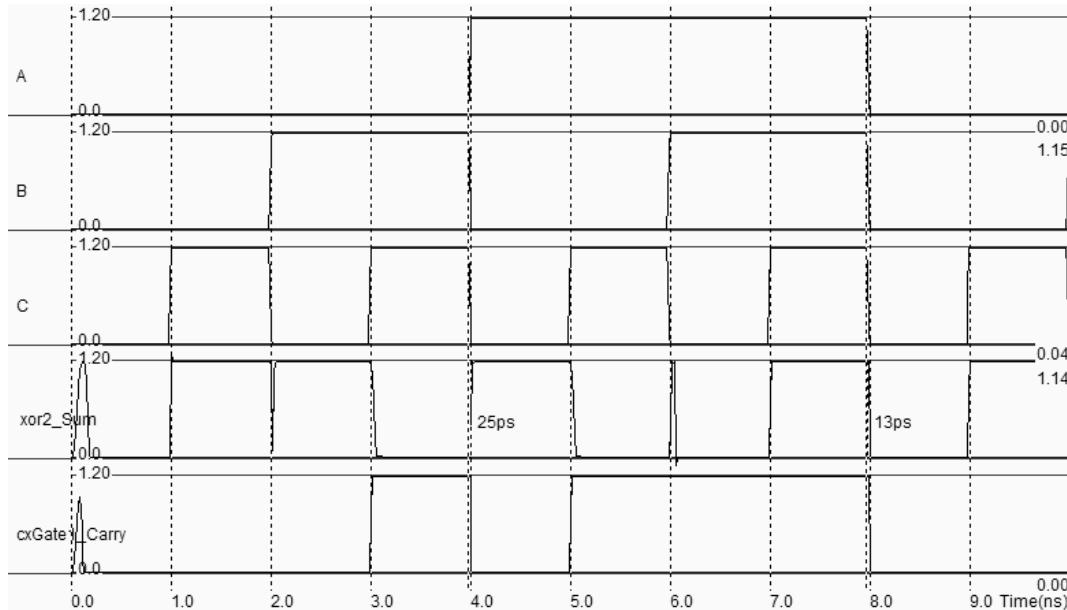
module fullAdder( C,B,A,Carry,Sum);
    input C,B,A;
    output Carry,Sum;
    xor #(16) xor2(w4,A,B);
    assign Carry=(A&B) | (C&(A|B));
    xor #(16) xor2(Sum,w4,C);
endmodule

// Simulation parameters in Verilog Format
always
#1000 C=~C;
#2000 B=~B;
#4000 A=~A;

```

*Fig. 7-18: the declaration of clocks in the Verilog text (FullAdder.TXT)*

The simulation of the full-Adder, shown in figure 7-18, complies with the initial truth-table and the logic simulation. Notice the glitch at time t=6.0ns due to internal transient switching of the logic circuit.



*Fig. 7-19: Simulation of a full-adder (File fullAdder.MSK).*

## 4. Ripple Carry Adder

In this section, the design of the adder is addressed in its simplest approach, where the carry of one stage propagates as an input for the next stage. This type of circuit based on a carry chain is called "ripple carry" adder.

### Structure of the Ripple Carry Adder

The 4-bit ripple-carry adder circuit includes one half adder and 3 full-adders in serial, according to the elementary addition shown in figure 7-20 [Belaouar]. Each stage produces the Boolean addition of three logical information. For example, the Boolean output  $s[1]$  is the addition of input signals  $a[1]$  and  $b[1]$ , together with the internal carry  $c[1]$ . Some examples of 4-bit addition are given in table 7-4. We give the output  $s$  in hexadecimal and decimal format.

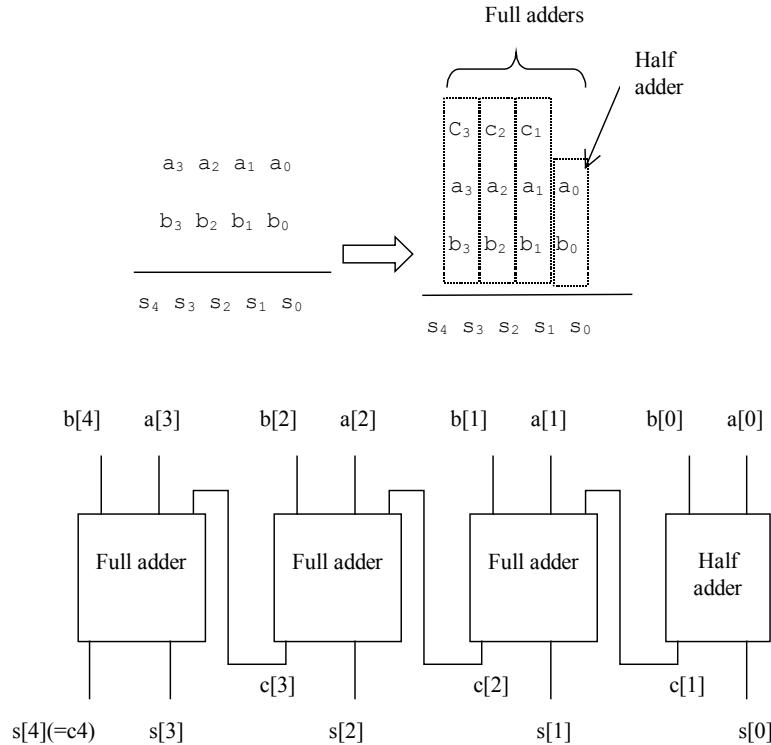


Fig. 7-20. Structure of the 4-bit ripple-carry adder

$a[0..3]$	$b[0..3]$	$s[0..4]$ (hexa)	$s[0..4]$ (Decimal)
0	0	00	0
0	F	0F	15
9	7	10	16
6	C	12	18
F	F	1E	30

Table 7-3. Adder result examples

The logic circuit shown in figure 7-4 allows a four-bit addition between two numbers  $a[0..3]$  and  $b[0..3]$ . The user symbols 'Hadd.sym' and 'Fadd.sym' are added to the design using the command **Insert → User Symbol**. In DSCH2, the numbers  $a$  and  $b$  are generated by keyboard symbols, which produces at the press of a desired key a 4-bit logic value. In the example shown in figure 7-21, the value of  $a$  is 1 (0001 in binary), and the value of  $b$  is F (1111 in binary form, or 15 in decimal). The result  $s$ , which combines  $s[0]$ ,  $s[1]$ ,  $s[2]$ ,  $s[3]$  and the last carry bit, is equal to 0x10 in hexadecimal, or 16 in decimal.

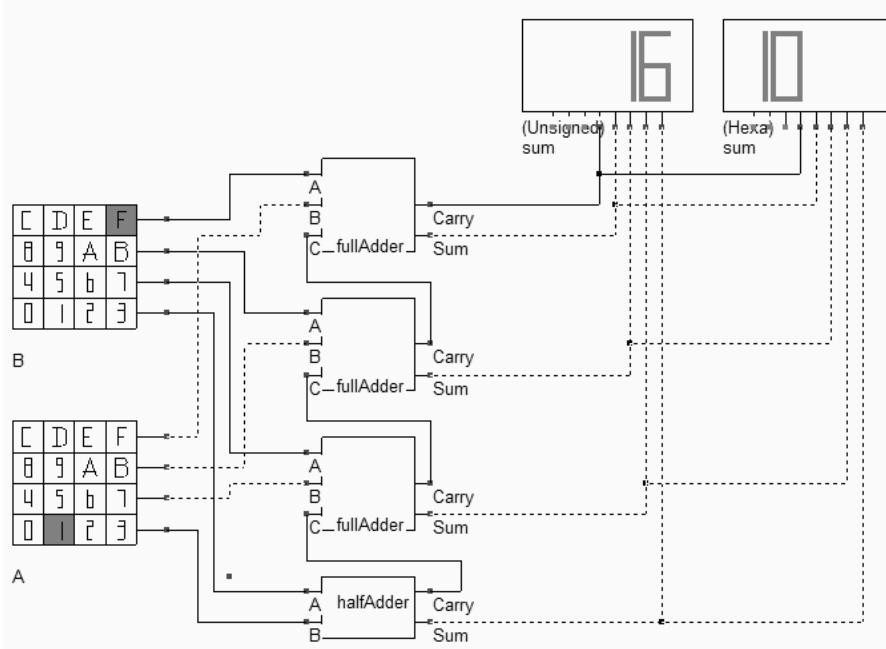


Fig. 7-21. Schematic diagram of the four-bit adder and some examples of results (Add4.SCH).

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, as shown in figure 7-22. The default option is the hexadecimal format. The unsigned integer format is used in the schematic diagram of figure 7-22 for the left display. Integer and fixed point display formats are used for arithmetic circuits. The ASCII character corresponding to the 8-bit input data may also be displayed.

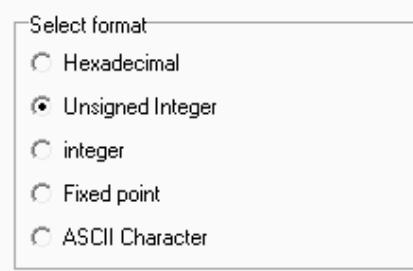


Fig. 7-22. The display symbol has five format options (Add4.SCH).

### Critical Path

The worst case delay of the circuit is calculated in Dsch by the command **Simulate→ Find critical Path→In the Diagram**. In the ripple carry circuit, the critical path passes through the carry chain. The predicted worst case delay is 0.8ns.

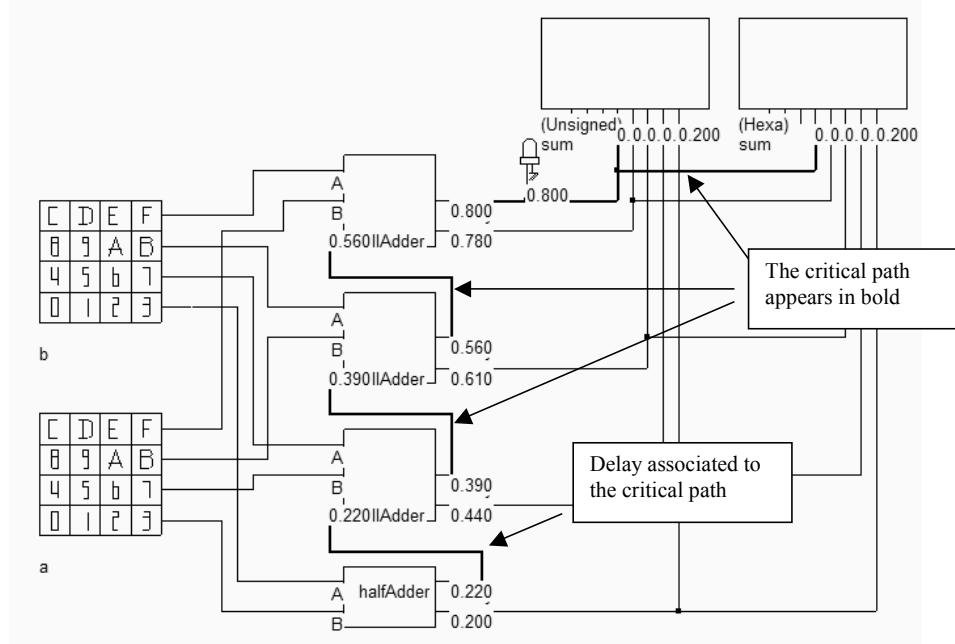


Fig. 7-23. The critical path of the four-bit adder (ADD4.SCH).

### A manual design of the 4-bit Ripple Carry Adder

We described in this paragraph an example of manual design of a 4-bit ripple carry adder. The elementary adder circuit **FullAdder.MSK** from figure 7-17 is reused. We tried to compact the interconnect network by rerouting the input interconnects over the cell, and shortened the carry wires. The modified layout is **fadd.MSK**. A proposed strategy for connecting adders together in order to build a ripple carry adder is detailed in figure 7-24. The cell placement, supply and I/O routing positioning ease the connection between blocks, leading to a compact design and short connections. Each adder is supplied by VDD and VSS rails that are connected all together on the right side of the bloc. The carry propagation is realized by a short interconnect flowing from the bottom to the top of each cell. The *a* and *b* data are routed on the left side of the cell, the result on the right side.

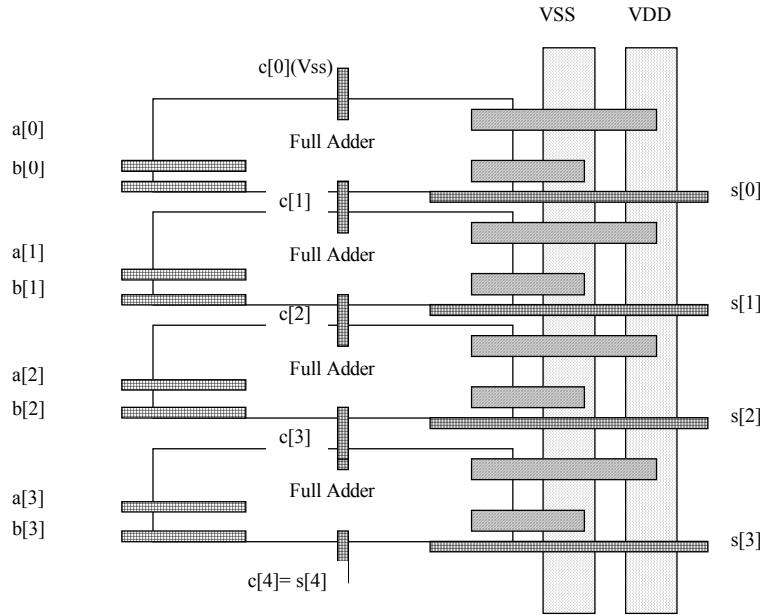


Fig. 7-24. The floor planning of the 4-bit ripple carry adder (ADD4.MSK).

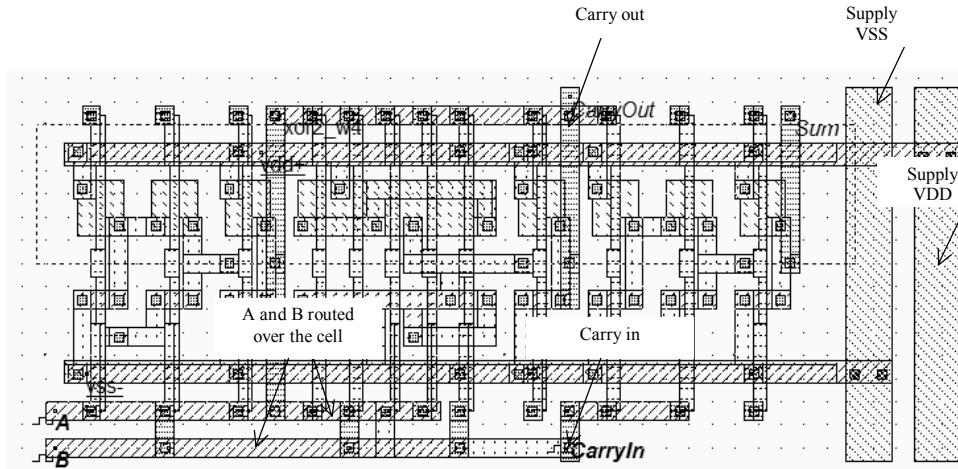


Fig. 7-25. Arranging the full adder to create a compact layout and to ease further connection (fadd.MSK)

Figure 7-26 details the four-bit adder layout based on the manual cell design of the adder. In Microwind2, the command **Edit → Duplicate X,Y** has been used to duplicate the full-adder layout vertically. The input and output names use brackets with an index ( $A[0]..A[3]$ ,  $B[0]..B[3]$  and  $Sum[0].. Sum [4]$ ), so that Microwind automatically displays the logic value corresponding to A,B and Sum. For example, at time  $t=1.5\text{ns}$  (Figure 7-27),  $A=3, B=1$ ,  $\text{sum}=4$ .

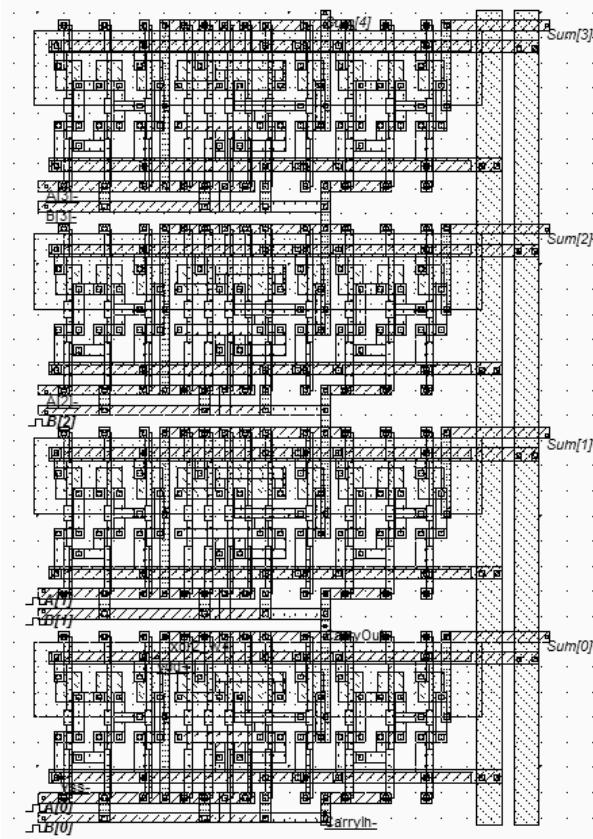


Fig. 7-26. Implementation of the four-bit adder (Add4.MSK).

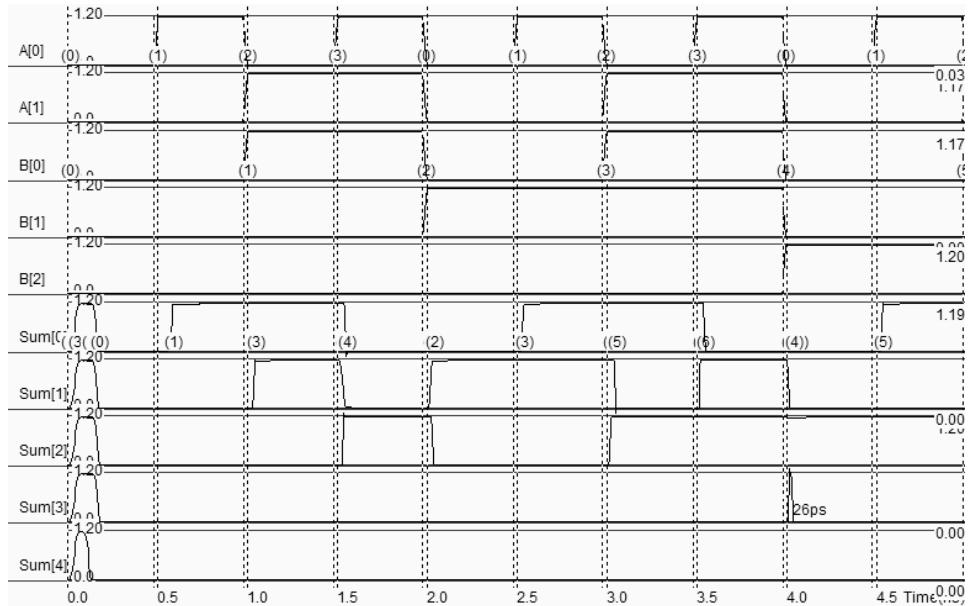


Fig. 7-27. Simulation of the four-bit adder (Add4.MSK).

A convenient method for characterizing the worst case circuit delay is to use the **Eye Diagram** simulation. This simulation mode superimposes output signals at any rise or fall edges of inputs A0..A3 or B0..B3. The cumulative drawing of the outputs is called the eye diagram. Figure 7-28 gives the eye diagram of the 4-bit ripple carry adder. It

can be seen that the worst case delay is around 80ps. Remember that the critical delay was 0.8ns in the logic simulation. How do we explain that difference? First of all, because the layout level simulation does not investigate all input configurations, specifically the ones where the critical delay is involved. The inputs A2, A3, and B3 are inactive, and set to 0. Secondly, the logic simulation assigned a default 70ps delay per interconnect. This average delay is overestimated in our case, as most interconnects are routed very short.

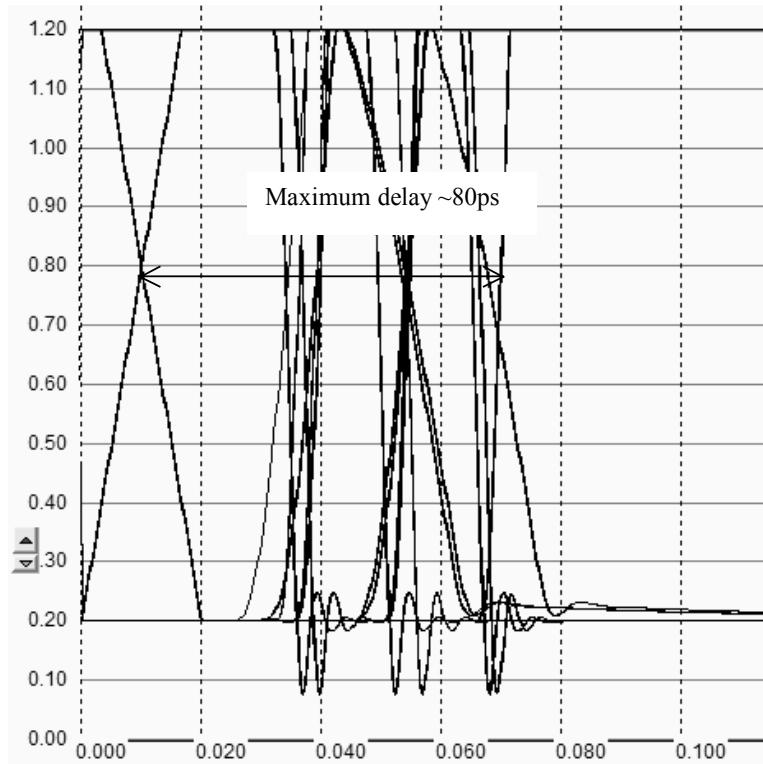
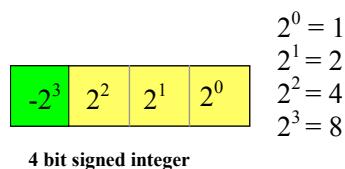


Fig. 7-28. Eye diagram of the four-bit adder (Add4.MSK).

## 5. Signed Adder

The signed integer format for a 4-bit input data is specified in table 7-6. The correspondence between unsigned and signed data for a 4-bit information is also reported. A specific symbol, namely **KbdSigned.SYM**, is available to support the generation of 4-bit signed input. The symbol may be loaded using the command Insert → User Symbol. Select the symbol **KbdSigned.SYM** in the list, as shown in figure 7-29. The display symbol on the left displays the 4-bit input data in unsigned integer form. The other symbol displays the same input information as a signed integer.



I[3]	I[2]	I[1]	I[0]	Unsigned integer	Signed integer
0	0	0	0	0	0
0	0	0	1	1	1
0	0	1	0	2	2
0	0	1	1	3	3
0	1	0	0	4	4
0	1	0	1	5	5
0	1	1	0	6	6
0	1	1	1	7	7
1	0	0	0	8	-8
1	0	0	1	9	-7
1	0	1	0	10	-6
1	0	1	1	11	-5
1	1	0	0	12	-4
1	1	0	1	13	-3
1	1	1	0	14	-2
1	1	1	1	15	-1

Table 7-6. Correspondence between signed and unsigned 4-bit data

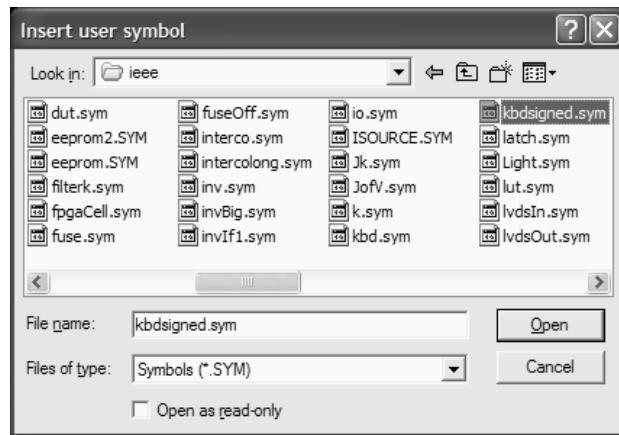


Fig. 7-29: Inserting the specific keyboard symbol which supports signed input (KbdSigned.SYM)

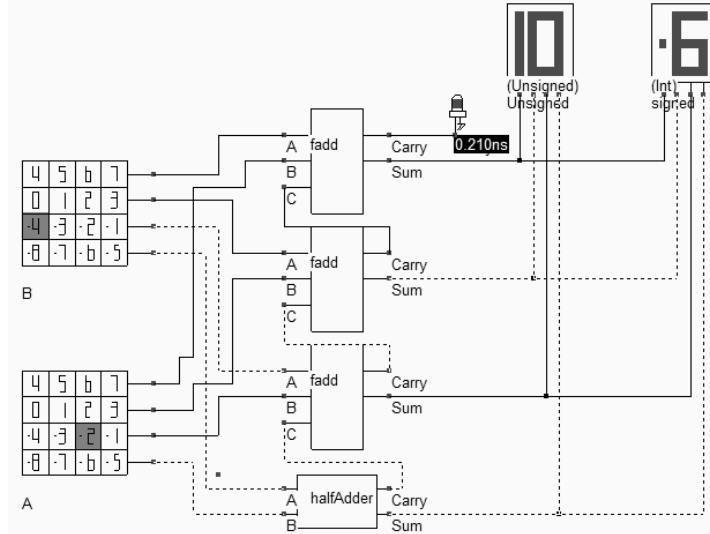


Fig. 7-30. The signed addition uses the same hardware as for the unsigned addition (ADD4Signed.SCH)

It can be seen from figure 7-30 that the unsigned adder circuit can be reused without any hardware modification for the addition of signed integers. The addition of  $a=-2$  with  $b=-4$  gives  $sum=-6$ .

## 6. Fast Adder Circuits

The main drawback of ripple carry adders is the very large computational delay due to the carry chain built in series. Several techniques have been proposed to speed up the addition, at the cost of more complex and power consuming circuits. We construct in this paragraph one type of high-speed adder and compare its performances to the ripple carry adder.

### Carry Look Ahead Adder

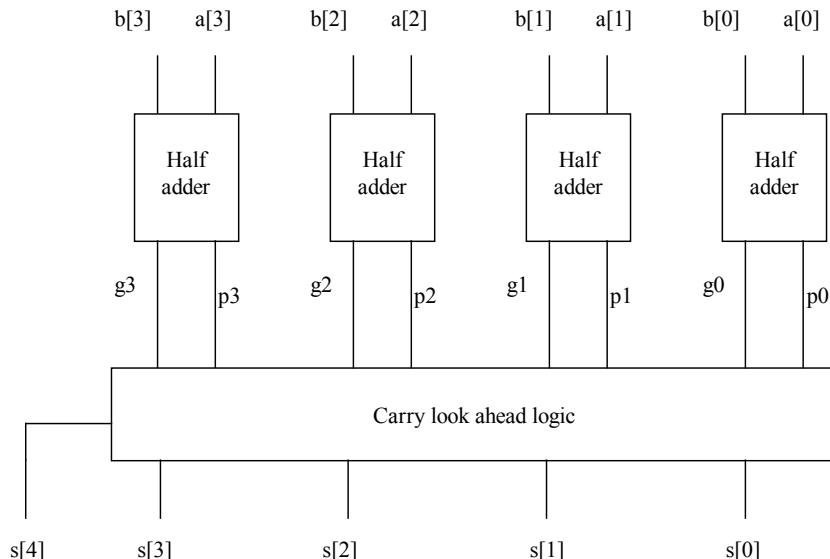


Fig. 7-31. The carry look-ahead logic circuit principles

The carry look-ahead adder computes the  $c_2$  and  $c_3$  carry functions by an optimized hardware, based on a complex gates. The start circuit is the elementary half-adder which produces  $p_i$  and  $g_i$  at each stage (equations 7-10 and 7-11). The sign "&" accounts for logical AND operator, "|" for OR and "^" for XOR. Then, the complex gate is built for each carry function, according to the Boolean expressions reported in equations 7-12.

$$\begin{aligned}
 g0 &= a[0] \& b[0] \\
 g1 &= a[1] \& b[1] \\
 g2 &= a[2] \& b[2] \\
 g3 &= a[3] \& b[3]
 \end{aligned} \tag{Equ 7-10}$$

$$\begin{aligned}
 p0 &= a[0] \wedge b[0] \\
 p1 &= a[1] \wedge b[1] \\
 p2 &= a[2] \wedge b[2] \\
 p3 &= a[3] \wedge b[3]
 \end{aligned} \tag{Equ 7-11}$$

$$s[0] = p0 \tag{Equ 7-12}$$

$$c1 = g0$$

$$s[1] = c1 \wedge p1$$

$$c2 = g1 | (p1 \& g0)$$

$$s[2] = c2 \wedge p2$$

$$c3 = g2 | (p2 \& (g1 | (p1 \& g0)))$$

$$s[3] = c3 \wedge p3$$

$$c4 = g3 | (p3 \& (g2 | p2 \& (g1 | (p1 \& g0))))$$

$$s[4] = c4$$

The equations 7-12 can be translated into layout using XOR gates and complex gates. The schematic diagram of the carry look-ahead adder is given in figure 7-32. Each half-adder circuit produces the  $gi$  and  $pi$  data. Complex gates produce the internal carry  $ci$ , while XOR gates generate the outputs  $s[i]$ .

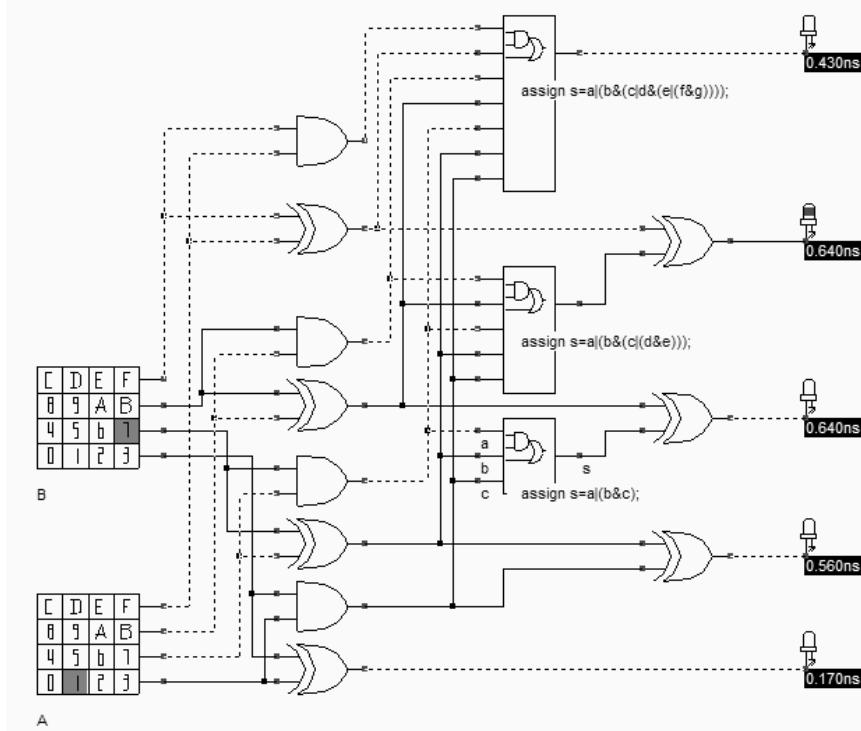


Fig. 7-32. The logic implementation of the carry look-ahead adder (Add4LookAhead.SCH)

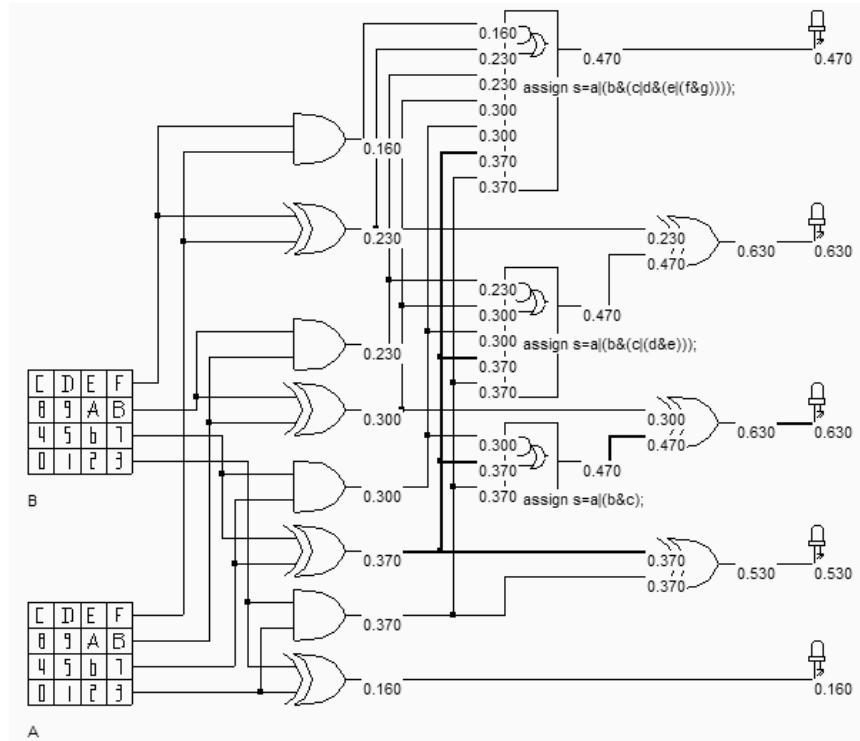


Fig. 7-33 Evaluation of the critical delay in the carry look-ahead adder (Add4LookAhead.SCH)

The key question is to know to which extend the critical delay has been reduced. The answer is given in figure 7-33. We notice a reduction of the critical delay from 0.81ns (Previous ripple carry circuit) down to 0.63ns (This look-ahead circuit). A remarkable point is the homogenous switching delay for most outputs, as compared to the ripple carry result which increases with the stage number.

### Complex Gate Implementation

Most logic cells involved in the construction of the carry look-ahead adder are conventional AND and XOR gates. However, the internal carry signals C<sub>2</sub>, C<sub>3</sub> and C<sub>4</sub> are combinations of AND and OR operators that are perfect candidates for complex gate implementation.

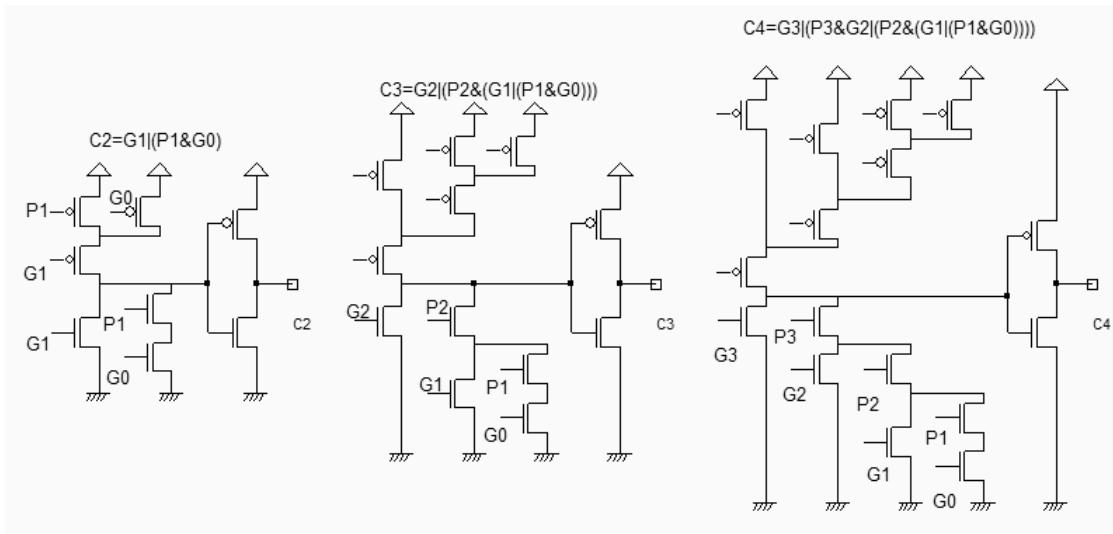


Fig. 7-34. The complex gates used in the carry look-ahead adder (Add4LookAheadCmos.SCH)

The circuit c3 can be generated by the CMOS cell compiler, through the command **Compile → Compile One Line**. The complex gate description is shown in figure 7-35. The layout generated from this description appears in figure 7-36. It is interesting to notice that the cell compiler could not implement the MOS devices using a continuous diffusion. Three separate p-diffusion areas have been used for implementing the pMOS devices.

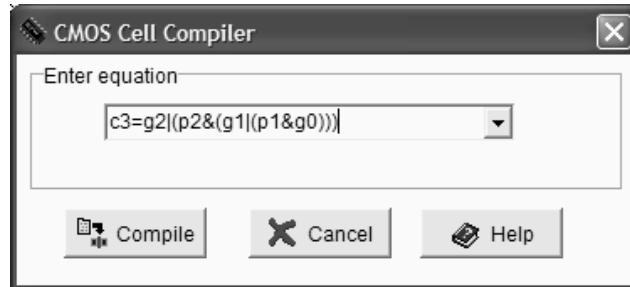


Fig. 7-35. Layout of the complex gate C3 (RippleCarryC3.MSK)

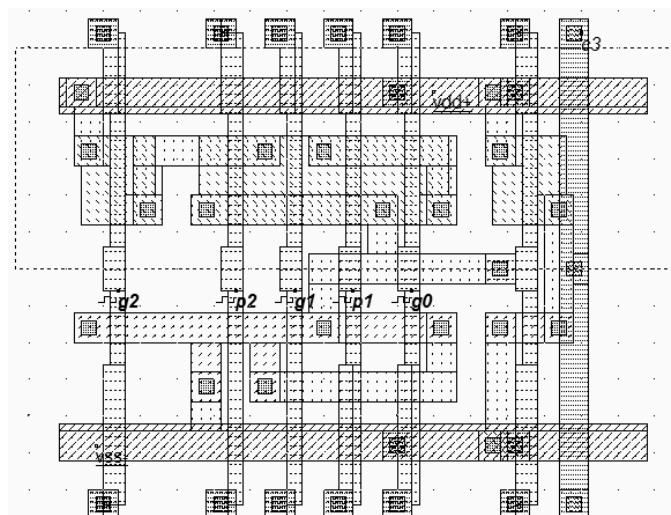


Fig. 7-36. Layout of the complex gate C3 (RippleCarryC3.MSK)

## 7. Subtractor Circuit

The subtractor circuit can be built easily with a full adder structure as for the adder circuit. The main difference is the needs for a 2's complement circuit which inverts the value of  $b$ , and the replacement of the half adder by a full adder, as the initial carry must be 1 (Figure 7-37). The logic circuit corresponding to the 4-bit subtractor is reported in figure 7-38. Some examples of subtractor results are also listed.

$a[0..3]$	$b[0..3]$	$s[0..4] = a - b$ (hexa)	$s[0..4]$ (Decimal)
0	0	00	0
0	1	0F	-1
1	0	01	1
7	9	0E	-2
C	6	06	6
F	F	00	0

Table 7-7. Subtractor result examples

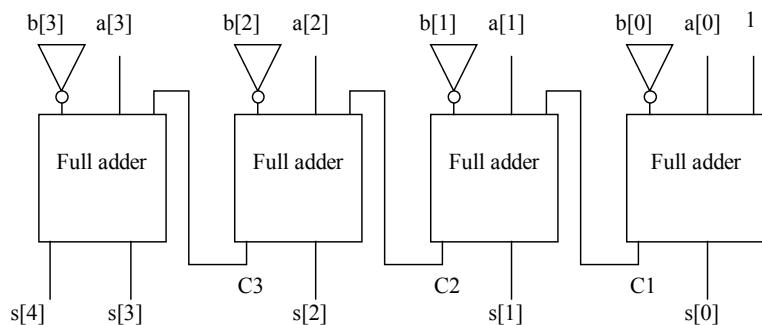
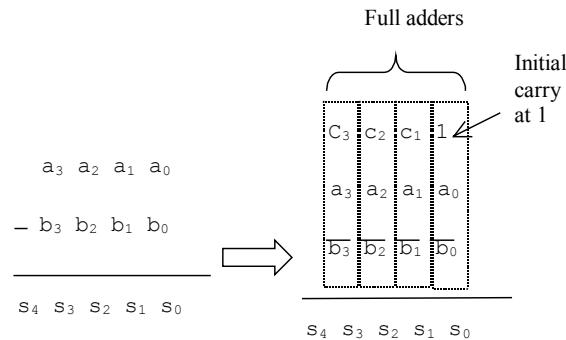


Fig. 7-36. Structure of the 4-bit subtractor

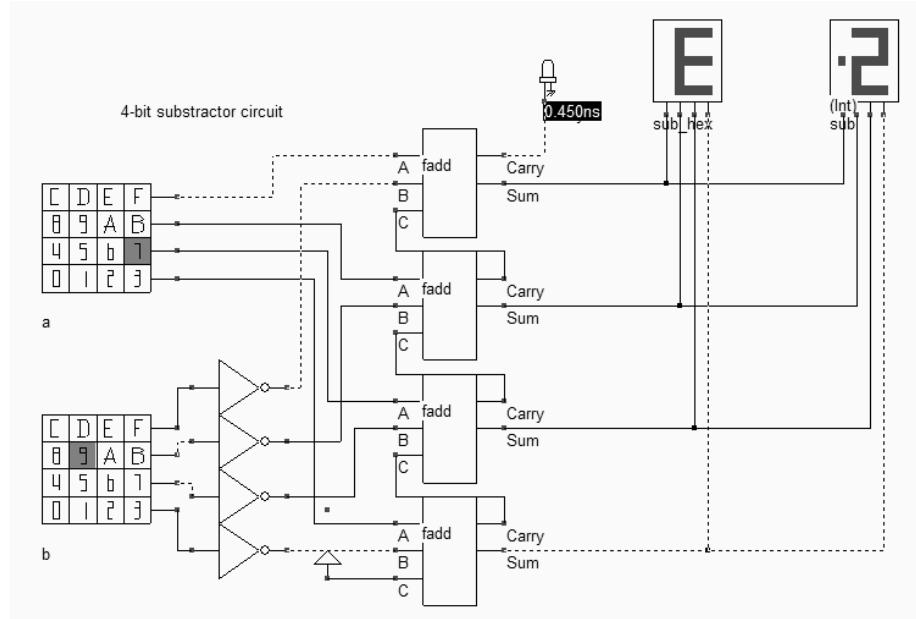


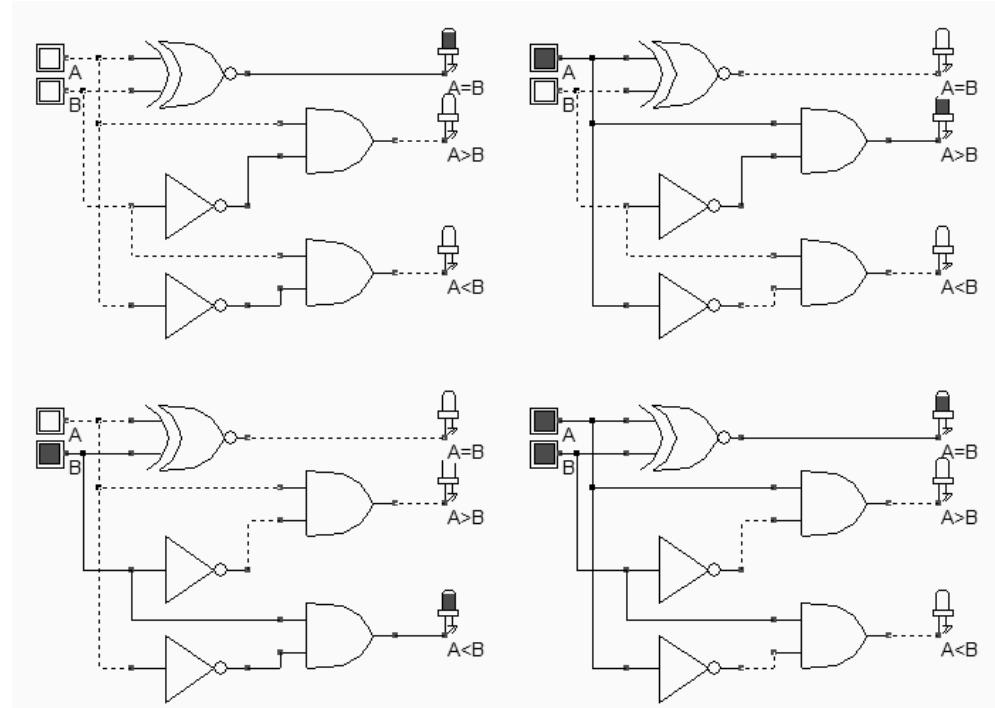
Fig. 7-37. Logic simulation of the 4-bit Subtractor (Sub4.SCH)

## 8. Comparator Circuit

### One-bit Comparator

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

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. 7-38 The truth table and schematic diagram of the comparator (CompTest.SCH).*

Once the logic circuit of the comparator is designed and verified at logic level, the conversion into Verilog is realized by the command **File → Make verilog File**. During the conversion, the node name of some signals has been changed. For example the node  $A < B$  has been modified into  $AiB$ . This is because some characters have another signification in Verilog, and cannot be part of a node name. Then, the Verilog text is converted into layout using Microwind. The layout of the comparator circuit is given in Figure 7-39. The XNOR gate is located at the left side of the design. The inverter and NOR gates are at the right side.

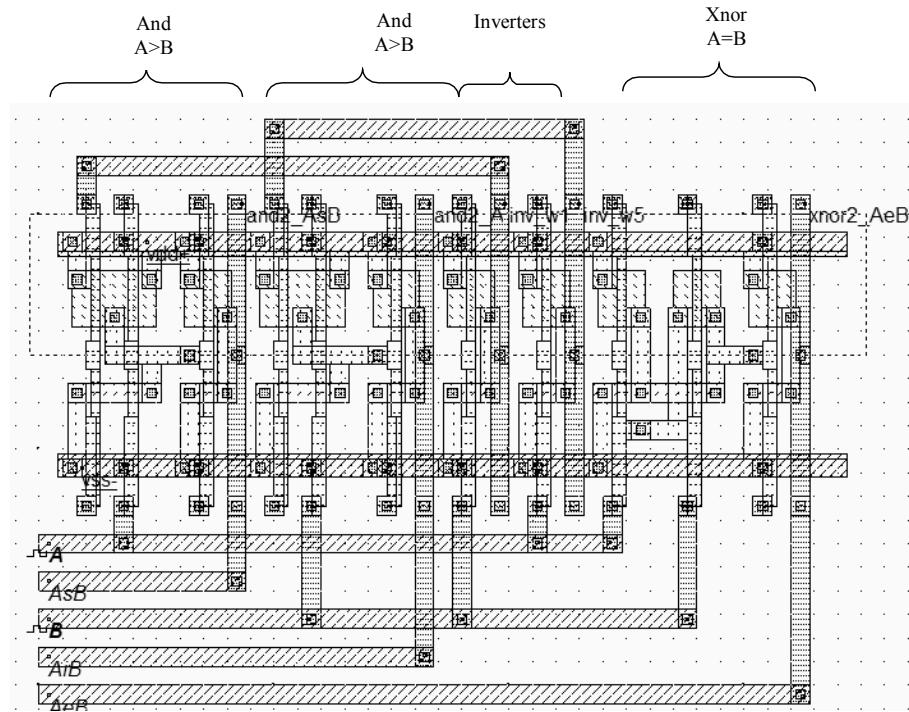


Fig. 7-39 The implementation of the comparator (Comp.MSK).

The simulation of the comparator is given in figure 7-40. After the initialization,  $A=B$  rises to 1. The clocks A and B produce the combinations 00,01,10 and 11.

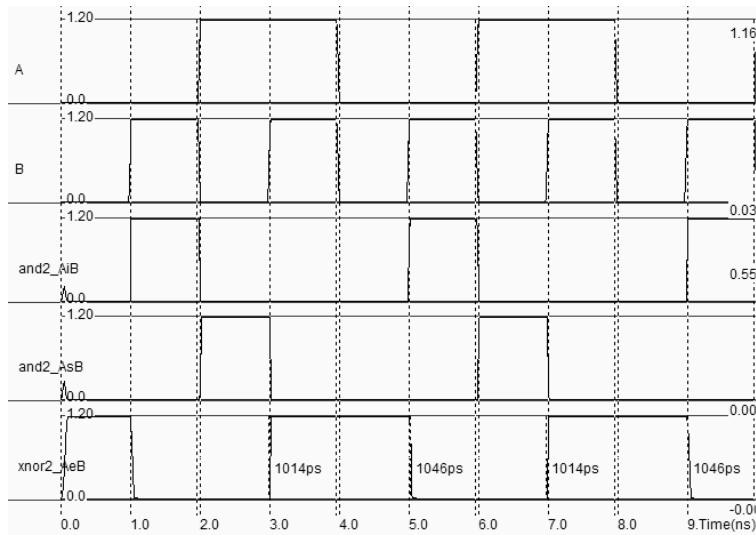


Fig. 7-40. Simulation of the comparator (COMP.MSK file).

### n-bit Comparator

An efficient technique for n-bit comparison is based on the use of adder circuits. In the schematic diagram of figure 7-41, the adder system is modified into a comparison system, by computing the Boolean function  $A-B$ . The ‘equal’

operator is built using simple AND functions. Consequently, each elementary comparator includes the full-adder layout, one inverter and one AND gate.

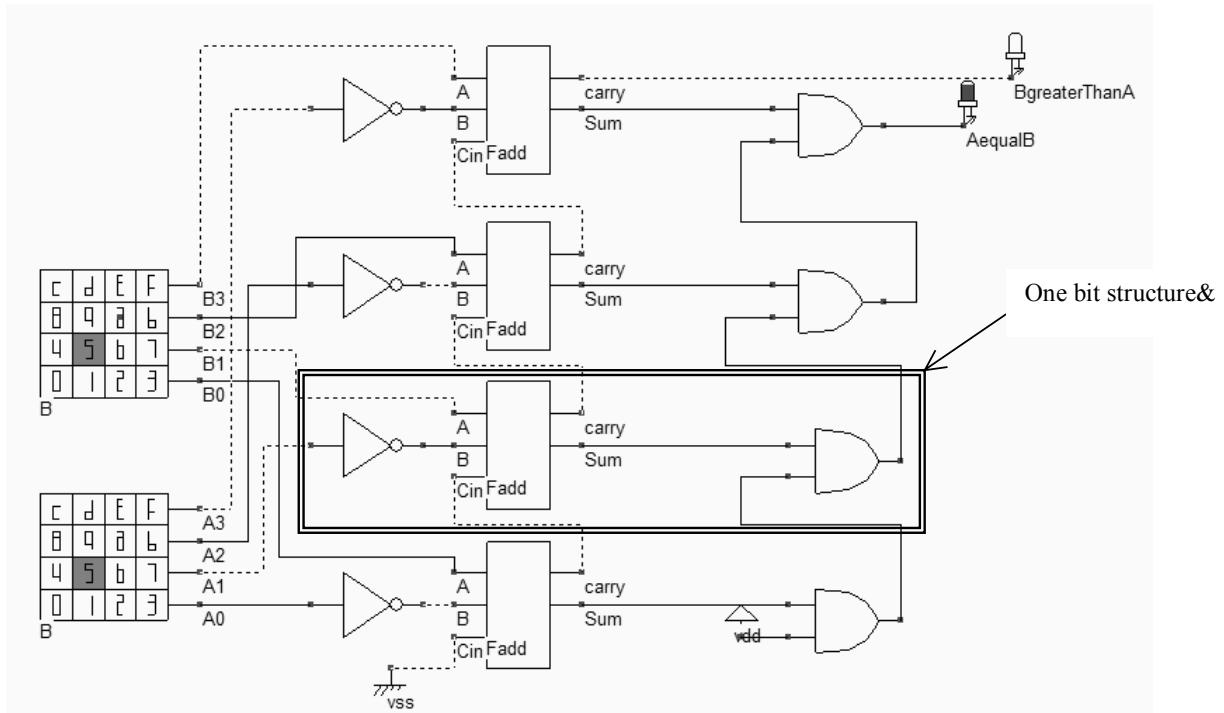


Fig. 7-41. Schematic diagram of a 4-bit comparator (COMP4.SCH).

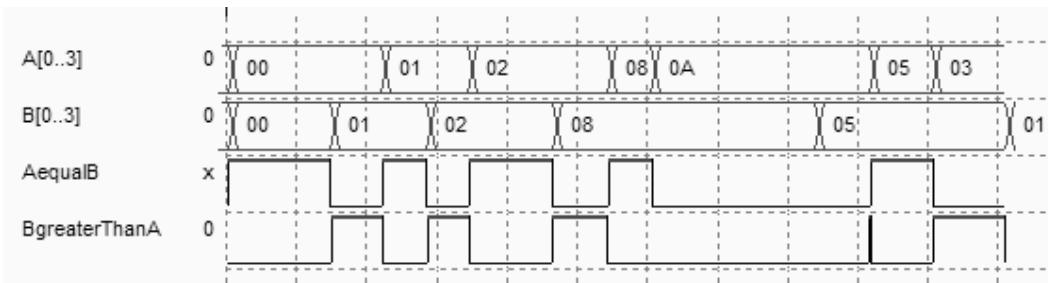


Fig. 7-42. Simulation of the 4-bit comparator (COMP4.SCH).

## 9. Student project: A Decimal Adder

This paragraph details a student project that performs the addition of two Binary Decimal Coded (BCD) numbers X and Y. The numbers range from 0 to 9, and the result (between 0 and 18) is visualized on hexadecimal displays. The specification of this project is described in the schematic diagram of figure 7-43.

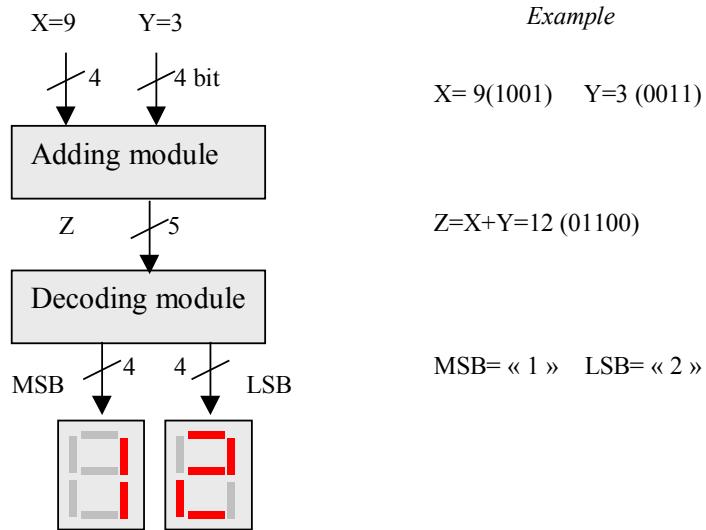


Fig. 7-43: Structure of the decimal adder project

#### 4-BIT ADDER

Four one-bit adders linked in cascade construct the 4-bit adder. You may use the adder symbol created previously (Fadd.SYM). The 4-bit adder from figure 7-20 may be regrouped into a new user symbol Add4.SYM, which will be reused in the project.

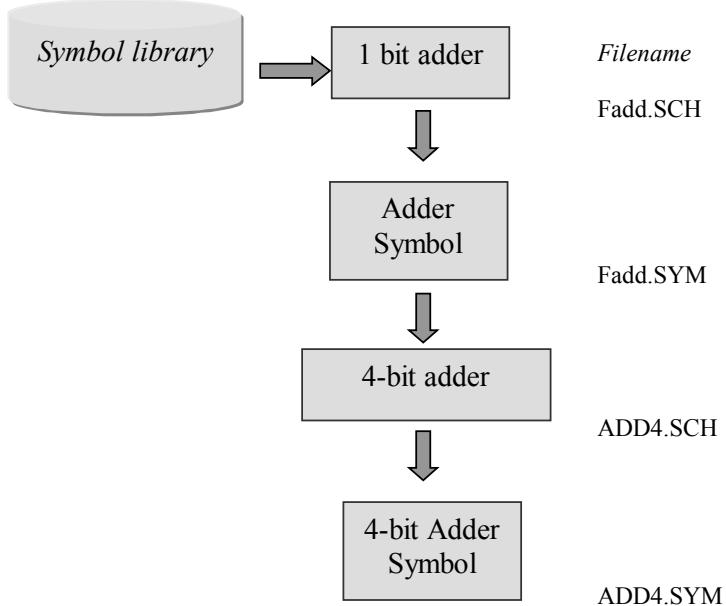


Fig. 7-44: Hierarchical construction of the 4-bit adder symbol

Use the command **File → Schema to New Symbol** to transfer the 4-bit adder schema into a user symbol. The schematic diagram is analyzed and a symbol is proposed regrouping all declared inputs and outputs. The button, clock, pulse and keyboard symbols are considered as inputs. The led and displays are considered as output. The user symbol with a name corresponding to the schematic diagram name, with the '.SYM' appendix, is stored in the current directory. In figure 7-45, the symbol ADD4.SYM is connected to two keyboards and one hexadecimal display to verify the correct behavior of the 4-bit adder.

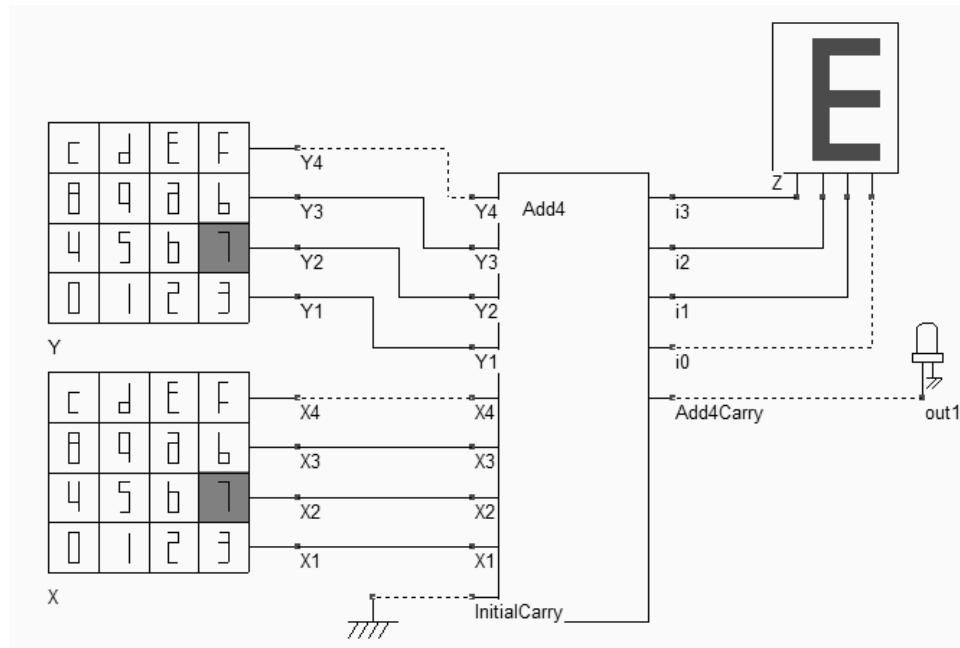
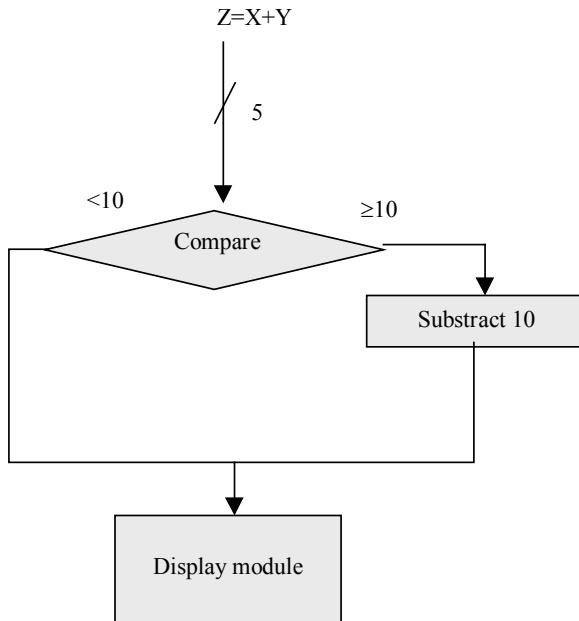


Fig. 7-45: Validation of the 4-bit adder symbol (ADD4Test.SCH)

## DECODER MODULE

The objective of the decoding module is to split the binary result of the addition into two BCD codes, one representing the tenth bit ranging from 0 to 1, the other representing the unit bit ranging from 0 to 9. The principle of the decoding circuit is shown in figure 7-46. Firstly, the result  $Z = X+Y$  is passed through a comparator. If  $Z < 10$ , the result is sent directly to the visualizing module. If not, the result is adjusted by subtracting 10.



Z=X+Y (Hex Binary)	SupOrEqualTo10	LessThan10
0 00000	0	1
1 00001	0	1
2 00010	0	1
3 00011	0	1
4 00100	0	1
5 00101	0	1
6 00110	0	1
7 00111	0	1
8 01000	0	1
9 01001	0	1
A 01010	1	0
B 01011	1	0
...		
1E 11110	1	0

Figure 7-46: Principles and truth table of the decoder module

## COMPARE TO 10

The function *SupOrEqualTo10* may be written using the following Boolean equation. One possible implementation is reported in figure 4-47. The positive logic was used for clarity, although negative logic is a better choice from delay and power consumption points of view.

$$\text{SupOrEqualTo10} = Z4 | (Z3 \& Z2) | (Z3 \& Z1) \quad (\text{Equ 7-13})$$

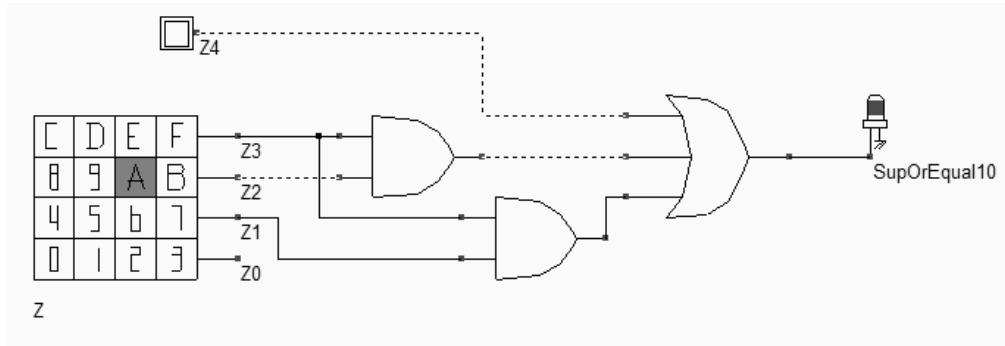


Fig. 7-47. A possible implementation of the function *SupOrEqualto10* (*SupEqu10.SCH*)

### SUBSTRACT 10

The subtractor module is enabled when the result  $Z = X+Y$  is greater or equal to 10. In that case, a subtract-by-10 operation is performed. The subtractor circuit is simply an adder with inverted input B, and an input carry set to 1. Consequently, the subtractor by 10 can be derived from the 4-bit Substractor circuit, as shown below. A **sub10.SYM** symbol is then created, with Z as inputs and LSB (Least significant bit) as outputs.

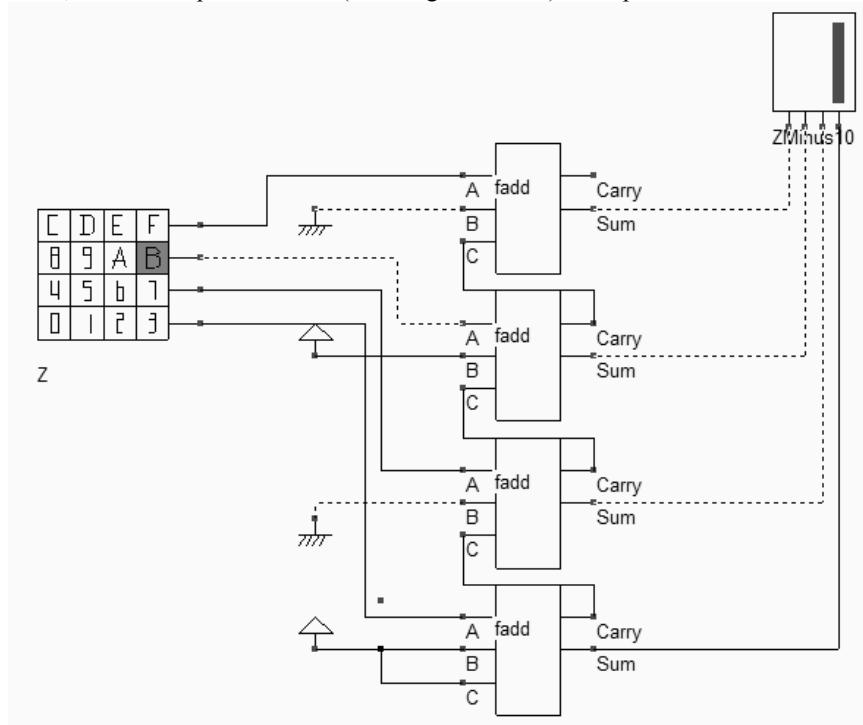


Fig 7-48. Subtraction by 10 (*SUB10.SCH*)

### Final BCD Adder

To complete the circuit, a multiplexor circuit decides whether the  $Z$  result or the  $Z_{\text{Minus}10}$  result is sent to the display. The multiplexor is made from 4 elementary multiplexor devices. When *SupOrEqual10* is asserted, the "1" appears in the

left display, and the  $Z_{Minus10}$  result appears in the right display. When  $SupOrEqu10$  is 0, a "0" appears in the left display and Z appears in the right display. The final circuit is shown in figure 7-49.

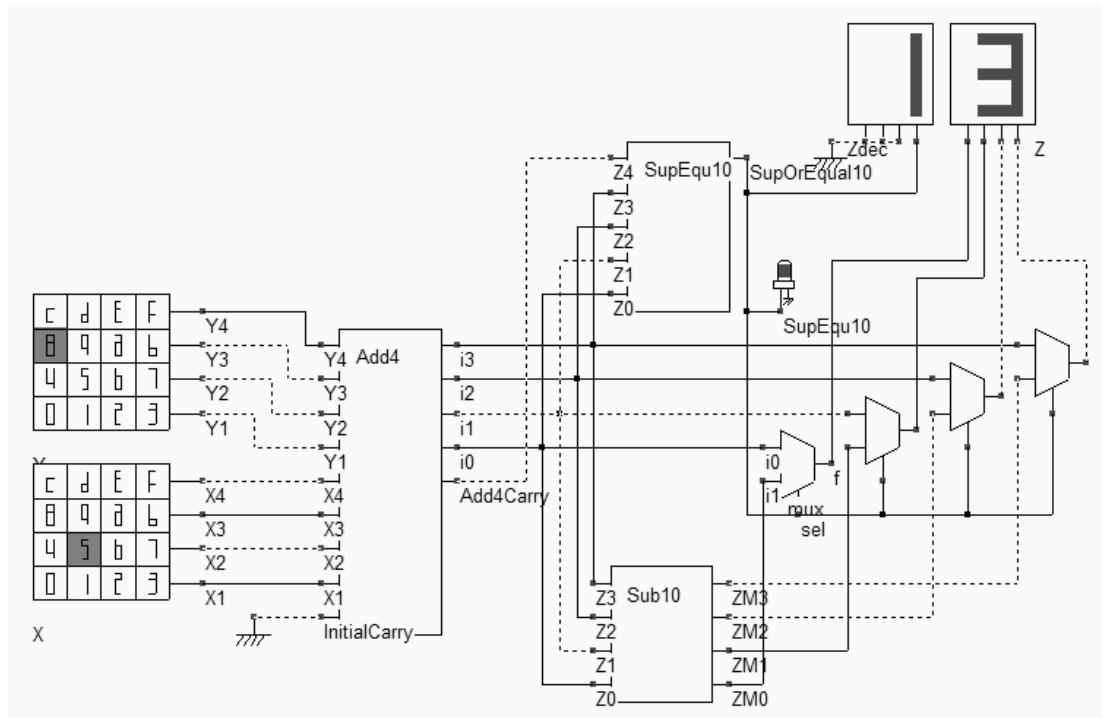


Fig. 7-49. Final circuit of the BCD adder (AdderBcd.SCH)

## 10. Multiplier

Let us illustrate the multiplication process through a small example based on 4-bit unsigned integer. The basic mechanism is a product  $A_i \& B_j$ , combined with the addition which involves a carry. The result propagates down to the result.

$$\begin{array}{r}
 A \quad 0110 \quad (6) \\
 B \quad 0111 \quad (7) \\
 \hline
 & 0110 \\
 & 0110 \\
 & 0000 \\
 \hline
 & 00101010 \quad (42)
 \end{array}$$

The multiplication of integer numbers A and B can be implemented in a parallel way using elementary binary multiplication circuits [Bellaouar]. Within each multiplication cell, the key idea is to compute the product  $P = A_i \& B_j$ , and add the previous sum and previous carry. The next sum is propagated to the bottom, while the next carry is connected to the multiply cell situated at the left side (Figure 7-50).

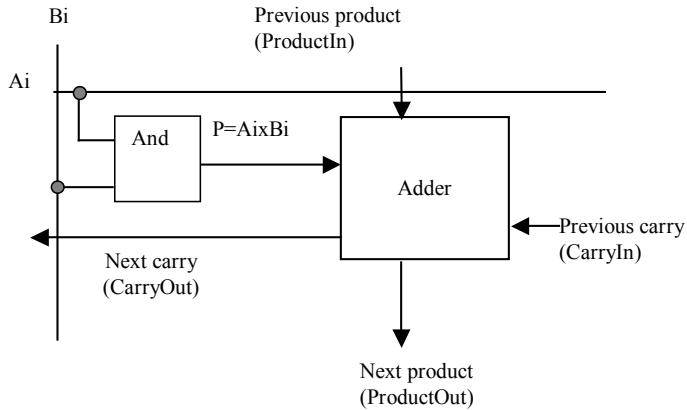


Fig. 7-50. Principles for the elementary multiplication

The elementary multiplication cell should verify the truth table given below. The cell can be made up of a full-adder cell and an AND gate, as shown in the schematic diagram given in figure 7-51.

Multiplier				
AixBi	ProductIn	CarryIn	CarryOut	ProductOut
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

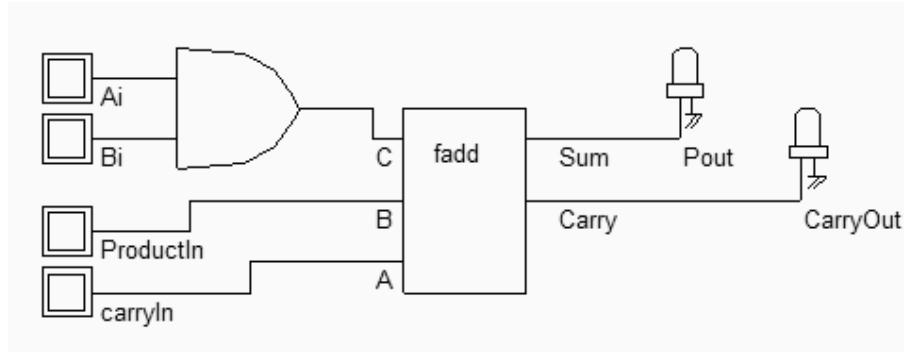


Fig. 7-51. Principles for the elementary multiplication (MUL1.SCH)

A 4x4 bit multiplication is proposed in Figure 7-52. The circuit multiplies input  $A$  (Upper keyboard) with input  $B$  (Lower keyboard) which produces an 8-bit result  $P$ . In the logic simulation, the 8-bit display is configured in decimal mode to ease the interpretation of the result.

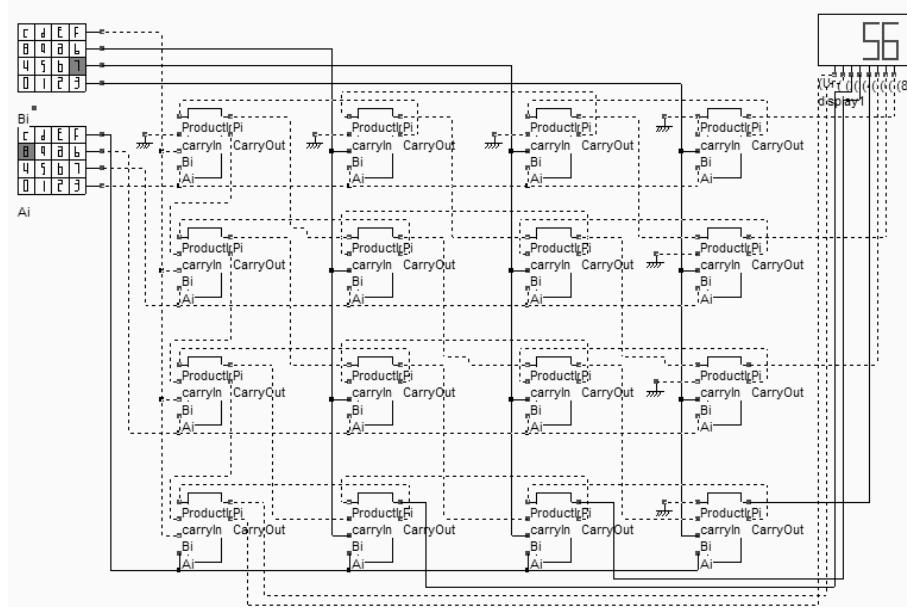


Fig. 7-52. Schematic diagram of the 4x4 bit multiplier (Mul44.SCH).

### Compiled Multiplier

The 4x4 bit multiplier can be translated into layout thanks to the Verilog compiler. By default, the circuit is constructed by placing all basic gates on a single row and performing the routing. The input and output signals are routed below the active area, and the internal wire routing is performed on the upper part of the active area. Notice that the routing is performed using metal3 for horizontal connections and metal2 for vertical connections. Industrial routing tools take advantage of the other metal layers (metal4,5,6,etc..) to create a more layout-effective implementation.

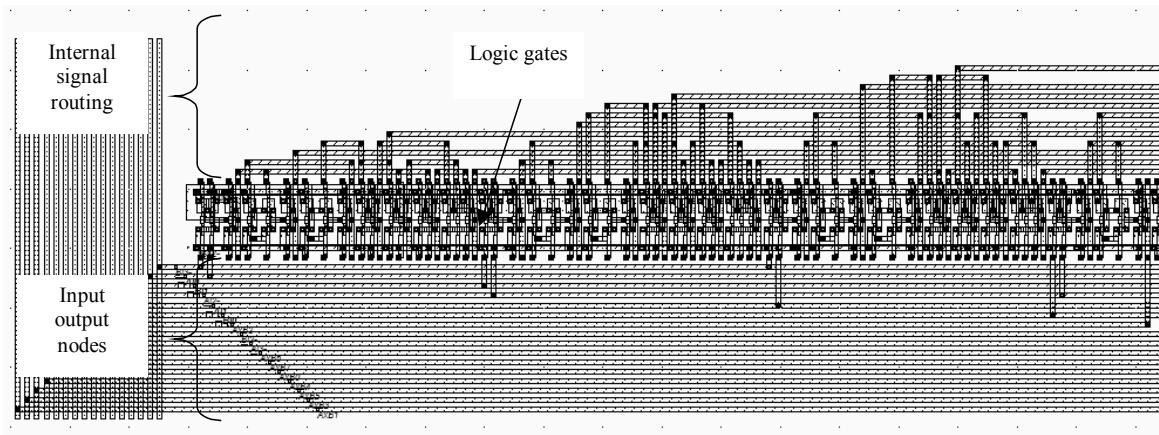


Fig. 7-53 Linear implementation of the 4x4 bit multiplier (Mul44.MSK).

### An compact Multiplier layout

A more efficient approach consists in using an AND gate and a full adder, and placing the interconnects in order to facilitate an iterative implementation. The inputs *CarryIn*, *Ai*, *Bi* and *ProductIn* and the outputs *ProductOut* and

*CarryOut* are organized in such a way that the array can be extended to a larger format. The general floorplan of the multiplier is shown in figure 7-54. An  $n \times n$  multiplier would require  $n^2$  elementary multiplier cells, each based on one AND gate and one full adder.

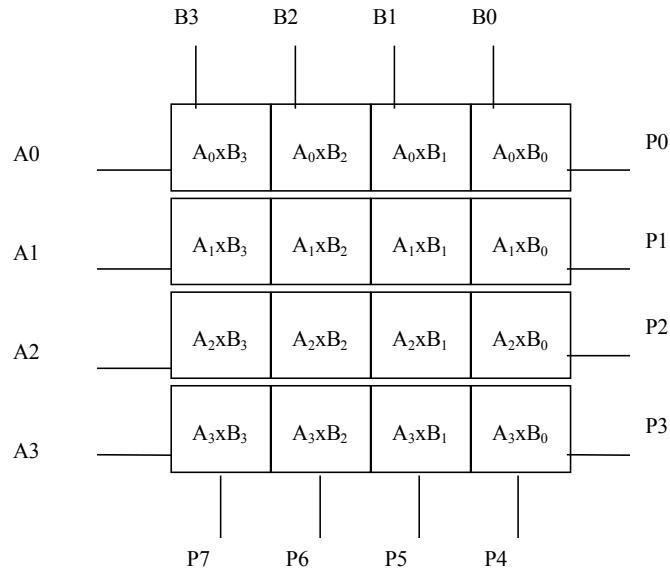
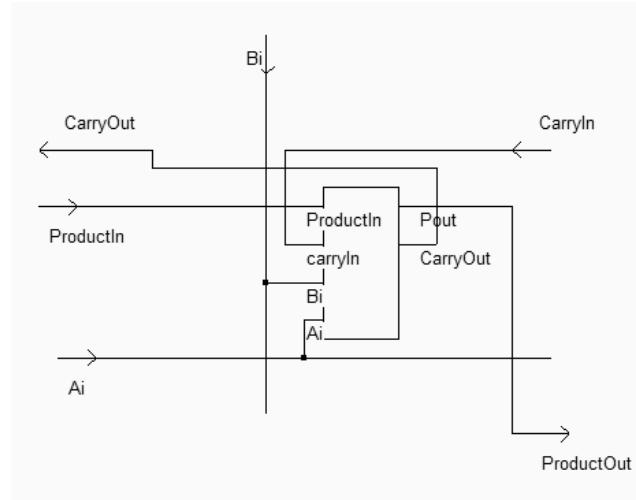


Fig. 7-54. Floor planning of the 4x4 bit multiplier

The main drawback of this architecture is the very important critical delay, which consists of the carry propagation through ten multiplier cells (Figure 7-55). Several algorithms exist for accelerating the multiplication (See reference [xxx]), more or less by a factor of two. Another problem is the design time needed to implement this array structure, and the severe risk of error due to manual arrangement of the cells and interconnects. In ultra-deep submicron technologies, compact 8x8 bit or 16x16 bit multipliers are proposed as intellectual property blocks (IP <Gloss>).

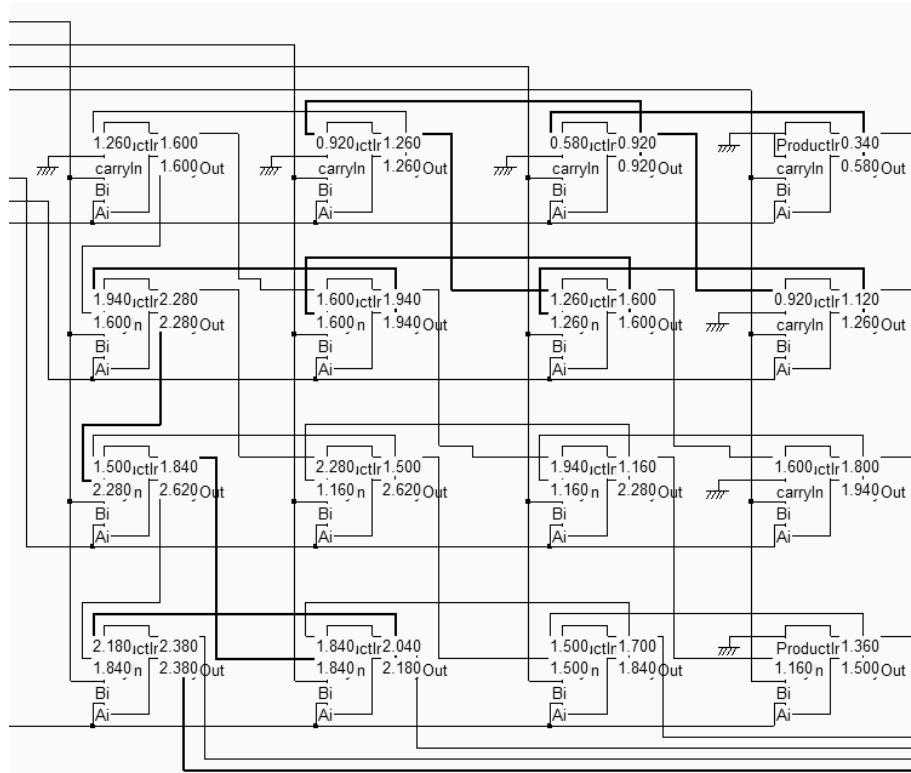


Fig. 7-55. Estimation of the critical delay path (Mul44.SCH).

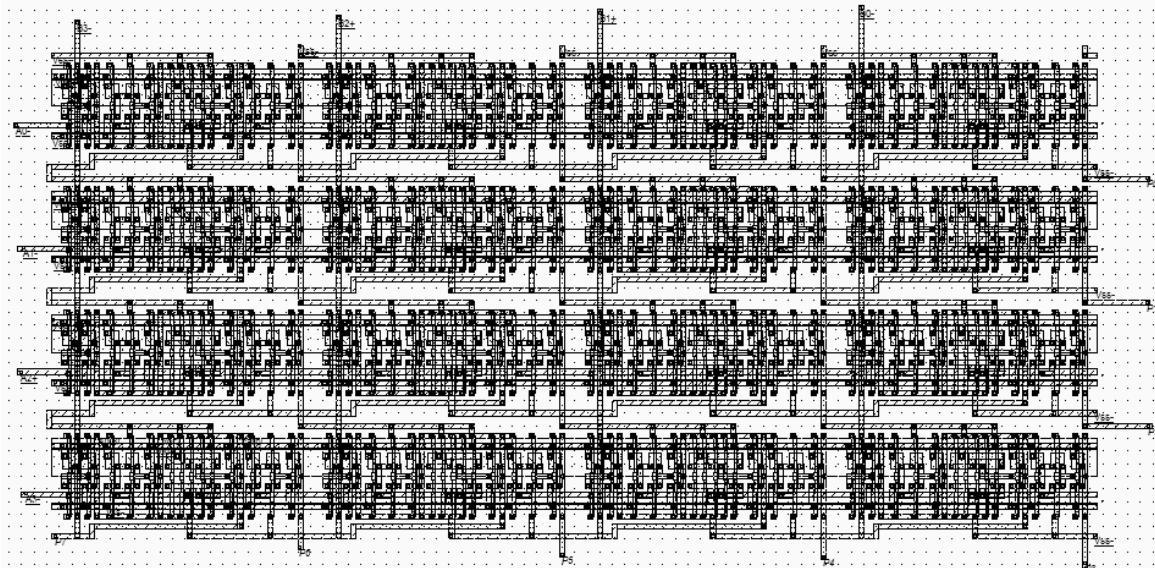


Fig. 7-56. Design of the 4x4 bit multiplier (Mul44Compact.MSK).

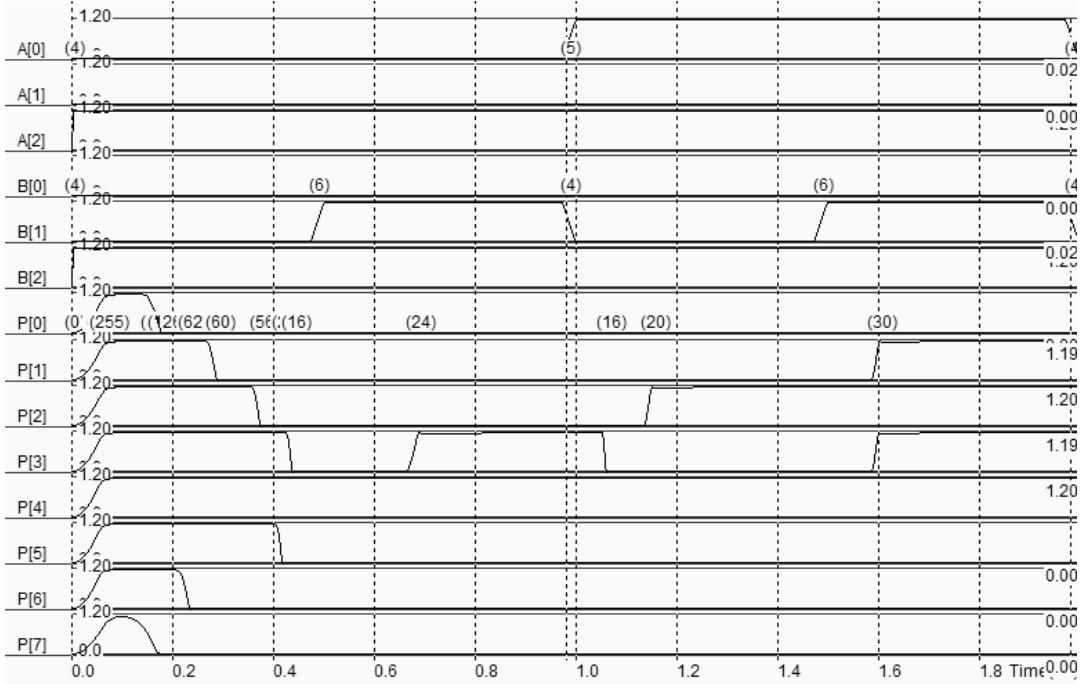


Fig. 7-57. Simulation of the 4x4 bit multiplier (Mul44Compact.MSK).

The simulation of the 4x4 parallel multiplier is given in figure 5-57. As for the adder, we use signal names with brackets, starting with index 0, to enable Microwind to compute the equivalent logic value of the A,B and P signals. Notice that the initialization phase is almost equal to 0.5nS. At time t=1.0ns, A=5, B=4, and P=20 after 150ps delay.

## 11. Arithmetic and logic Units (ALU)

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 7-58, 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*.

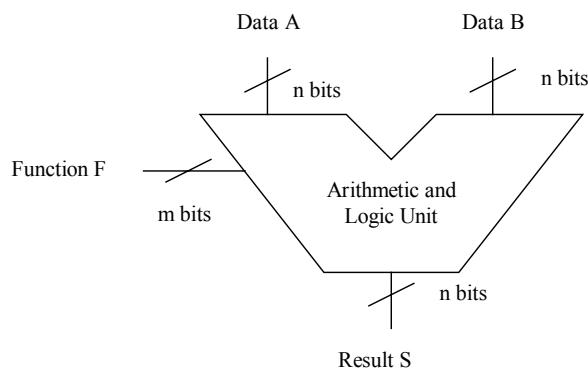


Fig. 7-58. Principles for the Arithmetic and Logic Unit

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 7-8. 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	Substractor	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 7-8. Some important instructions implemented in the ALU of the 8051 micro-controller

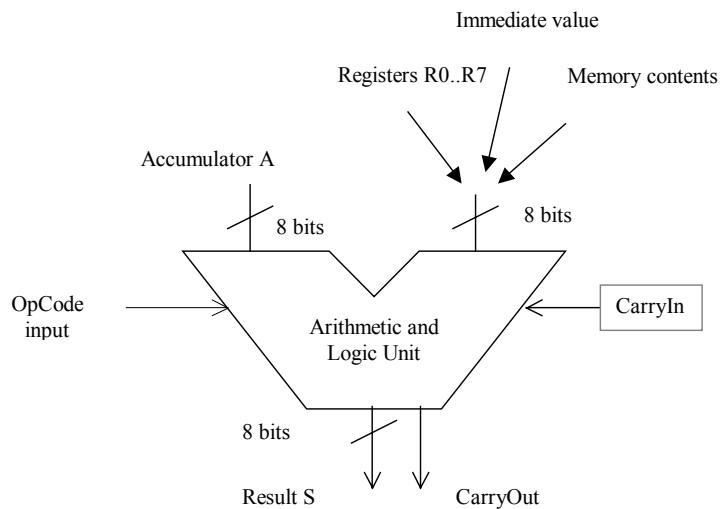


Figure 7-59. The arithmetic and logic unit of the 8051

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.

The description of the 8051 arithmetic and logic unit in DSCH2 is outlined below. The description language, originally in DELPHI [Borland], has been translated for clarity into VHDL [John]. The declaration of the ALU always starts by a description of the variables (figure 7-60) in the *Entity*, where we recognize the operation code *op\_Code*, with a 4 bit format fitted to the number of cases (The statement *When* appears 12 times in figure 7-60), the accumulator input *source\_a*, the other input *source\_b*, the previous *carry\_in*. The inputs *source\_a* and *source\_b* are in an 8-bit format. The input *carry\_in* is declared in the most common logic format (STD\_LOGIC). The standard logic values are mainly the two logic values '0' and '1'. We also use 'X' for unknown, 'Z' for high-impedance and '-' for "don't care". The result *result\_s* is in an 8-bit format, and *carry\_out* is a standard logic output.

```
entity ALU_8051 is
port(op_code : in UNSIGNED (3 downto 0);
      source_a : in UNSIGNED (7 downto 0);
      source_b : in UNSIGNED (7 downto 0);
      carry_in : in STD_LOGIC;
      result_s : out UNSIGNED (7 downto 0);
      carry_out : in STD_LOGIC;
end ALU_8051;
```

Figure 7-60. The ALU interface described in VHDL

```
architecture of ALU_8051 is
begin
  case op_code is
    when CLR => result_s <= "00000000";
    when CPL  => result_s(7 downto 0) <= not source_a(7 downto 0);

    when ADD => DO_ADD(source_a,source_b, carry_in, result_s,carry_out);

    when SUBB => DO_SUBB(source_a,source_b, carry_in, result_s,carry_out);

    when INC  => DO_ADD(source_a, "00000001",'0' , result_s,carry_out);

    when DEC  => DO_SUBB(source_a, "00000001" , '0' , result_s,carry_out);

    when XRL  => result_s(7 downto 0) <=
                  source_a(7 downto 0) xor source_b(7 downto 0);

    when ANL  => result_s(7 downto 0) <=
                  source_a(7 downto 0) and source_b(7 downto 0);
```

```

when ORL => result_s(7 downto 0) <=
            source_a(7 downto 0) or source_b(7 downto 0);

when RL => result_s(0) <= source_a(7);
            result_s(7 downto 1) <= source_a(6 downto 0);

when RR => result_s(7) <= source_a(0);
            result_s(6 downto 0) <= source_a(7 downto 1);

when others => carry_out <= '-';
end case;
end process;

```

Figure 7-61. The ALU circuit described in VHDL

The ALU circuit code is given in figure 7-61. For each value of the operation code, a specific circuit is constructed which links the inputs *source\_a* and *source\_b* to the output *result\_s*. The physical connection is created by the assignment "<=". The logical operators XOR, AND, OR and NOT are basic keywords of the VHDL language. The addition is described through a procedure called DO\_ADD, the subtraction through DO\_SUB. The complete code used to describe the ADD and SUBB operation is quite complex, due to the treatment of the carry. Part of that code is given in figure 7-62. We notice the use of arithmetic operators such as "+" for addition (And "-" for subtraction) and the formatting of the data to remove the most significant bit and to take into account the input *carry\_in*.

```

Result_1 := "0" & source_a (6 downto 0);
Result_2 := "0" & source_b (6 downto 0);
Result_3 := "0000000" & carry_in;
Result_s := Result_1+ Result_2+ Result_3;

```

Figure 7-62. The basic instruction included into the DO\_ADD procedure

Several synthesis tools are able to convert the VHDL architectural description into gate-level design, which can then be translated easily into layout. Neither DSCH nor Microwind can support such a description. The ALU may be designed using basic cells such as AND, XOR, Adder, Multiplexor using DSCH, and then be converted into layout.

### The ALU in DSCH2

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 7-63).

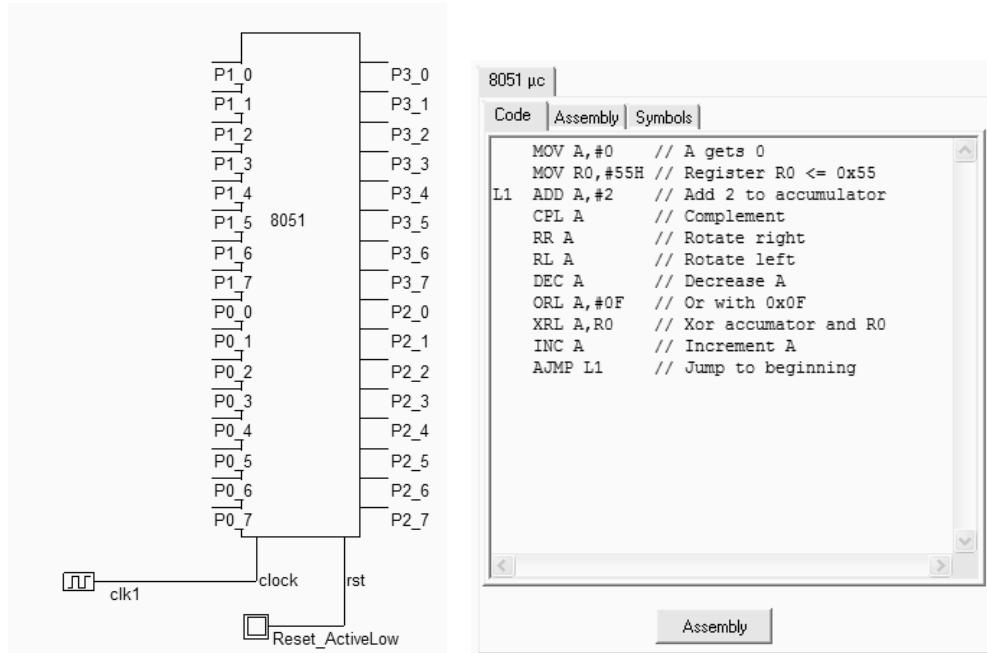


Figure 7-63. The 8051 symbol and its embedded software (8051.SCH)

After a double-click in the symbol, the embedded code appears. That code may be edited and modified (Figure 7-63). 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).

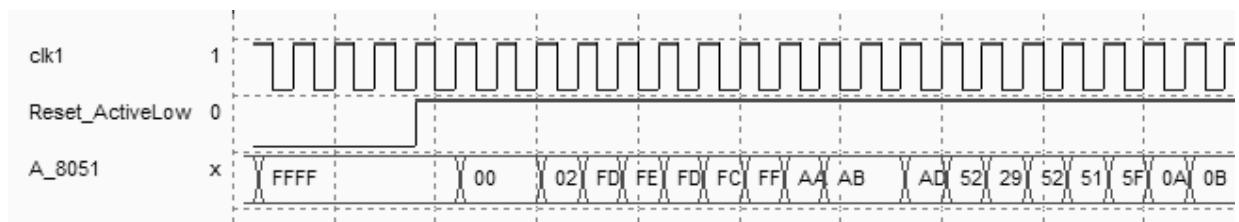


Figure 7-64. The simulation of the arithmetic and logic operation using the 8051 micro-controller (8051.SCH)

## 12. Conclusion

In this chapter, the design of basic arithmetic gates has been presented. The half adder and full adder have been presented. The full adder design has been conducted at layout level, with emphasis on advantages of manual design against automatic design regarding the silicon area efficiency. The comparator and multiplier circuits have also been described. A student project concerning the addition in binary-to-decimal format has been described in details, as well as an illustration of an arithmetic and logic unit used in the 8051 micro-controller.

## Exercises

- 7.1 What is the 16 bit data equivalent to -3? What is the 32 bit data equivalent to  $10^6$ ?
- 7.2 <Etienne: alternative circuit for full-adder>
- 7.3 Design a Half Adder using the AOI technique described in chapter 6.
- 7.4 Implement the carry look ahead adder of figure 7-32, using the CMOS cell compiler of Microwind. Do you confirm the switching speed gains forecast by logic simulation?
- 7.5 Compile into layout the binary-decimal-coded adder described in section 10. What is the average power consumption per MHz? From the analysis of the logic circuitry, which part could be redesigned in order to reduce the dynamic power consumption?
- 7.6 Are there alternative circuits to reduce the standby current of the binary-decimal-coded adder described in section 10?
- 7.7 Dsch2 also includes the model of the PIC16f84 micro-controller [Pic]. 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 `iowrf`), XOR (Instruction `xorwf`) and SUB (Instruction `subwf`) operations.

## References

- [Uyemura] John P. Uyemura "Introduction to VLSI Circuits & Systems", Wiley, 2002.
- [Belaouar] <tbd>
- [Weste] Chapter 8
- [Borland] [www.borland.com](http://www.borland.com)
- [John] Michael John, Sebastien Smith, "Application Specific Integrate Circuits", Addison-Wesley, 1997, ISBN 0-201-50022-1
- [Pic] <Add ref>

[8051] <Add ref>

# 8 Sequential Cell Design

This chapter details the structure and behavior of latches. The RS Latch, the D Latch and the edge-sensitive register are presented. The counters are also introduced, with an application concerning a 24-hour clock.

## 1. The Elementary Latch

Combinational circuits are able to add, multiply, shift, etc.. However, combinational circuits cannot store and retain values. The basis for storing an elementary binary value is called a latch. The simplest CMOS circuit is made from 2 inverters.

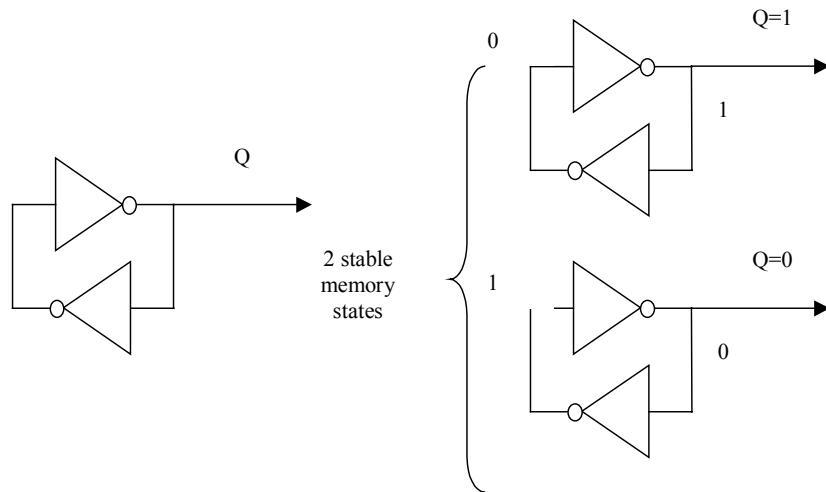


Figure 8-1: Elementary memory cell based on an inverter loop

Several techniques exist to modify the data inside the loop. One solution consists in adding a pass transistor, and make sure that the information coming from D overcomes the information retained by the feedback inverter. An other solution is based on NAND gate, which is extensively used in RS and D-latch structures. The *Set* and *Reset* signals are active low. A similar design, based on NOR gates is also shown in figure 8-2. The *Set* and *Reset* signals are active high in that circuit.

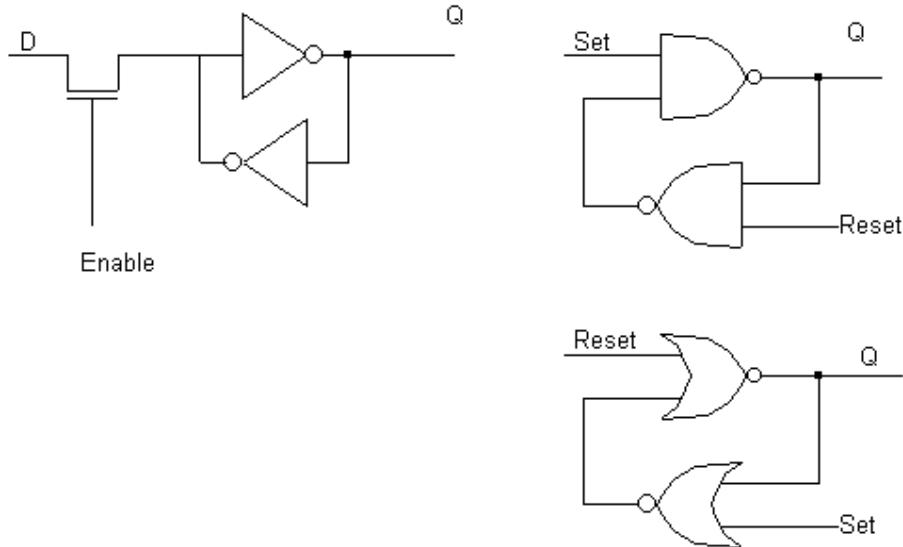


Figure 8-2: Methods to change the contents of the loop (latches.SCH)

## 2. RS Latch

The RS Latch, also called Reset-Set Flip Flop (RS FF), transforms a pulse into a continuous state. The RS latch can be made up of two interconnected NAND gates, as shown in figure 8-3. In the truth table, we see that 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. In this case,  $Q=nQ=1$ .

RS Latch (NAND)			
R	S	Q	nQ
0	0	1	1
0	1	0	1
1	0	1	0
1	1	Q	nQ



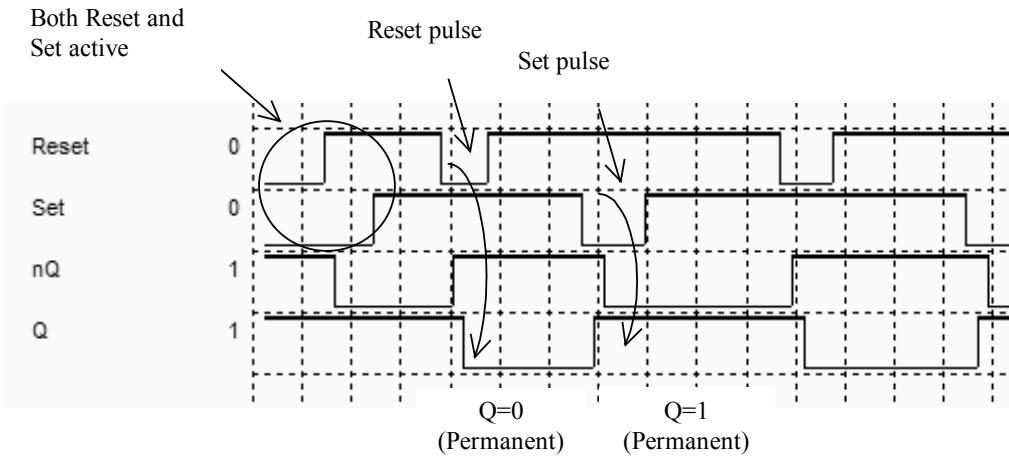


Figure 8-3: The RS-NAND latch and its typical simulation waveform (RSNand.SCH)

The simultaneous change from  $Reset=Set=0$  to  $Reset=Set=1$  (Figure 8.4) provokes what is called the meta-stable state, that corresponds to a high frequency oscillation. The meta-stability appears in the chronograms of figure 8-4 at the simultaneous rise of *Set* and *Reset*. At layout level, the parasitic oscillation does not exist, but a parasitic delay effect may be observed, up to 2 or 3 times the nominal gate delay.

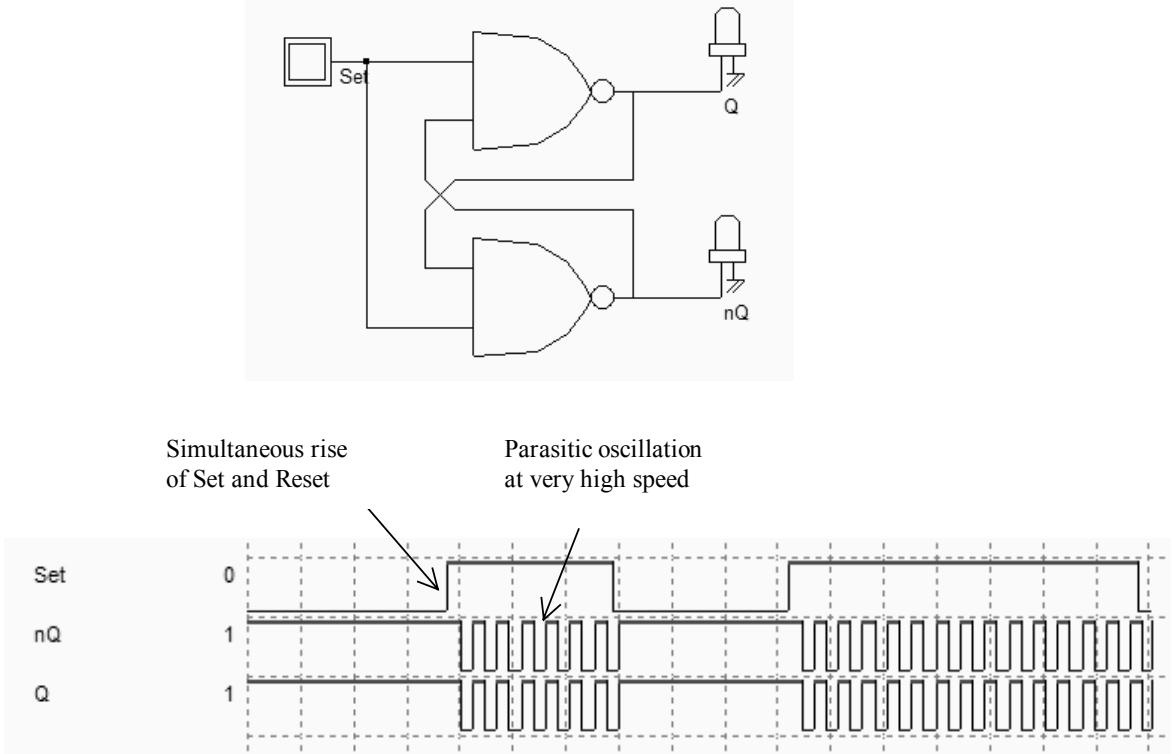


Figure 8-4: Illustration of the meta-stability at logical level (LatchMetaStable.SCH)

RS Latch (NOR)			
R	S	Q	nQ
0	0	Q	nQ
0	1	1	0
1	0	0	1
1	1	0	0

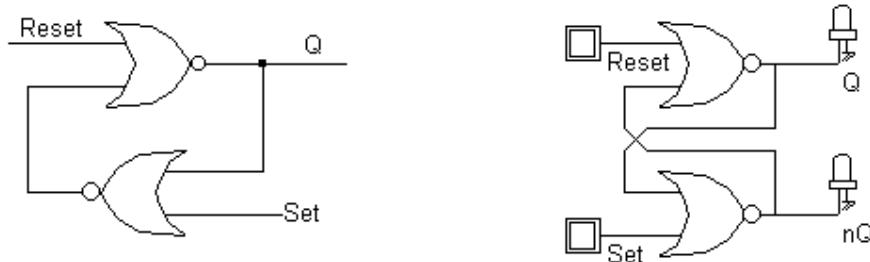


Figure 8-5. The truth table and schematic diagram of a RS-NOR latch (RsNor.SCH)

An alternative implementation of the RS latch is made from NOR gates (Figure 8-5). In that case, the *Reset* and *Set* inputs are active high. The cell transforms positive pulses into continuous states, as seen in the chronograms of figure 8-6.

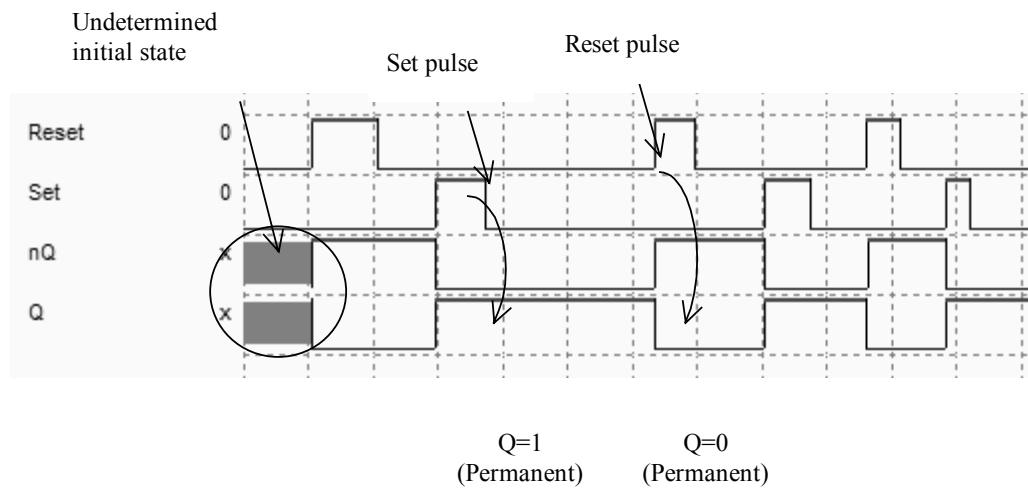
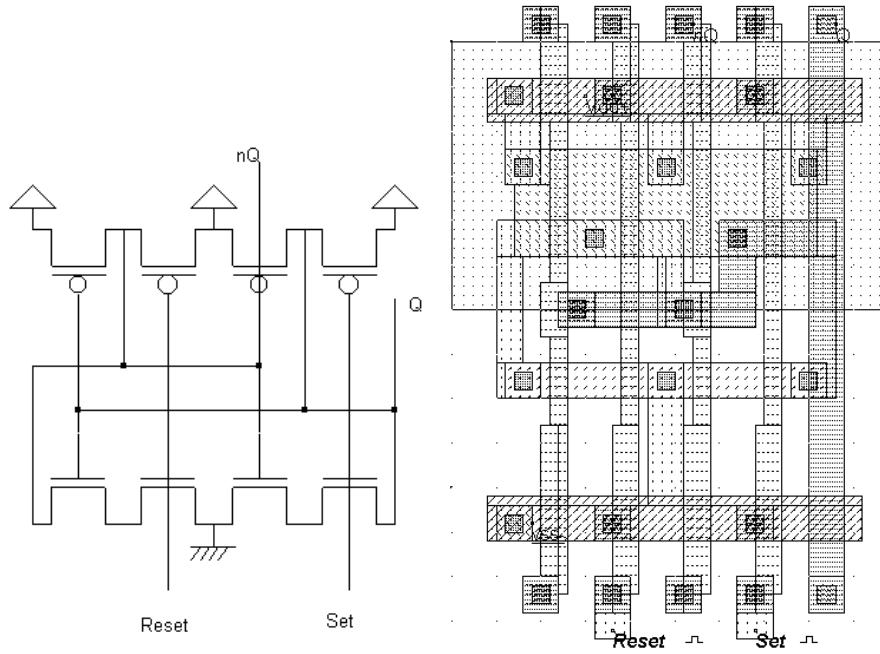


Figure 8-6. The typical simulation of a RS-NOR latch (RsNor.SCH)

## RS Latch Layout

You may create the layout of RS latch manually. The two NAND gates may share the VDD and VSS supply achieving continuous diffusions. The internal routing may also save routing area, leading to the layout shown in Figure 8-7 (RSNandManual.MSK).

Fig. 8-7. Manual design of the RS-NAND (*RsNandManual.MSK*)

An alternative approach is to compile the layout directly from the schematic diagram. Using DSCH, create the RS Nand logic circuit, and generate the Verilog text by using the command **File → Make Verilog File**. In Microwind2, click on the command **Compile → Compile Verilog File**. Select the Verilog text file corresponding the RS-Nand gate.

```
module RSNand( Reset,Set,Q,nQ);
  input Reset,Set;
  output Q,nQ;
  nand #(23) nand2(Q,Set,nQ);
  nand #(23) nand2(nQ,Q,Reset);
endmodule

// Simulation parameters
always
#100 Set=~Set;
#200 Reset=~Set;
```

Fig. 8-8 The Verilog description of the RS Nand gate

Click on **Compile**. When the compiling is complete, the resulting layout appears as shown in figure 8-8. The NAND implementation of the RS gate is completed. Compared to the layout shown in figure 8-7, the compiled version is a little larger, due to the fact that the layout conversion tool did not merge the supply connections.

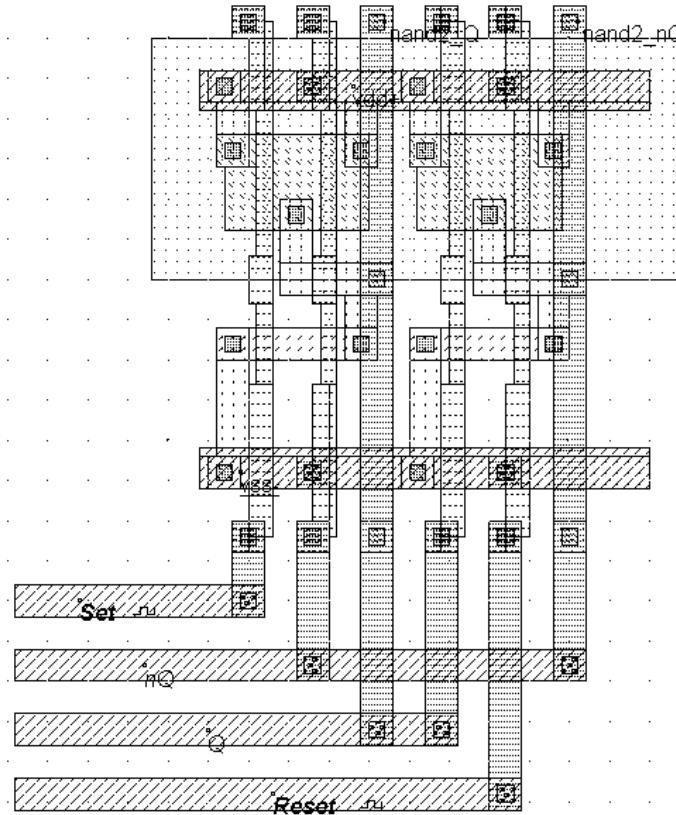


Figure 8-9. Automatic design of the RS-NAND (RsNand.MSK)

### Control of the RS-Latch

If we keep the clock properties assigned by default to *Set* and *Reset*, we get chronograms which are inappropriate to validate the RS latch behavior, particularly the memory effect. A better approach consists in declaring pulse properties. For example, an active pulse is created on *Reset* followed by an active pulse on *Set*. The pulse parameters are shown in figure 8-10. To obtain a negative pulse, click the small icon placed in the middle of the window ( ).

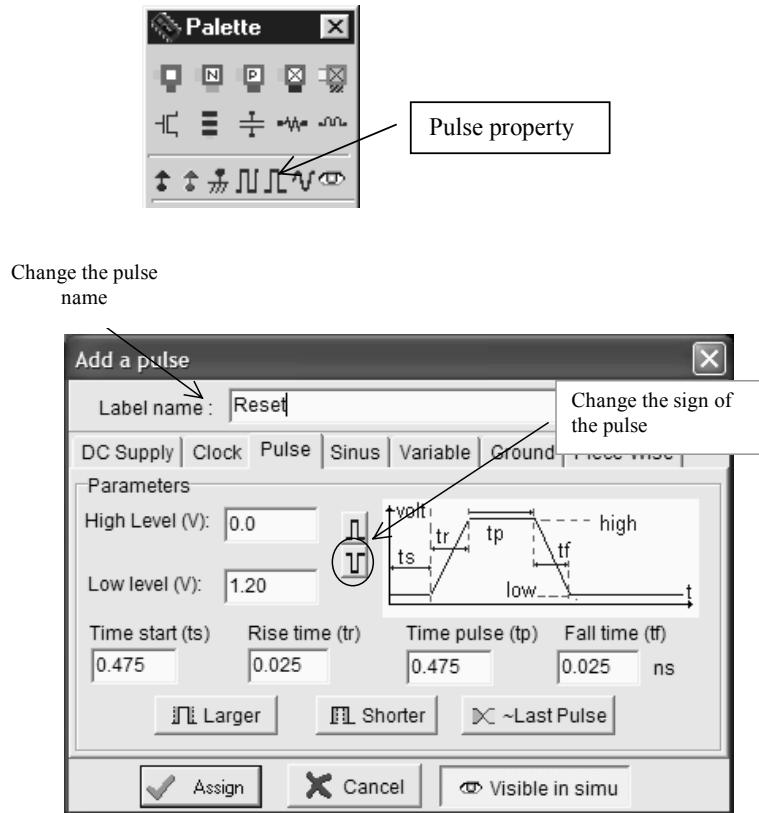


Figure 8-10. The pulse property in Microwind (RSNand.MSK)

The steps to add a pulse property are as follows. Select the Pulse icon in the palette. Click on the layout of the desired node. The screen shown in figure 8-10 appears. Change the label name, fix the sign of the pulse (positive by default), and click **Assign**.

Repeat the same procedure to assign a pulse property to node *Set*. The active level of the pulse is automatically delayed. Finally, click on **Simulate → Start Simulation**. The timing diagrams of figure 8-11 appear. A negative pulse on *Reset* turns *Q* to a stable low state. Notice that when *Reset* goes back to 1, *Q* remains at 0, which demonstrates the ability of the latch to transform a transient pulse into a permanent state. When a negative pulse occurs on *Set*, *Q* goes high, *nQ* goes low. The combination *Reset=Set=1* corresponds to the memory state.

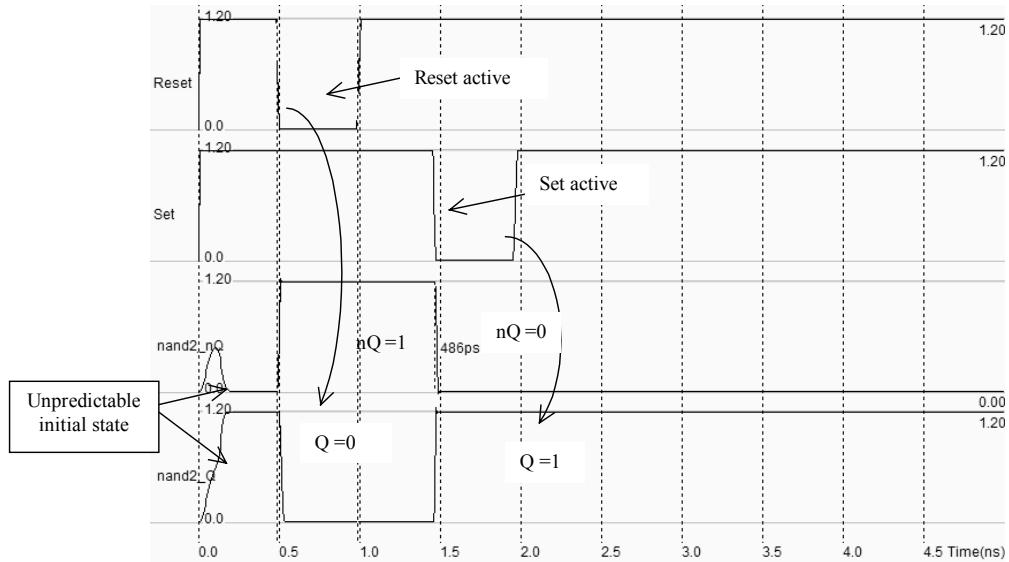
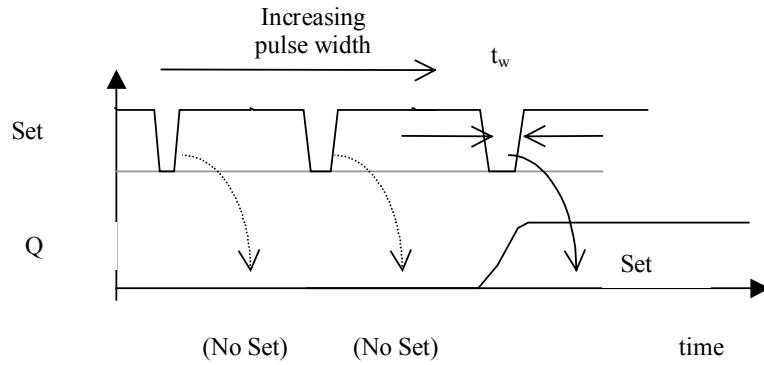


Figure 8-11. Simulation of the RSNAND latch (RSNand.MSK)

### Minimum Pulse Width

One important characteristic of the RS cell is the minimum pulse width that provokes the *Set* or *Reset* effect. The pulse width is usually named  $t_w$ . One possible procedure consists in increasing the pulse width until the output  $Q$  is set to its proper value.

Figure 8-12. Definition of the minimum pulse width  $t_w$  that sets the latch correctly

To conduct this analysis, we program the input *Set* as a Piece-Wise-Linear signal (PWL <Gloss>). This property is derived from the pulse property, but not limited to one single shot. The signal is described using pairs of (time,voltage) information, listed in an array. To access to the PWL property, click the Pulse first, then change the selection in the property window.

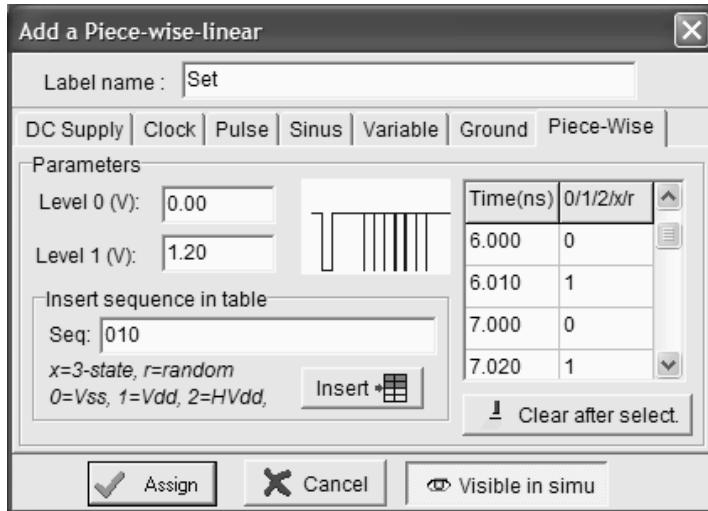


Figure 8-13. Setting a series of negative pulses with increased width, step 10ps on the Set signal using a Piece-Wise-Linear waveform (RSNandTw.MSK)

For this study, the latch outputs should be connected to a load in order to conduct the simulations in a realistic environment. A 10fF virtual capacitor is connected to  $Q$  and  $nQ$ . In figure 8-14, the pulse width which leads to a correct setting of the output  $Q$  is approximately 40ps. We used the Monte-Carlo simulation mode (**Simulation → Simulation Parameters**) to use a random set of technological parameters which accounts for the process variations. Depending on the process parameters, the result is slightly changed. The parameter  $tw$  is usually specified in the data sheet of the RS latch, and more generally that of all latches. Notice that this value is strongly dependent on the loading conditions. We use the BSIM4 model for an accurate analog simulation. In that case, however, the results are similar with model LEVEL 3.

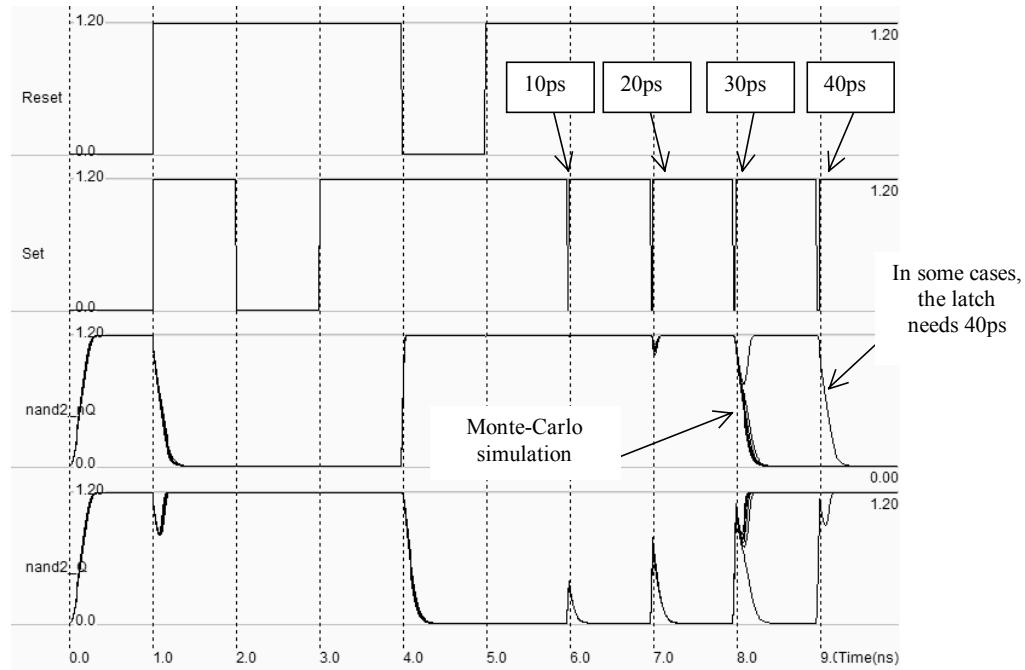


Figure 8-14 Finding the minimum pulse width of the RSNand latch (RSNandTw.MSK)

### 3. D Latch

The vast majority of CMOS integrated circuits use a single clock to synchronize the sequences of operations. This technique is called synchronous design, and is very popular as it allows automated and safe design. A well known D-latch controlled by a clock is shown in figure 8-15. When the clock input is high, the latch output Q follows the changes of the input D. The latch is transparent. Now, when the clock input goes low, the latch is in memory mode, meaning that it produces the value stored in the loop at the output Q. The latch holds the memory state as long as the two inverters are supplied.

D Latch			
D	Clock	Q	nQ
0	0	Q	nQ
0	1	0	1
1	0	Q	nQ
1	1	1	0

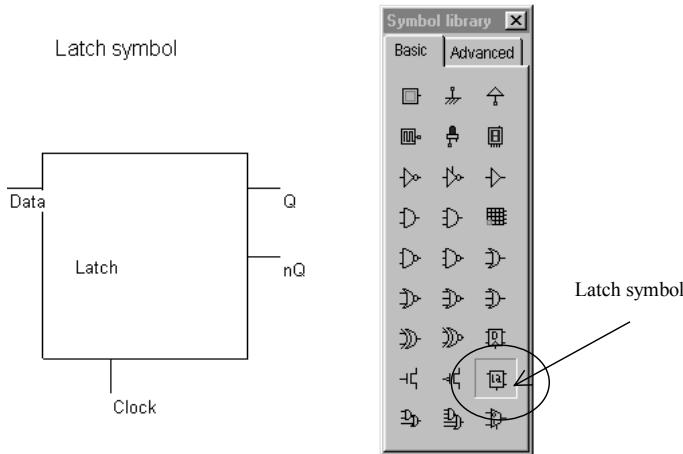


Figure 8-15 The truth-table and symbol of the Dlatch

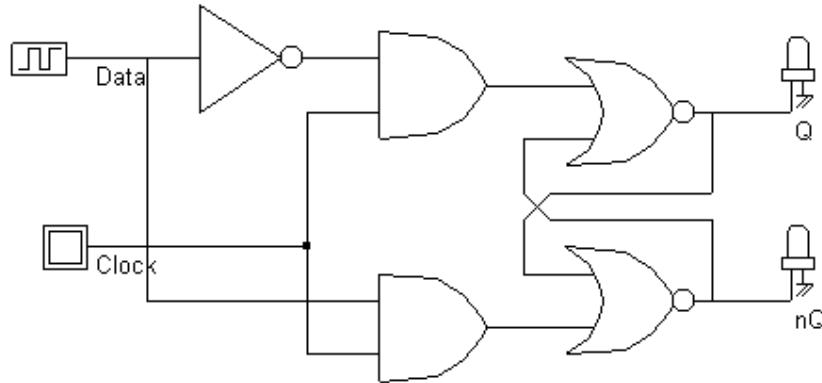


Figure 8-16 The schematic diagram of a D Latch (File DLATCH.SCH).

The truth table and the symbol of the static D-latch, also called Static D-Flip-Flop, are shown in figure 8-15. The schematic diagram of the D-latch is shown in figure 8-16. When performing the logic simulation,  $Q$  and  $nQ$  start with an undetermined state (Appearing in gray in Figure 8-17). Once the clock is active,  $Q$  and  $nQ$  turn to a deterministic state, as the data input  $D$  is transferred to  $Q$ , and its opposite to  $nQ$ . When the clock returns to level 0, the latch keeps its last value.

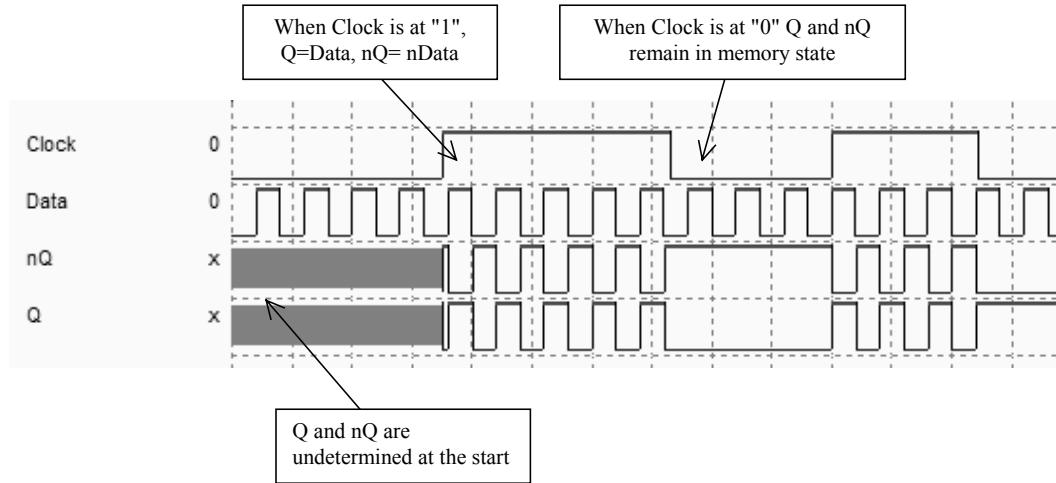


Figure 8-17 Logic simulation of the D Latch (File DLATCH.SCH)

### A Complex Gate implementation of the D-Latch

In order to obtain a compact design, complex gates should be used rather than discrete AND cells which require a NAND gate with an Inverter. The two circuits are compared in figure 8-18: the direct implementation uses 22 MOS devices, the complex gate one only 14. The complex gate solution also leads to shorter propagation delay. Notice the description of the complex function using the expression:

$$s = \sim((a \& b) | c) \quad (\text{Eq} 8-1)$$

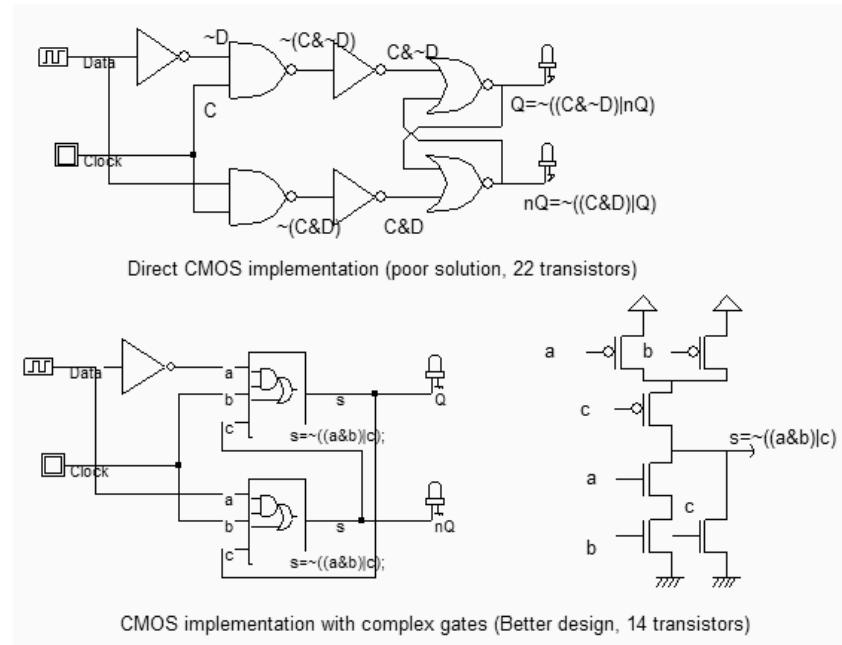


Figure 8-18 CMOS implementation of the D Latch (Dlatch.SCH)

Generate the Verilog text by using the command **File → Make Verilog File**. In Microwind2, click on the command **Compile → Compile Verilog File**. In the example given in figure 8-19, the file 'DLatchCompile.TXT' contains the D-Latch structural description, where we recognize the two complex gates and the Inverter. Also included in the Verilog text are the declarations of the control signals.

```
module DLatchCompile( Data,Clock,Q,nQ);
    input Data,Clock;
    output Q,nQ;
    assign      Q=~(w1&Clock)|nQ;
    assign      nQ=~((Data&Clock)|Q);
    not inv(w1,Data);
endmodule

// Simulation parameters in Verilog Format
always
#1000 Data=~Data;
#1000 Clock=~Clock;
```

Figure 8-19 Generating a Verilog file corresponding to the D-latch (File DLatchCompile.TXT)

When the compiling is complete, the resulting layout appears as shown in Figure 8-20. The layout of the D-latch includes, from left to right, the two complex gates and the inverter. The internal wire is routed on the top of the cell, all I/Os being routed on the bottom.

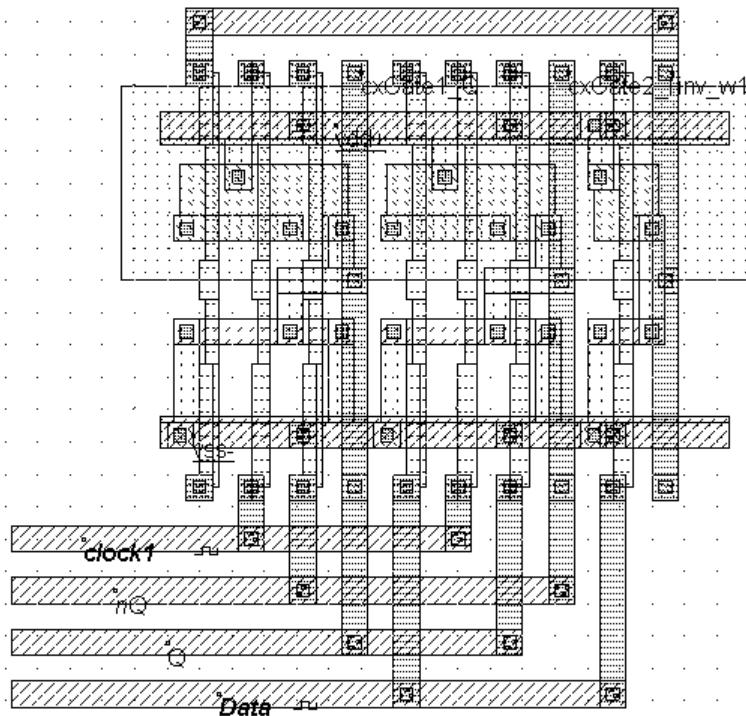


Figure 8-20 Compiling of the DLatch (File DLatchCompile.MSK)

The simulation is reported in figure 8-21. 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 several situations in one single simulation. 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* stay in memory state.

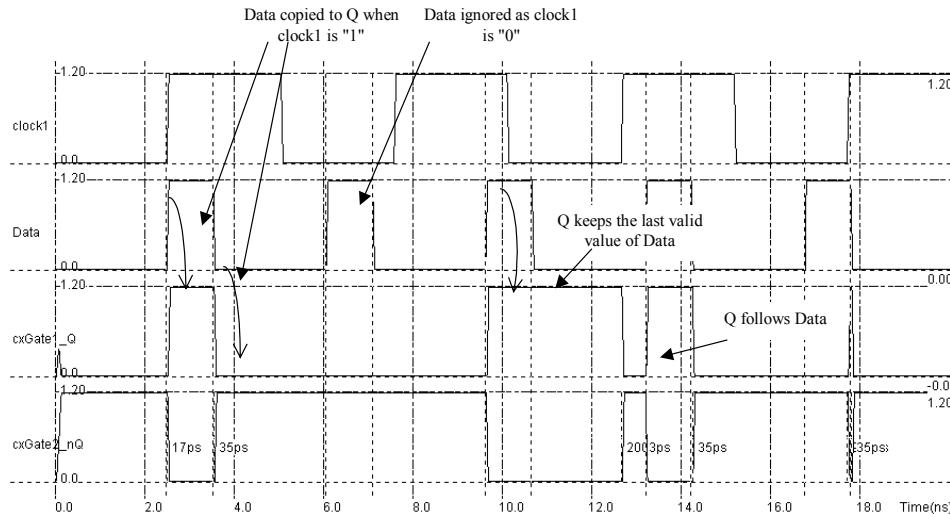


Figure 8-21: The compiled Dlatch at work (*DlatchCompile.MSK*).

### Timing Analysis

When the latch is transparent, the delay between the change of *D* and the change of *Q* is named  $t_{PD}$  (Figure 8-22). The outputs *Q* and *nQ* are usually loaded with a 10fF capacitance to account for the connection to other logic gates through metal interconnects. The delay between the rise of *D* and the rise of *Q* may differ from the delay evaluated between the fall of *D* and the fall of *Q*.

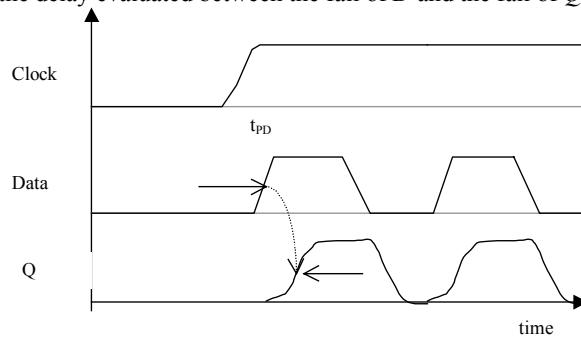


Figure 8-22 Definition of the propagation delay  $t_{PD}$  within D Latch

The minimum pulse width for the clock to pass the input *Data* to the output *Q* properly is labeled  $t_w$ . An illustration of the test setup to characterize  $t_w$  is given in figure 8-23. We change the clock property into a pulse property and increase the width of the clock pulse step by step, until the data passes to the output.

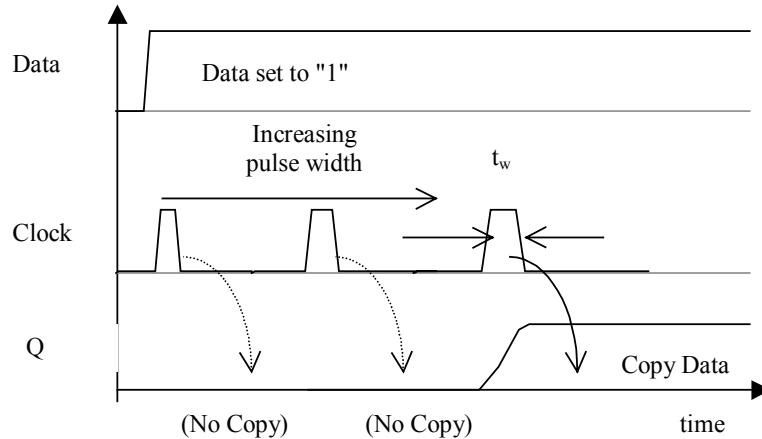


Figure 8-23 Simulation setup to characterize the minimum clock pulse of the D Latch

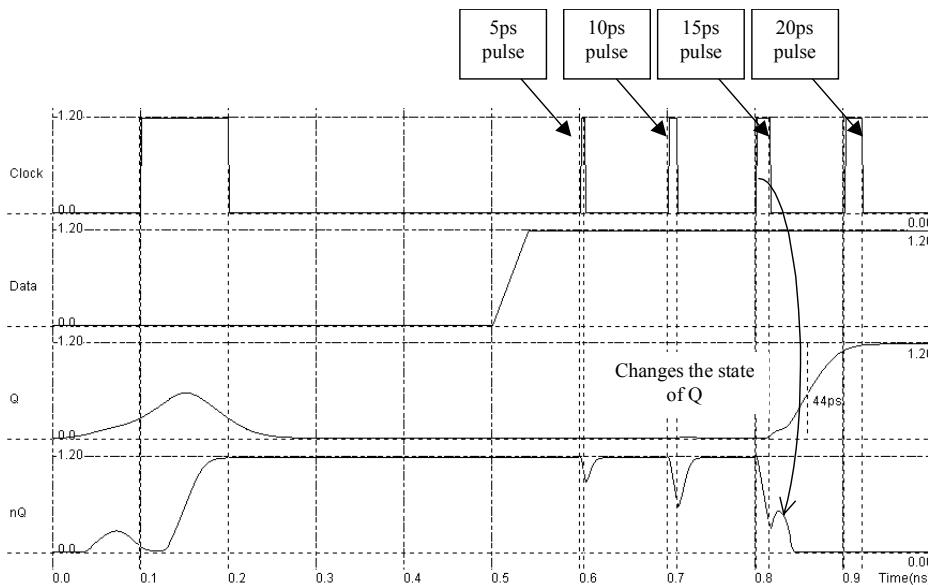


Figure 8-24 characterization of the minimum clock pulse of the D Latch (DlatchLevel.MSK)

We find that a 15ps pulse is required to enable the *Data* information to be transferred to *Q*. In this design, with a 10fF parasitic load, a safe value for  $t_w$  would be 20ps. However, the latch should be tested in extreme conditions (Worst case technological parameters, worst case temperature), and the largest value of  $t_w$  should be kept. In figure 8-24, the delay  $t_{PD}$  between the rise of *D* and the rise of *Q* is around 45ps.

### Compact D-Latch Layout

A very compact design of the D-latch is obtained by using the original two-inverter memory loop and by adding the minimum hardware to change its state : one n-channel pass transistor to inject the new *Data*, and one p-channel to perform the feedback, or to suppress the loop effect when writing a new data. The resulting schematic diagram is shown in figure 8-25. Only 6 MOS devices are required to construct the D-latch. Recall that the complex gate implementation was based on 14 devices.

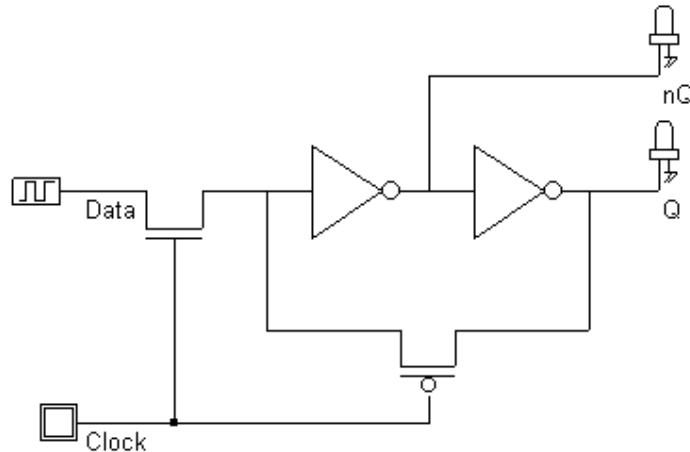


Figure 8-25: The very compact D-latch including 2 inverters and 2 pass transistors (Dlatch.SCH).

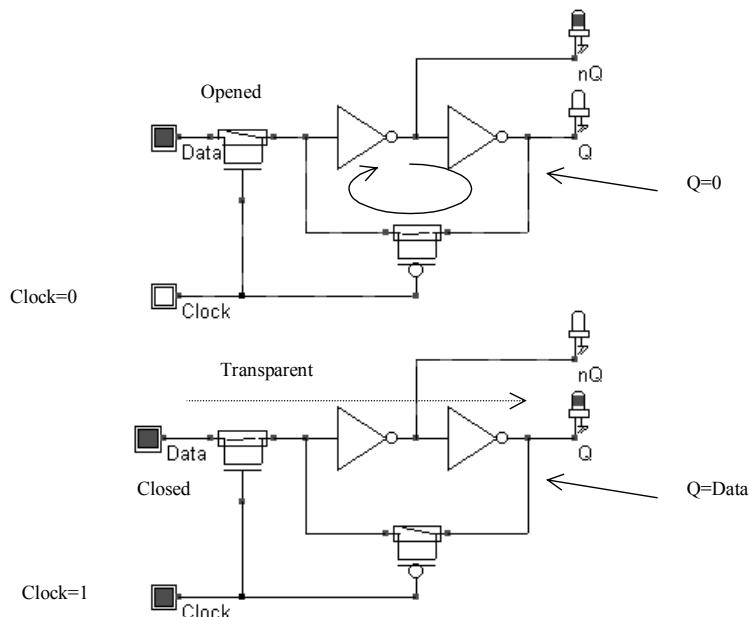


Figure 8-26: Simulation of the very compact D-latch (Dlatch.SCH).

An illustration of the latch at work is given in figure 8-26. When clock is at 0 (Upper configuration), the loop is active thanks to the pMOS, and the Data information is isolated from the loop. When clock is at 1 (Lower configuration), the latch is transparent , and the loop is broken by the pMOS.

Usually, latches proposed in cell libraries propose designs based on transmission gates rather than single pass transistors. The single pass transistor approach leads to more compact designs. However, the voltage amplitude is degraded, so the noise margin is reduced, and the switching speed is slower. Consequently, low power designs would prefer pass transistor approach, while high speed circuits would use pass transistors.

### D-Latch Limitations

The main limitation of the D-latch is its inadequacy to build shift registers or counters. On a positive level of the clock, the whole series of D-latches is transparent to the input data.

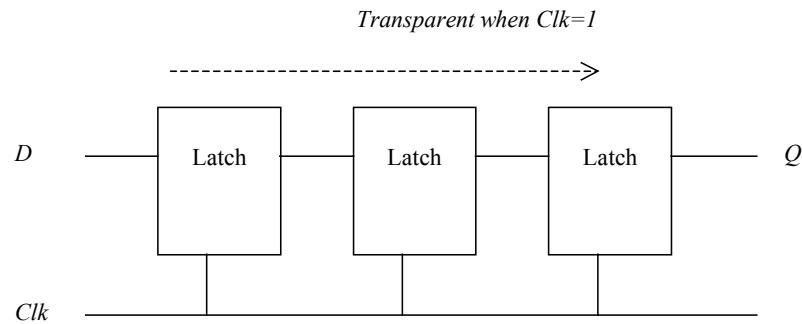


Figure 8-27: Incorrect shift register design using level sensitive D latch.

### 4. EDGE-TRIGGERED D-Register

To overcome the limitation of latches when trying to build registers, we use edge triggered latches, where the information flows from the input  $D$  to the output  $Q$  only at a rise edge of the clock (Figure 8-28). The latch is commonly known as D-Flip Flop (DFF) or D-register (Dreg).

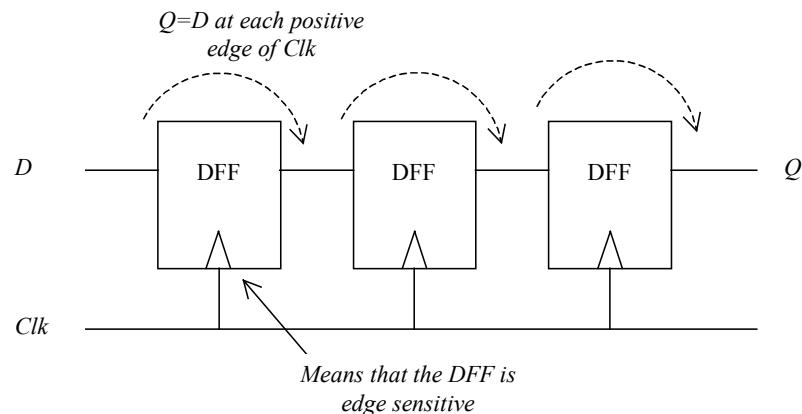


Figure 8-28: Correct shift register design using edge triggered flip-flop.

A well known edge-triggered flip flop is the JK latch. However, the JK is rarely used due to its complexity. In practice, 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.

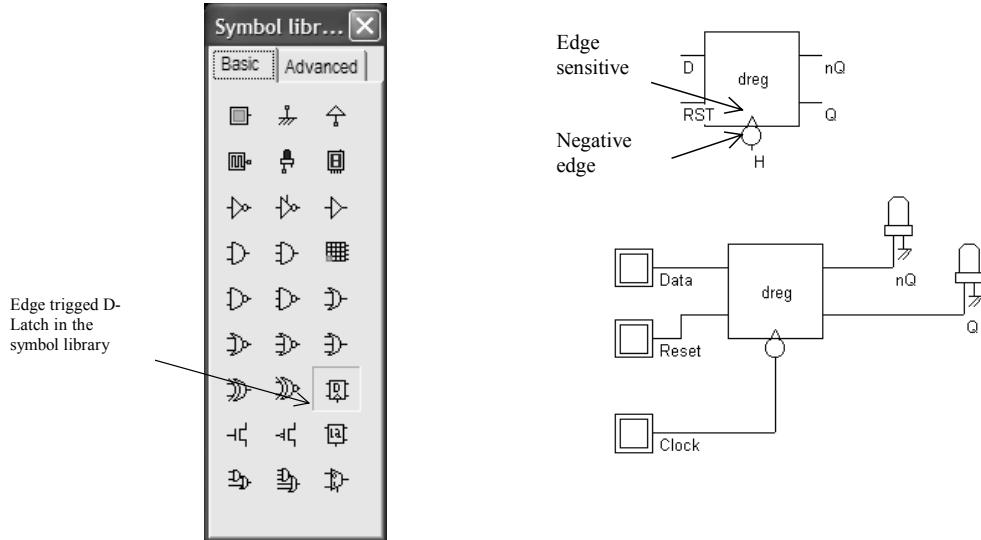


Figure 8-29: The edge triggered D-Flip Flop symbol

The symbol of the Edge-triggered D-flip flop may be found in the symbol palette at the location shown in figure 8-29. The symbol and its typical connection are also given in that figure. Notice the triangle at the clock input that recalls the sensitivity of outputs  $Q$  and  $nQ$  to the edge of the clock. The circle indicates that this Dreg cell is sensitive to a fall edge of the clock.

### Behavioral Model of the DReg

In DSCH, the Dreg symbol is declared by a behavioral model shown in figure 8-30. Notice the difference between previous Verilog descriptions, where the structure of the cell was explicit (AND, Inv, complex gates, XOR, etc..), and this type of declaration which is purely a software description. At that stage, we do not know how the cell is constructed. Another example of behavioral model is the 8051 micro-controller described at the end of chapter 7. The behavioral models are very attractive for fast simulations, but do not accurately account for the physical effects such as delay or consumption.

In the behavioral description of table 8-1, the  $q$  and  $nq$  output signals are declared as registers. This specific variable is equivalent to a memory cell. In DSCH, a register  $reg$  is assigned an elementary memory, which may contain three possible values "1", "0" or "x". The line "always @ (negedge clk)" is equivalent to a test on a fall edge of the signal  $clk$  (Figure 8-30) In that case, the instructions between the begin and the end keywords are executed. The constant "default\_delay" has a value which is updated depending on the loading conditions and the technology. Notice the asynchronous reset of the Dreg when  $rst=1$ .

```
MODULE DReg(d,clk,rst,q,nq);
  input d,clk,rst;
  output q,nq;
  reg q,nq;

  always @ (negedge clk)
```

```

begin
    #(default_delay) q=d;
    #(default_delay) nq=~d;
end
if (rst)
begin
    q=0;
    nq=1;
end
endmodule

```

Table 8-1: Behavioral model of the edge-sensitive D-flip flop used in DSCH

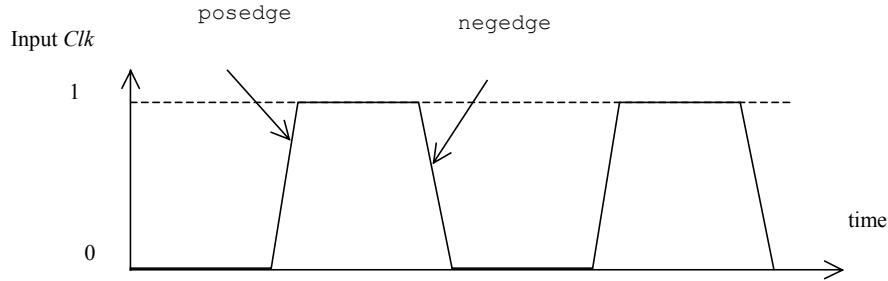


Figure 8-30: Definition of clock edges in Verilog

### Schematic Diagram of the DReg

One very compact implementation of the edge-triggered Dreg is reported below. For clarity, the *Reset* circuit has been removed. The schematic diagram is based on inverters and pass-transistors. It is constructed from two memory loop circuits in series. The cell structure includes a master memory cell (left) and slave memory cell (right).

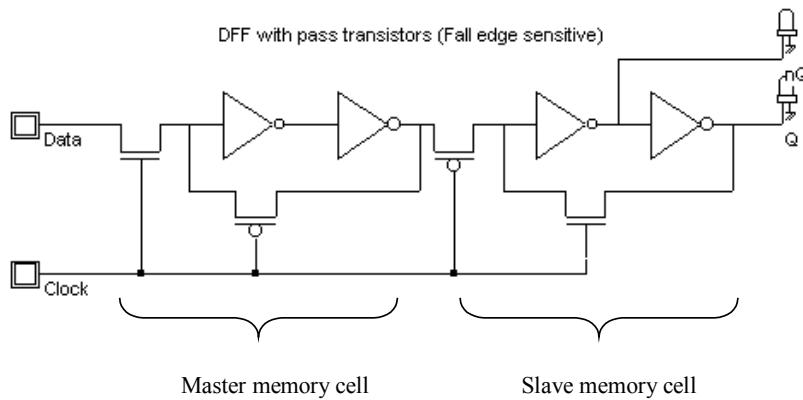


Figure 8-31: A compact implementation of the edge-sensitive Dreg (Dreg.SCH)

In figure 8-32, *clock* is high, the master latch is updated to a new value of the input *D*. The slave latch produces the previous value of *D* on the output *Q*. 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.

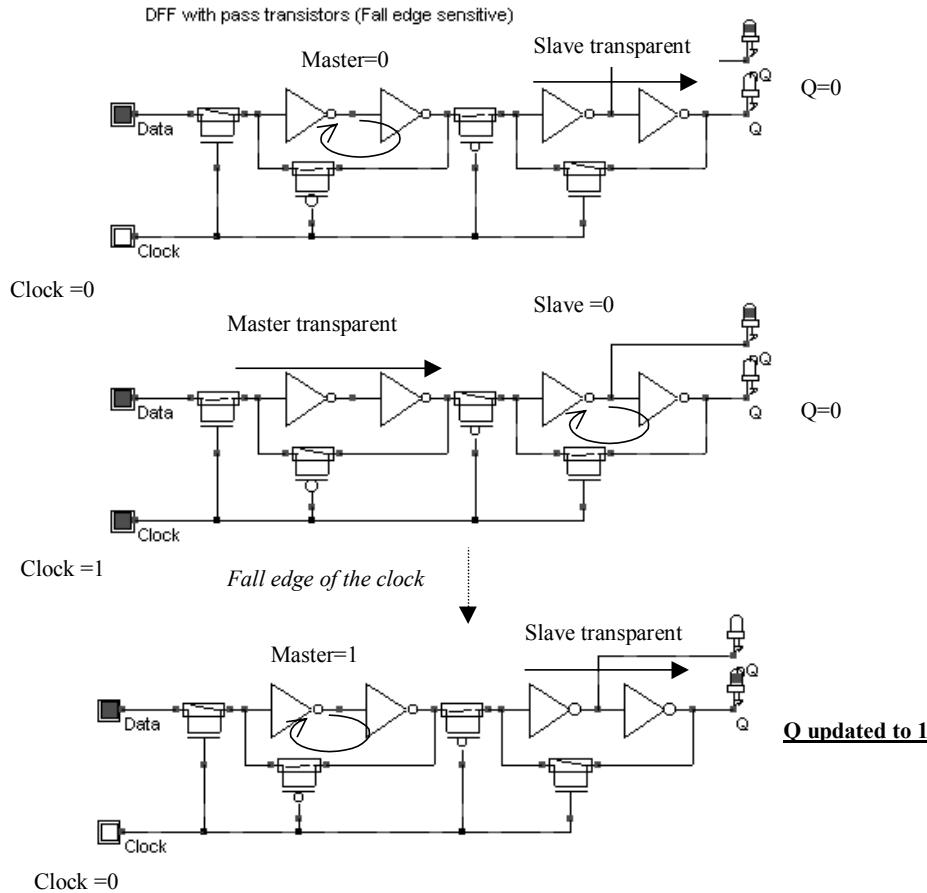


Figure 8-32 The edge-triggered latch and its logic simulation (Dreg.MSK)

### Adding a Reset function to the DReg

The reset function is obtained by a direct ground connection of the master and slave memories, using nMOS devices. This added circuit is equivalent to an asynchronous Reset, which means that  $Q$  will be reset to 0 when *Reset* is set to 1, without waiting for an active edge of the clock.

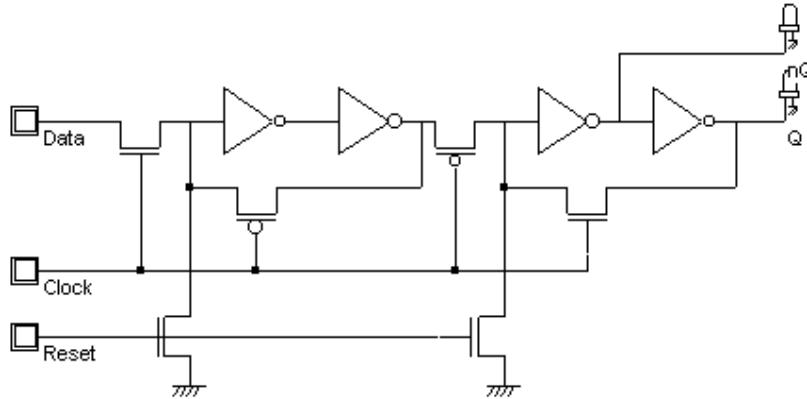


Figure 8-33 Reset function added to the DReg (Dreg.MSK)

### Timing Analysis

Three delay parameters are important in the case of Dreg cells (Figure 8-34): the setup time  $t_{SU}$ , between a valid change of  $D$  and the active edge of the clock, the hold time  $t_H$ , between the active edge of Clock and a change of  $Data$ , and the propagation delay  $t_{PD}$ , between the active edge of  $Clock$  and an updated setup of  $Q$ .

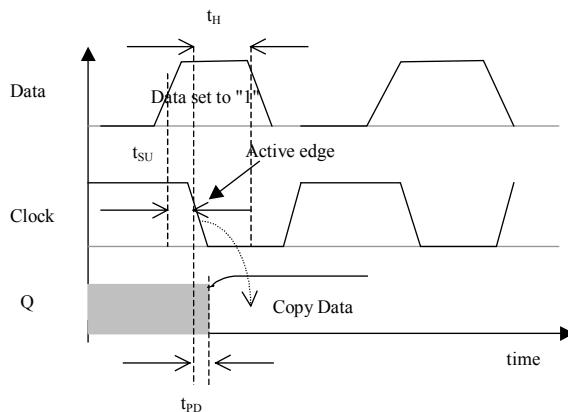
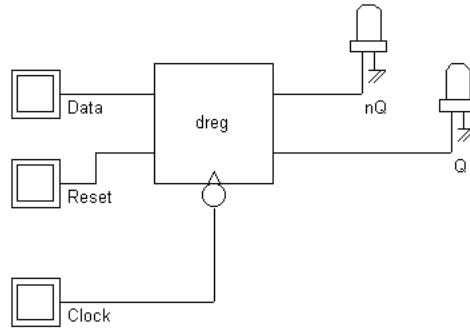


Figure 8-34 Important timing parameters in the DReg cell

### Implementation of the DReg

We create a schematic diagram including the register symbol proposed in the symbol palette of DSCH. Three buttons are added to the circuit to control the inputs  $Data$ ,  $Reset$  and  $Clock$ , as shown in figure 8-35. Using the command **Make Verilog File**, we create the corresponding Verilog description. As can be seen, the register is built up from one single call to the primitive “dreg”.



```

module dregCompile( Clock,Reset,Data,Q,nQ );
  input Clock,Reset,Data;
  output Q,nQ;
  dreg #(19) dreg1(Q,nQ,Data,Reset,Clock);
endmodule

// Simulation parameters in Verilog Format
always
#1000 Clock=~Clock;
#2000 Reset=~Reset;
#3000 Data=~Data;

```

Figure 8-35 The Verilog text corresponding to the Dreg circuit (DregCompile.SCH)

In Microwind, the Verilog text is converted into layout as shown in figure 8-36. The *dreg* primitive is converted into a complex structure including the master and slave memories, each with two inverters and two pass-transistors, as well as one pass transistor for the Reset function.

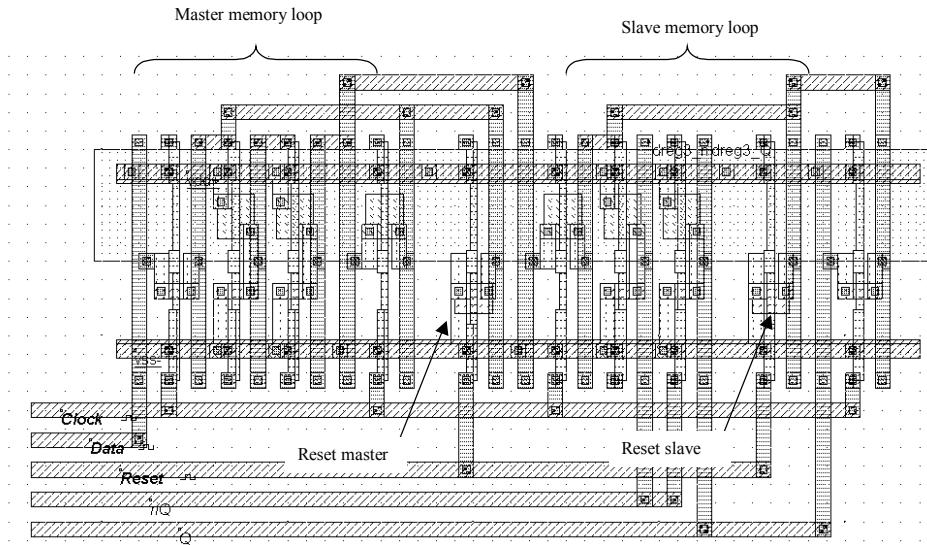


Figure 8-36 Compiling the Dreg with Microwind (DregCompile.MSK)

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. The data *D* is 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 8-37. 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.

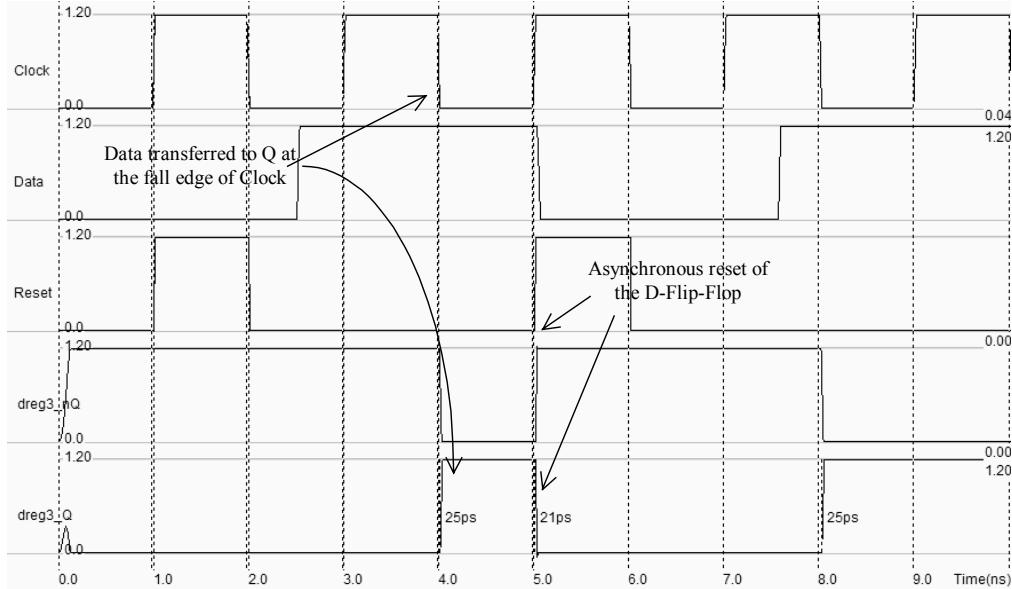


Figure 8-37 Simulation of the DREG cell (DregCompile.MSK)

### Improved Dreg Design

In cell libraries, most latch designs are based on transmission gates rather than single pass transistors. The single pass transistor approach leads to more compact designs, dissipating less power. However, the voltage amplitude is degraded, so the noise margin is reduced, and the switching speed is slower. The *Dreg* cell implemented with transmission gates requires one inverter to generate the *nClock* signal (Figure 8-38). Notice the NAND gate for the *Reset* function which is active on a low level of *nReset*.

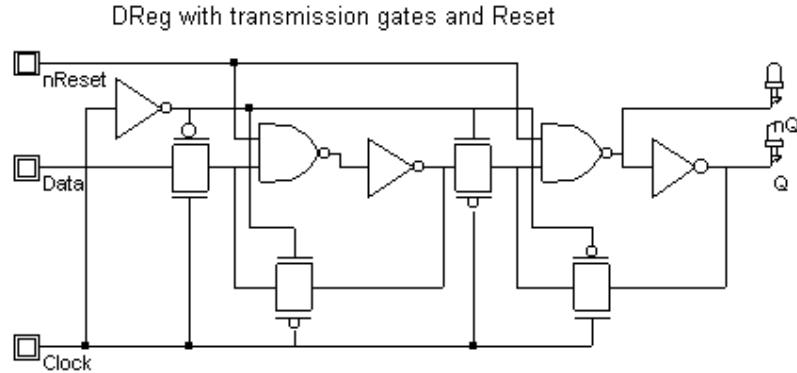


Figure 8-38 Dreg with transmission gates (DregTgate.SCH)

In the case of long distance routing between the Dreg output and its cell destination, the output node  $Q$  is rarely linked to the storage node directly (Figure 8-39). This is crucial for reliability in case of coupling noise, as introduced in chapter 5. Around one millimeter of coupled interconnects is sufficient to provoke crosstalk noise approaching the commutation point of the inverters. With a latch design with direct feedback from  $Q$  to the storage loop (Figure 8-39-a), a strong noise injected by capacitance coupling in an adjacent line may change the state of the register. If the signal  $Q$  is isolated from the storage loop (Figure 8-39-b), the noise vanishes and the memory state is not altered. A modified circuit with isolated outputs is proposed in figure 8-40. Notice the supplementary inverters that increase the propagation delay of the circuit and increase the power consumption.

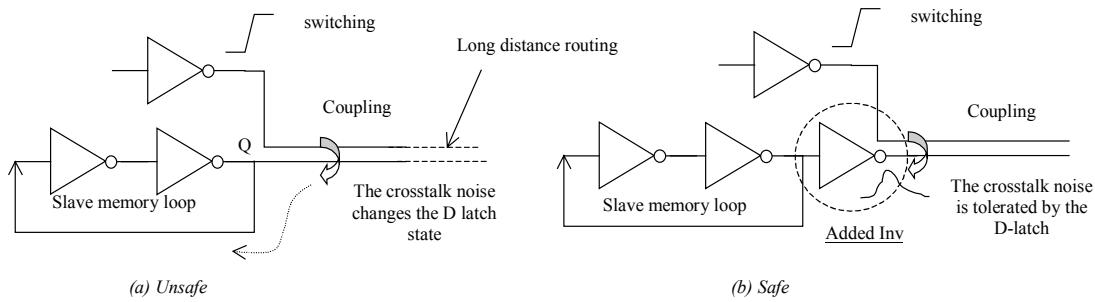


Figure 8-39 The storage loop directly connected to the output node  $Q$  may be affected by a coupling noise.

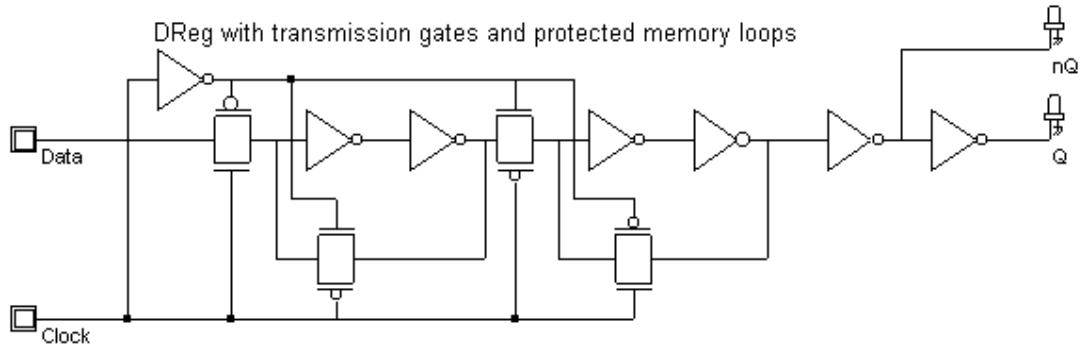


Figure 8-40 The isolated Dreg circuit (DregTgate.SCH)

## 5. Clock Divider

The one-bit counter is able to produce a signal featuring half the frequency of a clock. The most simple implementation consists of a *Dreg* where the output  $nQ$  is connected to  $D$ , as shown in figure 8-41. In the logic simulation, the input *clock* changes the state of *ClockDiv2* at each fall edge. The *reset* signal is active high, and sticks the output to 0.

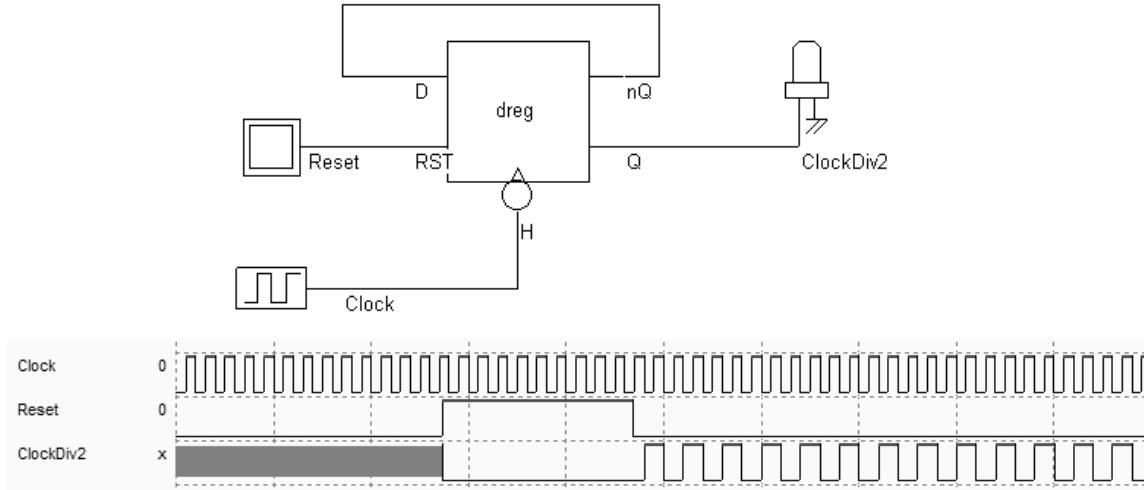


Figure 8-41 Logic simulation of the divider-by-two (ClockDiv2.SCH)

### Maximum operating frequency

The most important parameter to be characterized in the clock divider is the maximum frequency  $f_{max}$  up to which the cell divides properly. Let us extract this frequency  $f_{max}$  using the compiled version of the clock divider.

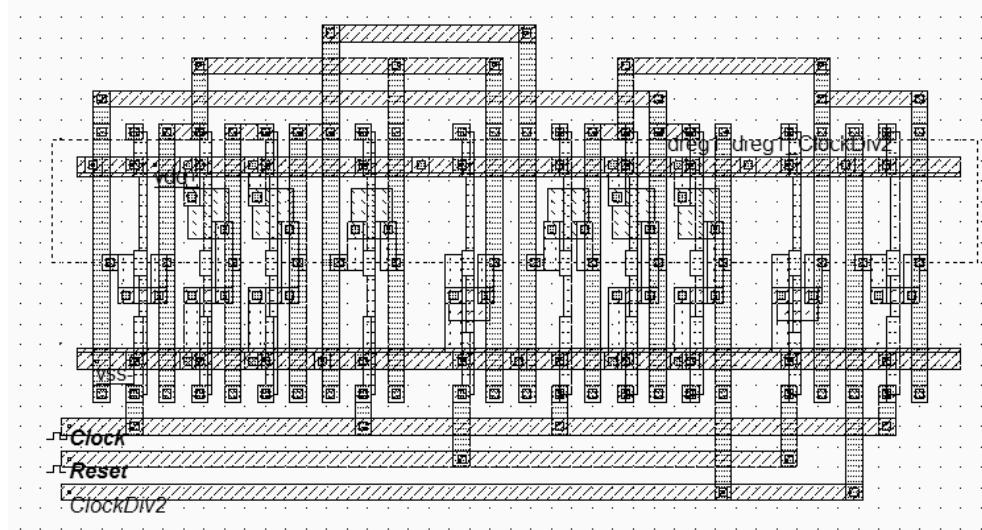


Figure 8-42 The compiled layout of the clock divider (ClockDiv2.MSK)

Firstly, the default clock assigned to the signal *Reset* should be changed into a pulse property, to avoid a cyclic reset of the divider. Secondly, the input *Clock* is reprogrammed as a piece-wise-linear where the input frequency starts from 1GHz and rises up to 10GHz.

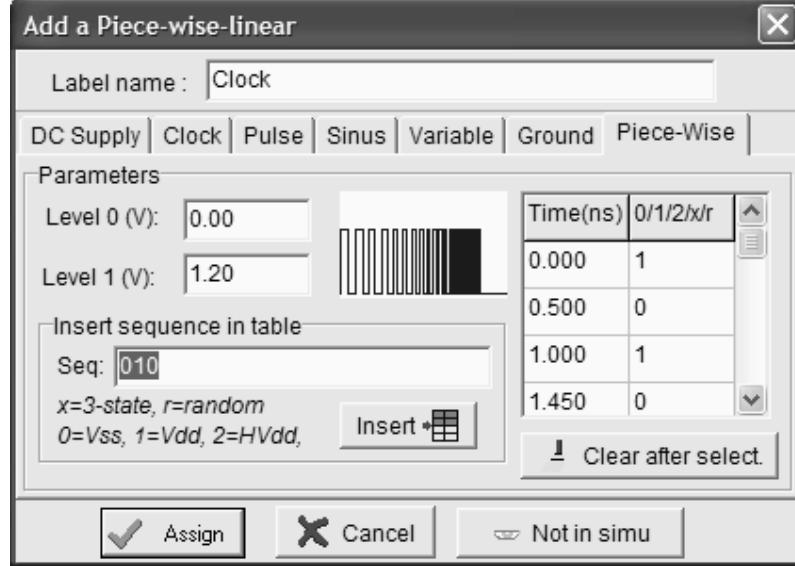


Figure 8-43 The piece-wise-linear description of the Clock with increased frequency (ClockDiv2.MSK)

The recommended simulation mode is **Frequency vs. time**. The frequency variation of the output node *ClockDiv2* appears in the upper part of figure 8-44. It can be seen that the divider circuit works correctly up to 2.5GHz, that is an input frequency of 5.0GHz.

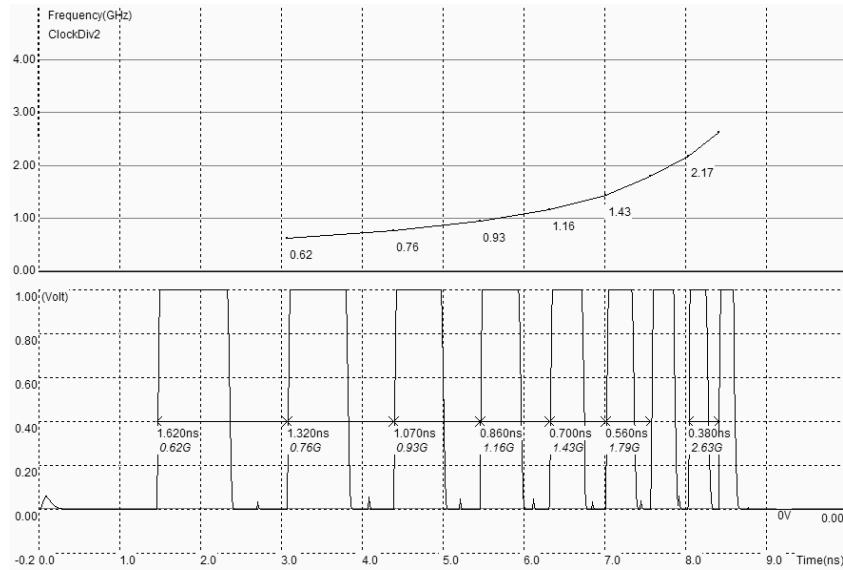


Figure 8-42 Analog simulation of the divider-by-two for the evaluation of the maximum operating frequency (ClockDiv2.MSK)

### Asynchronous Counter 0..15

The *Dreg* cell connected as a one-stage counter cell may be cascaded to create a larger counter circuit.

The clocking of each stage is simply carried out by the previous counter stage output, to form an asynchronous counter circuit. The 4-stage binary counter displays numbers from 0 to 15, using a chain of four *Dreg* cells, as illustrated in figure 8-43. Note the *Reset* signal common to all stages.

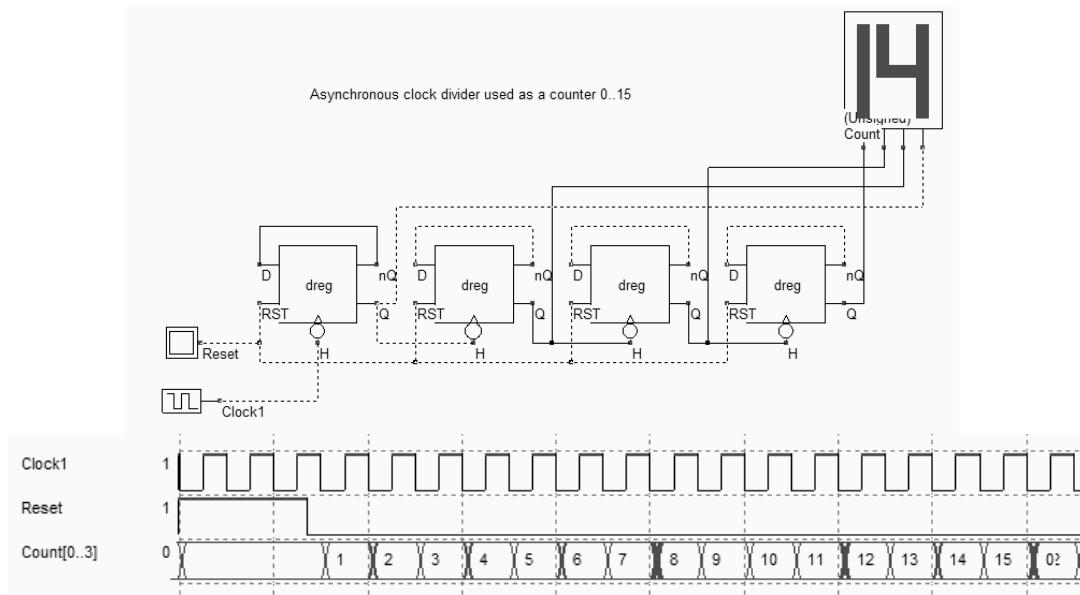


Figure 8-43. The asynchronous counter circuit principles and logic simulation (CountA16.SCH)

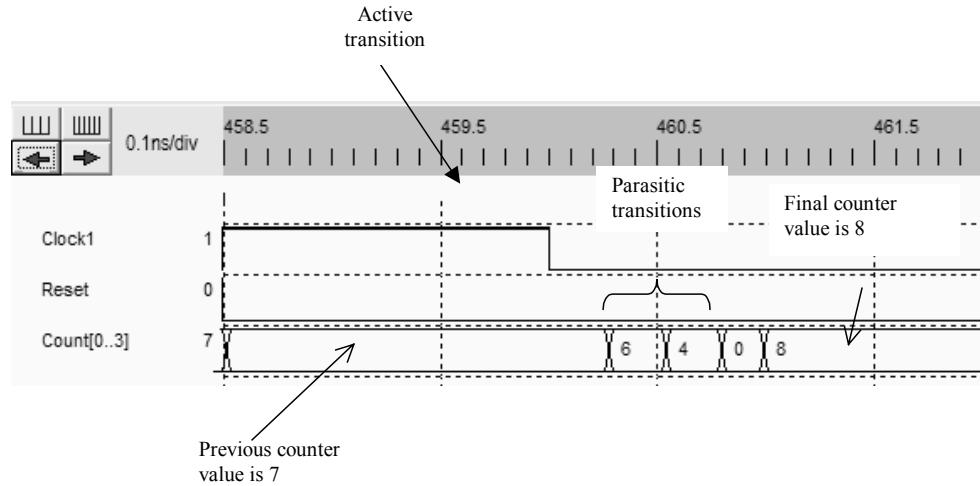


Figure 8-44 Intermediate counter values during asynchronous switching (CountA16.SCH)

The asynchronous counter has several intermediate values during switching due to cascaded delays in the chain. Before the last stage is correctly settled, several parasitic values appear in the chronograms. In figure 8-45, a divide-by-13 counter is proposed. It uses an AND gate connected to the *Reset* control, to provoke a general reset of the *Dreg* cells once the desired number has been attained.

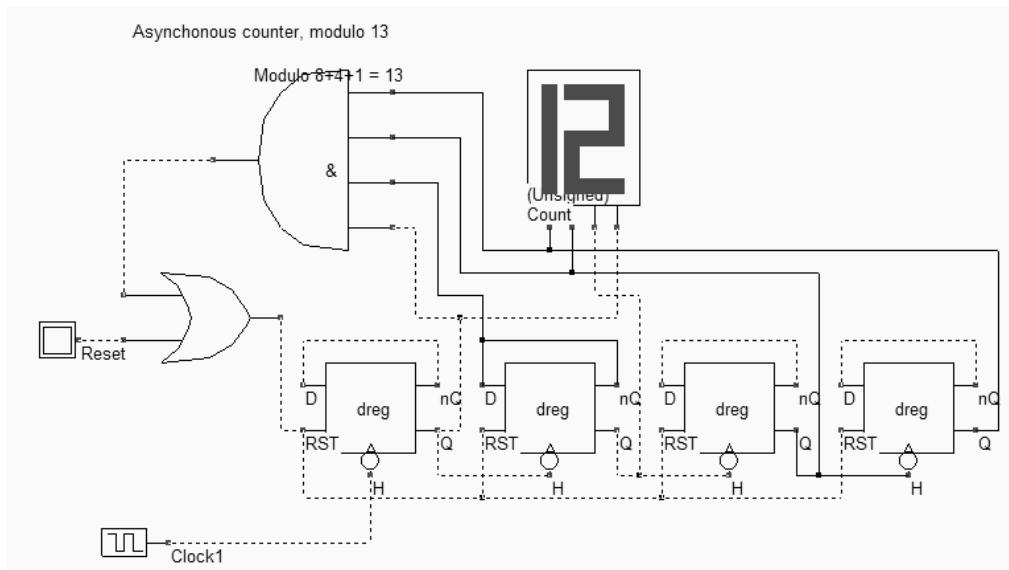


Figure 8-45 Counter 0..12 using an evaluation circuit for a forced Reset (CountA13.SCH)

An undesired *Reset* may be provoked by the AND gate in the instability time interval during which the counter chain changes its value. This is why asynchronous counters are dangerous to use, and synchronous counters are preferred.

## 6. Synchronous Counters

The main difference between asynchronous and synchronous counters is the clock connection. In the case of a synchronous counter, the signal *clock* is shared by all stages of the counter. The synchronous counter shown in figure 8-46 [Weste] uses one adder and one *Dreg* for each stage. The counter performs up and down counting. The *Reset* signal is shared by all *Dreg* cells.

In the simulation, we see that the counter increments the output when Up/Down is at zero. Otherwise, the outputs is decremented.

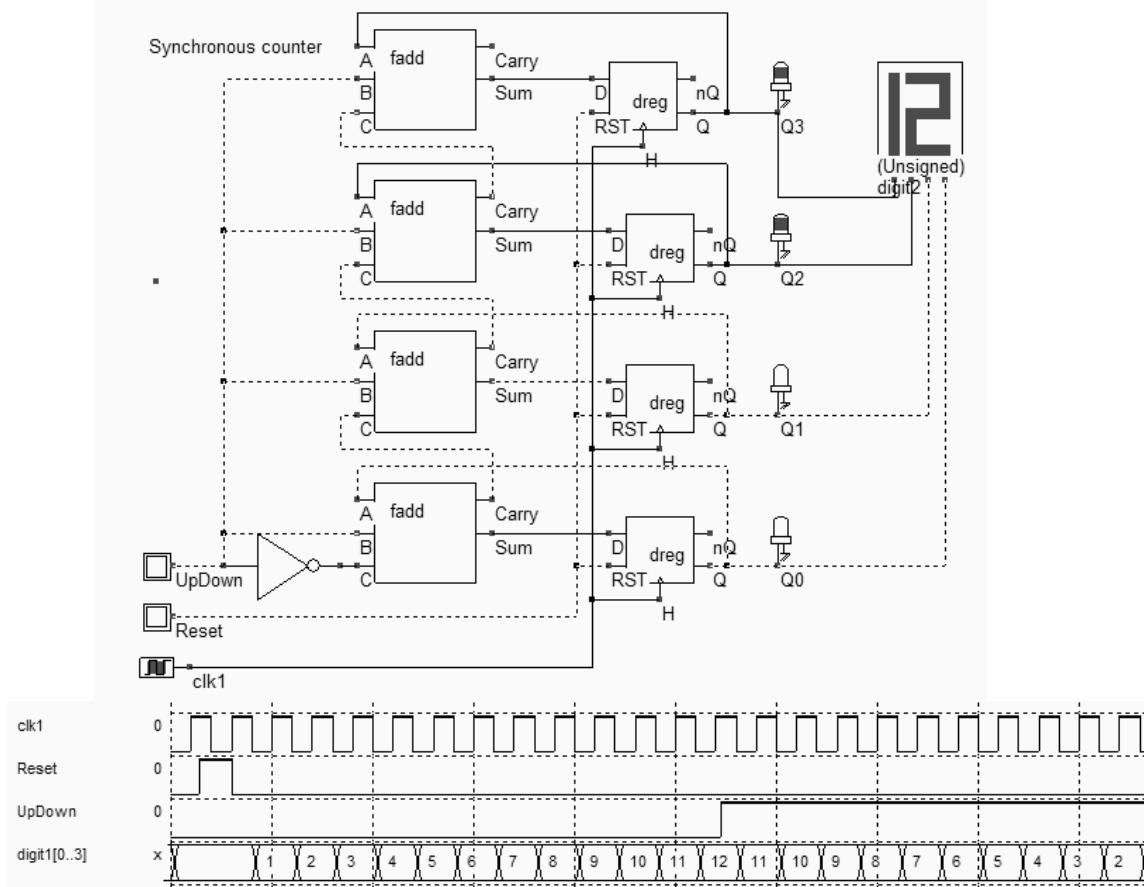


Figure 8-46 Design of a synchronous Up/Down counter using a Full adder and a Dreg (CounterUpDown.SCH)

### Counter Model

The behavioral model of a counter is given in table 8-1. The model description uses the Verilog format, which is very similar to the internal format used to describe models in DSCH.

```
module countUp(clock,reset,count);
    input clock,reset;
    output [3:0] count;
    reg [3:0] count;      // declares a register type for Count

    always
    begin
```

```

if (reset==0) // asynchronous Reset of the counter
  count = 0;
@(negedge clock); // At a fall edge of the clock
if (count==9)
  count = 0;          // After 9, 0
else
  count = count+1;   // otherwise increment the counter
end
endmodule

```

*Table 8-1 The behavioral model of the counter*

The *count* value is declared as a register, which is equivalent to a variable in a programming language.

The count register is assigned immediate values such as 0, or assigned an incremented value, which requires an adder circuit. Notice that Verilog understands the addition "+", the subtraction "-", and several other operators. The value of *count* remains unchanged until a new assignment is executed. The register *count* is changed on a negative edge of the input clock.

## 7. Shift registers

Shift registers are used in many integrated circuits to convert serial logic data into parallel data. In many cases, serial data links are preferred for communication between blocs, as it reduces the number of pins and the size of buses with a positive impact on the interconnection cost. However, the serial link is slower than the parallel interface. Edge sensitive *Dreg* cells are very useful for converting serial data stream into parallel data stream. In Verilog, the shift operation symbol is "<<" for shift left and ">>" for shift right.

In the circuit shown figure 8-47, four Dreg cells are chained by connecting the Q output to the D input of the next stage. The serial data enters by the input *DataIn*. At each fall edge of *Clock*, the register copies the input onto the output. In the upper configuration, a '1' has propagated to the first stage. At the next active edge of the clock, it has propagated to the second stage, while a new one passes through the first stage. Then, a zero is presented on *DataIn*. At the end, the four serial information (1,1,0,0) appears in (*D0,D1,D2,D3*).

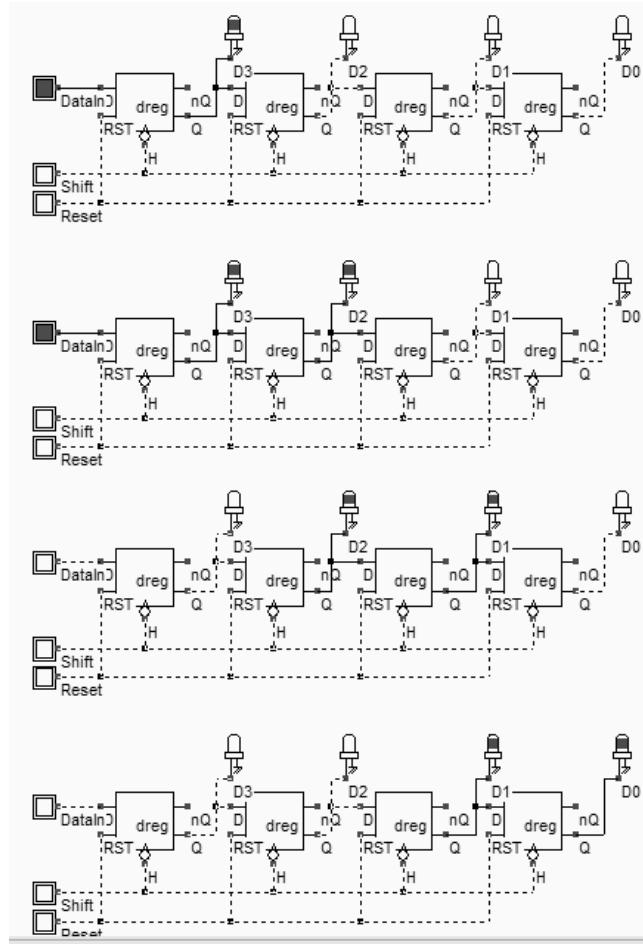


Figure 8-47 Using Dreg cells to build shift registers (ShiftReg4.SCH)

### Programmable delay line

Shift registers may also be used to construct a programmable delay line. The principles of this circuit are to pass the input signal directly, or to delay the information for one period of clock through a *Dreg* cell. The decision circuit is a multiplexor, controlled by the user. The simulation confirms that the input data string *DataIn* is delayed over one, two, three or four clock periods, depending on the number of active *AddDt*.

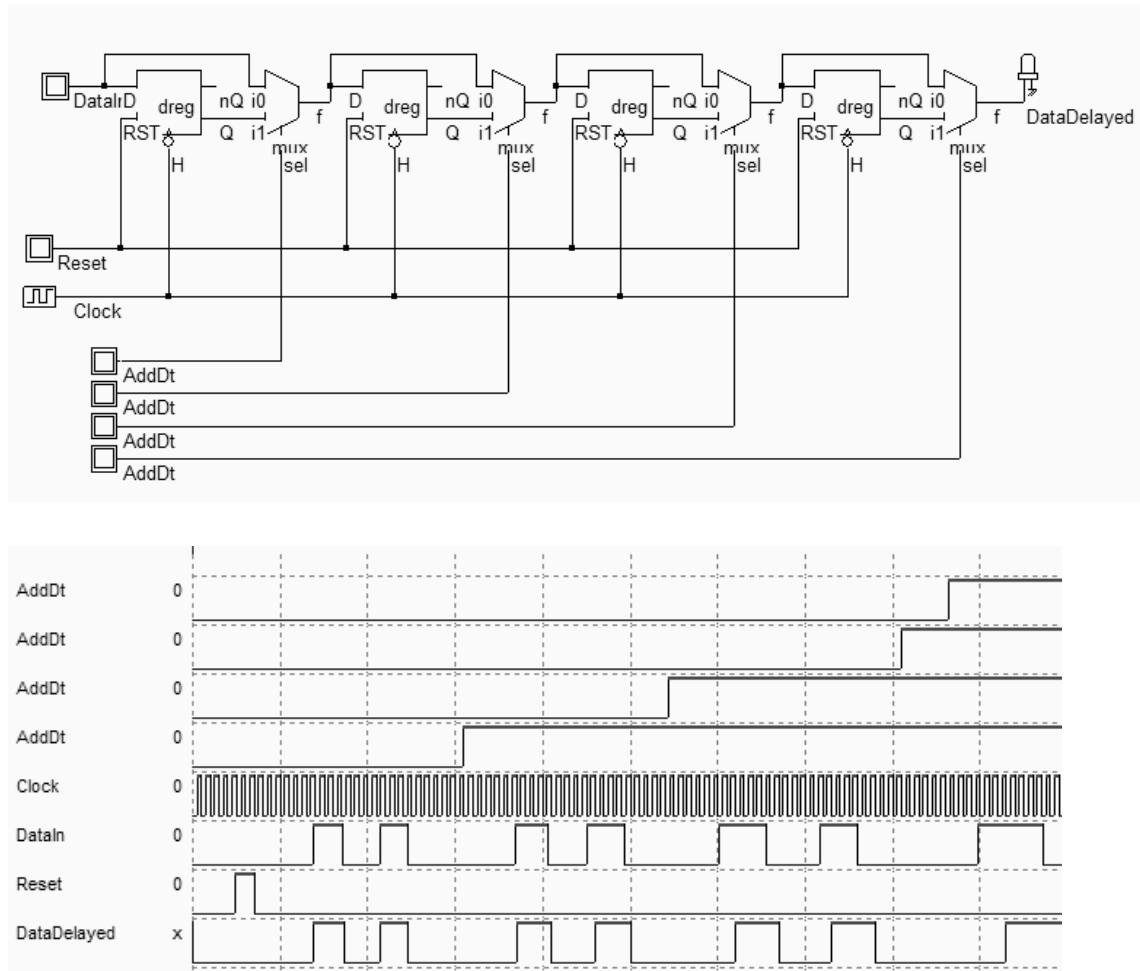
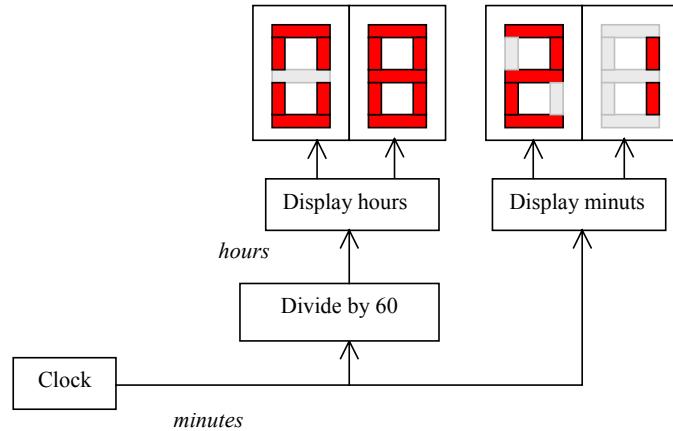


Figure 8-48 A programmable delay line using Dreg (DelayLine.SCH)

## 8. A 24 Hours Clock

We describe in this chapter a circuit to generate a clock that counts hours and minutes. The basic components are described in figure 8-49. The main clock is divided by 60 to create minutes, which is again divided by 60 to create hours. Several counters are needed for this circuit: a counter up to 10 and 6 for minutes, and a counter up to 24 for hours.



*Figure 8.49 Principles of a 24 Hours clock circuit*

## Basic Cell

A possible implementation is based on synchronous counters. In that case, the basic element, called *T-latch*, has two ways of resetting the output  $Q$ . The synchronous reset ( $\sim\text{Sclear}$ ) is effective on a fall edge of the clock, while the asynchronous *Reset* assigns a zero to  $Q$  independently of the clock. The synchronous counter makes an extensive use of the synchronous *Reset*, while the asynchronous *Reset* is kept for initializing the whole hardware, and for starting with a determined state.

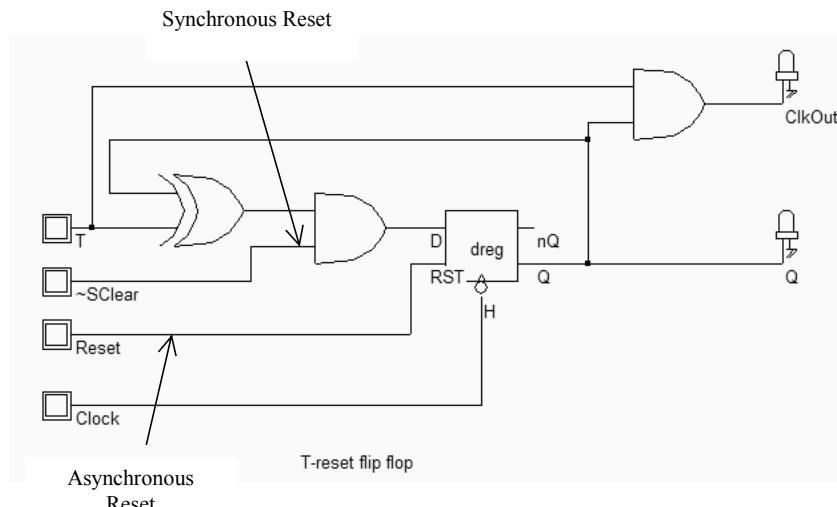


Figure 8-50 The T-Flip flop used fro synchronous counters (ClkBascT.SCH)

The T-Flip flop architecture is derived from the synchronous counter of figure 8-46. The full-adder circuit is replaced by a half adder as the decrement function is useless. The XOR and AND cells of the half adder circuit may be identified in the T-flip flop design.

## Synchronous Counter

Consider the counter design shown in figure 8-51. Four T-Flip-Flop circuits have been cascaded to create a synchronous counter up to 15. The problem is to build an interrupting circuit which resets the registers once the number 9 is attained. This is done with a NAND gate situated on the right lower side of the schematic diagram. The NAND output is connected to the synchronous *Clear* of the latches. When asserted, it should be held during one complete period of clock before the circuit is reset. The circuit starts to count if *Enable* is active (High level) and if the asynchronous *Reset* is inactive (Low level). The synchronous clear  $\sim$ *Clear* should also be high. In that condition, the circuits count from 0 to 9 and then reset. Notice in the chronograms of figure 8-52 the aspect of *Sup9* which corresponds to a clean pulse during one period of clock. This signal will be used to control the next stages of the counters.

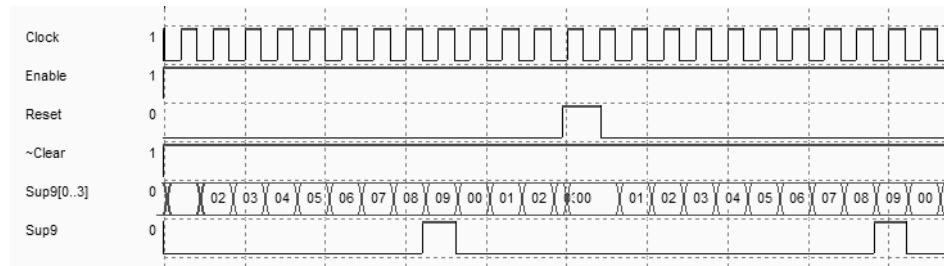
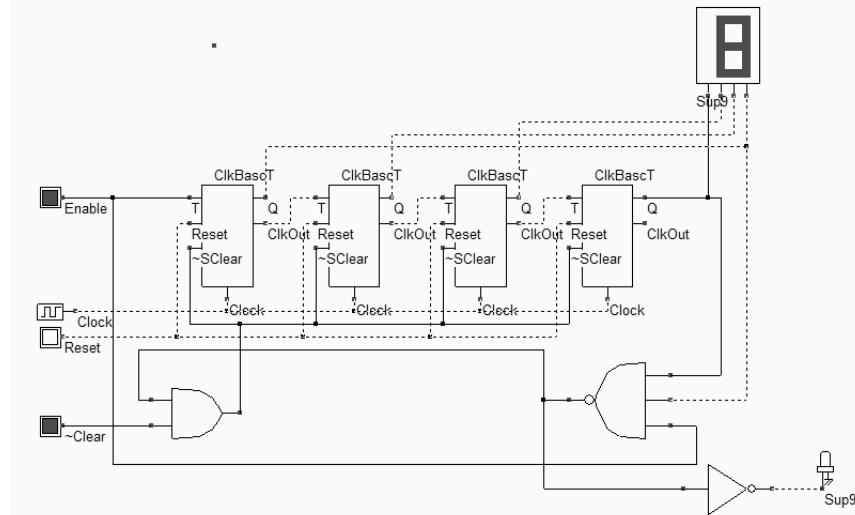


Figure 8-51 Divider by 10 (*ClkDiv\_10.SCH*)

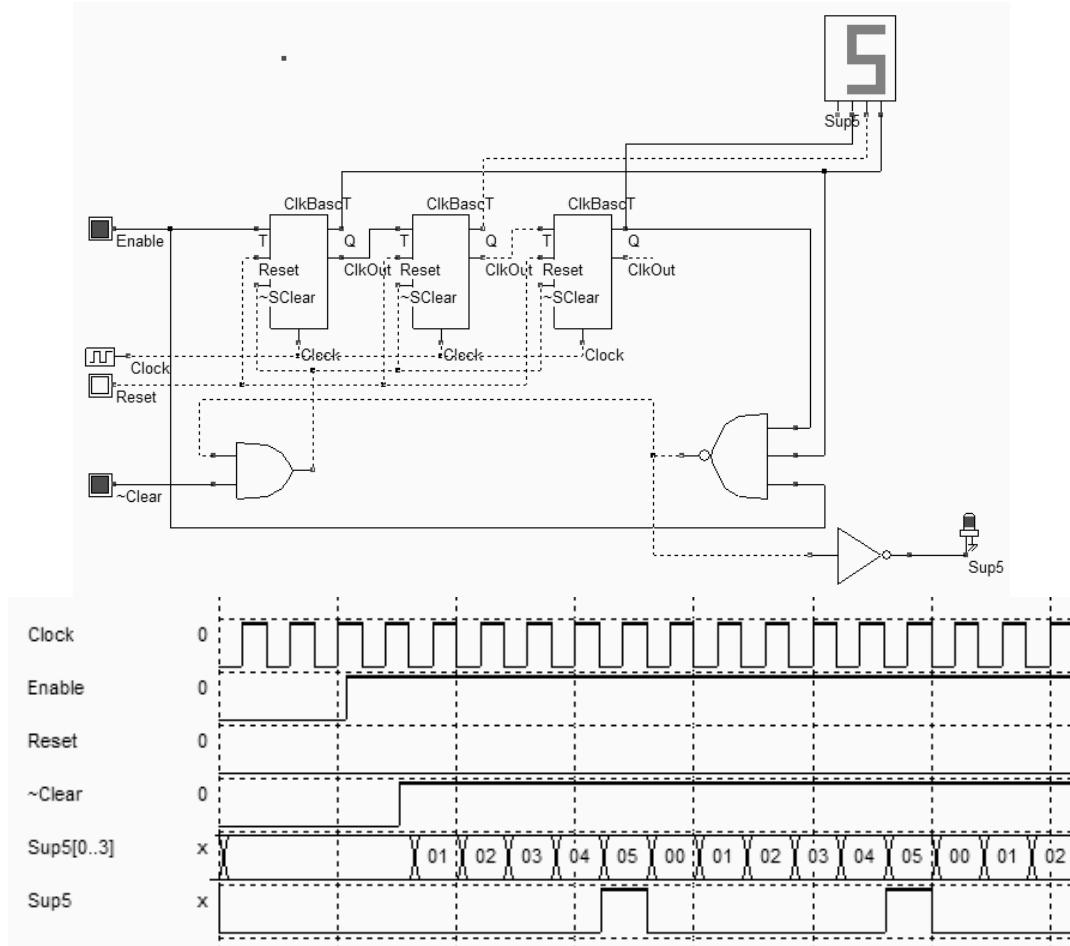


Figure 8-52 Divider by 6 (ClkDiv\_6.SCH)

The synchronous counter shown in figure 8-52 is used to count from 0 to 5. The structure is very similar to the counter from 0 to 9, except that only three T-flip flop circuits are needed. We combine the two counters to create the minute counter from 0 to 59. The most important connection is between the output *Sup9* and the Enable input of *Clk\_Div6*. When *Sup9* is asserted, the counter from 0 to 5 is activated, and the next clock edge will increase the slave counter. Otherwise, only the master counter from 0 to 9 is working.

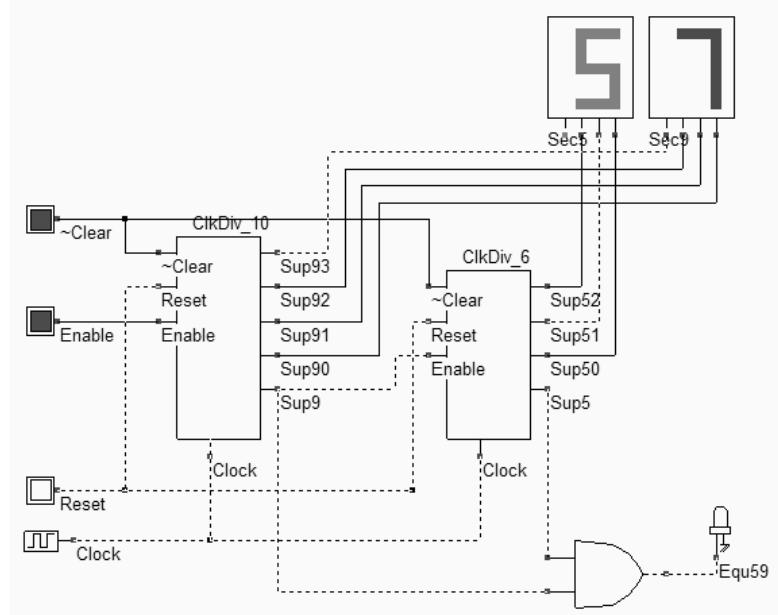


Figure 8-53 The minute counter (ClkDiv\_60.SCH)

The circuit for counting hours reuses the circuits *ClkDiv\_10* and needs a new circuit *ClkDiv\_3*. The signal *Equ23* is generated when the number 3 appears in the master counter [0..9] and the number 2 appears in the slave counter [0..2] (Figure 8-54).

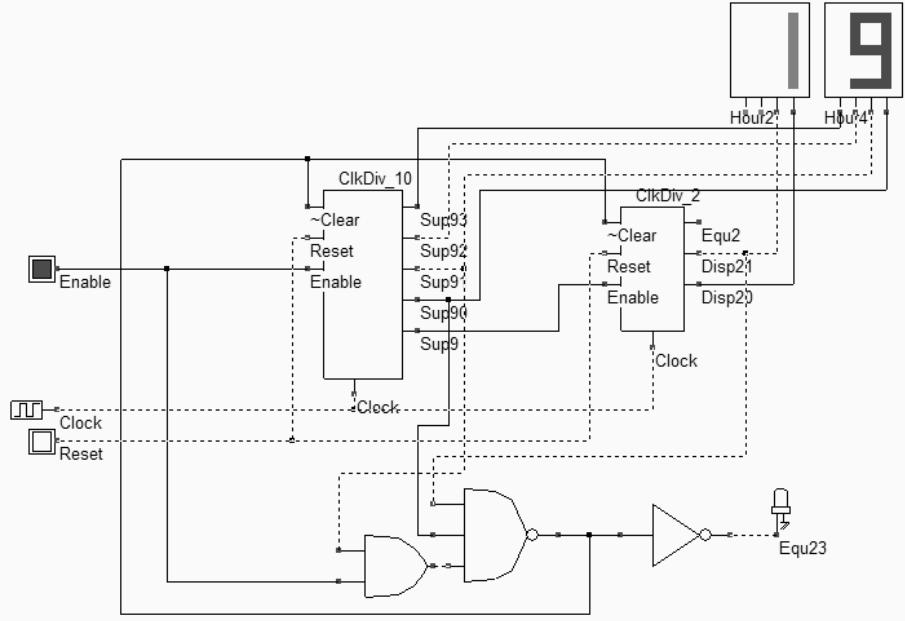


Figure 8-54 The hour counter (ClkDiv\_24.SCH)

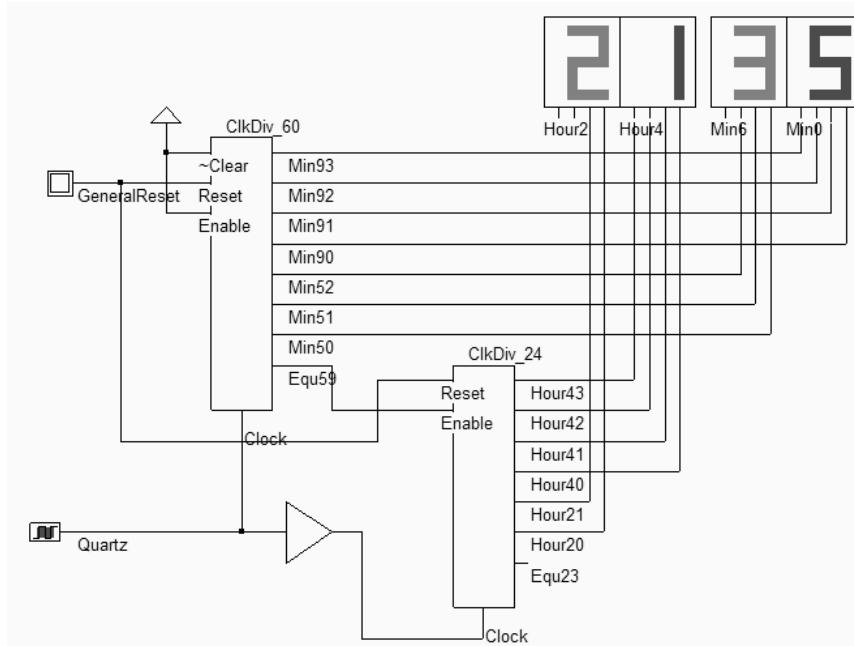


Figure 8-55 The complete schematic diagram of the 24H00 clock counter (Clk24H00.SCH)

The final schematic diagram of the clock is shown in figure 8-55. The circuit works properly up to a maximum frequency. If we increase the frequency of the main clock, the clock resets erratically. This is due to the cumulated delay, which is the sum of all delay stages. Also notice the buffer on the clock signal, used to delay the fall edge of the clock on the *ClkDiv\_24* circuit, as the enable signal is issued from the *ClkDiv\_60* circuit through *Equ59* with a significant delay.

## 9. Conclusion

In this chapter, we have introduced the elementary latch based on two inverters, and the RS latch with its NAND and NOR implementation. The D-latch has also been detailed, and the timing performances have been analyzed at layout level. The edge-triggered register was described from a behavioral and structural point of view. Several versions of this register have been reviewed, and some main applications have been illustrated: the clock divider, the asynchronous counter, the shift register and the programmable delay line. At the end of this chapter, synchronous counter circuits have been presented, with an application to a complete 24 hours clock.

## References

[xxx] V. Stojanvoic, V. Oklobdzija "Comparative Analysis of Master Slave Latch & Flip Flops for High Performance and Low Power Systems", IEEE Journal of Solid State Circuits, Vol 3, April 1999, pp 536-548

[Weste] <tbd>

[Uyemura] <tbd>

## EXERCISES

- 8.1 Compare the current consumption and switching performances of two versions of D-latch: the complex gate implementation and the AND/NOR implementation.
- 8.2 Using DSCH, realize the following schematic diagram, and simulate its behaviour with a clock at the input *In*. What is the functionality of that circuit?

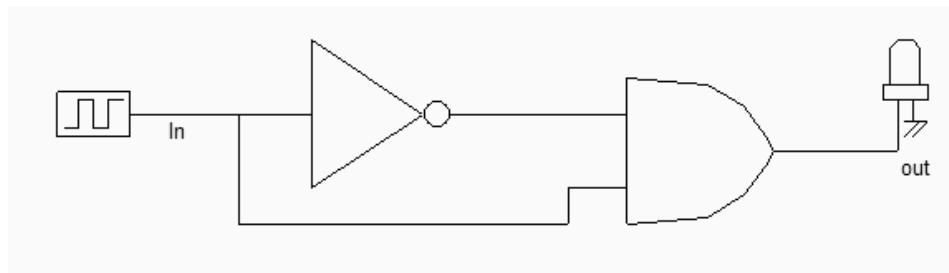


Figure 8-56 Circuit to be analyzed

- 8.3 How does this latch work? Implement this schematic diagram using Microwind. What is the effect of :
- Degrading the strength of inverter *Inv1*?
  - Adding a physical capacitor on node *n1*?
  - Adding more inverters between *Inv1* and *And1*?

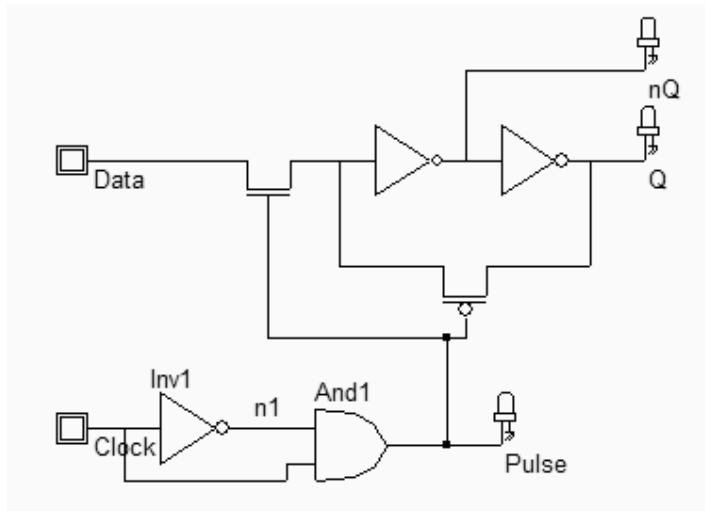


Figure 8-57 Latch to be analyzed

# 9

## Field Programmable Gate Array

This chapter introduces the principles, implementation and programming of configurable logic circuits, from the point of view of cell design and interconnection strategy.

### 9.1 Introduction

Field programmable gate arrays (FPGA) are specific integrated circuits that can be user-programmed easily. The FPGA contains versatile functions, configurable interconnects and an input/output interface to adapt to the user specification. FPGAs allow rapid prototyping using custom logic structures, and are very popular for limited production products. Modern FPGA are extremely dense, with a complexity of several millions of gates which enable the emulation of very complex hardware such as parallel microprocessors, mixture of processor and signal processing, etc... One key advantage of FPGA is their ability to be reprogrammed, in order to create a completely different hardware by modifying the logic gate array. The usual structure of FPGA is given in figure 9-1.

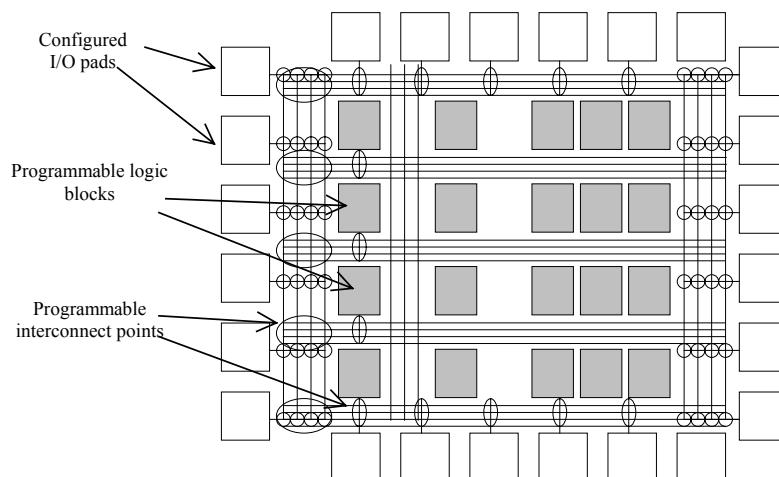


Figure 9-1: Basic structure of a field programmable gate array

One example of a very simple function (3-input XOR) implemented in a FPGA is given in figure 9-2. Three pads on the left are configured as inputs, one logic block is used to create the 3-input XOR and one pad on the right is used as output. The propagation of signals is handled by interconnect lines, connected together at specific programmable interconnect points.

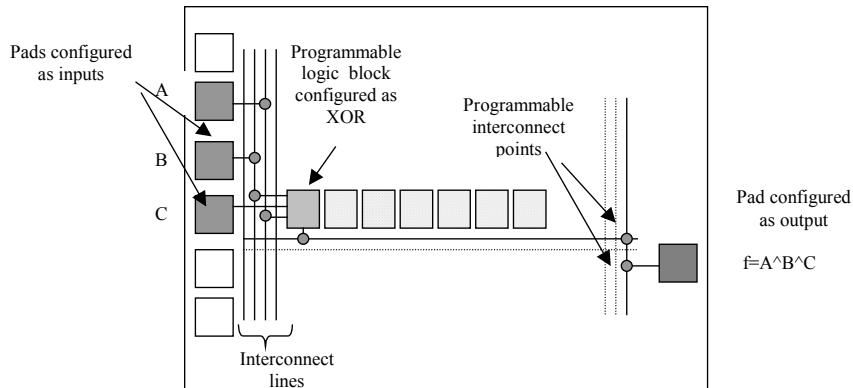


Figure 9-2: Using a field programmable gate array to build a 3-input XOR gate

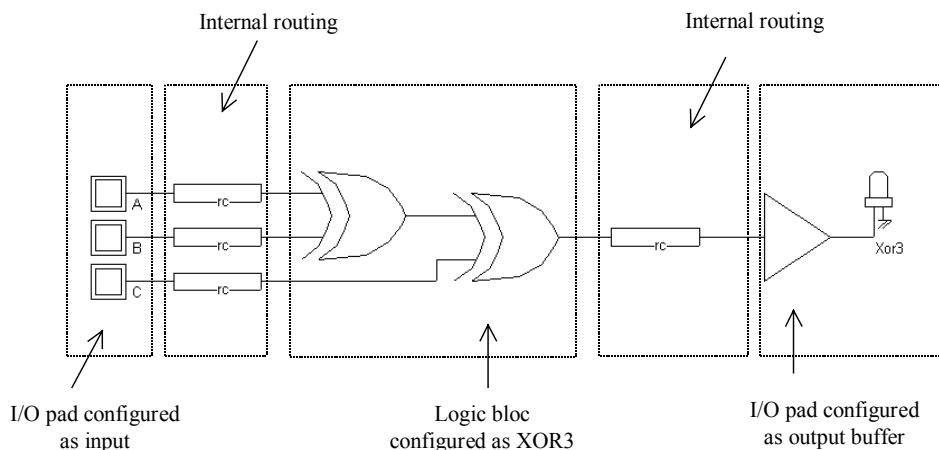


Figure 9-3: Equivalent circuit for the FPGA configured in XOR3 gate

Three pads are configured as inputs and represent the logical information A,B and C (Figure 9-3). An internal routing path is created to establish an electrical link between the I/O region and the logic bloc. Internally, the logic bloc may be configured in any combination of sequential basic function. Each logic bloc usually supports 3 to 8 logic inputs. In our example, the bloc is configured as a 3-input XOR. Then, other internal routing wires are configured in order to carry out the signal to an I/O pad configured as an output. The global propagation delay of such architecture is evidently very high, if compared to a 3-input XOR gate that may be found in the cell library. This is usually the price to pay for configurable logic circuits.

Notice that FPGA not only exist as simple components, but also as macro-blocs in system-on-chip designs (Figure 9-4). In the case of communication systems, the configurable logic may be dynamically changed to adapt to improved communication protocol. In the case of very low power systems, the configurable logic may handle several different tasks in series, rather than embedding all corresponding hardware that never works in parallel.

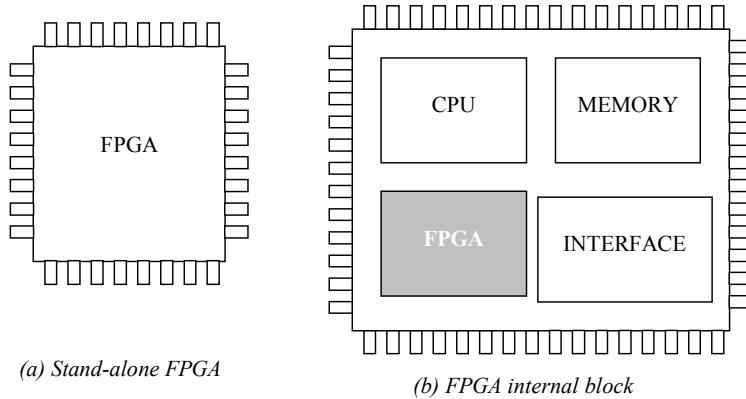


Figure 9-4: FPGA exist as stand-alone Ics or blocs within a system-on-chip

## 9.2 Configurable Logic Circuits

The programmable logic block must be able to implement all basic logic functions, that is INV, AND, NAND, OR, NOR, XOR, XNOR, etc... Several approaches are used in FPGA industry to achieve this goal. The first approach consists in the use of multiplexor, the second one in the use of look-up tables.

### MULTIPLEXORS

Surprisingly, a two-input multiplexor can be used as a programmable function generator, as illustrated in table 9-1. Remember that the multiplexor output  $f$  is equal to  $i_0$  if  $en=0$ , and  $i_1$  if  $en=1$ . For example, the inverter is created if the multiplexor input  $i_0$  is equal to 1,  $i_1$  is equal to 0, and enable is connected to  $A$ . In that case, the output  $f$  is the  $\sim A$ . The figure 9-5 describes the use of multiplexor to produce the OR, AND, NOT and BUF functions.

Function	Boolean expression for output $f$	$i_0$	$i_1$	$en$
BUF ( $A$ )	$f=A$	0	$A$	1
NOT ( $A$ )	$f=\sim (A)$	1	0	$A$
AND ( $A, B$ )	$f=A \& B$	0	$B$	$A$
OR ( $A, B$ )	$f=A   B$	$B$	1	$A$

Table 9-1: use of multiplexor to build logic functions

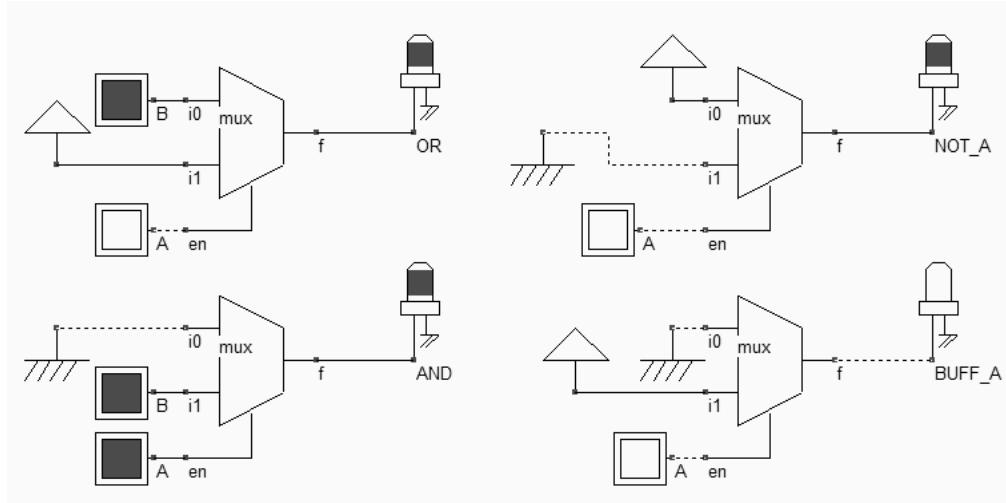


Figure 9-5: use of multiplexor to build logic functions (fpgaMux.SCH)

Although NOT, AND and OR are directly available, other functions such as NAND, NOR and XOR cannot be built directly using a single 2-input multiplexor, but need at least two multiplexor circuits.

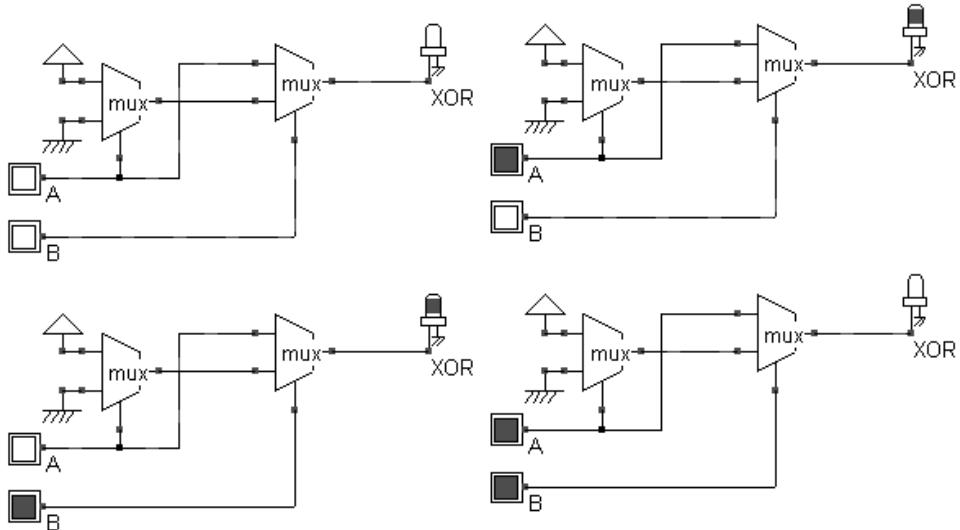
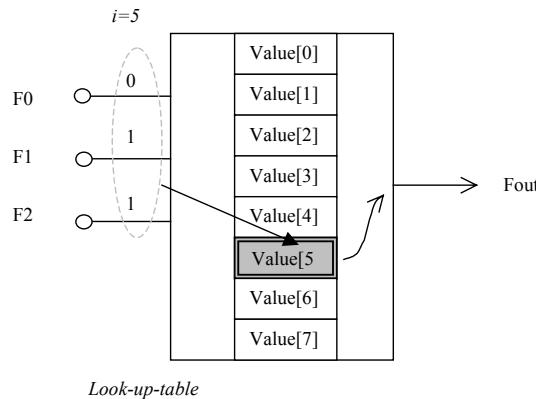


Figure 9-6: The XOR gate built from 2 multiplexor circuits (fpgaMux.SCH)

The XOR function is shown in figure 9-6. The 4-input XOR gate would require 6 multiplexor cells. Remember that each multiplexor cell consists of a minimum of 6 transistors for a buffered output, and has 3 delay stages (The two inverters and the pass transistor). The XOR4 implementation would comprise a total of 18 delay stages, which is far too important. Therefore, the multiplexor approach is not very efficient for many logical functions.

## Look Up Table

The look-up table (LUT) is by far the most versatile circuit to create a configurable logic function. The look-up table shown in table 9-2 has 3 main inputs  $F0, F1$  and  $F2$ . The main output is  $Fout$ , which is a logical function of  $F0, F1$  and  $F2$ . The output  $Fout$  is defined by the values given to  $Value[0]..Value[7]$ . The three values  $F0, F1, F2$  create a 3-bit address  $i$  between 0 and 7, so that  $Fout$  gets the value of  $Value[i]$ . In the example of table 9-2, the input creates the number 5, so  $Value[5]$  is routed to  $Fout$ . The table below gives  $Value[i]$  for the most common logical functions of  $F0, F1$  and  $F2$ .



Function	Value [0]	Value [1]	Value [2]	Value [3]	Value [4]	Value [5]	Value [6]	Value [7]
$\sim F0$	0	1	0	1	0	1	0	1
$\sim F1$	0	0	1	1	0	0	1	1
$\sim F2$	0	0	0	0	1	1	1	1
$F0 \& F1$	0	0	0	1	0	0	0	1
$F0   F1   F2$	0	1	1	1	1	1	1	1
$F0 \wedge F1 \wedge F2$	0	1	1	0	1	0	0	1

Table 9-2: Link between basic logic functions and the information stored in  $Value[0]..Value[7]$

In the case of the 3-input XOR, the set of values of  $Fout$  given in the truth-table of table 9-3, must be assigned to  $Value[0]..Value[7]$ . In the schematic diagram shown in figure 9-7, we must assign manually the  $Fout$  truth-table to each of the 8 buttons. Then  $Fout$  produces the XOR function of inputs  $F0, F1$  and  $F2$ .

F2	F1	F0	$Fout = F0 \wedge F1 \wedge F2$	Assigned to
0	0	0	0	$Value[0]$
0	0	1	1	$Value[1]$
0	1	0	1	$Value[2]$
0	1	1	0	$Value[3]$
1	0	0	1	$Value[4]$
1	0	1	0	$Value[5]$
1	1	0	0	$Value[6]$
1	1	1	1	$Value[7]$

Table 9-2: Truth-table of the 3-input XOR gate for its implementation in a look-up-table

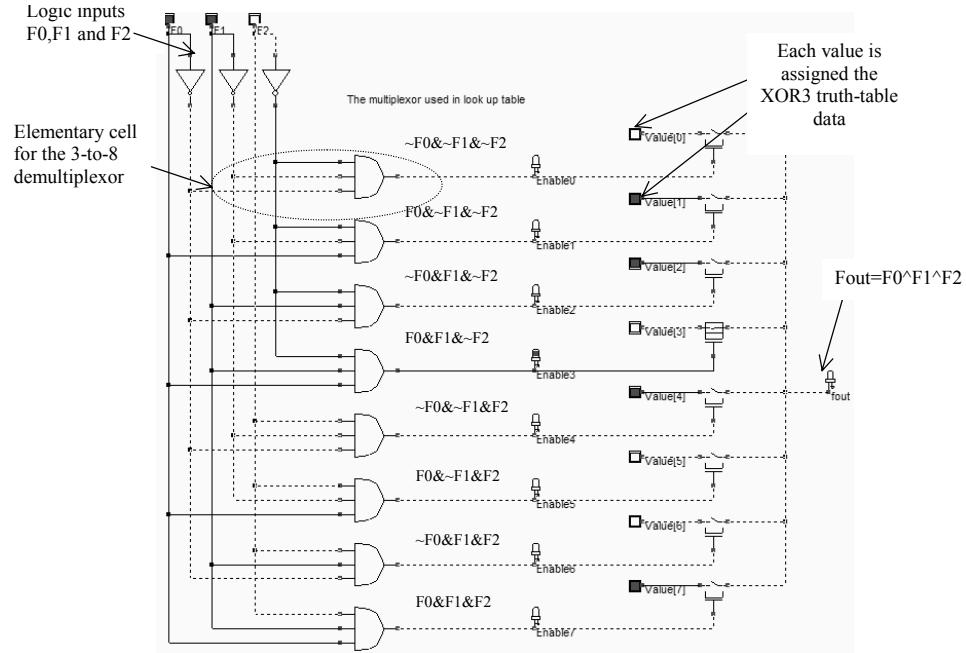


Figure 9-7: The output  $f$  produces a logical function  $F_{out}$  according to a look-up-table stored in memory points  $Value[i]$  (*FpgaLutStructure.SCH*)

### Memory Points

Memory points are essential components of the configurable logic blocks. The memory point is used to store one logical value, corresponding to the logic truth table. For a 3-input function ( $F_0, F_1, F_2$  in the previous LUT), we need an array of 8 memory points to store the information  $Value[0]..Value[7]$ . There exist here also several approaches to store one single bit of information. The one that is illustrated in figure 9-8 consists of D-reg cells. Each register stores one logical information  $Value[i]$ . The Dreg cells are chained in order to limit the control signals to one clock  $ClockProg$  and one data signal  $DataProg$ . The logical data  $Value[i]$  is fully programmed by a word of 8 bits sent in series to the signal  $DataProg$

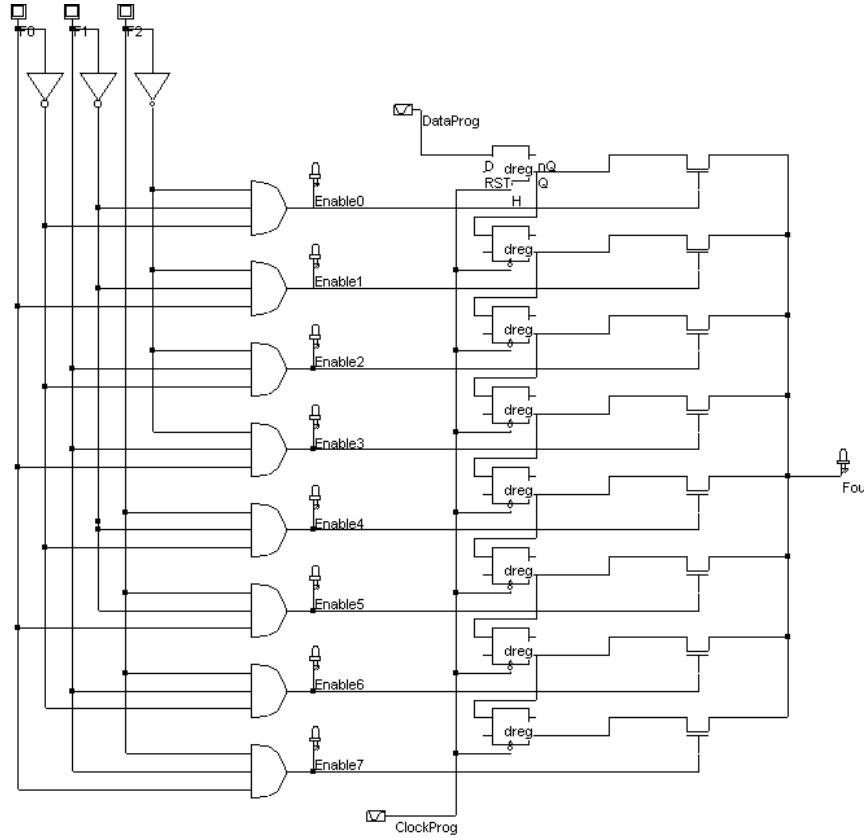


Figure 9-8: The look-up information is given by a shift register based on D-reg cells (FpgaLutDreg.sch).

The configuration of the 3-input LUT into a 3-input XOR gate obeys to a strict protocol described in figure 9-9. A series of 8 active edges is generated by the *ClockProg* signal (*Dreg* is active on fall edges). This is done by configuring a pulse generator with series of 0-1 as shown below.

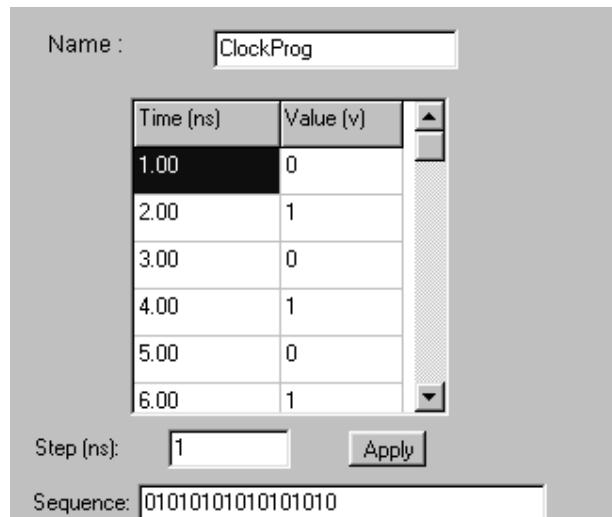


Figure 9-9: Programming the *ClockProg* pulse to generate 8 active edges (FpgaLutDreg.sch)

At each active edge, the shift register is fed by a new value presented sequentially at input *DataProg*. As the D-reg is active on fall edge, data may be changed on each rise edge. Notice that the last *Dreg* corresponds to *Value[7]*. Therefore, *Value[7]* must be inserted first, and *Value[0]* last. This means that the *DataProg* pulse must describe the truth table in reverse order, as shown below.

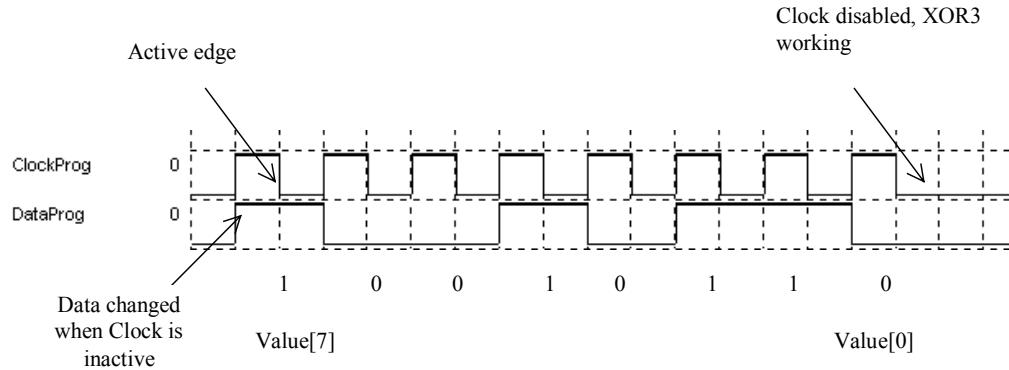


Figure 9-10: At the end of the 8-th clock period, the LUT is configured as a 3-input XOR (*FpgaLutDreg.sch*)

Most FPGA designs use *Dreg* to store the LUT configuration. Notice that the configuration is lost when the power supply is down.

### Fuse and Antifuse

To retain the configuration even without power supply, non volatile memories must be used. A one-time programmable non-volatile memory is the fuse [Sharma][Uyemura]. Usually, a contact between metal layers is used as a fuse, as an over-current would blow its structure, as illustrated in figure 9-11. Although this technique induces severe damages close to the contact, no specific technological layer is required as it is a CMOS compatible approach.

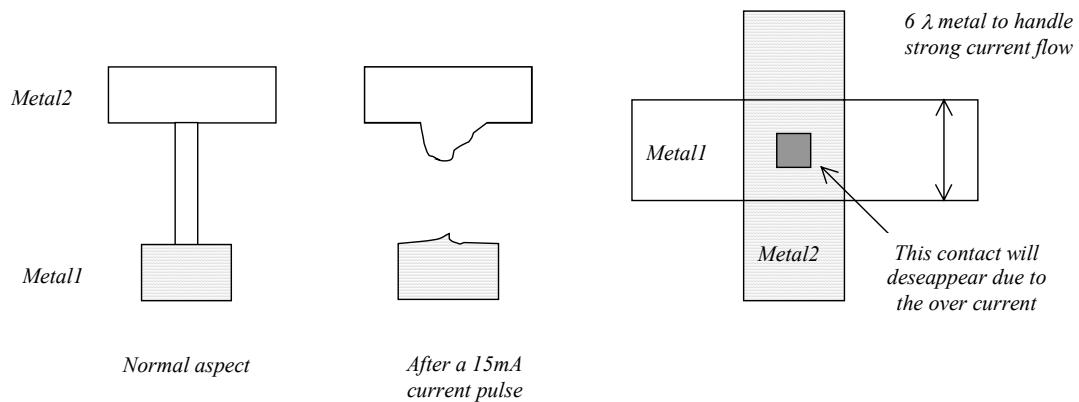


Figure 9-11 Contact fuse

A driver with large channel width (Several  $\mu\text{m}$ ), supplied by the highest available voltage (VDDH) performs the drive of very strong current pulse. The schematic diagram of the fuse circuit is shown in figure 9-12. When the command *BlowFuse* is active, both nMOS and PMOS devices are on, leading to a short circuit current. This current must be higher than 15mA to destroy the contact.

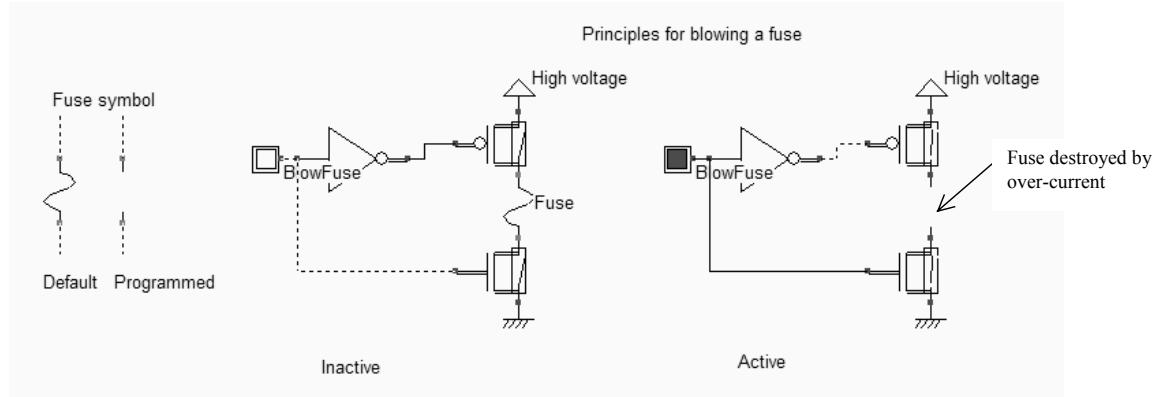


Figure 9-12: Fuse circuit programming (FuseCircuits.SCH)

In contrast to the fuse, the normal state of the antifuse is to be opened. In the example shown in figure 9-13, a thin insulator interrupts the contact between metal1 and metal2. A very high voltage applied between metal1 and metal2 (Typically 10V) breaks the oxide and provokes a conductive path between the metal layers. The use of very high voltage on the chip requires a careful use of high-voltage MOS, and of specific I/O pads, to ensure that no part of the circuit is damaged.

Another popular structure, called ONO (Oxide, Nitride, Oxide) leads to a resistive path when programmed. The typical value of the resistance is 500 ohm. Statistically, the spread of the resistance is much larger for the SiO<sub>2</sub> than for the ONO fuse [Smith], which makes the ONO fuse more attractive, at the price of supplementary process steps.

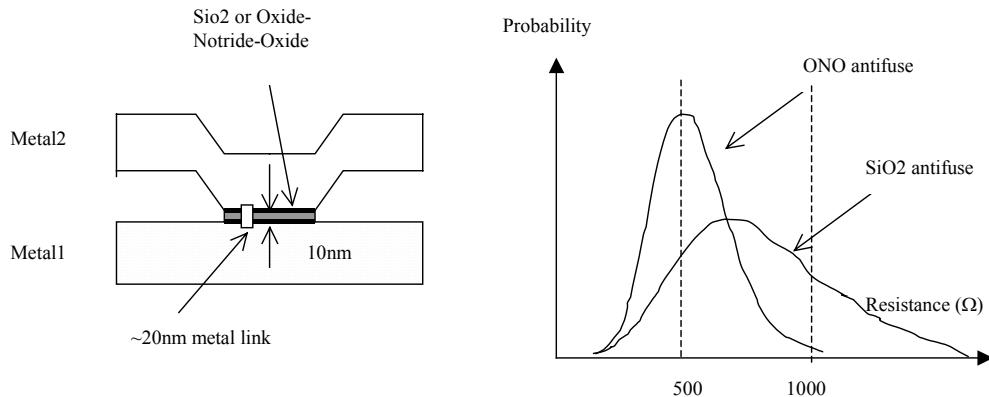


Figure 9-13: the antifuse principles and the comparative resistance spread for ONO and SiO<sub>2</sub>

Other types of non-volatile memories are being used for hardware programming of FPGA array: EEPROM and FRAM memories. These memories are not altered when the power supply is down, and can be re-programmed a large number of times. These types of memory cells are detailed in chapter 10.

### Implementation in DSCH

In DSCH, a look-up-table symbol is proposed in the symbol menu (Figure 9-14). It is equivalent to the schematic diagram of figure 9-8. An important property of the LUT symbol is its ability to retain the internal programming as a non-volatile memory would do. The user's interface of the LUT symbol is given in figure 9-14. There are three ways of filling the look-up-table: one consists in defining each array element with a 0 or a 1. The number corresponds to the logic combination of inputs F2,F1,F0. For example n°4 is coded 100 in binary, corresponding to F2=1,F1=0 and F0=0. A second solution consists in choosing the function description in the list. The logic information  $F_{out}$  assigned to each combination of the inputs updates the look-up-table. A third solution is also proposed: enter a description based on inputs F0,F1 and F2, and the logic operators " $\sim$ " (Not), " $\&$ " (And), " $\mid$ " (Or) and " $\wedge$ " (Xor). Then click the button **Fill LUT** to transfer the result of the expression to the table.

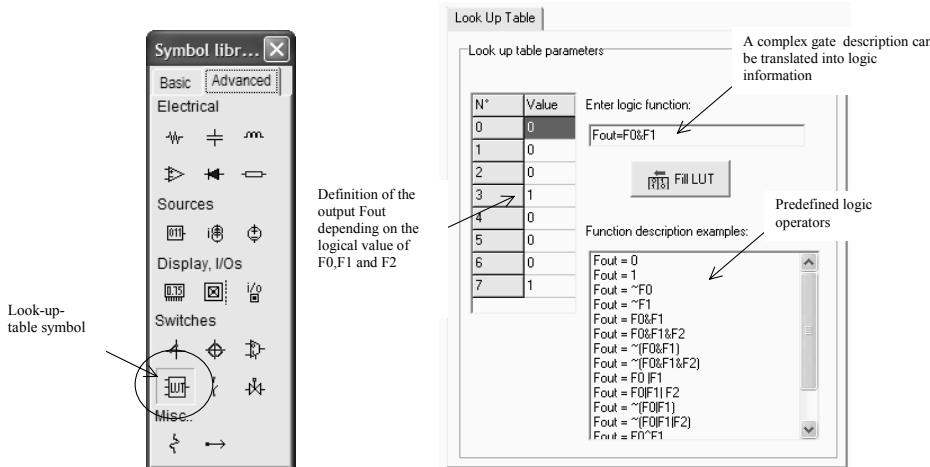


Figure 9-14: The look-up-table symbol

### 9.3 Programmable Logic Block

The programmable logic block consists of a look-up table, a D-register and some multiplexors. There exist numerous possible structures for logic blocks. We present in figure 9-15 a simple structure which has some similarities with the Xilinx XC5200 series (See [Smith] for detailed information on its internal structure). The configurable block contains two active structures, the Lut and the D-reg, that may work independently or be mixed together.

The output of the look-up-table is directly connected to the block output *Fout*. The output can also serve as the input data for the D-register, thanks to the multiplexor controlled by *DataIn\_Fout*. The *DataOut* net can simply pass the signal *DataIn*, in that case the cell is transparent. The *DataOut* signal can also pass the signal *nQ*, depending on the multiplexor status controlled by *DataIn\_nQ*

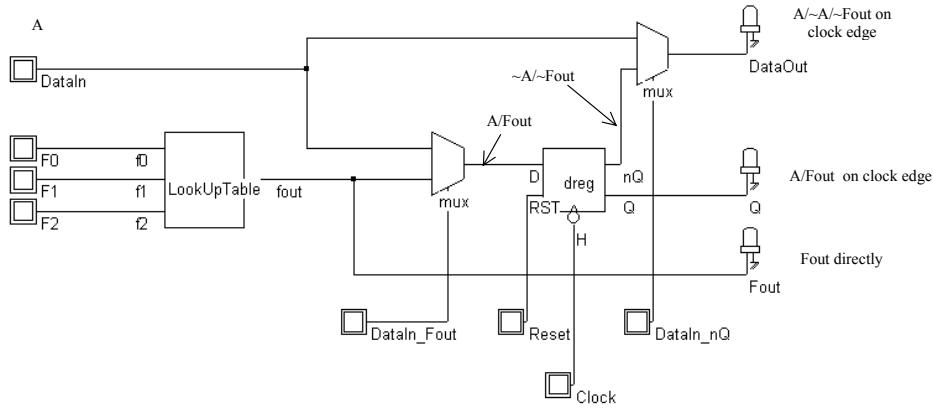


Figure 9-15: Simple configurable logic block including the Look Up Table and a D-register (FpgaCell.SCH)

The block now consists of the LUT and the D-register. We chain the information *DataIn\_Fout* and *DataIn\_nQ* on the path of the shift register by adding 2 supplementary *Dreg* cells. Each *Dreg* still uses the same clock *ClockProg* and chained input data *DataProg*. The complete circuit is shown in figure 9-16.

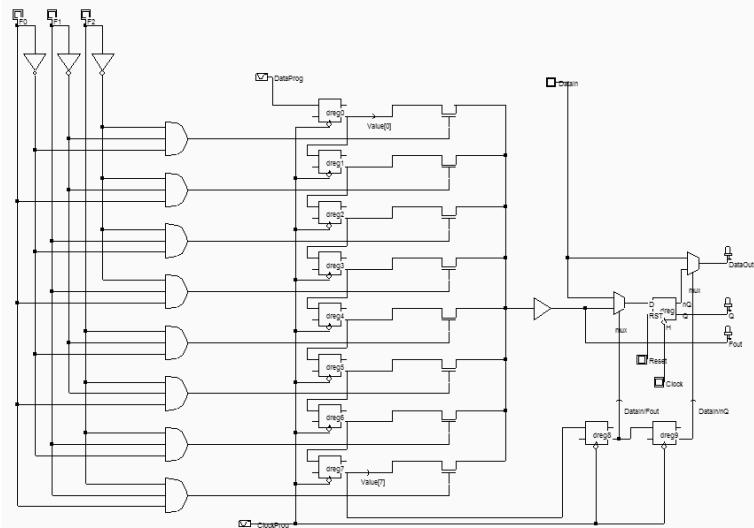
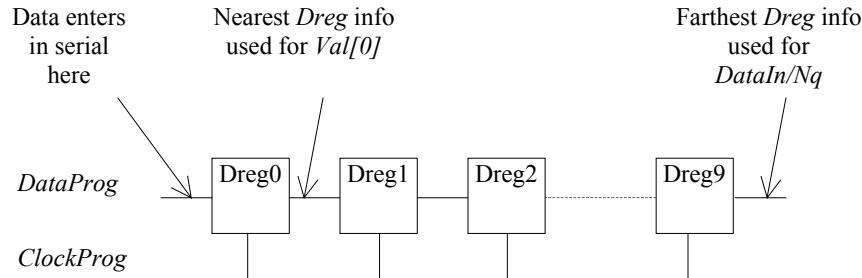


Figure 9-16: The Look Up Table, the D-register and the shift register including the 2 multiplexor cells (FpgaBlockStructure.SCH)

The configuring of the block is achieved thanks to 10 active clock edges on *ClockProg*, and 10 serial data on *DataProg*. The chain of *Dreg* starts at *Dreg0* (Upper *Dreg* in figure 9-16, which produced *Value[0]*) and stops at *Dreg9* (Right side of figure 9-16 which produced *DataIn/nQ*). The information that flows at the far end of the register chain is defined at the first cycle, while the closest register is configured by the data present at the last active clock edge.,



Clock cycle	1	2	3	4	5	6	7	8	9	10
DataProg	DataIn/ Nq	Datain/ Fout	Val [7]	Val [6]	Val [5]	Val [4]	Val [3]	Val [2]	Val [1]	Val [0]

Table 9-3: Serial data information used to program the LUT memory points

## 9.4 Interconnection between blocks

The interconnection strategy between logic blocks is detailed in this paragraph. We shall focus on the programmable interconnect point and the programmable switching matrix. Then, we will discuss the global implementation of the structure.

### Programmable Interconnect Point

The elementary programmable interconnect point (PIP <Gloss>) may be found in the **Advanced** set of **Switches** symbols (Figure 9-17). It consists of a configurable bridge between two interconnects.

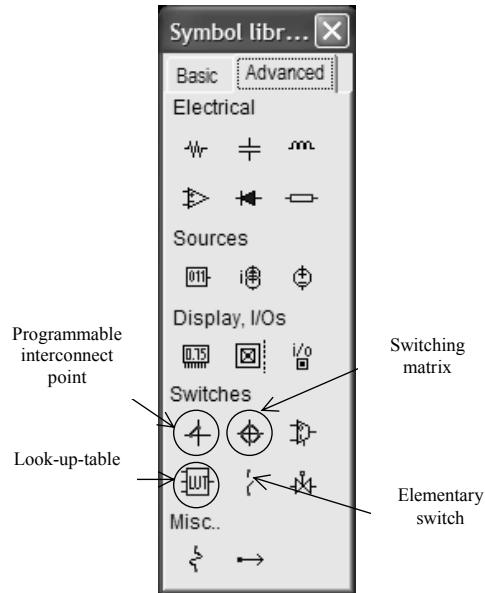


Figure 9-17: The programmable interconnect point (PIP) in the palette of symbols

The PIP may have two states: "On" and "Off". You may switch from "On" to "Off" by a double click on the symbol (Screen shown in figure 9-18) and a click on the button **On/off**.

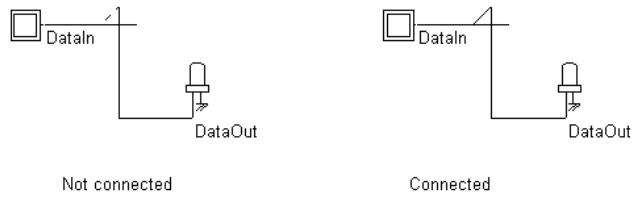
Programmable Interconnect Point (PIP)



4

Off                  On

Programmable Interconnect Point (PIP)



Click here  
to change  
the state

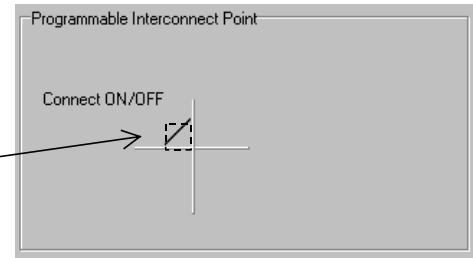


Figure 9-18: Changing the state of the PIP (FpgaPip.SCH)

The bridge can be built from a transmission gate, controlled once again by a *D-reg* cell (Figure 9-19).

When the register information contains a 0, the transmission gate is off and no link exists between *Interco1* and *Interco2*. When the information held by the register is 1, the transmission gate establishes a resistive link between *Interco1* and *Interco2*. The resistance value is around  $100 \Omega$ .

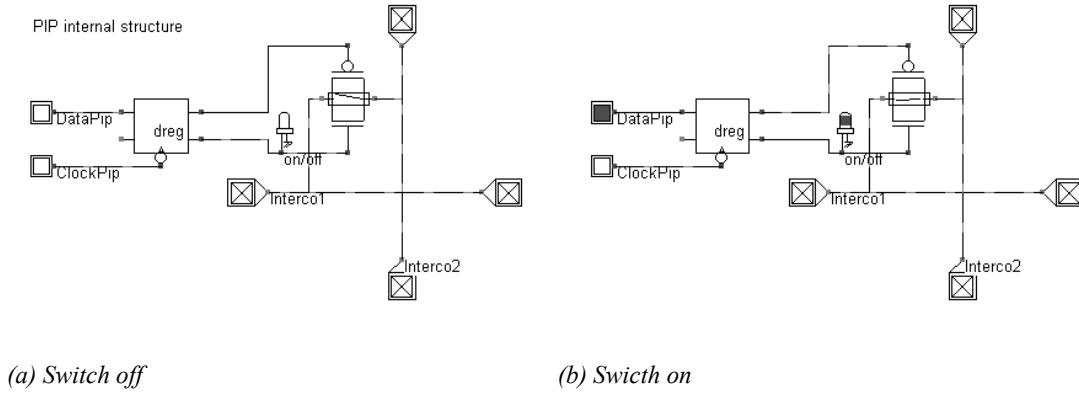


Figure 9-19: Internal structure of the PIP and illustration of its behavior when Off (a) and On (b)  
(FpgaPip.SCH)

The regrouping of programmable interconnect points into matrix is of key importance to ensure the largest routing flexibility. Examples of  $3 \times 3$  and  $3 \times 2$  PIP matrix are shown in figure 9-20. The link between *In1* and *Out1*, *In2* and *Out2*, *In3* and *Out3* is achieved by turning some PIP on. A specific routing tool usually handles this task, but the manual re-arrangement is not rare in some complex situations. In DSCH, just press the key "O" to switch the PIP On and off.

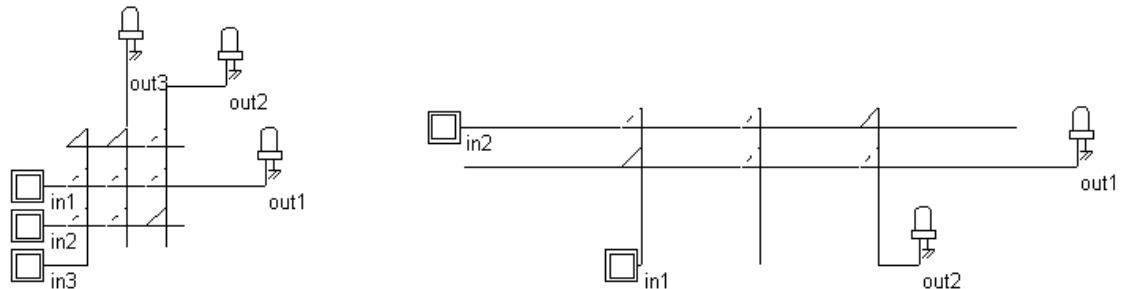


Figure 9-20: Matrix of programmable interconnect points (FpgaPip.SCH)

## Switching Matrix

The switching matrix is a sophisticated programmable interconnect point, which enables a wide range of routing combinations within a single interconnect crossing. The aspect of the switching matrix is given in figure 9-21. The matrix includes 6 configurable bridges between the two main interconnects.

The switching matrix symbol may be found in **Advanced** set of **Switches** symbols. By a double click on the matrix symbol, you get access to the 6 **On/Off** switches.

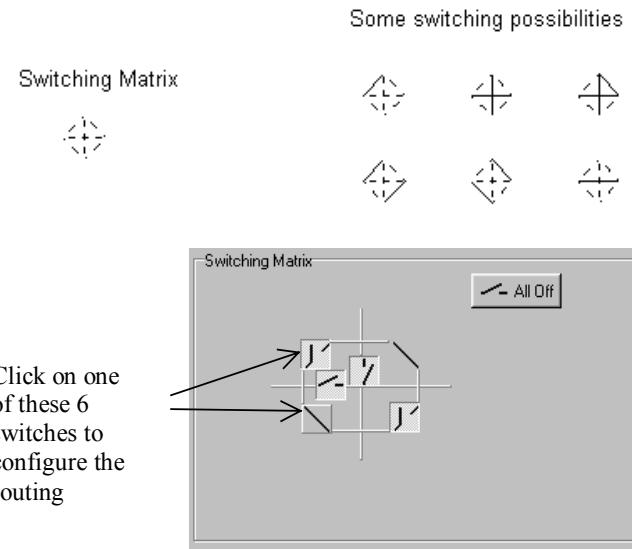


Figure 9-21: Changing the state of the matrix (FpgaMatrix.SCH)

To ease the programming of the matrix, short cuts exist in DSCH. You can change the state of the matrix by placing the cursor on the desired symbol and pressing the following keys:

- To switch off the matrix, press the key "o".
- To switch on the matrix, press the key "O".
- To enable an horizontal link, press the key "-".
- To enable a vertical link, press the key "|".

Examples of 3x2 and 3x3 switching matrix are given in figure 9-22. The routing possibilities are numerous, which improves the configurability of the logic blocs.

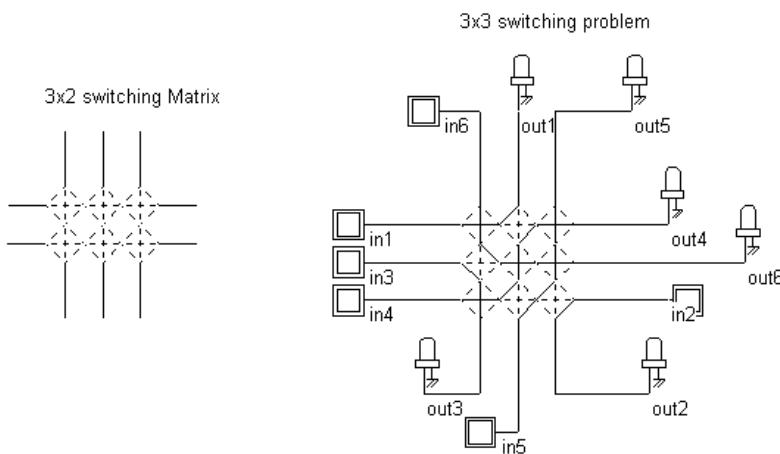


Figure 9-22: 3x2 switching matrix and example of routing strategy between 6 inputs and outputs (fpgaMatrix.SCH)

### Implementation of the Switching Matrix

From a practical point of view, the switching matrix can be built from a regrouping of 6 transmission gates (Figure 9-23). Each transmission gate is controlled by an associated *Dreg* cell, which memorizes the desired configuration. The *Dreg* cells are chained so that one single input *DataIn* and one clock *LoadClock* are enough to configure the matrix.

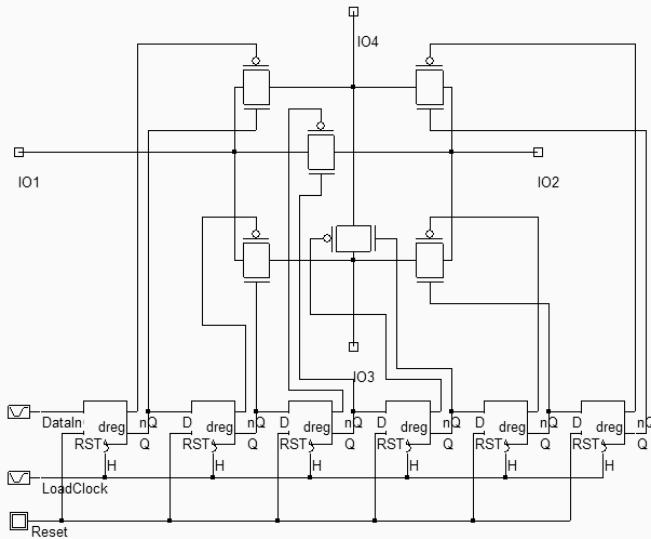


Figure 9-23: The transmission gates placed on the routing lines to build the matrix (FpgaMatrix3.SCH)

### Array of Blocs

The configurable blocs are associated with programmable interconnect points and switching matrix to create a complete configurable core. An example of double configurable block and its associated configurable routing is proposed in figure 9-24.

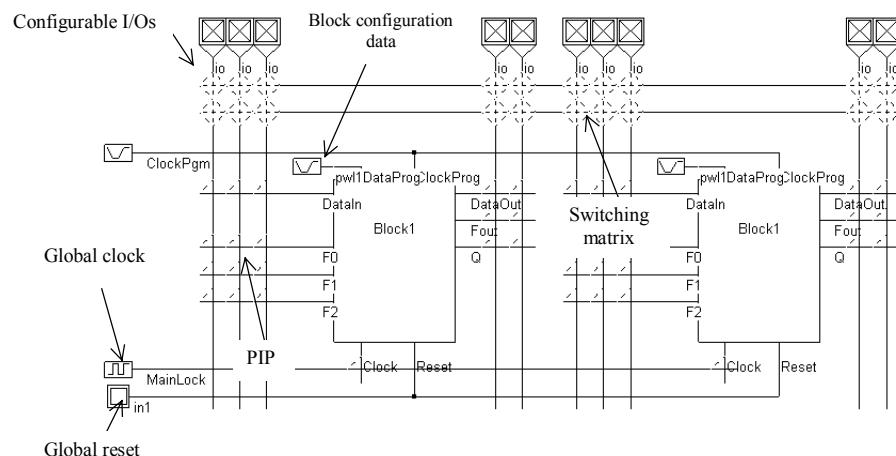


Figure 9-24: Configurable blocks, switching matrix, configurable I/Os and arrays of PIP (fpga2blocks.SCH)

## Full-Adder Example

The truth table and logical expression for the full-adder are recalled in Table 9-3. The implementation of the CARRY and SUM function is achieved by programming two look-up tables according to the truth-tables reported in table 9-3.

FULL ADDER					
A	B	C	SUM	CARRY	RESULT
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	1	0	1
0	1	1	0	1	2
1	0	0	1	0	1
1	0	1	0	1	2
1	1	0	0	1	2
1	1	1	1	1	3

Table 9-3. The full-adder truth-table

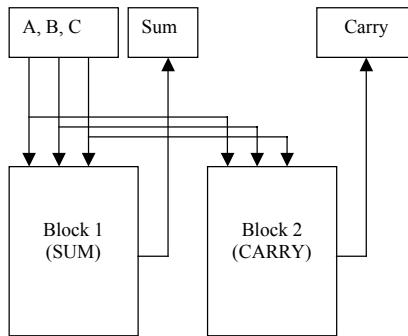


Figure 9-25: The SUM and CARRY functions to realize the full-adder in FPGA (fpgaFullAdder.SCH)

The general diagram of the Full adder implementation is given in figure 9-25. One programmable logic block *Block1* supports the generation of the sum for given logic values of the inputs *A,B* and *C*. The information needed to configure *Block1* as a *Sum* function (3-input XOR) is given in table 9-4. Notice that we only use the LUT in this programmable logic block. The *Dreg* is not active, and we only exploit the output of the LUT *Fout*, which is configured as the *Sum*.

The signal *Sum* propagates outside the block to the output interface region by exploiting the interconnect resources and switching matrix. The other programmable logic block *Block2* supports the generation of the signal *Carry*, from the same inputs *A,B* and *C*. The programming of *Block2* is also given in table 9-4. The result *Carry* is exported to the output interface region as for the *Sum* signal. Again, in this block, only the LUT is active.

Block 1 (Sum of F0, F1 and F2)									
Cycle 1	2	3	4	5	6	7	8	9	10
DataIn	Datain	Val[7]	Val[6]	Val[5]	Val[4]	Val[3]	Val[2]	Val[1]	Val[0]
Nq	Fout	0	1	0	0	1	0	1	0

Block 2 (Carry of F0, F1 and F2)									
Cycle 1	2	3	4	5	6	7	8	9	10
DataIn	Datain	Val[7]	Val[6]	Val[5]	Val[4]	Val[3]	Val[2]	Val[1]	Val[0]
Nq	Fout	0	1	1	1	0	1	1	0

Table 9-4. Serial data used to configure the logic blocks 1&2 as SUM and CARRY

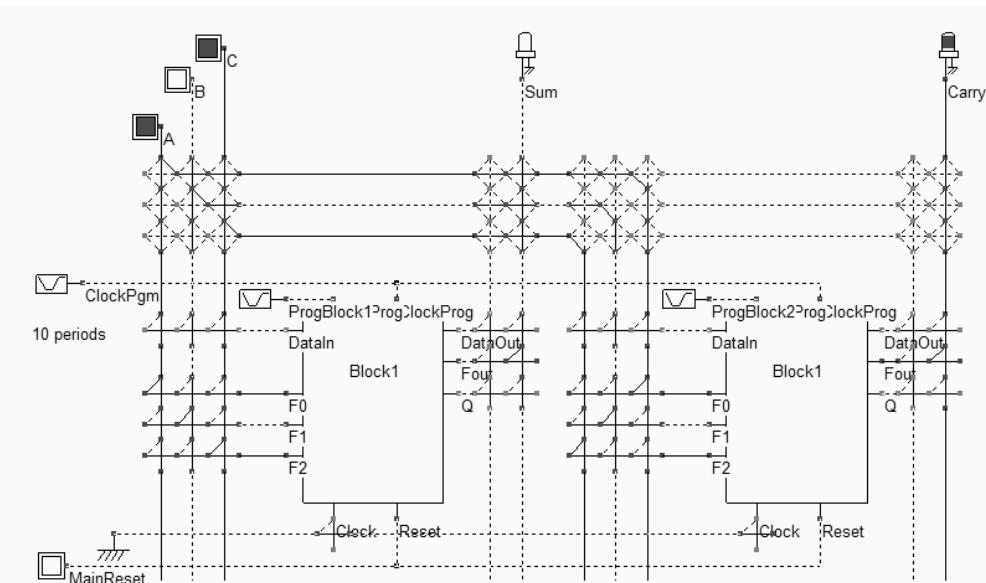


Figure 9-26: Simulation of the full-adder implemented in 2 configurable blocks (fpgaFullAdder.SCH)

The programming sequence is contained in the piece-wise-linear symbols *ProgBlock1* and *ProgBlock2*. As seen in the chronograms of figure 9-27, the program clock *ClockPgm* is only active at the initialization phase, to shift the logic information to the memory points inside the blocks which configure each multiplexor. The routing of the signals *A*, *B* and *C* as well as *Sum* and *Carry* has been done manually in the circuit shown in figure 9-26. In reality, specific placement/routing tools are provided to generate the electrical structure automatically from the initial schematic diagram, which avoids manual errors and limits conflicts or omissions.

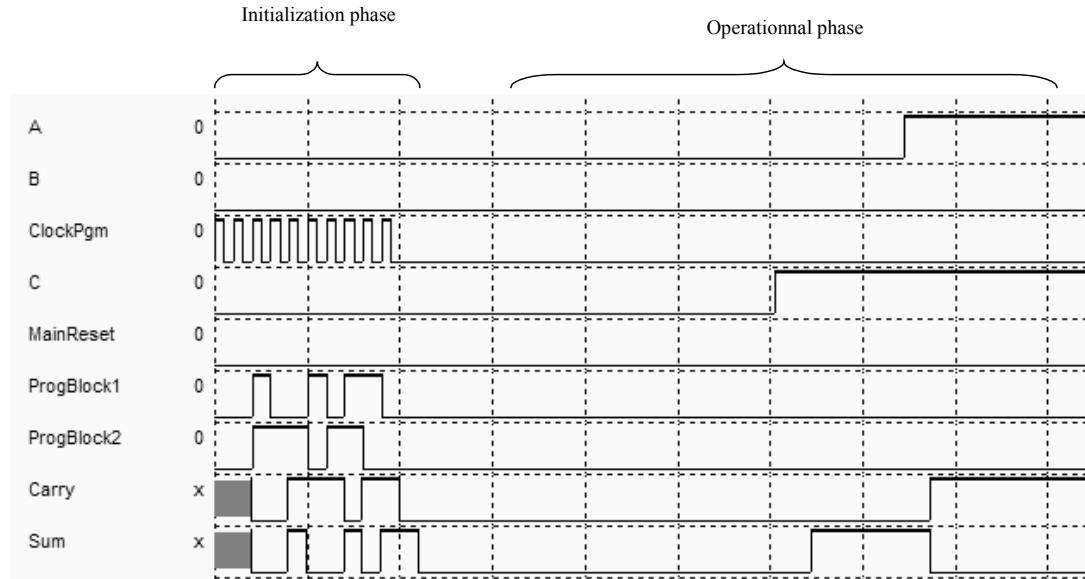


Figure 9-27: Chronograms of the full-adder FPGA (*fpgaFullAdder.SCH*)

### Clock Divider Example

A second example is proposed as an application of the FPGA circuits. It concerns the clock division. We recall in figure 9-28 the general structure and the typical chronograms of the clock division by four, which requires two *Dreg* cells, with a feedback from the output  $\sim Q$  to the input  $D$ .

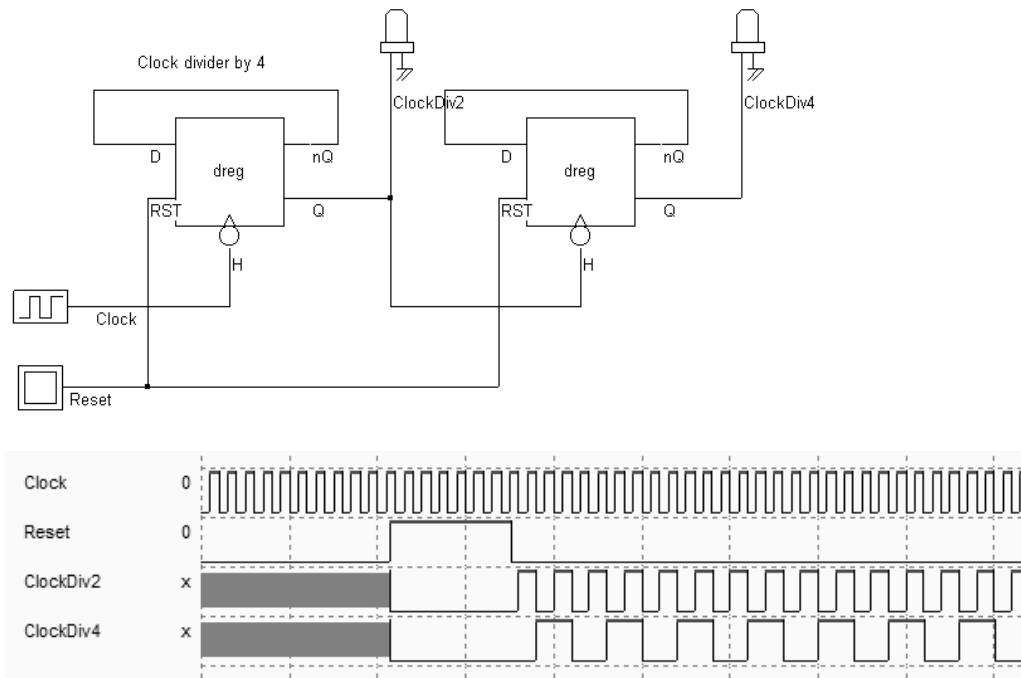


Figure 9-28: Diagram and typical simulation of the clock divider by 4 (*ClockDiv4.SCH*)

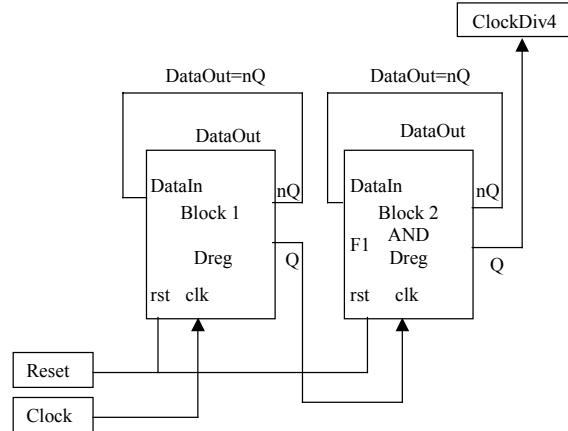


Figure 9-29: Implementation of the clock divider in 2 configurable blocs (FpgaDiv4.SCH)

The general diagram of the clock divider implementation is given in figure 9-29. Each programmable logic block is configured as a single stage clock divider. The information needed to configure *Block1* as a simple *Dreg* function is given in table 9-5. This serial data information creates a direct path from *DataIn* to input *D* of the *Dreg* cell, while *nQ* propagates to *DataOut*, as detailed in figure 9-30.

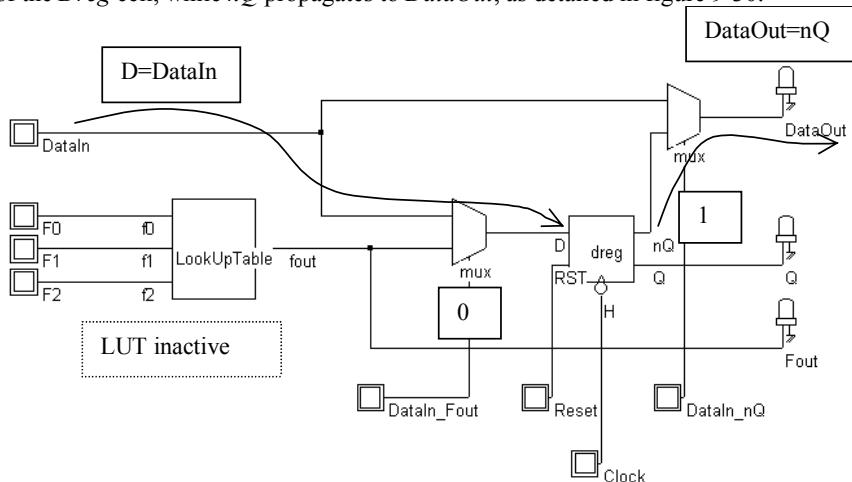


Figure 9-30: Use of the configurable block as a DReg (FpgaDiv4.SCH)

Block 1 (DataOut=nQ, D=DataIn)										
Cycle 1	2	3	4	5	6	7	8	9	10	
DataIn	Datain	Val[7]	Val[6]	Val[5]	Val[4]	Val[3]	Val[2]	Val[1]	Val[0]	Nq
1	0	0	0	0	0	0	0	0	0	

Block 2 (DataOut=nQ, D=DataIn)										
Cycle 1	2	3	4	5	6	7	8	9	10	
DataIn	Datain	Val[7]	Val[6]	Val[5]	Val[4]	Val[3]	Val[2]	Val[1]	Val[0]	Nq
1	0	0	0	0	0	0	0	0	0	

Table 9-5 Serial data used to configure the logic blocks 1&2 as clock dividers (FpgaDiv4.SCH)

Outside the programmable block, the signal  $nQ$  propagates to the input  $DataIn$ . Notice that the look-up table is inactive in this configuration. The other programmable logic block *Block2* is also programmed as a *Dreg* circuit with a feedback from  $nQ$  to  $DataIn$ .

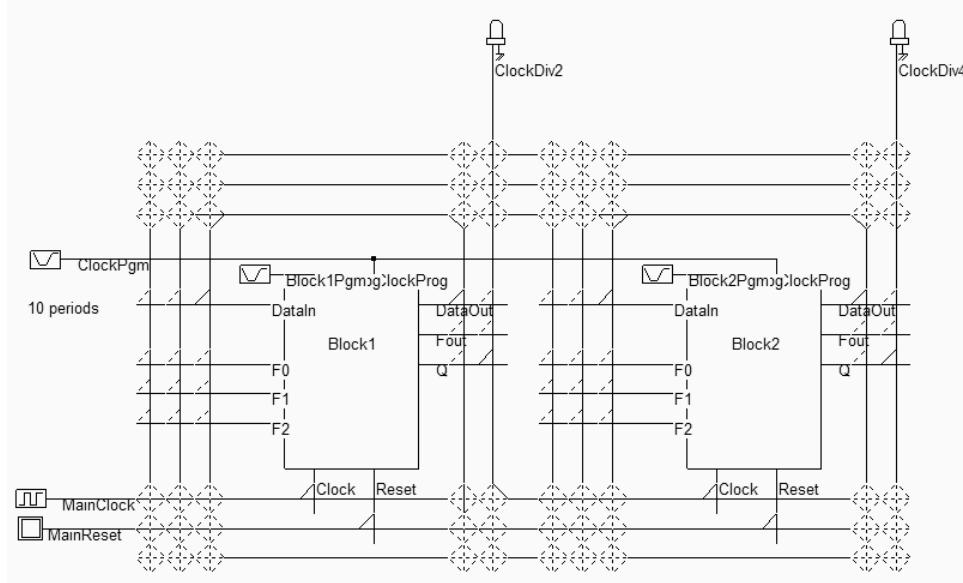


Figure 9-31: Routing of the clock divider in 2 configurable blocs (FpgaDiv4.SCH)

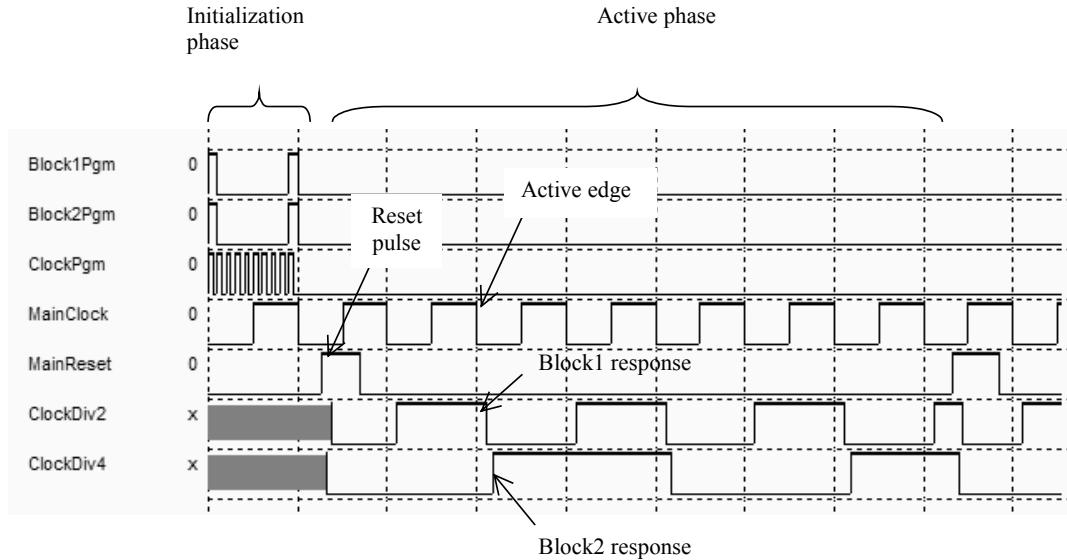


Figure 9-32: The chronograms of the clock divider circuit (ClockDiv4.SCH)

The simulation of the counter is proposed in figure 9-32. The first nanoseconds are dedicated to the programming of the blocks. Once properly configured, the counter starts to work according to the specifications of figure 9-28. Notice the very important delay in responding to the active edges, which is due to the intrinsic complexity of the configuration block, and to the long interconnect delay through the connection points and switching matrix.

## 9.5 Conclusion

In this chapter, we have given a brief introduction to field programmable gate arrays, from the point of view of cell design. Firstly, the use of multiplexor and look-up-tables for building configurable logic circuits has been illustrated. Secondly, the programming of memory points using chained D-registers and fuse has been described. Thirdly, we have described the programmable interconnect points and switching matrix, with their implementation in DSCH. Finally, the implementation of a full adder and a clock divider have been performed using two configurable logic blocks, programmable interconnect points and switching matrix.

## References

- [Smith] Michael. J. S. Smith "Application Specific Integrated Circuits", Addison Wesley, ISBN 0-201-50022-
- [Sharma] <add>
- [Uyemura] <add>

## EXERCISES

- 9.1 Using DSCH (file exo9\_1.sch), configure the 16 switching matrix in order to connect: switch1 to Lights L3 and L5, switch2 to Lights L1 and L6, switch3 to Lights L2 and L4, switch4 to Lights L7 and L8.

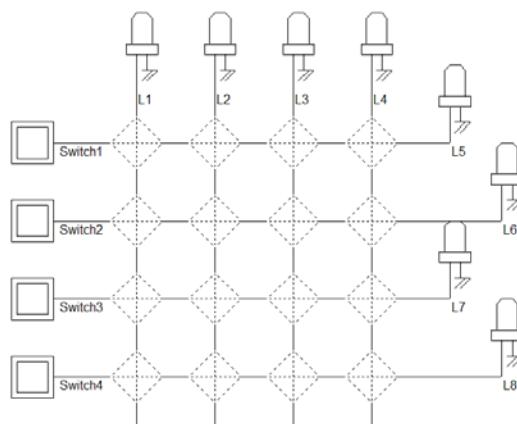


Figure 9-33: Routing exercise

Answer: see file ch91.sch

9.2 Store the 8 following bits (01110111) (reading from left to right) in the look-up-table, as in figure 9-4. How many active edges on *ClockProg* do you need to configure the LUT ? Which logical function have you realized ?

*Answer:* a) 8      b)  $F_{out} = F_2 + \overline{F_1 \cdot F_0}$

9.3 Store the 8 bits (00000111) (reading from left to right) in the LUT of figure (9-8). Demonstrate that you have realized the following logical function:  $\overline{F_2 + (F_1 \cdot F_0)}$ . Using 2 LUT and one inverter, create the *D\_Latch* below. Give the serial data sequence for *DataProg*.

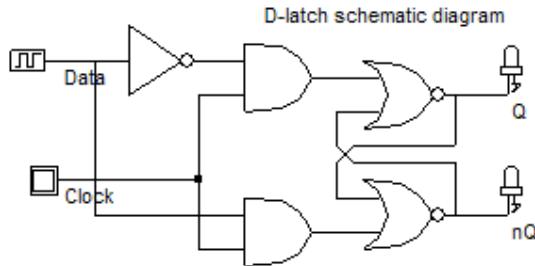


Figure 9-34: Implementing a *D\_latch* in PFGA

*Answer:* a)  $F_0 \cdot \overline{F_1} \cdot F_2 + \overline{F_0} \cdot F_1 \cdot F_2 + F_0 \cdot F_1 \cdot F_2 = \overline{F_2 + (F_1 \cdot F_0)}$  b) LUT N°1:  $F_0 = \overline{\text{Data}}$ ,  $F_1 = \text{Clock}$ ,  $F_2 = nQ$ , *DataProg*=00000111, LUT N°2:  $F_0 = \text{Data}$ ,  $F_1 = \text{Clock}$ ,  $F_2 = Q$ , *DataProg*=00000111

9.4 How many programmable logic blocs do you need to create the following asynchronous counter (Figure 9-35)? Give the programmable sequences for each block.

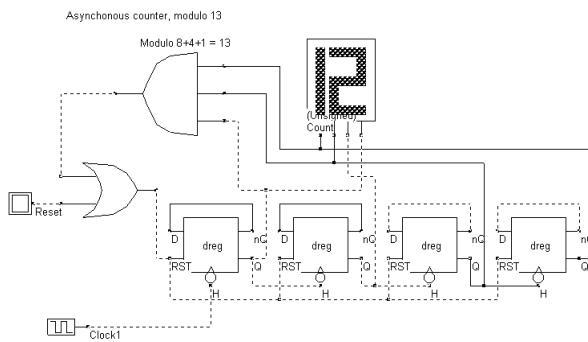


Figure 9-35: Implementing an asynchronous counter in PFGA

*Answer:* See ch94.SCH, where only 3 programmable logic blocs are used, as explained in figure 9-36.

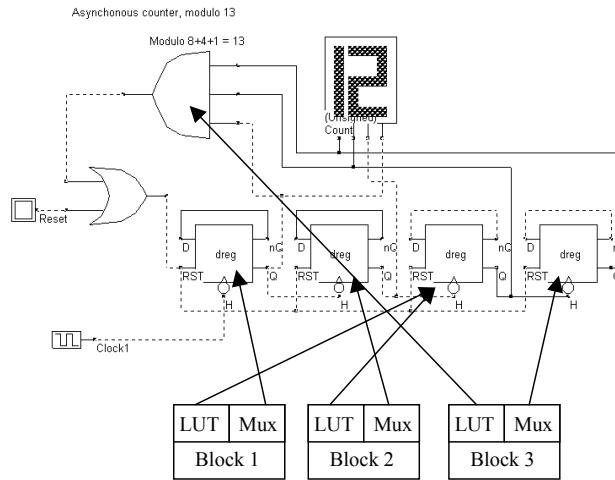


Figure 9-36: A compact solution for implementing an asynchronous counter in FPGA

9.6 Using programmable logic blocs create a one-bit comparator. The truth table is given below.

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

Table 9-8: The comparator

# 10      Memories

This chapter describes the memory cell architecture. After a general introduction, we detail the principles and implementation of static RAM, dynamic RAM, read-only memories, electrically erasable memories, and Ferromagnetic memories.

## 10.1 The world of Memories

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.

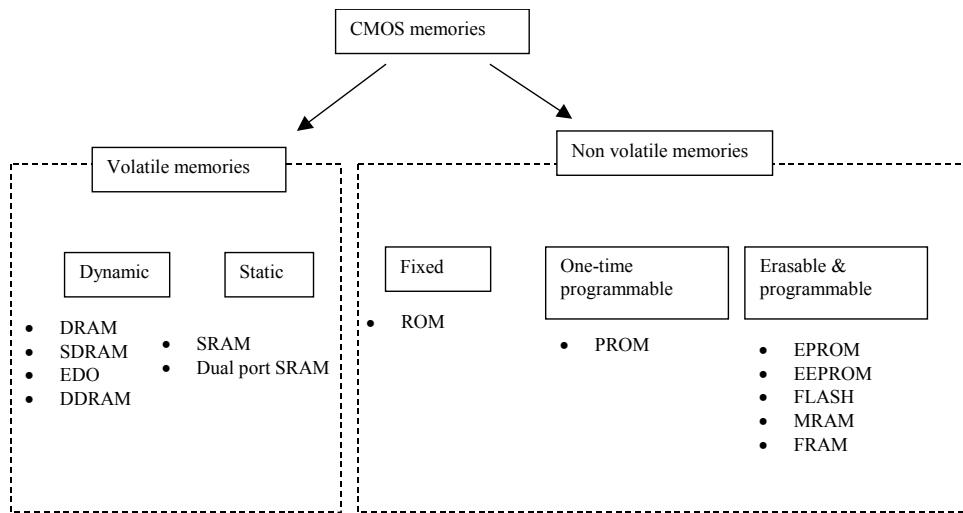


Figure 10-1 Major classes of CMOS compatible memories

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

- In volatile circuits (Figure 10-1 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 10-1 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 <Gloss>) and ferroelectric RAM (FRAM<Gloss>) memories.

Millions of elementary memories are used by microprocessors for executing software. Micro-code, operating systems and low level software are usually stored in non-volatile memory. The software execution requires fast-access random access memory such as static or dynamic RAM. Memory exist as stand-alone components (Figure 10-2-a), but also as embedded blocks in system-on-chip, such as those shown in figure 10-2-b.

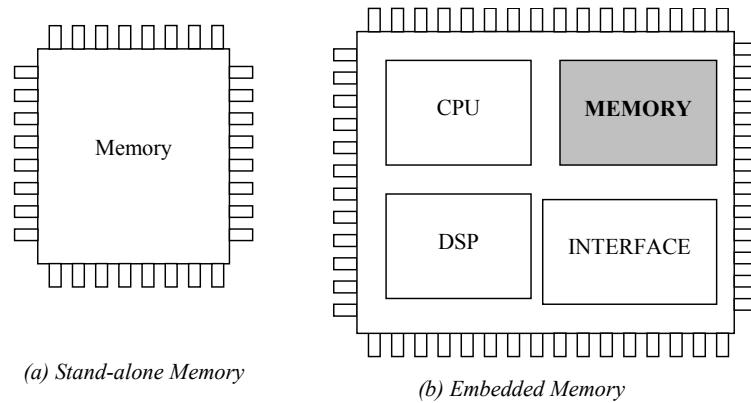


Figure 10-2: The memory exists as a stand-alone component or as embedded block

### Memory organization

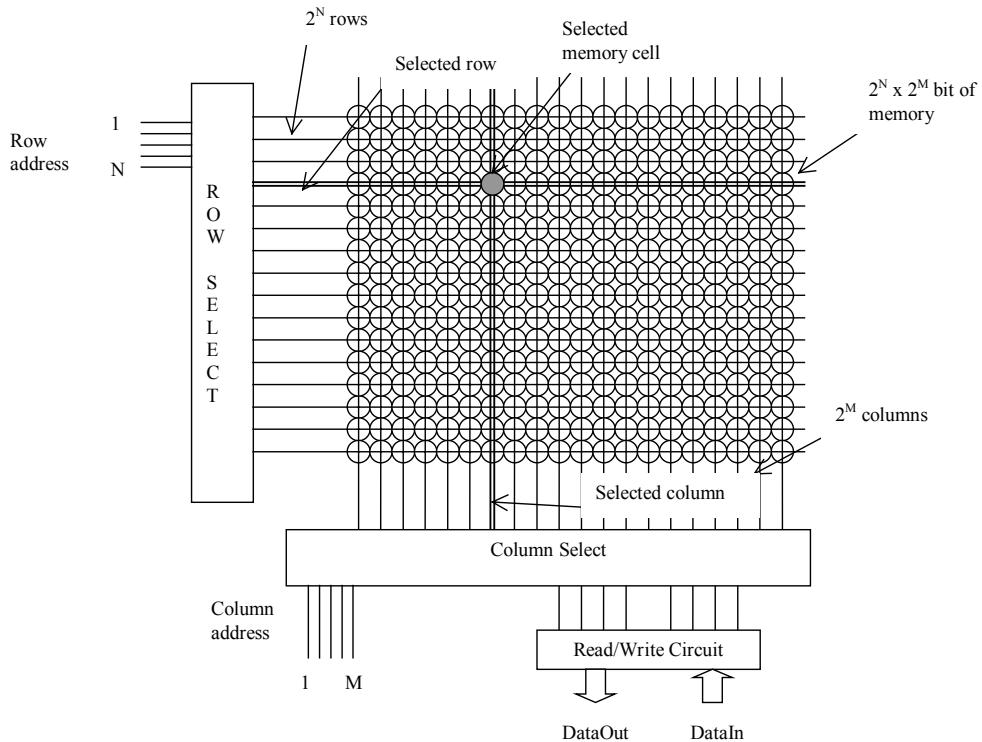


Figure 10-3 Typical memory organization

Figure 10-3 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). Several organizations exist: 1024x1024 bit, 128Kx8bit, 64Kx16bit, 32Kx32 bits, etc.. For example, the organization 128Kx8bit consists in selecting 8 columns in parallel. In that case, the size of *DataOut* and *DataIn* bus is 8 bit.

### Access Time

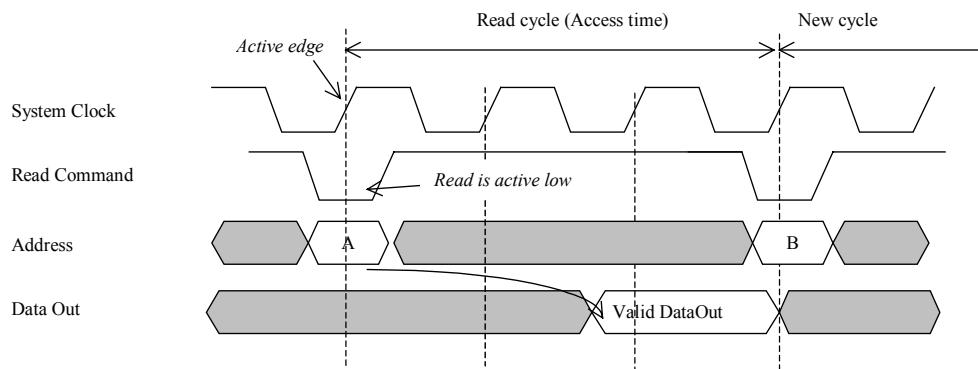


Figure 10-4: Read Access Time

The typical timing diagram of a memory block is shown in figure 10-4. A fast system clock, around 1GHz period, synchronizes the whole sequence. On an active level of the *Read* command (Usually a low level), the read cycle begins. It may take several clock cycles before the data is available. In the case of figure 10-4, two clock cycles are necessary before the valid data is proposed at the *DataOut* bus. The typical access time for Mega-bit memories ranges between 1ns and 10ns.

## 10.2 Static RAM Memory

### Introduction

The static RAM is a very important class of memory. It consists of two cross-coupled inverters, which form a positive feedback with two possible states illustrated in figure 10-5. This cell is also the base of many sequential circuits, as detailed in chapter 8.

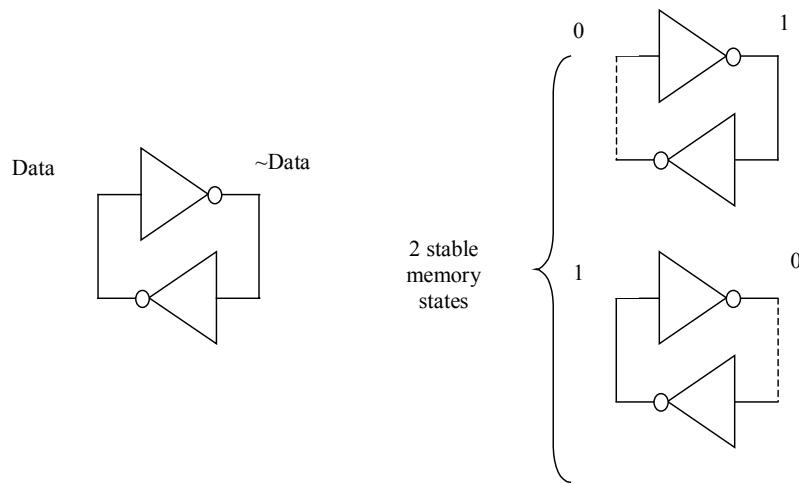


Figure 10-5: Elementary memory cell based on an inverter loop

### Trying the Five Transistor Cell

The state of the memory cell can be changed by a pass transistor. In that case, the memory cell has five transistors. The signal that controls the gate of the pass transistor is usually called the *word line*, or *WL*. The data information that flows vertically is called the *Bit Line* (*BL*). The schematic diagram of the static memory array is shown in figure 10-6.

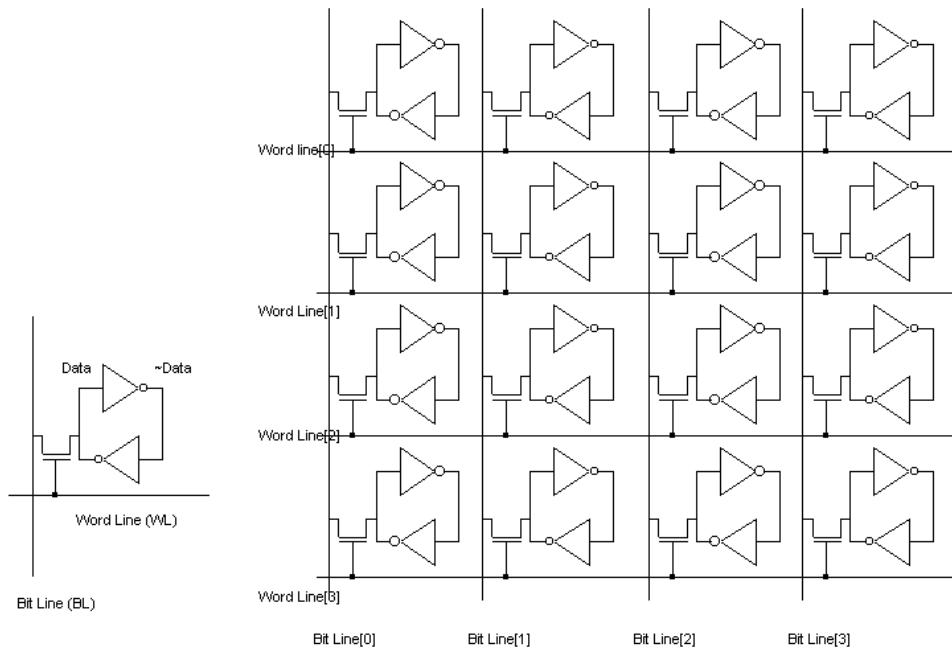


Figure 10-6: Building a memory array with five transistor static memory cell (RAMStatic5T.SCH)

The 5 transistor static RAM layout is given in Figure 10-7. Remember that the RAM cell will be copied hundreds of times in X and Y, that silicon area is money, and that up to 80% of the system-on-chip transistors may be used for

memory. Consequently, the RAM layout should be as compact as possible to produce a memory array with the smallest possible silicon area. The steps to build the 5T memory cell are presented in the next figures. We start by designing one nMOS and two compiled inverters (Figure 10-7). The first action is to flip the inverter situated on the left in order to share the supply network (Figure 10-8).

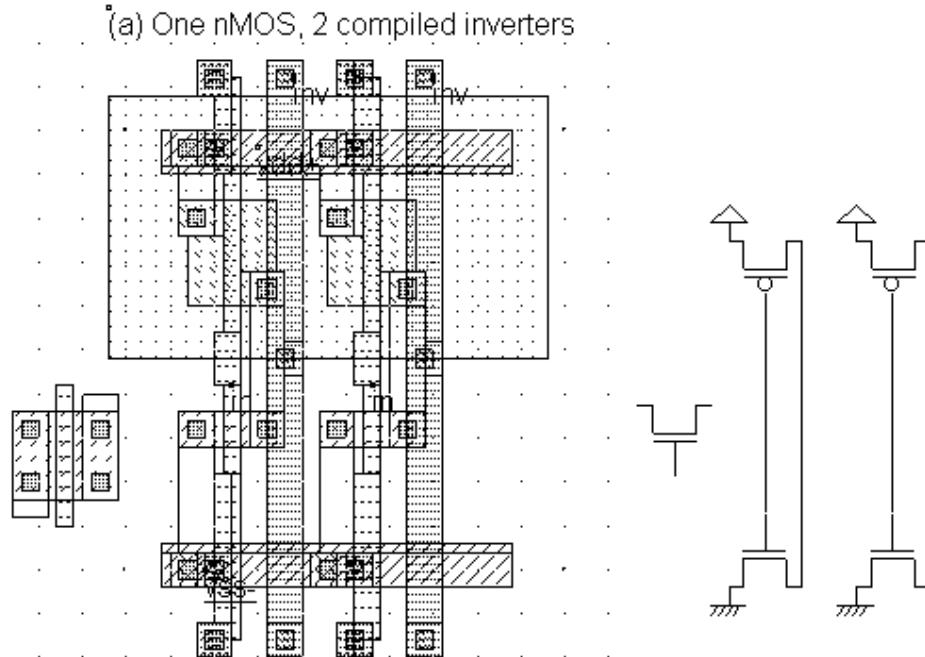


Figure 10-7: Initial steps for the design of the 5T memory cell (RAMStatic5T.MSK)

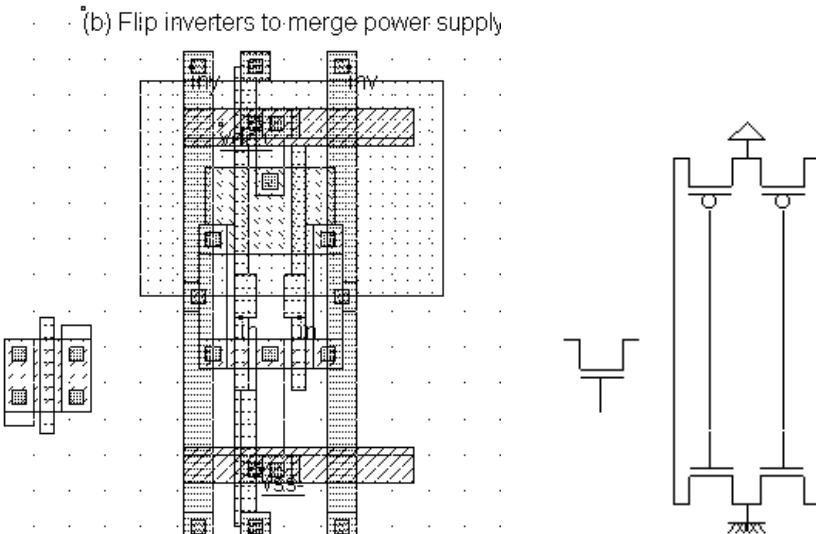


Figure 10-8: Merging the supply contacts to compact the cell design (RAMStatic5T.MSK)

(c) Cross-couple inverters and merge nMOS

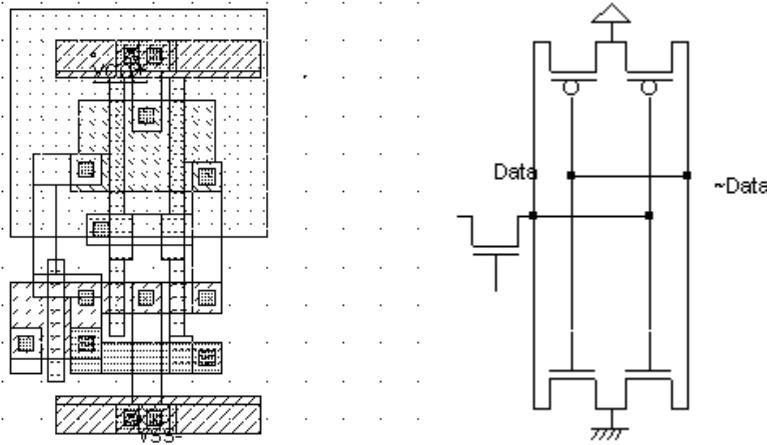


Figure 10-9: Construction of the cross-coupled interconnects (RAMStatic5T.MSK)

The next step, illustrated in figure 10-9, consists in building the cross-coupled interconnects. We need to verify that the design rules are not violated thanks to the Design Rule Checker that can be found in the **Analysis** menu. One interconnect is routed in between the nMOS and pMOS regions, while the other interconnect is routed in the lower part of the cell. Another strategy would consist in enlarging the separation between nMOS and pMOS to rout both cross-coupled interconnects in the middle of the cell.

(d) Compact power lines, add WL, BL contacts

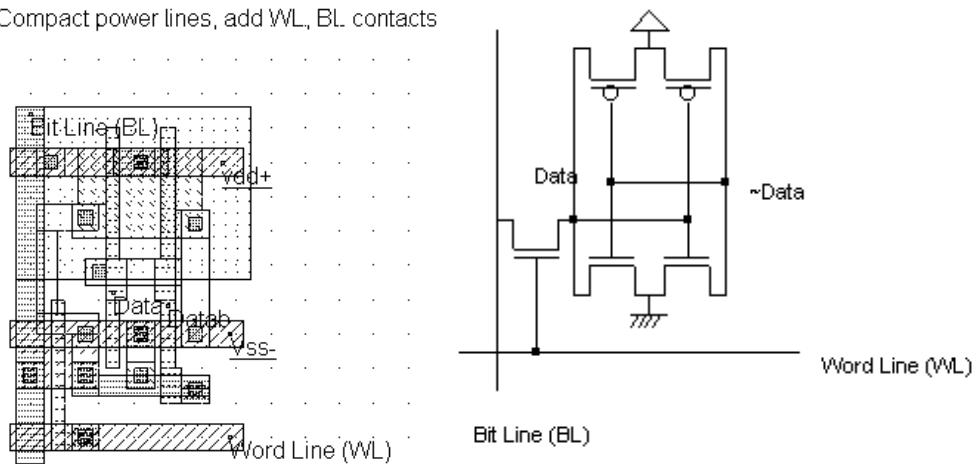


Figure 10-10: Final design of the 5T memory cell (RAMStatic5T.MSK)

The final aspect of the cell is shown in figure 10-10. The **Bit line** signal is made with metal2 and crosses the cell from top to bottom. The supply lines are horizontal, made with metal3. This allows easy matrix-style duplication of the RAM cell.

### Simulation of the Five Transistor Cell

The best way to stimulate the RAM cell is to assign a pulse on the *Bit Line*. We insert the sequence 01xx10xx, where "0" means a 0V (Ground voltage), "1" 1.2V (VDD core supply), and "x" means a floating node. When the pulse is in "x" state, the node is no more controlled by the pulse, and the memory cell data may take the lead on the vertical *Bit Line*. The word line is simply assigned a clock.

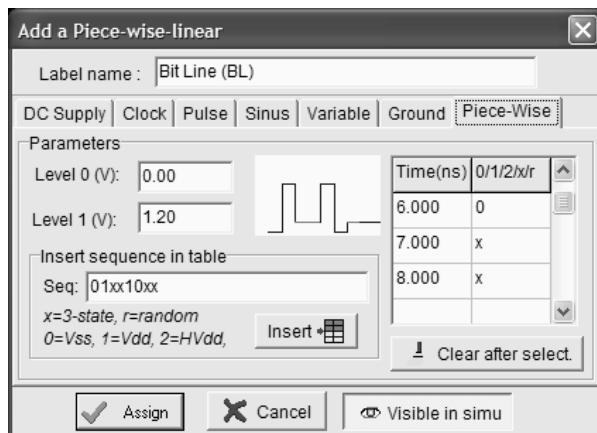


Figure 10-11: Using a pulse to write and read the 5T memory cell (RamStatic5T.MSK)

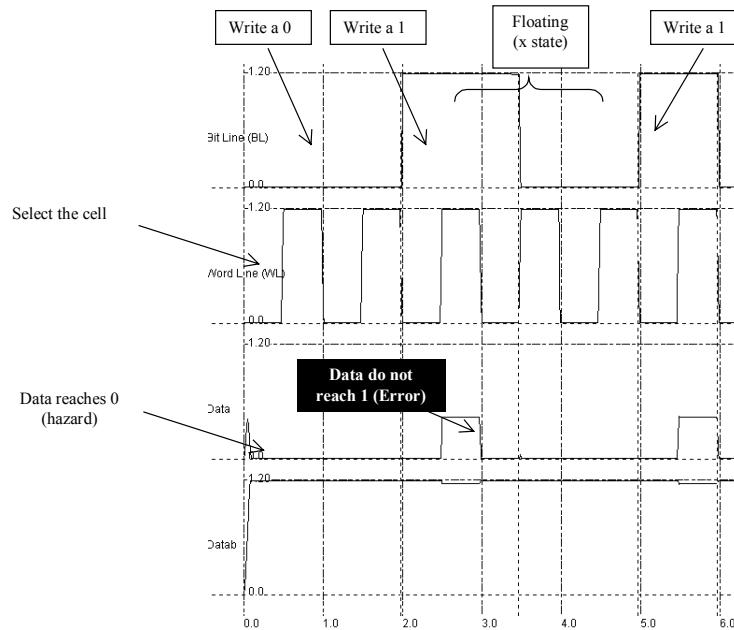


Figure 10-12: Simulation of the 5T memory cell which exhibits a design error(RamStatic5T.MSK)

The simulation of the 5T static RAM is shown in figure 10-12. We have attempted to write a "1" at time 2.0ns, but when the word line WL is asserted (time 2.5ns), we do not observe a change in the memory state, as we would expect. This is due to the wrong sizing of the n-MOS pass transistor, which cannot compete with the n-MOS device of the inverter. Although the pass nMOS width is larger than the nMOS of the inverter, the logic information "1" coming from the bit line turns down to a weak "1" once it passes through the nMOS device (Figure 10-13).

The width of the pass transistor must be enlarged to pass the "1" strongly enough to win the competition against the "0" forced by the nMOS device of the inverter. Consequently, the layout must be changed until the memory state is controlled without any hazard.

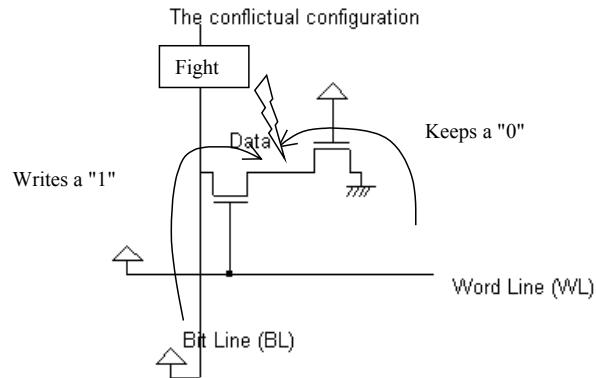


Figure 10-13: Conflict in the static memory cell which is at the origin of the design error(RamStatic5T.MSK)

### Corrected Five Transistor Cell

To obtain a minimum safety margin, the following changes must be conducted (Figure 10-14):

- The switching point of the inverter must be lowered. Consequently, the pMOS width must be significantly reduced.
- The pass transistor must be stronger. In other words, the width must be increased. The drawback is a larger cell area.
- The inverter strength must be reduced. The length must be increased. The drawback is a slower cell response.

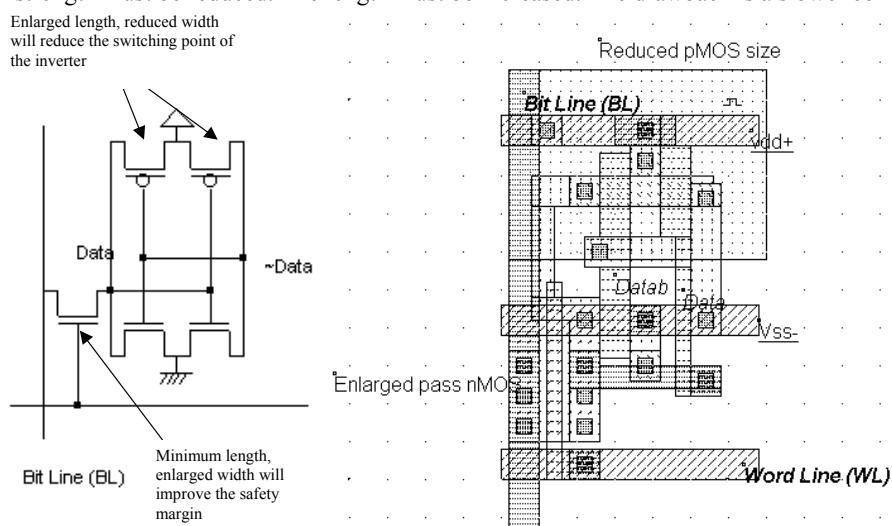


Figure 10-14: Layout of the modified 5T memory cell (RamStatic5Tb.MSK)

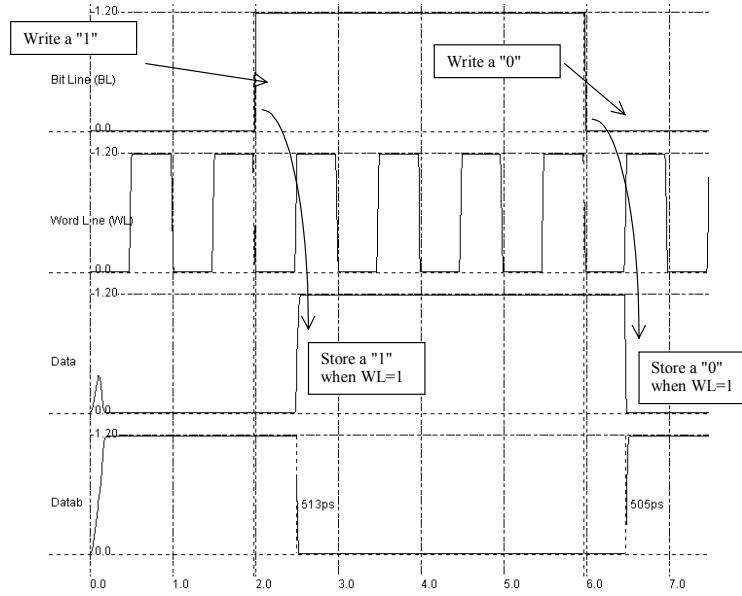


Figure 10-15: Correct simulation of the 5T memory cell (RamStatic5Tb.MSK)

The new layout is shown in figure 10-14. The simulation using model 3 still does not work. However, the simulation using BSIM4 does (Figure 10-15). How should we interpret this result? The modified design is just at the limit between a correct and a wrong design. Any variation (Process, temperature, real MOS dimension) could act on the result. This means that we must change again the design in order to obtain a safe behavior which would not be affected by a small change of parameter. In a 5T implementation, this must be done by further increasing of the pass Mos width, which enlarges the cell dimensions even more. It becomes evident that the 5T design is useless.

### The 6 transistor Memory Cell

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 10-16. 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 10-16. 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.

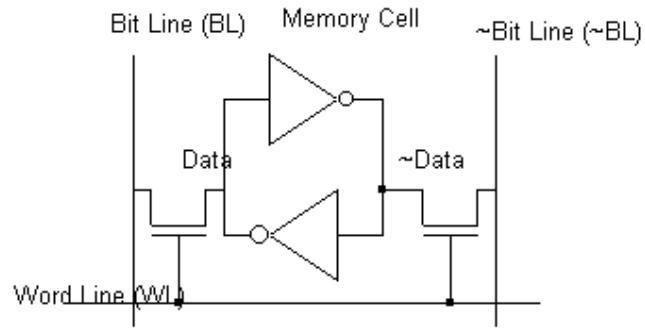


Figure 10-16: The layout of the 6 transistor static memory cell (RAM6T.SCH)

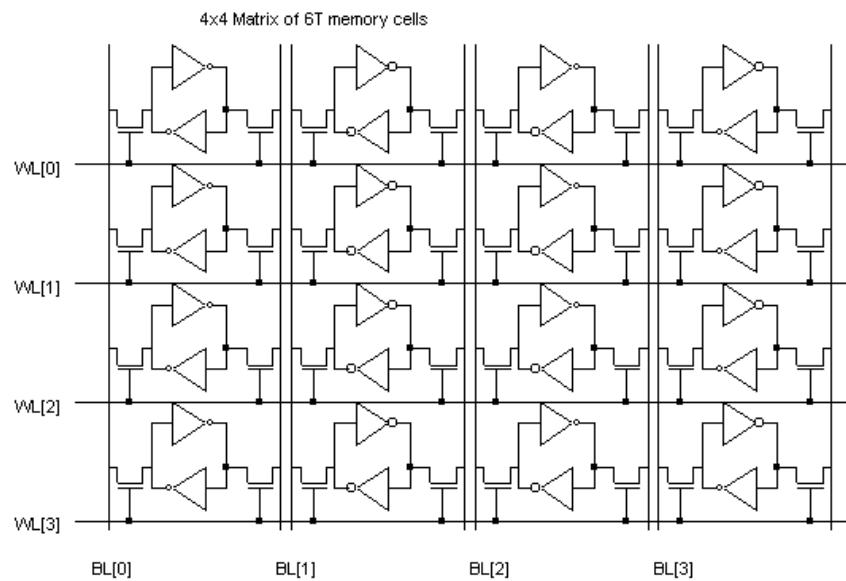


Fig. 10-17 An array of 6T memory cells, with 4 rows and 4 columns (RAM6T.SCH)

### The 6T Cell Layout

The RAM layout is given in Figure 10-18. The *BL* and  $\sim$ *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.

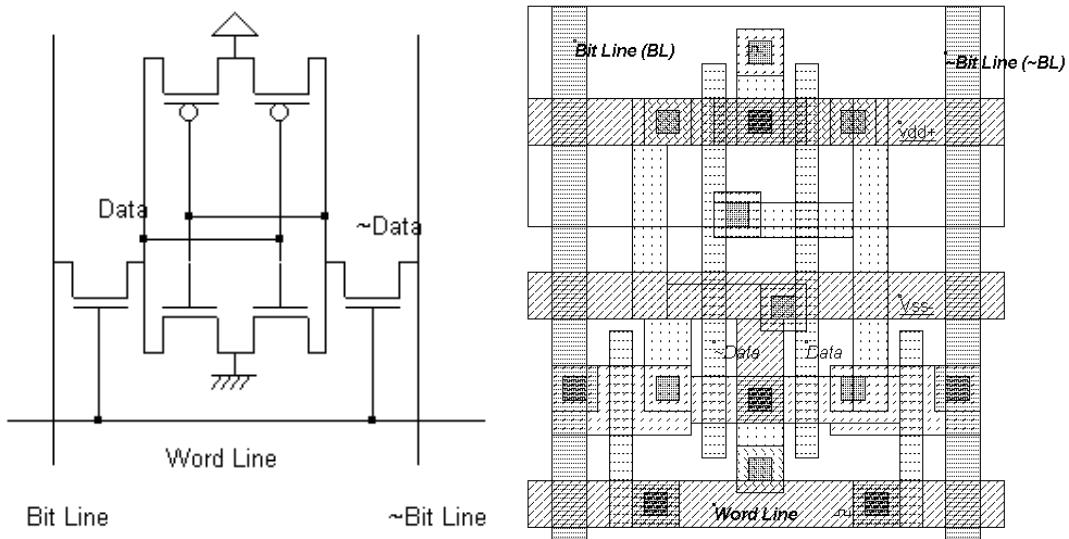


Fig. 10-18. The layout of the static RAM cell (RAM6T.MSK).

The cross-section shows the nMOS devices and the connection to VSS using metal3, situated in the middle of the cell. The *BL* and *~BL* lines, in metal2 are on both sides. The word line controls the access between the bit lines and the internal memory information.

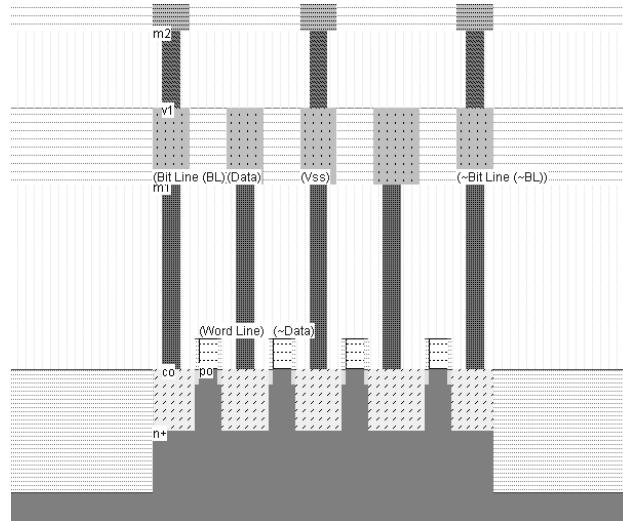


Fig. 10-19. Cross-section of the static RAM cell in the n-channel MOS region (RAM6T.MSK).

The size of the static RAM is given in the menu **Layout size**, accessible through the command **File → Properties**. As shown in figure 10-20, the layout dimensions are 41x46 lambda.

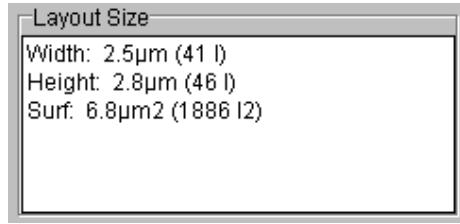


Fig. 10-20. The size of the static RAM cell can be found in the Properties menu(RAMStatic6T.MSK).

### The 6T Memory Simulation

**WRITE CYCLE.** Values 1 or 0 must be placed on *Bit Line*, and the data inverted value on  $\sim$ *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  $\sim$ *Data* propagates to  $\sim$ *Bit Line*.

**SIMULATION.** The simulation parameters correspond to the read and write cycle in the RAM. The simulation steps proposed in figure 10-21 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  $\sim$ *Bit Line* signals are controlled by pulses. The floating state is obtained by inserting the letter "x" instead of 1 or 0 in the description of the signal.

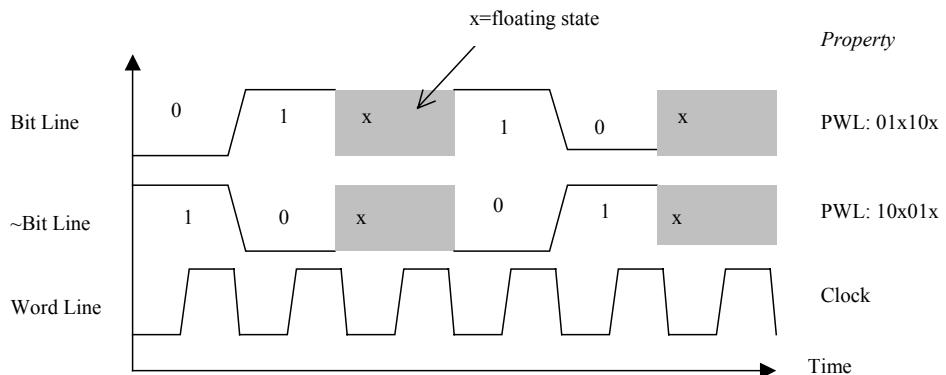


Figure 10-21. Proposed stimulation patterns for the simulation of the 6T static Ram memory (RamStatic6T.MSK)

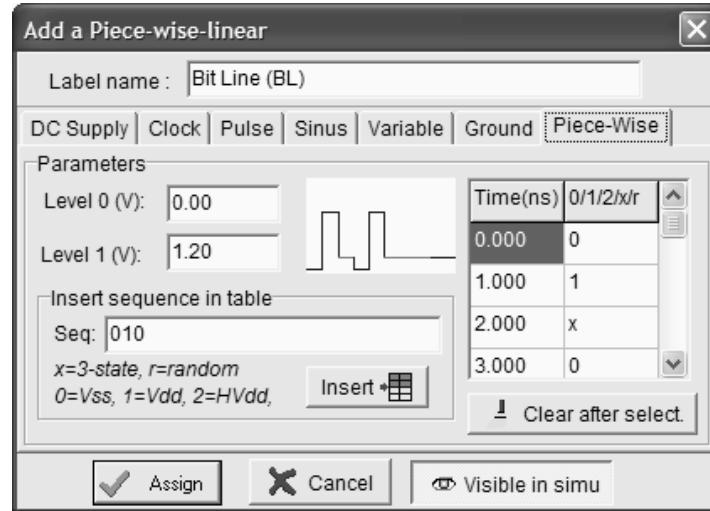


Fig. 10-22. The bit Line pulse uses the "x" floating state to enable the reading of the memory cell (RamStatic6T.MSK)

The simulation of the RAM cell is proposed in figure 10-23. 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.

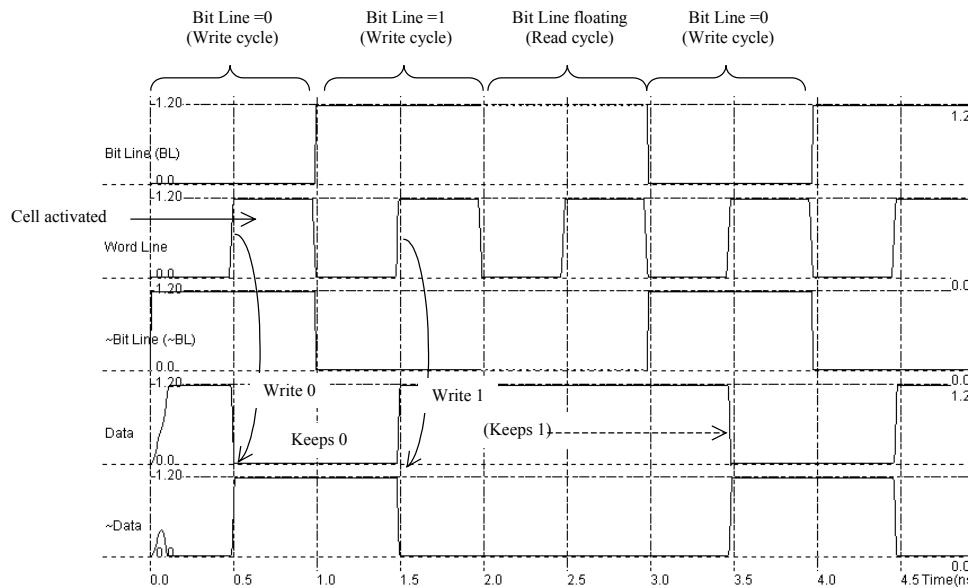


Fig. 10-23. Write cycle for the static RAM cell (RamStatic6T.MSK).

## 10.3 A 64 Bit Static RAM

### A Poor 64 bit RAM array

We can duplicate the RAM cell into a 8x8 bit array using the command **Edit -> Duplicate XY**. Select the whole RAM cell and a new window appears. Enter the value « 8 » for X and « 8 » for Y into the menu. Click on « **Generate** ». The array of 8x8 memory cells is generated.

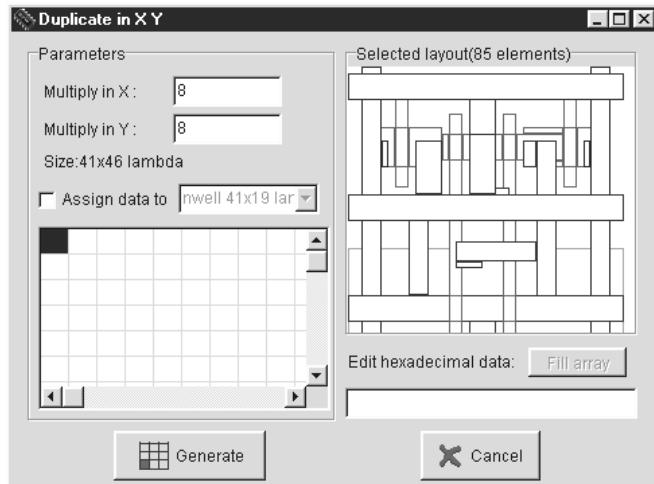


Fig. 10-24. Duplicating the RAM Cell in X and Y (Ram8x8.MSK)

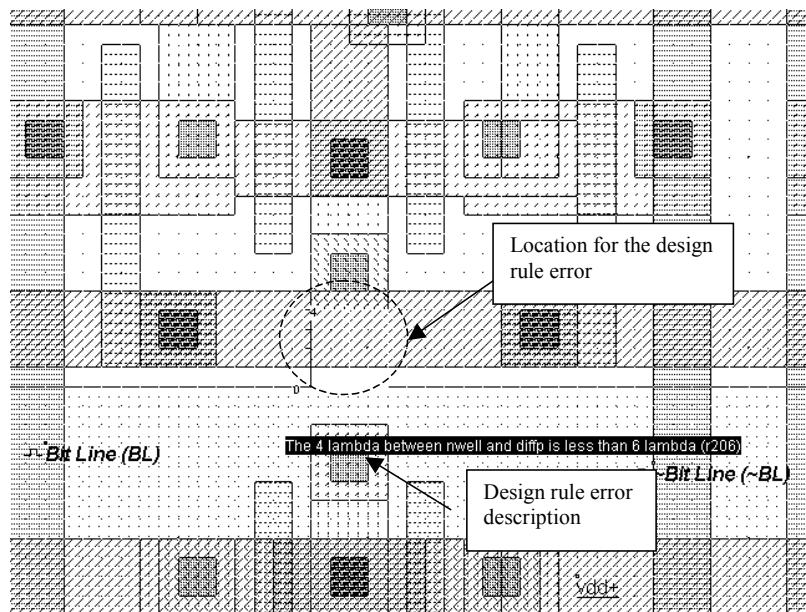


Fig. 10-25. Design rule error between polarization and n-well (Ram8x8.MSK)

Following the duplicate command, it is recommended to verify the design rules, due to the proximity in X and Y of other cells. In figure 10-25, there is a violation of the minimum distance between the polarization contact and the next n-well zone. It can be noticed that we lose an important area at the interface between the pMOS devices and the upper nMOS devices. A good idea is to flip the cells in order to share the VDD polarization and bulk contacts.

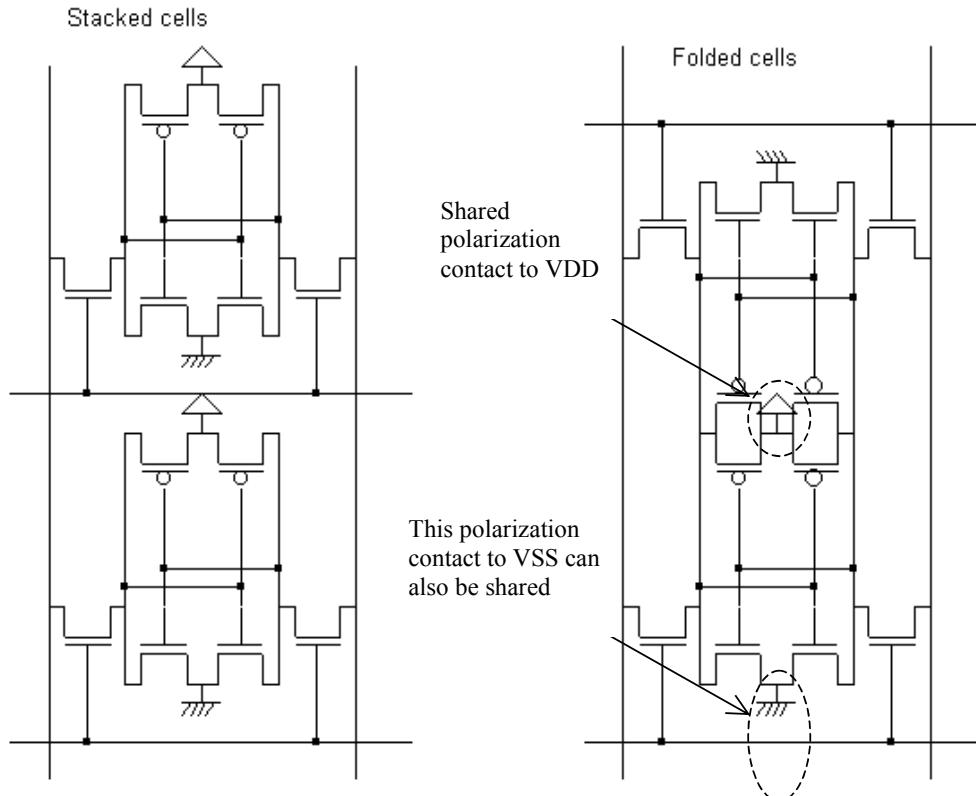


Figure 10-26. Sharing the VDD well and polarization helps reducing the cell area (Ram8x8.MSK)

Compared to the layout of figure 10-25, which corresponds to a 48 lambda height (Once the design rules have been complied with), the height of the same static RAM with shared VDD supply becomes equal to 43 lambda, which corresponds approximately to a 10% gain. There are many ways to further improve the 6T cell density.

### A Compact 8x8 RAM array

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. An example of very compact memory cell [Sharma] is given in figure 10-27.

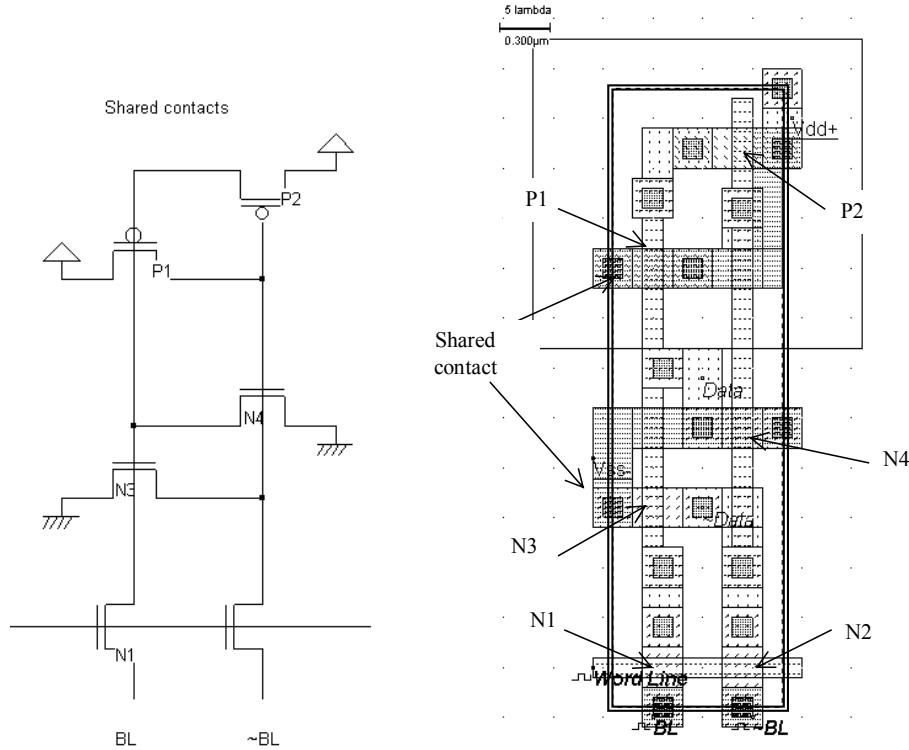


Figure 10-27. Sharing all possible contacts lead to a very compact cell design (Ram6Tcompact.MSK)

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 10-28).

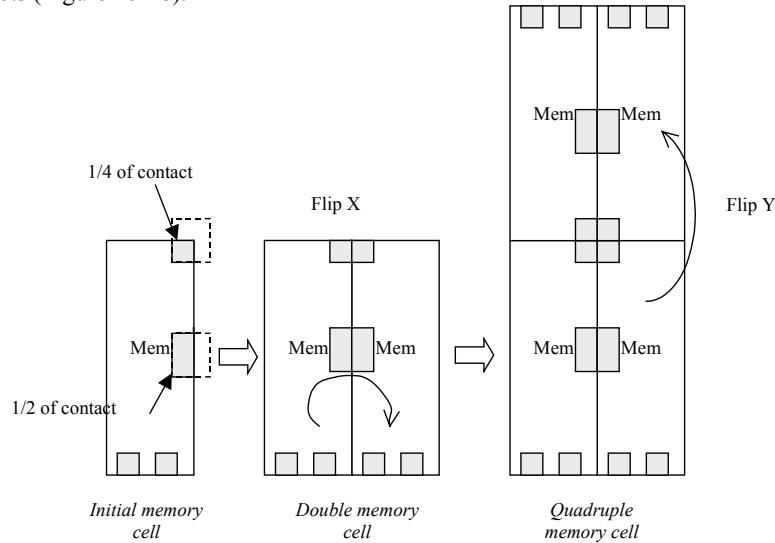


Figure 10-28. The 4 cell memory pattern with shared contacts (Ram6Tcompact.MSK)

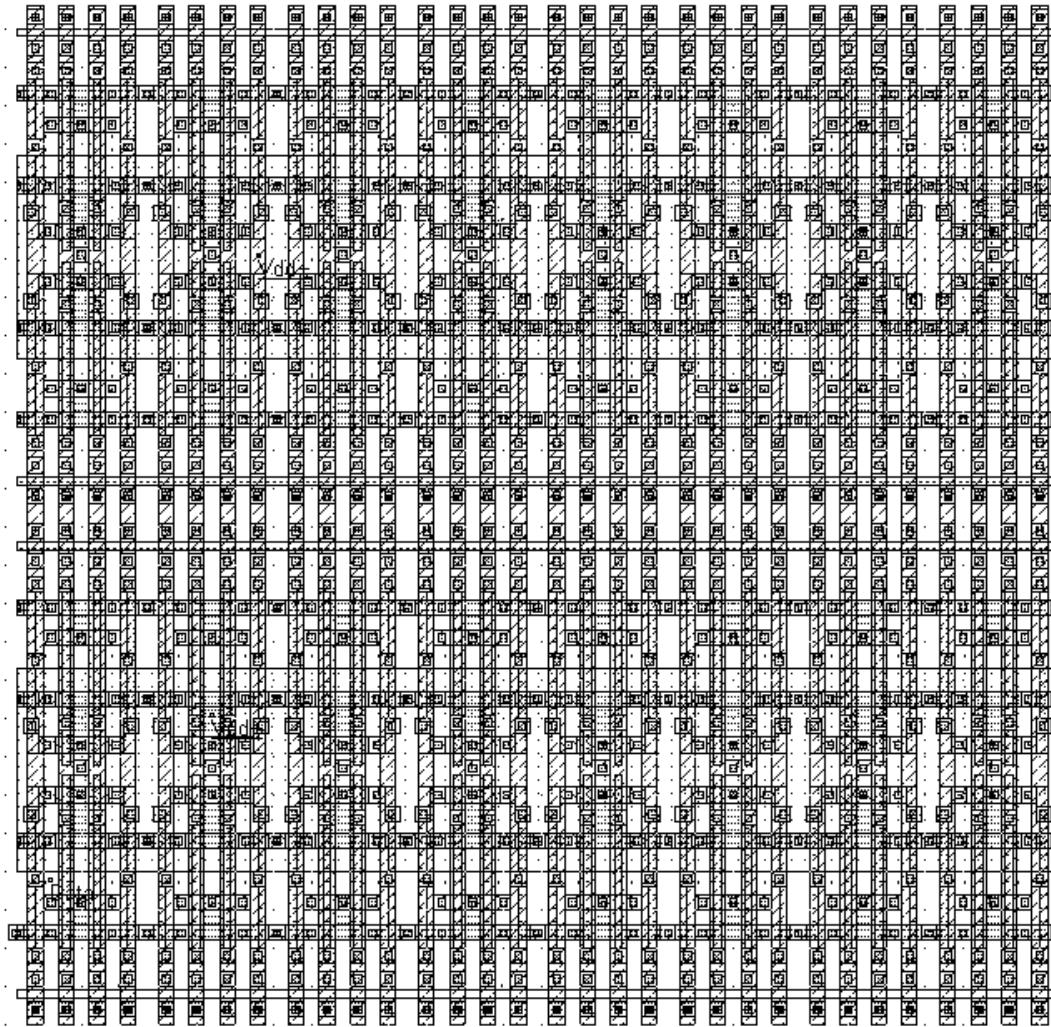


Figure 10-29. Compact 16x4 array of memory cells with shared contacts (Ram16x4Compact.MSK)

As the cell is very narrow, we organize the layout in a 16x4 bit arrangement (16 cells in X, 4 cells in Y). The Ram8x8 block based on the cell "RamStatic6T" leads to a  $434\mu\text{m}^2$  layout ( $0.12\mu\text{m}$ ), while the Ram16x4 block based on "Ram6Tcompact" leads to a  $262 \mu\text{m}^2$  layout, that is a 45% gain. The result is very dense, as shown in figure 10-29. However, specific technological features are developed ( $45^\circ$  layout, direct diffusion to poly contact, pseudo pMOS, local rule violation) to further reduce the cell area by a factor of 2 and more [Sharma].

### Checking the Fault Margin

Although the simulation of the compact memory cell works fine, the question remains whether we are close to an erroneous writing, or not. In other words, do we have a sufficient margin? One answer consists in changing the sizing of the critical transistors and run the simulation. We consider that a 20% margin is sufficient to warranty a correct behavior. Clearly, the pass transistor should be designed large for both fast write and quick read operations [Harzasti]. However, for the objective of a small silicon area and low power consumption, the pass transistor should be kept small. There is clearly a compromise to find.

Secondly, for a safe operation, the pMOS device of the inverter should be small, to lower the commutation point. Meanwhile, a poor pMOS drive would slow down the read operation. We investigate the role of the pMOS width in figure 10-30. The reference size, called (a) in the figure, is the one provided in RAM6Tcompact.MSK. The reference pMOS size is  $W_p=4$  lambda,  $L_p=2$  lambda. The reference pass transistor has a 2 lambda gate length and 4 lambda gate width.

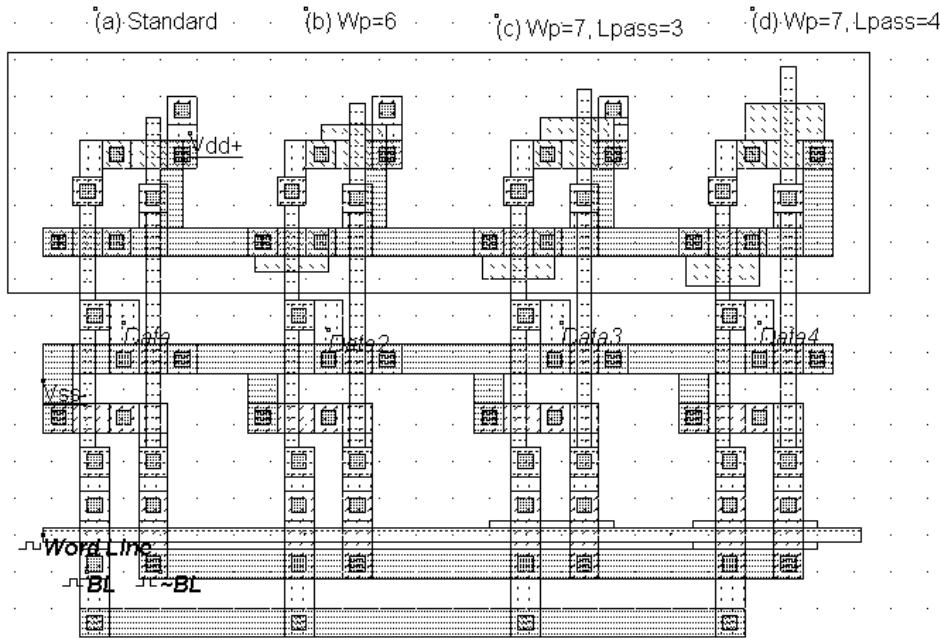


Figure 10-30: Changing the design sizing of critical MOS devices to investigate the fault margin (Ram6Tmargin.MSK)

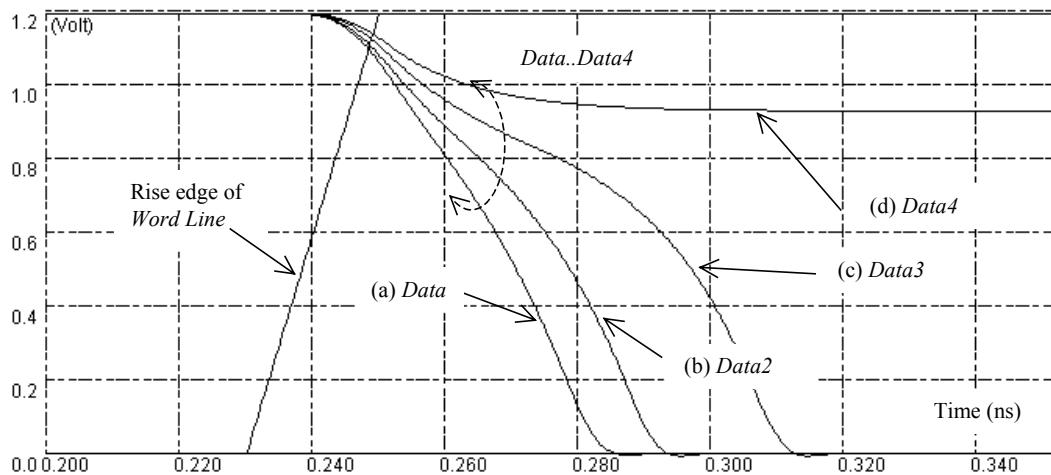


Figure 10-31: Simulation of the four designs when writing a "0" (Ram6Tmargin.MSK)

In design (b) of figure 10-30, the commutation point of the inverter is raised by enlarging the pMOS device, getting closer to a write failure. In (c) the commutation point is increased even more, and the drive of the pass transistor is reduced by enlarging the channel. In (d) the pass transistor channel is altered even more.

The simulation of figure 10-31 consists in writing a "0" in the memory cell. This corresponds to a conflict between the pass transistor which is connected to "0" and the lower inverter which keeps a "1", because  $\sim Data=1$ . The conflict is solved once the *Data* voltage reaches the commutation point  $V_c$  of the upper inverter, which changes the state of the loop. We note a correct waveform for designs (a) and (b), a slow change for design (c) and an erroneous state "1" for design (d).

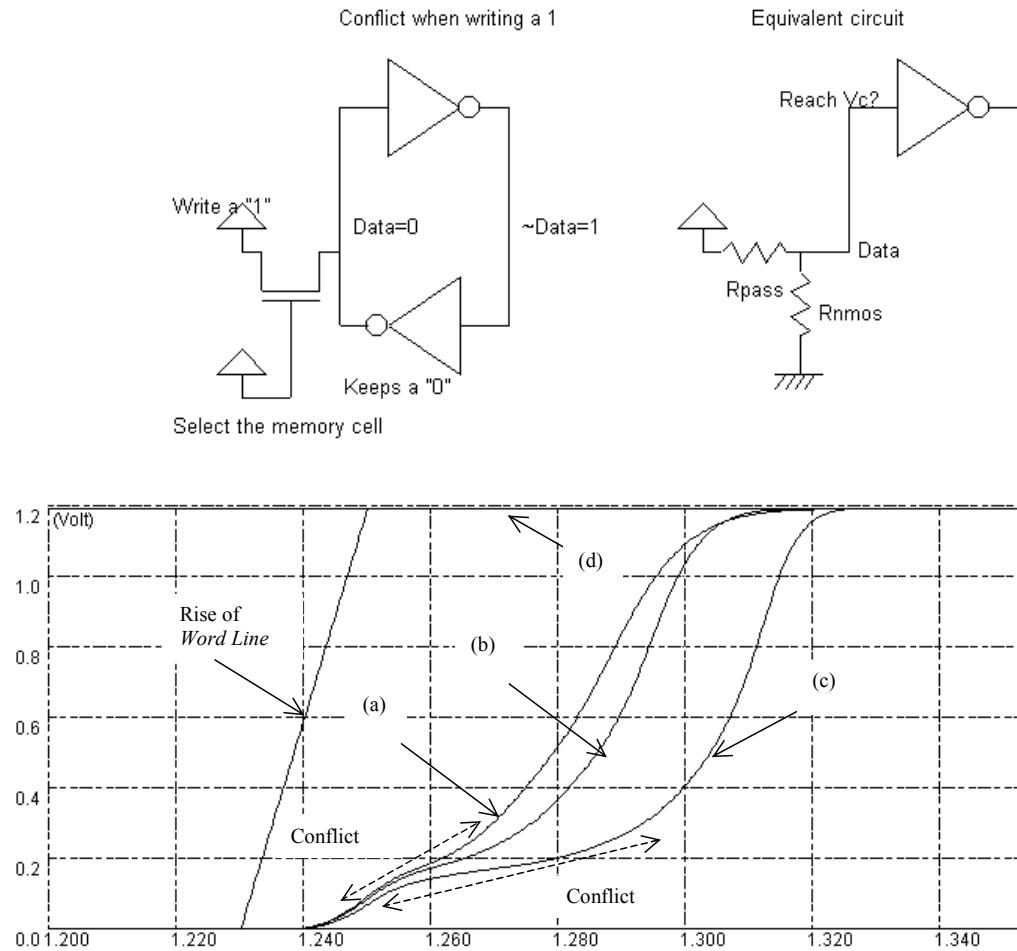


Figure 10-32: Simulation of the four designs when writing a "1" (Ram6Tmargin.MSK)

The simulation of figure 10-32 consists in writing a "1" in the memory cell. This operation is critical as the pass transistors compete with the inverter and the memory may remain at "0", which was fatal for the 5-transistor design of figure 10-10. Again, we note a correct waveform for designs (a) and (b), a slow change for design (c). Design (d) is stuck at "1". The margin between design (a) and (c-d) is sufficient to consider the standard design as a safe design.

### 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.

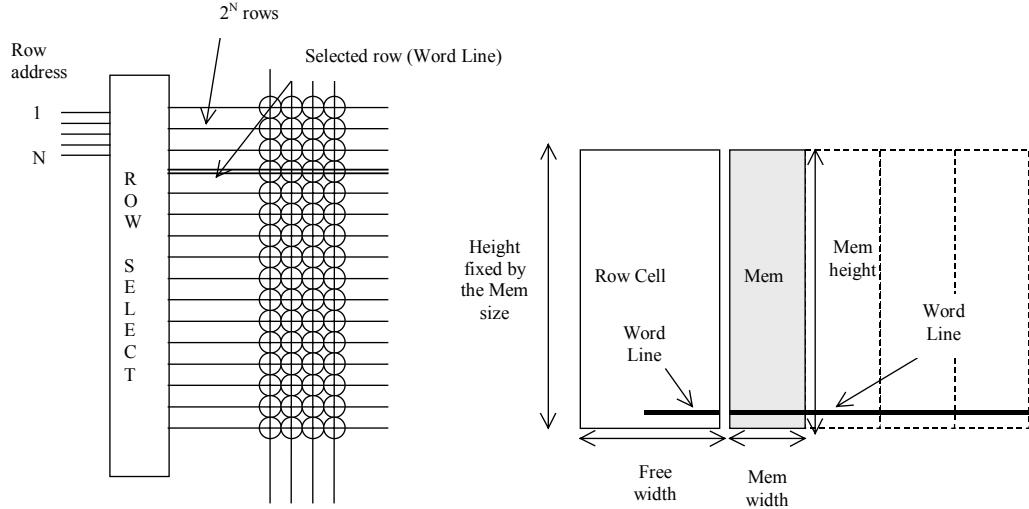


Fig. 10-33 The row selection circuit

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 10-34, 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.

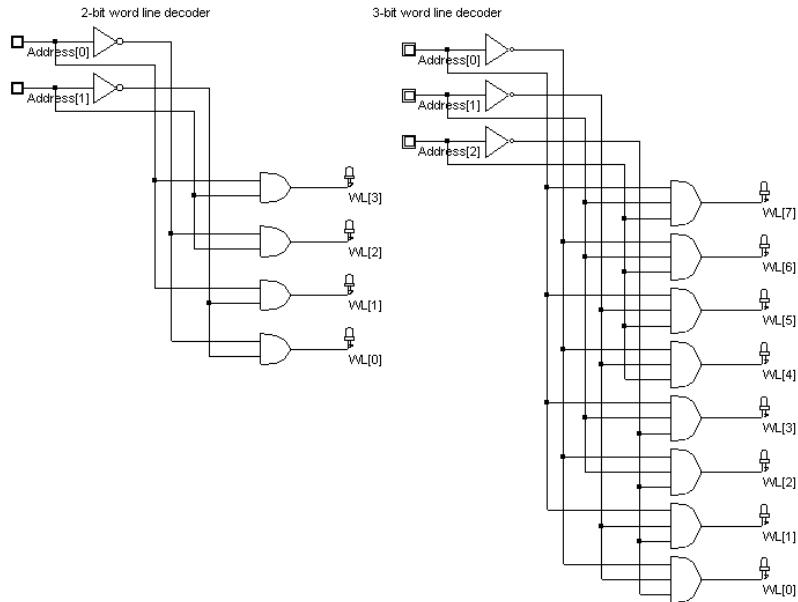


Fig. 10-34. The row selection circuit in 2 bit and 3 bit configuration (RamWordLine.SCH)

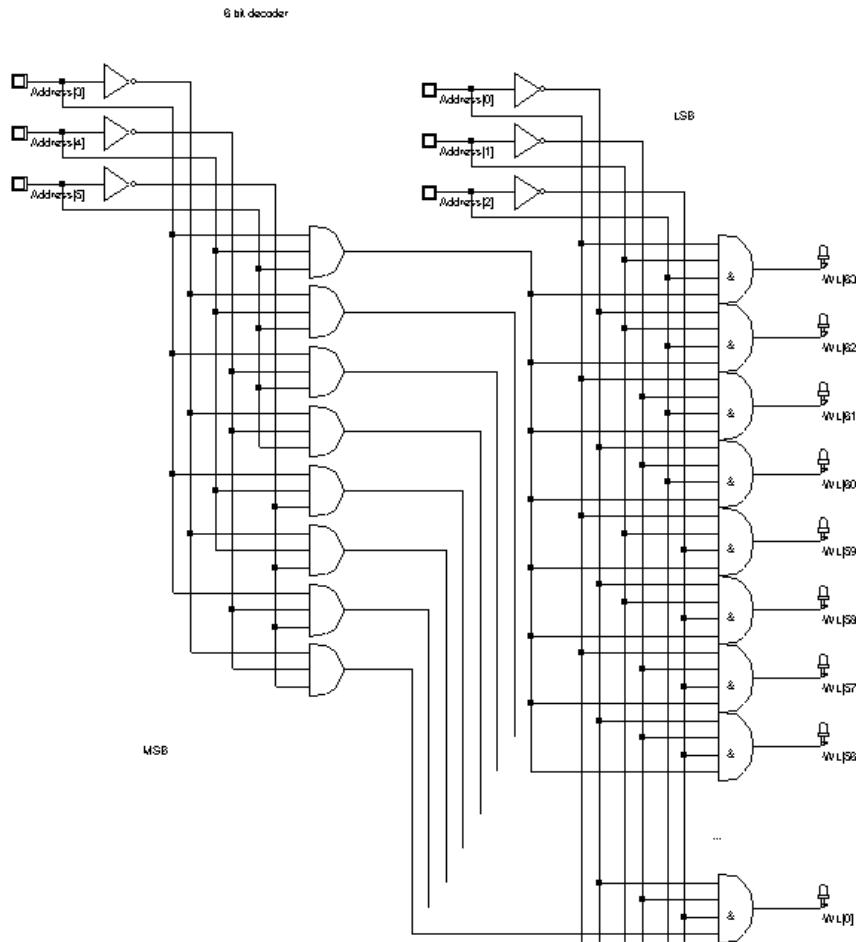


Fig. 10-35. The 6-bit row selection circuit using groups of 3 bits (RamWordLine.SCH)

The 6-bit decoder of figure 10-35 is built from two stages of 3-bit decoders. A total of 64 word lines are generated using this circuit. All word lines have not been shown for clarity's sake. The row selection circuit loads a very significant capacitance which is the sum of *Word Bit* of each elementary memory cell. Consequently, the AND gate is designed using a NAND gate followed by a strong buffer (Figure 10-36).

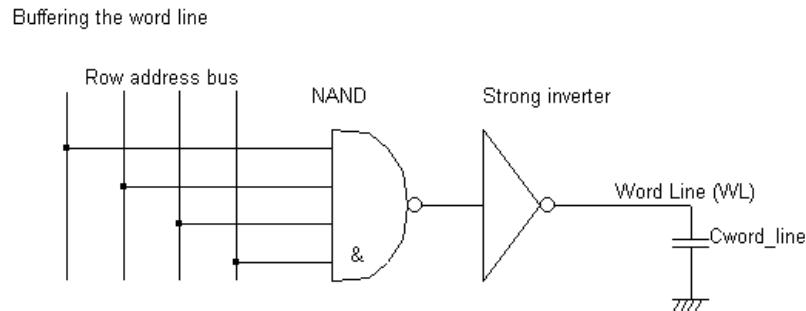


Figure 10-36. Buffering the word line command (RamWordLine.SCH)

The Row Selection circuit height should be adjusted to that of the RAM cell height. When making the final assembly between blocks, the command **Edit → Move Step by Step** is very useful. This command helps to move a selected block with a lambda step (Figure 10-37).

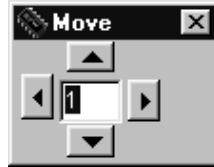


Figure 10-37 Moving a portion of layout step by step

The row selection layout has the particularity to be very regular. It encrypts a binary pattern through the addition of contacts, as illustrated in figure 10-38. The binary encoding can be realized in a semi-automatic way in Microwind. The idea is to create a pattern consisting of a vertical metal box (Address lines), crossing a horizontal metal2 box, and a via to build an electrical link between these two nodes. The basic pattern is duplicated 6 times in X and 3 times in Y for each *word line* circuit. The via layer is copied only if a physical layer is needed at the intersection between vertical address lines and horizontal interconnects. The copying of the via is controlled by a Boolean table sown in figure 10-39.

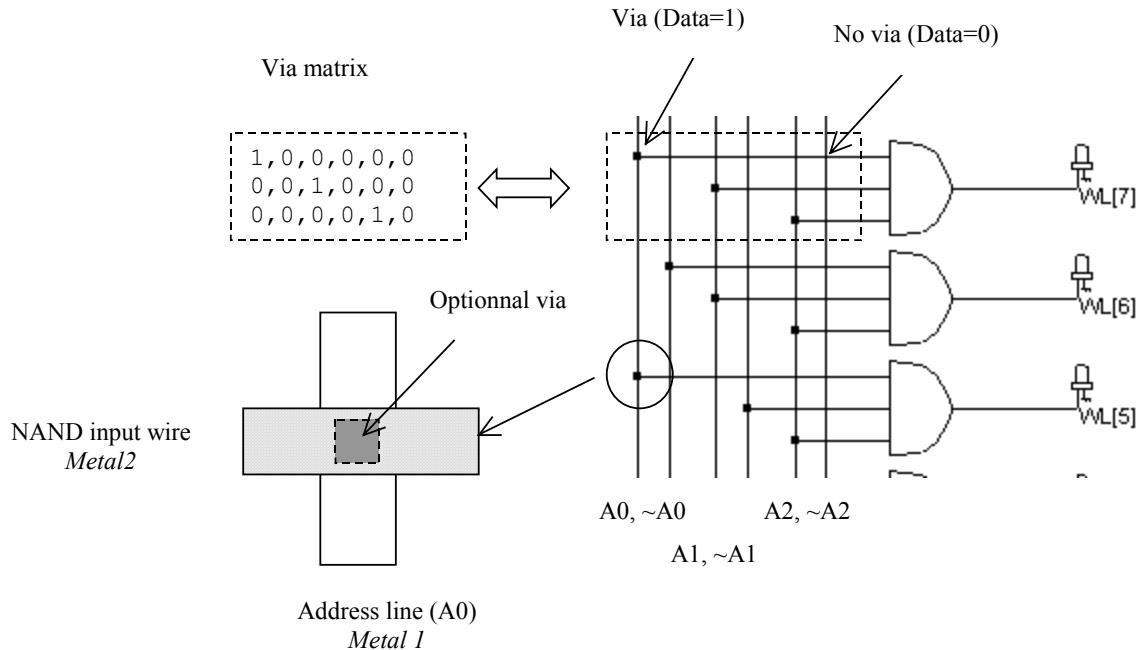


Figure 10-38 The address uses contacts on selected address lines to program the word line according to each address.

The data matrix is filled by zeros and ones. A zero indicates that no via will be generated, a one indicates that the via will be copied. The tick *Assign data* must be asserted, and the via box must be chosen in this case. In the Boolean matrix, the "0" may be changed into a "1" by a double click at the desired location.

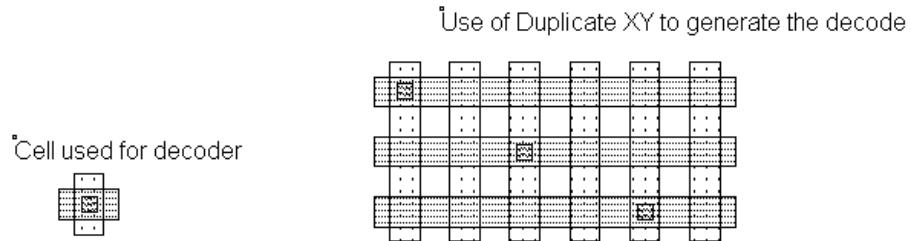
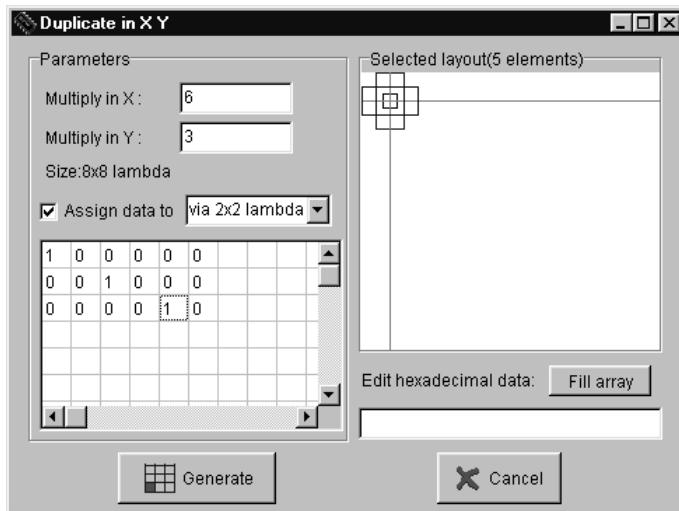


Figure 10-39 Generating the matrix of contacts using the Duplicate XY command (RamDecoder.MSK)

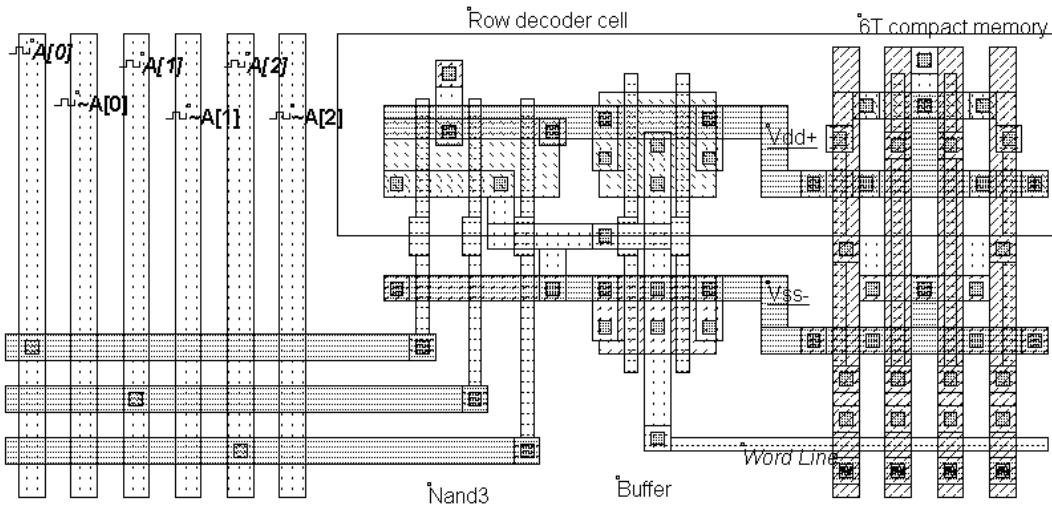


Figure 10-40 The decoder circuit and its link to the memory array (RamDecoder.MSK)

The aspect of the row decoder circuit is shown in figure 10-40. On the left side, we recognize the regular interconnect matrix and its associated via. In the middle, the 3-input NAND gate and a buffer are designed, with the appropriate whirring and supply connections to fit exactly with the rigid layout of the static RAM cell.

### Column Selection Circuit

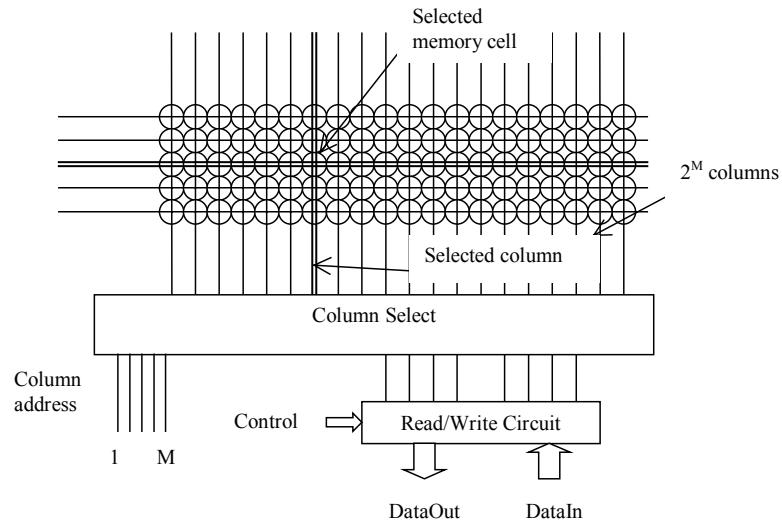


Figure 10-41. The column selection circuit principles

The column decoder selects a particular column in the memory array to read the contents of the selected memory cell (Figure 10-41) 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 10-42 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 10-42) the *DataIn* is amplified by the buffer, flows from the bottom to the top and reaches the memory through *BL* and  $\sim\text{BL}$ . If *Write* is off, the 3-state inverter is in high impedance, which allows the information to be read on *DataOut*.

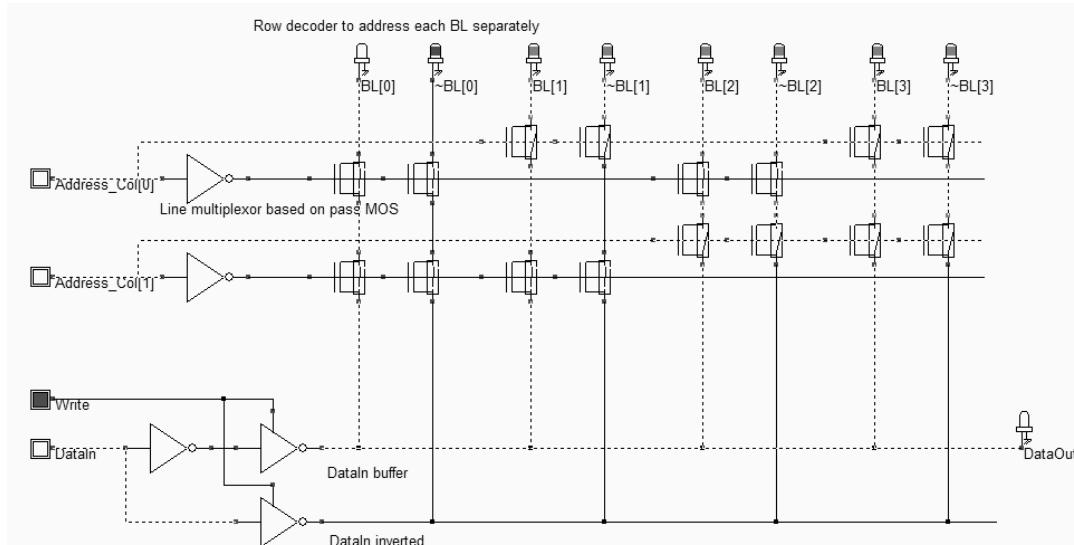


Figure 10-42. Row selection and Read/Write circuit (RamColumn.SCH)

In many cases, the *DataIn* and *DataOut* signals have a wide format, usually 8 or 16 bits. In the schematic diagram of figure 10-43, the *DataIn* and *DataOut* bus is 2-bit wide. Only one address line *Address\_Col[0]* is required to select the appropriate columns. From a layout point of view, the nMOS transistors should fit the narrow width of the memory cell. This is usually done by stacking *BL* and  $\sim$ *BL* pass MOS devices on top of each other. Furthermore, the pass transistors should be designed with a large width to avoid any bad surprise at the write cycle.

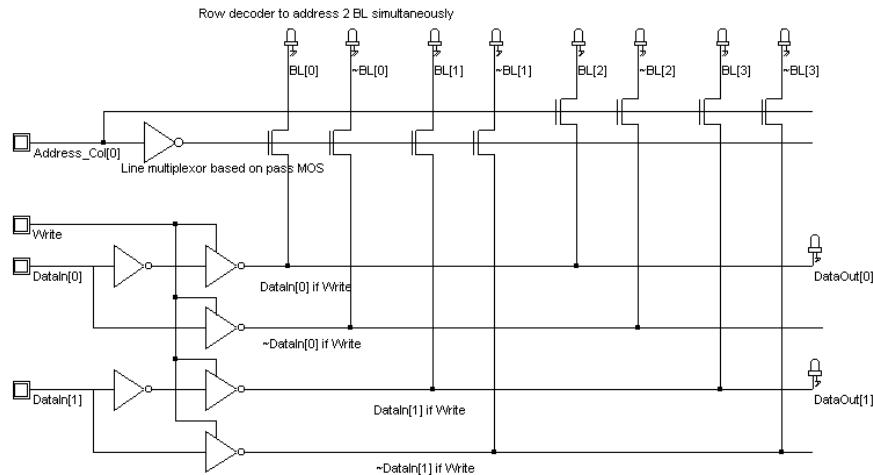


Figure 10-43. Row selection and Read/Write circuit with a 2-bit *DataIn* and *DataOut* information (RamColumn.SCH)

### A Complete 64 bit SRAM

The 64 bit SRAM memory interface is shown in figure 10-44. 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 10-45.

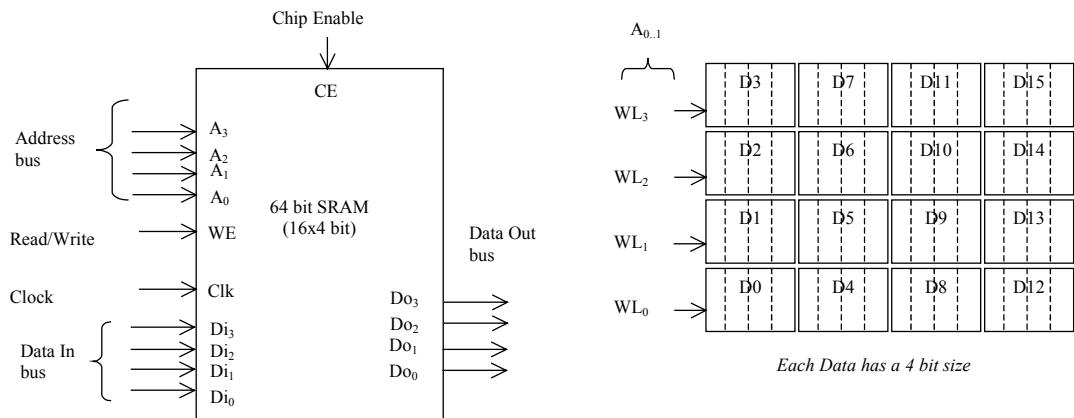


Figure 10-44. The architecture of the 64 bit RAM (RAM64.MSK)

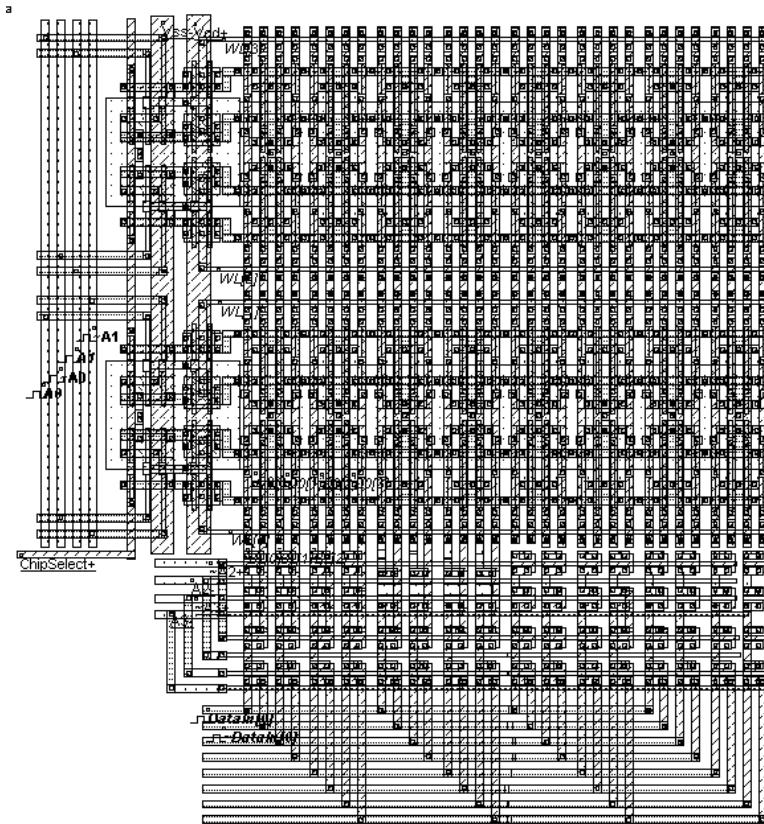


Figure 10-45. The complete RAM layout (RAM64.MSK)

### Emulating a 1024x1024 memory array

The 64 bit static RAM undoubtedly has a very short access time, which is not representative of the real case memory circuits. In reality, 1Mb memory arrays are often created by designing a large matrix of 1024 cells in X and 1024 in Y. The major consequence in terms of read access performances is the addition of a very huge parasitic load on each vertical bit line, which is shared by 1024 cells. Notice that the word line is also shared by 1024 cells, which slows down considerably the line selection process.

Unfortunately, Microwind is not able to handle large complexity designs such as the 1024x1024 static RAM array. An error message would appear, and neither the extraction nor the simulation would run. The idea is to emulate the behavior of the 6T cell in a reduced configuration but with an equivalent parasitic load corresponding to the 1 mega-bit array, by using a virtual capacitor. You may find the virtual capacitor symbol in the palette. The question is to evaluate the approximated value of the total bit line and word line capacitance created by a 1Mb array. This can be done by evaluating the parasitic capacitance of one column in the 64bit RAM array, then by dividing by the number of cells, and finally by placing a virtual capacitance equivalent to the cell value multiplied by 1023.

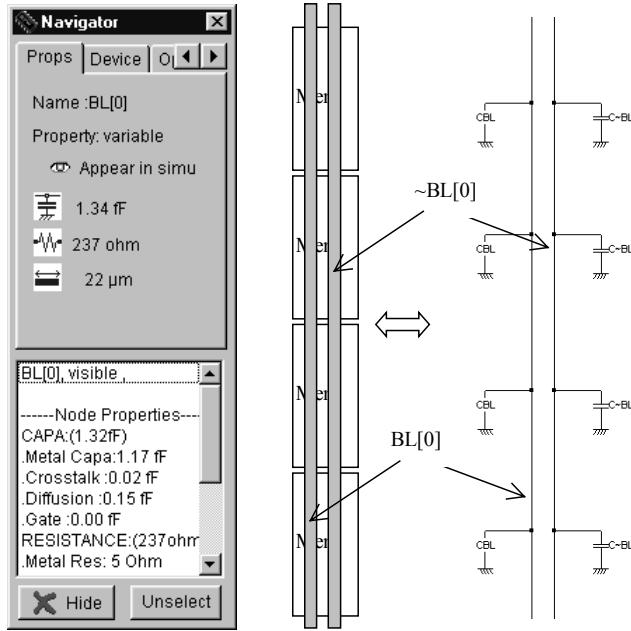


Figure 10-46 Information on the BitLine parasitic capacitance (Ram64.MSK)

The parasitic capacitance of the *BitLine* [0] is about 1.2fF (0.12 $\mu$ m technology). This means that each cell loads the *BitLine* by 0.3fF. The equivalent parasitic capacitance corresponding to 1024 cells would be around 300fF. In the file Ram64Loaded, not only is the simulation slow, but the 300fF load is almost equivalent to a voltage source, and may provoke a logical error, as shown in figure 10-47. First, we write a zero on *Data0* ( $t=0$ ns), and a one on *Data1* ( $t=1$ ns). When we read *Data0*, the bit lines are still at a high level on *BL[0]* and a low level on *~BL[0]* which is a heritage of the previous write cycle. At time  $t=2$ ns, the bit line levels corrupt the reading of the zero, and force the cell to an erroneous one.

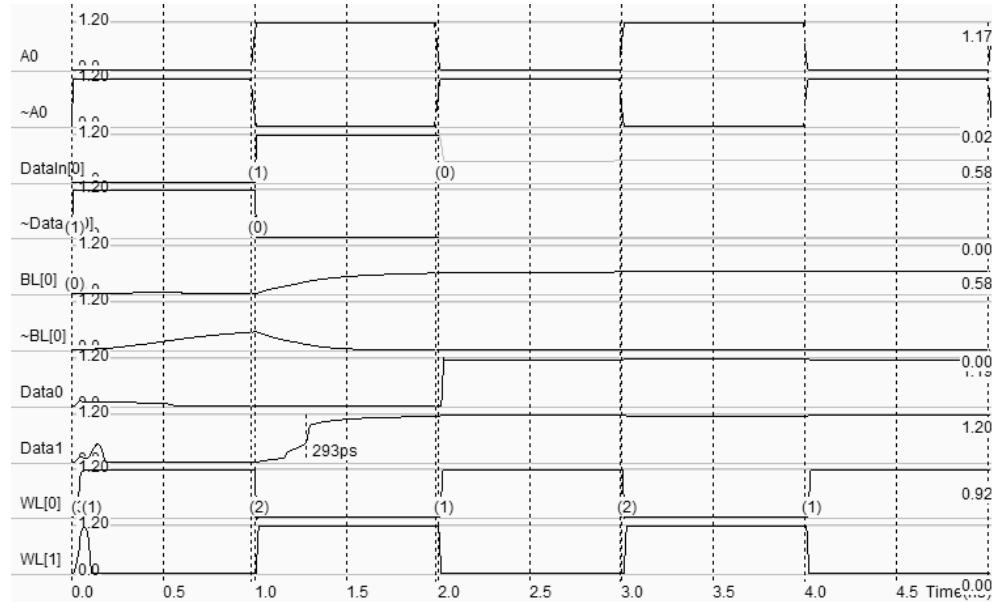


Figure 10-47 The 300fF loading provokes a reading fault at time 2.0ns (Ram64Loaded.MSK)

### Precharge Circuit

To avoid logical errors due to the Bit line parasitic load capacitance and ensure safe read and write operations, a modification of the memory array and timing sequence are required. The usual voltage of precharge is  $VDD/2$ . Before reading or writing into the memory, the bit lines are tied to  $VDD/2$  using appropriate pass gates. This represents a precharge operation. When reading a 0, the  $BL$  and  $\sim BL$  diverge from  $VDD/2$  (Figure 10-48) and reach the "1" and "0" levels after a short time. When reading a 1,  $BL$  reaches "1" while  $\sim BL$  reaches "0". As the static RAM cells are based on active devices (Two ring inverters), the SRAM memories usually provide the fastest reading and writing access times.

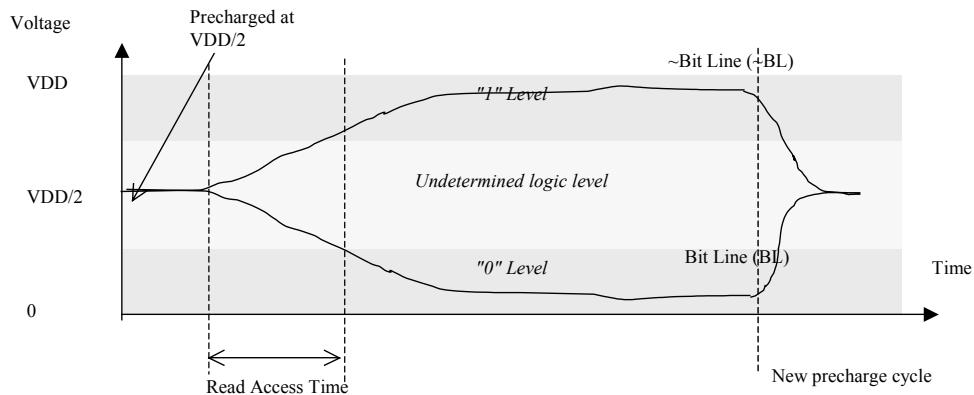


Figure 10-48: Read cycle using a precharge circuit

A simple precharge circuit consists of an n-channel MOS or p-channel MOS (Both switch the voltage  $VDD/2$  without degradation). The drain is connected to  $VDD/2$ , the source to the bit line (Figure 10-49).

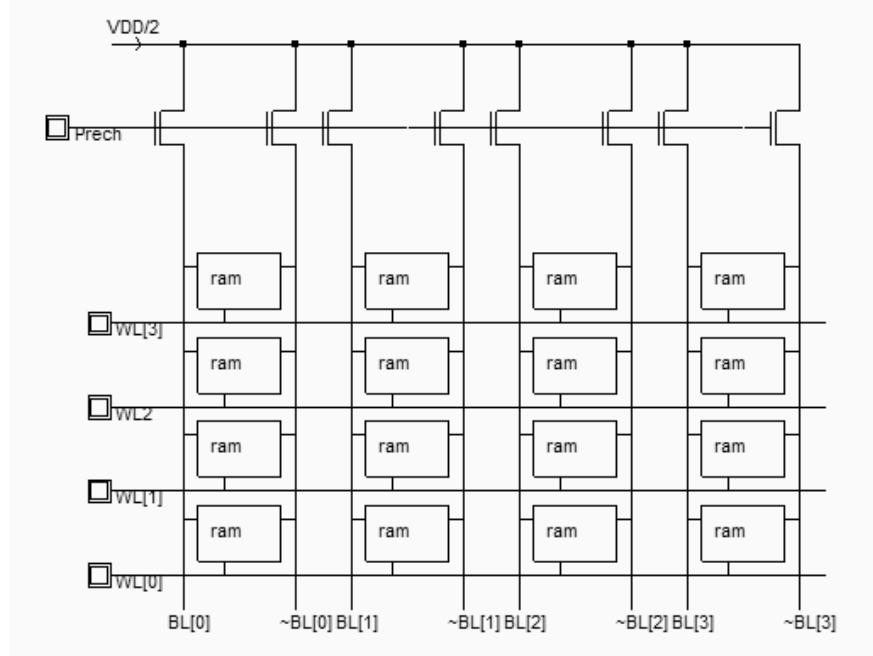


Figure 10-49: Connecting a precharge circuit to all bit lines (RamColumn.SCH)

### Analog Amplifier

To further speed up the read process, analog amplifiers are used. The tiny difference is rapidly converted into a logic level, without waiting until  $BL$  and  $\sim BL$  reach their final voltage (Figure 10-50).

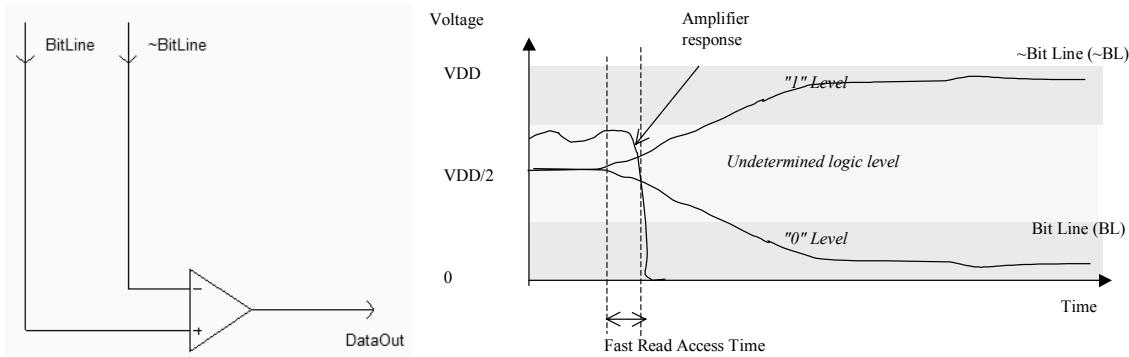


Figure 10-50: Shorter read access time thanks to a precharge circuit

The two commonly used operational amplifier designs are shown in figure 10-51. The first amplifier is a current mirror amplifier (See chapter 11 for more details on this circuit). When *Enable* is on, the *DataOut* signal saturates either to a low or high level, depending on the voltage difference  $V_{BL}-V_{\sim BL}$ . An alternative design for the operational amplifier is also proposed. The positive feedback in the amplifier, that is the cross-coupled link between *DataOut* and the pMOS device, permits faster sense operation than the basic circuit.

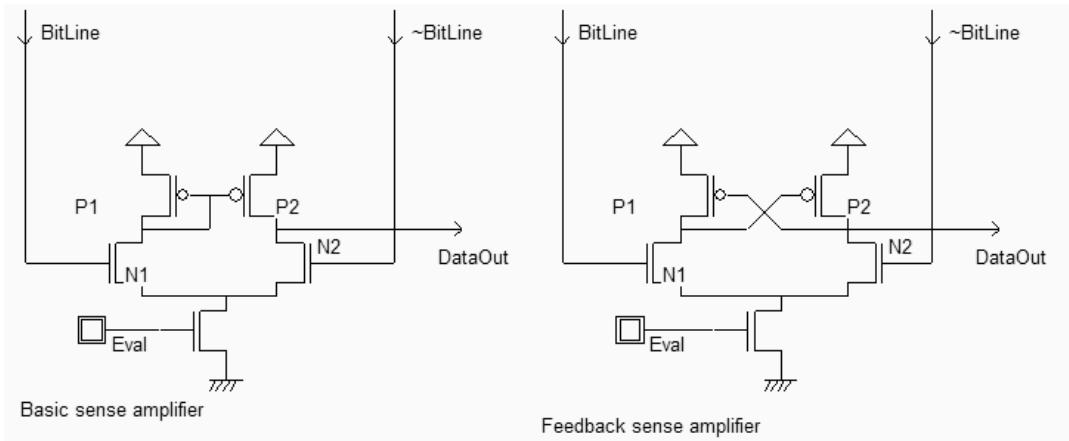


Figure 10-51: Very short read access time thanks to an operational amplifier (RamSenseAmpli.SCH)

The complete logic circuit including the *Write/Read* control, the precharge and the sense amplifier is shown in figure 10-52. When the precharge is active, all bit lines are charged to  $VDD/2$ , while all word lines are low. When the precharge is turned off, one of the word lines (*WL*) is active. A write operation (*Write/Read*=1) forces *BL* and  $\sim BL$  to the desired value given by *DataIn*. A read operation (*Write/Read*=0) turns the write buffers off and turns the sense

amplifier on. The open-loop differential amplifier compares the value of  $BL$  and  $\sim BL$  and gives the logic result on *ReadData[0]*.

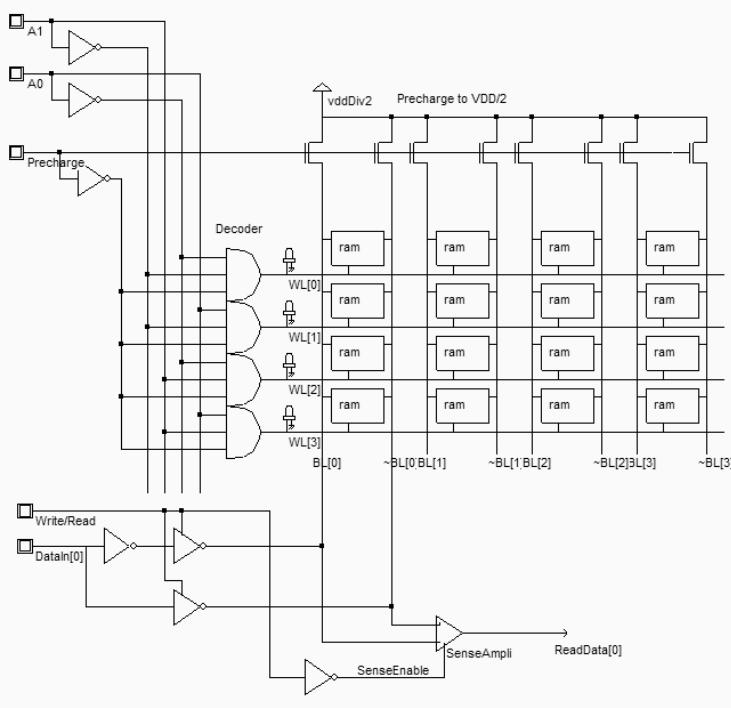


Figure 10-52: The sense amplifier used for read operation (Ram16Sense.SCH)

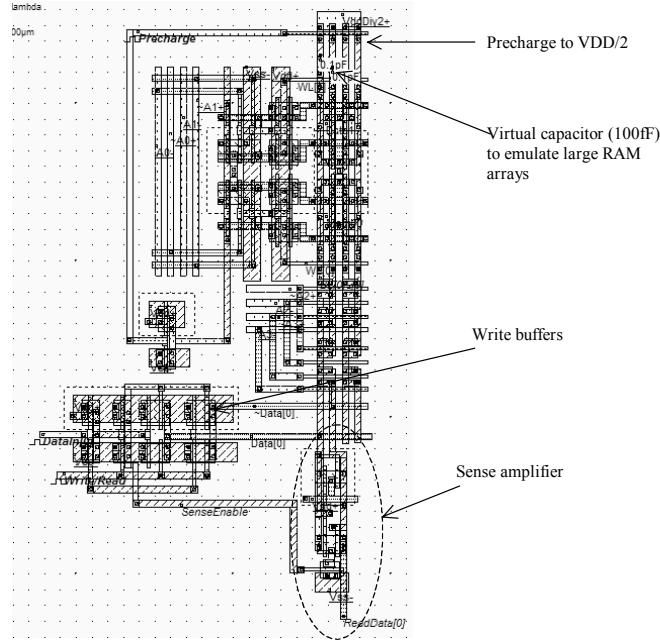


Figure 10-53: Layout of a portion of RAM with the control logic and the sense amplifier (RamSenseAmpli.MSK)

An implementation of the feedback sense amplifier is proposed in figure 10-54. The layout includes a portion of the 64 bit RAM, and the control logic. Parasitic loads corresponding to a 1Mb implementation are added to the vertical bit lines using a virtual capacitor of 0.3pF.

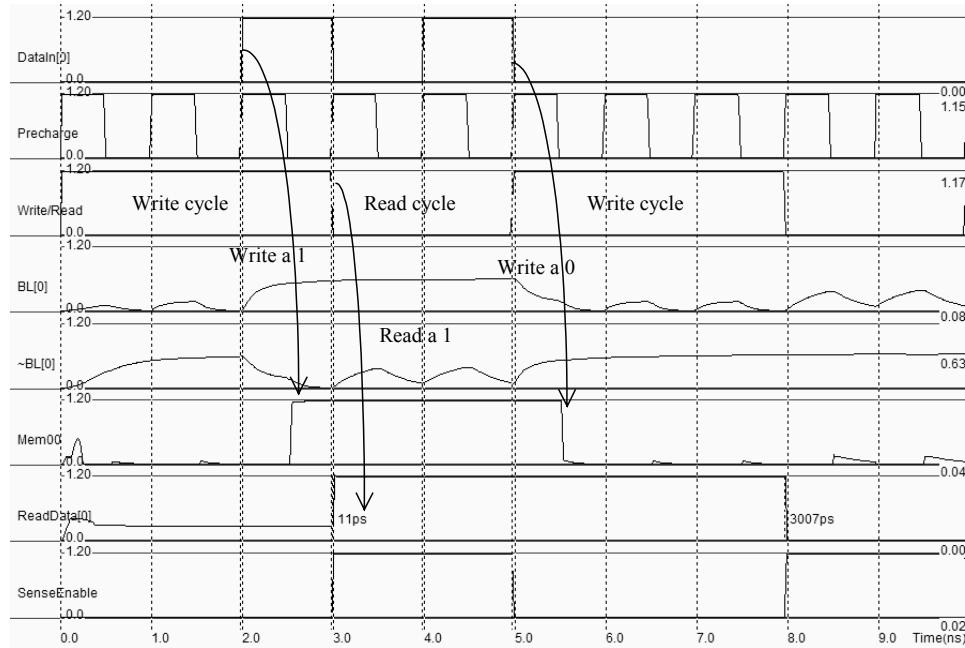


Figure 10-54: Simulation of the sense amplifier (RamSenseAmpli.MSK)

The simulation shown in figure 10-54 includes two *Write* and two *Read* cycles. The first write cycle (0-3ns) writes a 1 in the desired memory cell (*Mem00*). The reading operation (3-5ns) confirms that the memory value is "1" (*ReadData[0]*). The second write cycle (5-8ns) writes a 0 in the desired memory cell (*Mem00*). The reading operation (8-10ns) confirms that the memory value is "0" (*ReadData[0]*). Notice that the write operation (*Write/Read=1*) forces *BL* and  $\sim$ *BL* to the desired value given by *DataIn*. The precharge effect is clearly seen during the read operation.

### Standby Current

The static power consumption of the embedded SRAM is quite high. The origin of this current is the leakage path in the active inverters, due to non negligible off currents illustrated in the schematic diagram Figure 10-55. In the 64 bit memory, the cumulated standby currents reach around  $10\mu A$ . A direct extrapolation to a 1 mega-bit memory would lead to around  $100mA$ , which is incompatible with lower power operations. In practice, Static RAM use low leakage MOS devices whenever possible, or even use MOS devices with enlarged channel length to limit the leakage current.

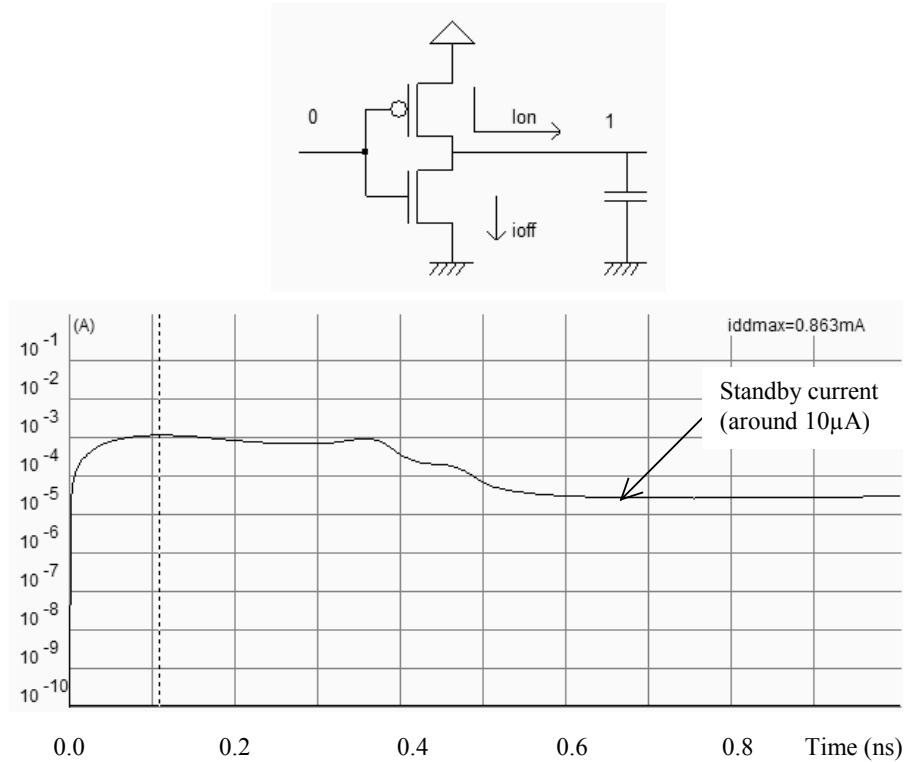


Figure 10-55: Evaluation of the standby current in a 64 bit SRAM memory

## 10.4 Dynamic RAM

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 10-56.

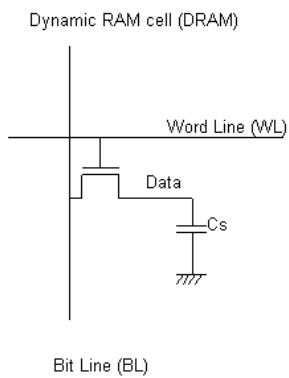


Figure 10-56: The 1 transistor dynamic RAM cell (RAM1T.SCH)

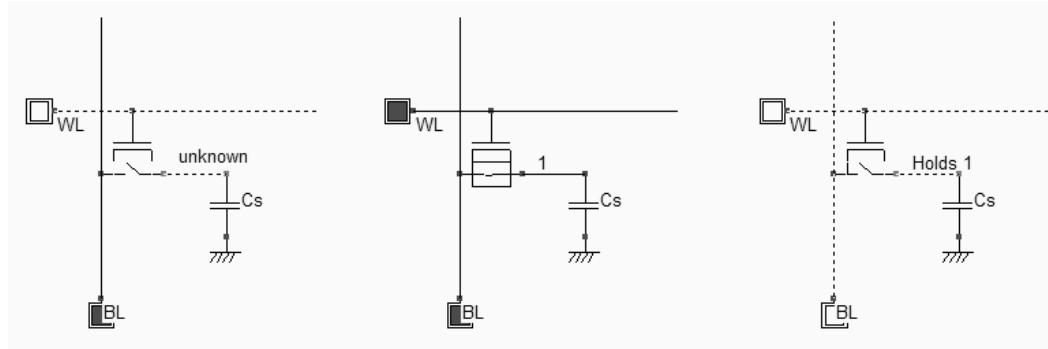


Figure 10-57: Simulation of the Write cycle for the 1 transistor dynamic RAM cell (RAM1T.SCH)

The write and hold operation for a "1" is shown in figure 10-57. The data is set on the bit line *BL*, then the word line *WL* is activated, and *Cs* is charged. As the pass transistor is n-type, the analog value reaches  $VDD - Vt$ . When *WL* is inactive, the storage capacitor *Cs* holds the "1".

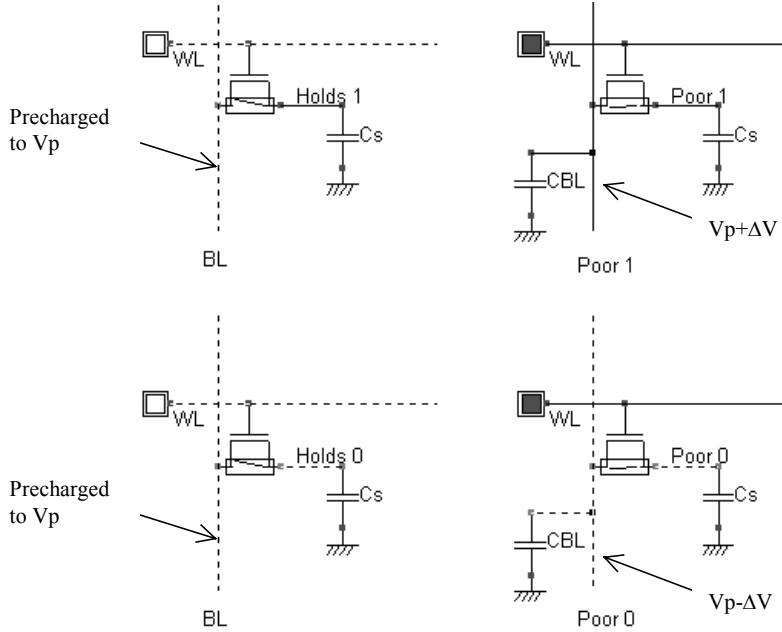


Figure 10-58: Simulation of the Read cycle for the 1 transistor dynamic RAM cell (RAM1T.SCH)

The reading cycle is destructive for the stored information. Suppose (Figure 10-58) that *Cs* holds a 1. The bit line is precharged to a voltage  $V_p$  (Usually around  $VDD/2$ ). When the word line is active, a communication is established between the bit line, loaded by capacitor *CBL*, and the memory, loaded by capacitor *CS*. The charges are shared between these nodes, and the result is a small increase of the voltage  $V_p$  by  $\Delta V$ , thanks to the injection of some charges from the memory. The formulation for this voltage difference can be evaluated in the case of a "1" stored in the memory by putting into equations the conservation of charges.

$$C_{BL} \cdot V_p + C_s \cdot Vdd = (C_{BL} + C_s) V_{final} \quad (\text{Equ. 10-2})$$

The left term of equation 10-2 corresponds to the sum of the charges stored in the bit line precharged at  $V_p$ , and the charges stored in the memory cell. The right term is the charge shared once the two capacitances (The bit line and the memory) are charged with the final voltage. The voltage increase can be rewritten as follows:

$$\Delta V = V_{final} - V_p = \frac{C_s}{C_{BL} + C_s} (V_{dd} - V_p) \quad (\text{Equ. 10-3})$$

Notice that  $C_s$  is usually much smaller than  $C_{BL}$ , meaning that  $\Delta V$  is very small. Now, if the  $C_s$  was holding a zero, the activation of the word line would result in a small decrease of the precharge voltage, to  $V_p - \Delta V$ .

In summary, the bit line voltage  $V_p + \Delta V$  means that the memory state was 1, the voltage  $V_p - \Delta V$  means that the memory state was 0. We say "was" because the memory information is destroyed by the read cycle. What the DRAM memory must do is to convert the  $+/-\Delta V$  into 1/0, and rewrite the memory for a future read cycle.

### DRAM Memory Cell

The DRAM memory cell should be as small as possible, but with the highest possible value for the storage capacitor  $C_s$ . The first idea, shown in figure 10-59, consists in using the parasitic junction capacitance as the storage capacitor  $C_s$ . The polysilicon gate is shared by all word lines in the same row, and the metal interconnect is shared by all bit lines in the same column. The capacitor  $C_s$  is around 0.1fF in 0.12μm technology. Notice that the bit line contact may be shared by two memory cells to improve the density.

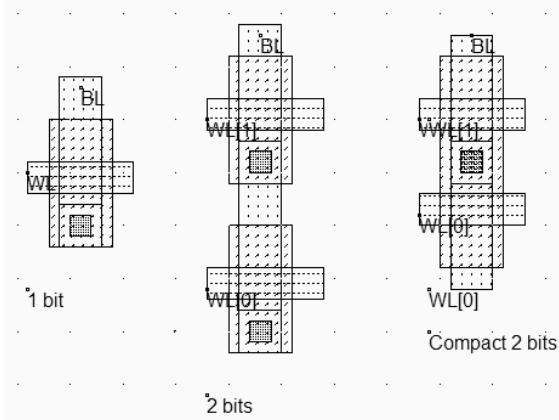


Figure 10-59: A DRAM memory design using parasitic junction capacitance (DramJunction.MSK)

There are two main problems with this design: first, the capacitance is very small, because the junction capacitance does not have a very high value, secondly, a leakage exists between the capacitor  $C_s$  and the bit line, through the access transistor, even when a zero voltage is applied on the bit line. Consequently, the charges stored in the capacitor tend to evacuate, partly to the ground through the parasitic diode ( $Idiode$ ), partially to the bit line through the channel current  $i_{off}$ , which means that the memory information is retained only for less than one μsecond (Figure 10-60).

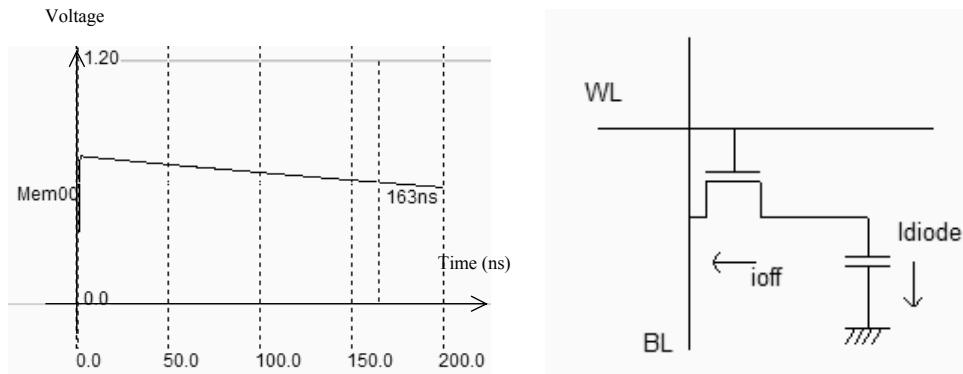


Figure 10-60: Leakage current in the dynamic RAM based on a junction capacitance (DramJunction.MSK)

The leakage current may be reduced by using low leakage MOS devices with non-minimal channel length, but the best technique is to increase by 2 or 3 orders of magnitude the storage capacitor. 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 10-61 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 increased height between the substrate surface and metal1, and the use of high permittivity dielectric oxide. The silicon dioxide SiO<sub>2</sub> has a relative permittivity  $\epsilon_r$  of 3.9. Other oxides, compatible with the CMOS process have a higher permittivity (Higher "K") : Si<sub>3</sub>N<sub>4</sub> with  $\epsilon_r$  equal to 7, and Ta<sub>2</sub>O<sub>5</sub> with  $\epsilon_r$  equal to 23.

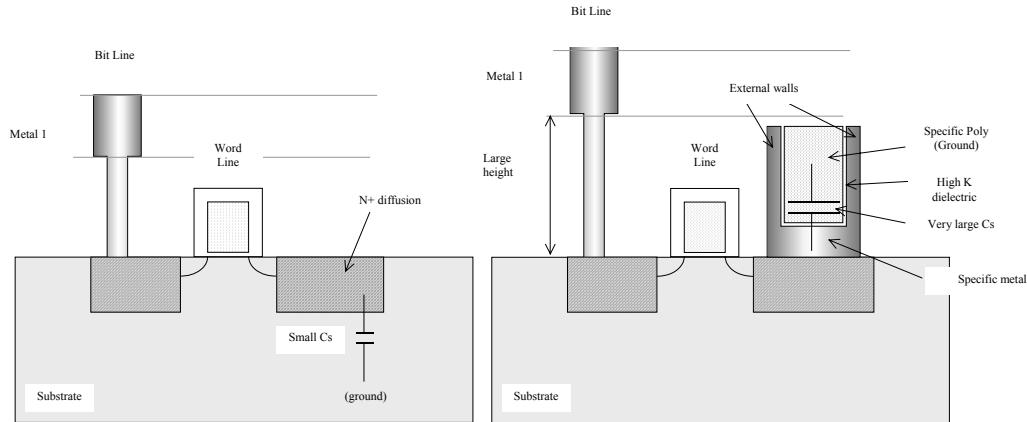


Figure 10-61: Increasing the storage capacitance (Left: junction capacitor, right, embedded capacitor)

The drawback of these methods is the addition of specific process steps to build the 3D capacitor, including delicate fabrication of high dielectric materials. The additional processing steps for the embedded DRAM represent approximately a 25% cost over the basic process. In Microwind, the high capacitance memory can be generated using the option layer, as shown in figure 10-62. The option layer is placed at the intersection between the n-diffusion area and the VSS metal line.

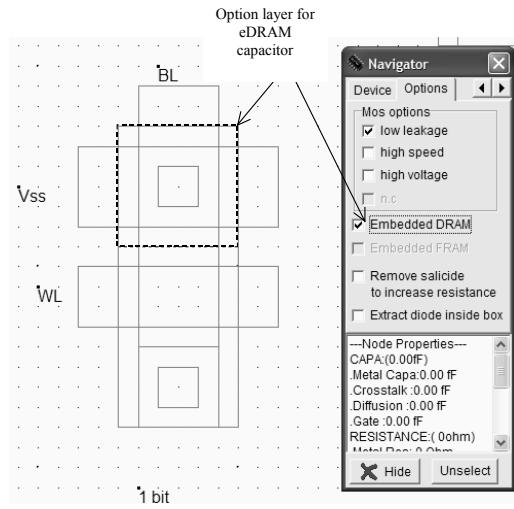


Figure 10-62: The option layer configured for the embedded capacitor (DramCell.MSK)

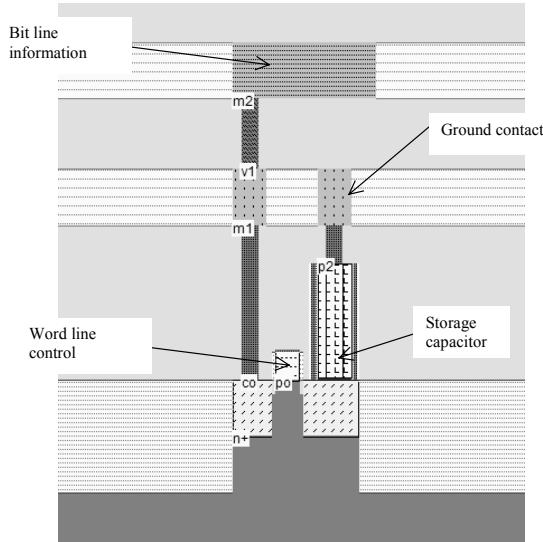


Figure 10-63: Cross-section of the DRAM cell with an embedded capacitor (DramCell.MSK)

The cross-section of the DRAM capacitor is given in figure 10-63. 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 ( $\text{SiO}_2$  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  $20\text{fF}$  in this design. Higher capacitance values may be obtained using larger option layer areas, at the price of a lower cell density. A DRAM array is shown in figure 10-64, together with a vertical cross-section at the capacitor location.

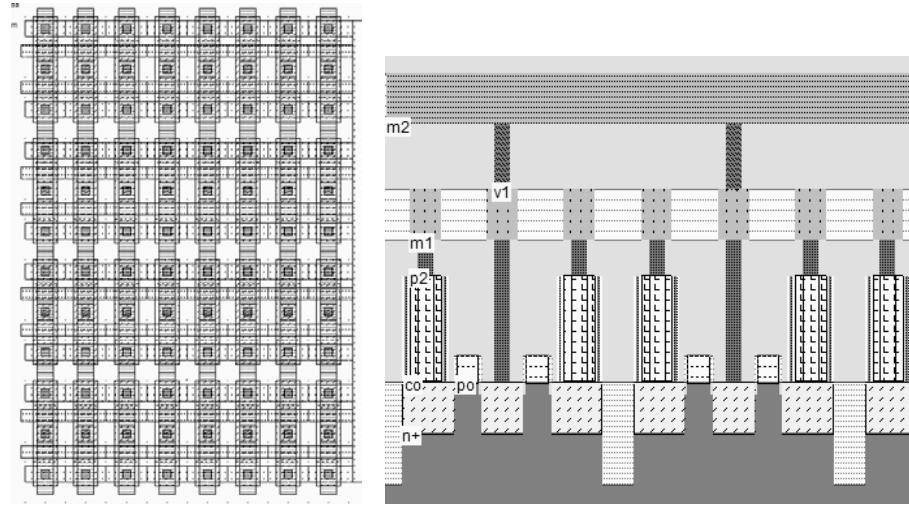


Figure 10-64: The stacked capacitor cell compared to the diffusion capacitor cell (DramEdram.MSK)

The charges stored in the capacitance of the dynamic RAM do not keep intact for very long, because of the leakage current ( $I_{off}$ ) of the pass transistor. However, the time during which the stored voltage may be considered as constant is more than two orders of magnitude higher for an embedded capacitor than for a simple diffusion.

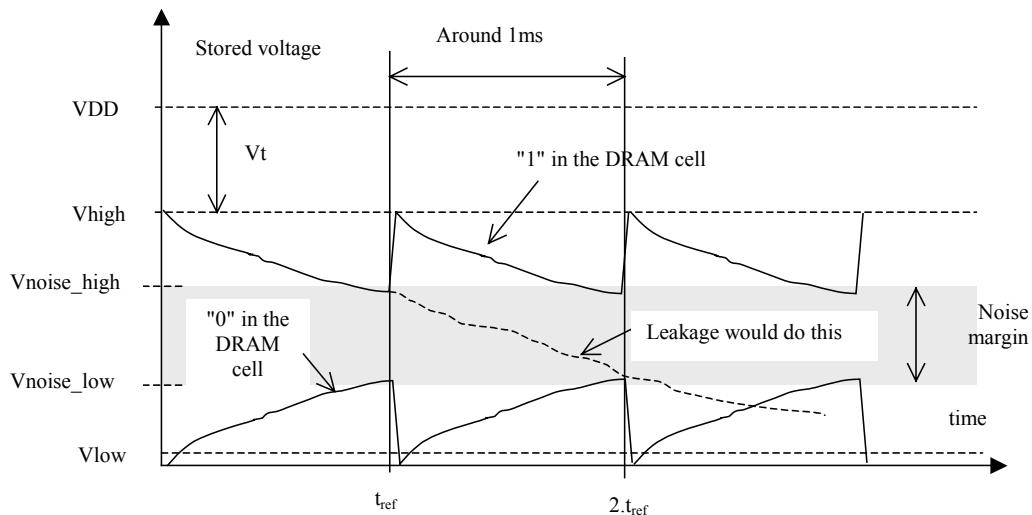


Figure 10-65: The refresh cycle for the DRAM

The capacitor discharge is the order of 1ms, as illustrated in figure 10-65. Before the memory voltage enters the undetermined zone ( $V_{noise\_high}$  when the memory contents was 1), the memory needs to be refreshed. The periodical refreshment of the memory is part of the DRAM memory control circuit.

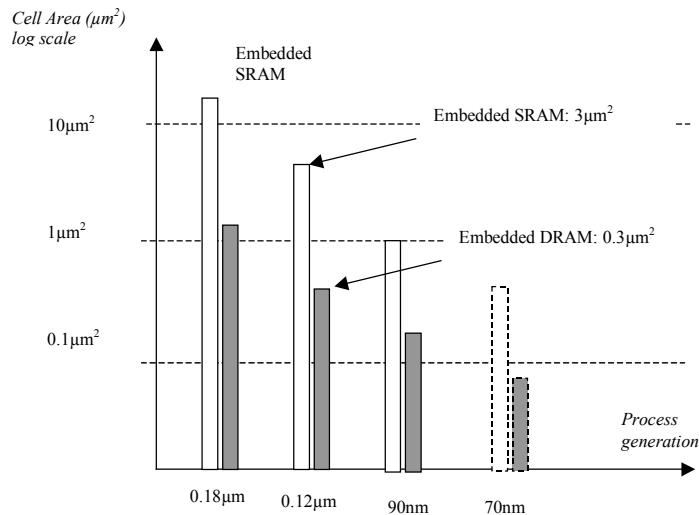


Figure 10-66: The DRAM size compared to the SRAM size

The size of the dynamic memory point is always significantly smaller than the size of the static memory point. This is of key advantage when large memory arrays are needed. However, the dynamic memories are slower to read because of the passive structure of the memory point. Static RAM memories are used for high speed data exchange while DRAM memories are used for mass memory storage.

## 10.5 ROM memory

The most simple non-volatile memory consists of a layout in which the data is permanently written through a specific layout arrangement and cannot be changed by the user once fabricated. The two logic states are shown in figure 10-67. The basic patterns for a "0" consists of an open circuit, while the "1" corresponds to the existence of a n-channel MOS. We assume that the vertical bit lines are precharged at 1 by default. In the case of an open circuit, the bit line remains at 1, and *dataOut* gives 0. If an nMOS device exists, the *BL* is tied to ground and *dataOut* gives 1. The logic programming is part of the fabrication process. ROM memories are used for storing microprocessor programs, mathematical values such as a sampled sinusoidal data for waveform generators, etc.. The size of ROM memories is usually small as compared to other embedded memories, because the contents cannot be changed.

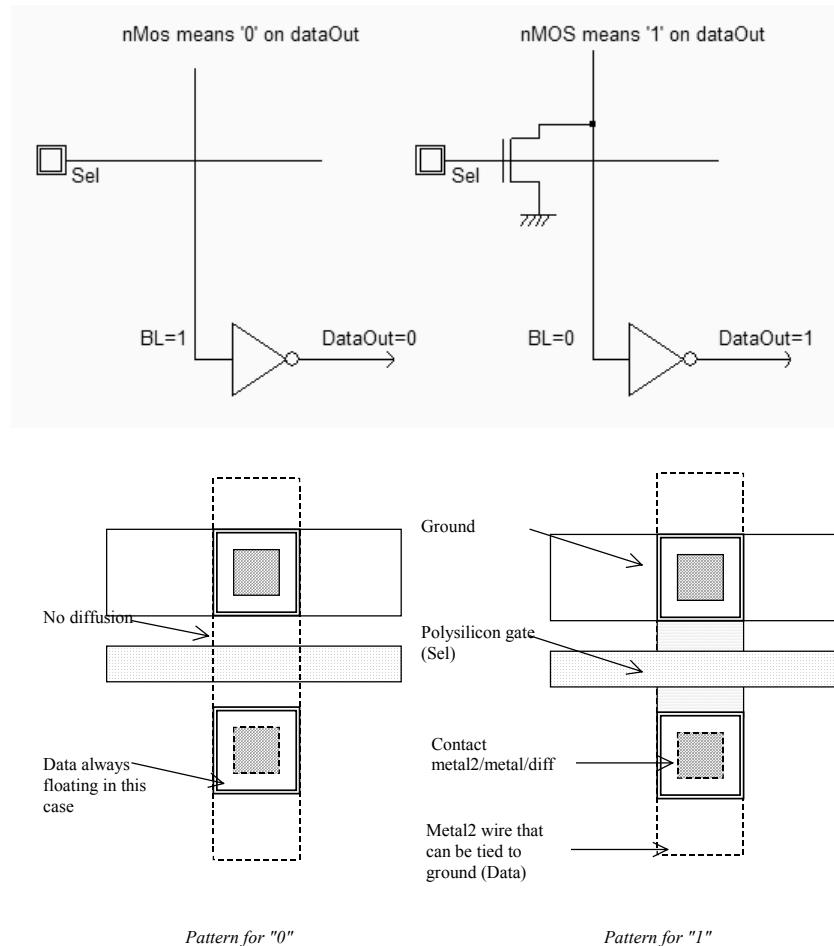


Figure 10-67 Changing the status of the memory point by a diffusion layer

We may create an array of ROM memory that stores the string "Hello!", which corresponds to the following binary data stream (Table 10-2).

Address	Character	ASCII code in hexadecimal	Equivalent in binary
00	" "	20	00100000
00	"h"	68	01101000
01	"e"	65	01100101
02	"l"	6C	01101100
03	"l"	6c	01101100
04	"o"	6f	01101111
05	"!"	21	00100001

Table 10-2 Encoding the string "Hello" in a ROM memory

The ROM architecture proposed in figure 10-68 is a NOR-like circuit [Haraszti]. The upper PMOS transistor precharges the vertical bit lines to VDD. Depending on the address, a high voltage is applied on one word line ( $WL[I]$  in the example), while all other word lines are kept at a low potential. The selected word line turns all programmed

transistors on, which result in a discharge towards VSS, while bit lines with unprogrammed transistors remain on high voltage. The inverters situated on the lower part of the ROM array refresh and invert the bit line information which is sent to the display. Notice that the precharge effect is not simulated at logic level. When the precharge is off, all bit line nodes are considered in 3-state, without any consideration of a "low 3-state" level or "high 3-state" level which is taken into account in industrial logic simulators.

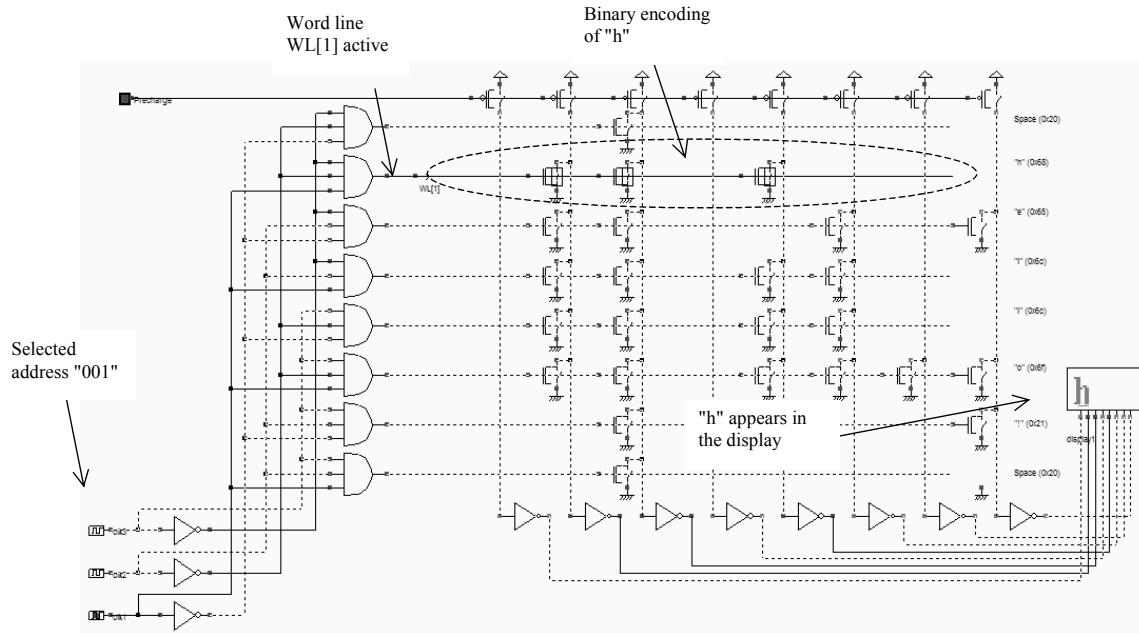


Figure 10-68 Delivering the contents of the ROM memory (Rom8x8.SCH)

In DSCH, the hexadecimal display has been configured to display the character corresponding to the ASCII input information (Figure 10-69).

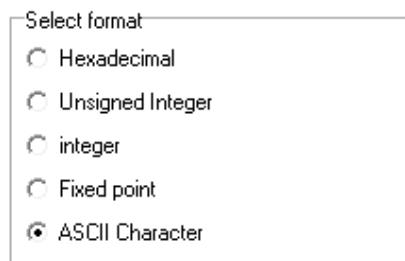


Figure 10-69 The hexadecimal display configured in ASCII mode (Rom8x8.SCH)

### Layout considerations

The basic cells are shown in figure 10-70. The transistor situated on the left ("1") creates a low resistance path between the bit line and the ground when the word line is high. The layout corresponding to "0" is not a transistor, as the diffusion has been removed. The polysilicon ensures the continuity of the word line information, while the metal 2 ensures the continuity of the bit line information.

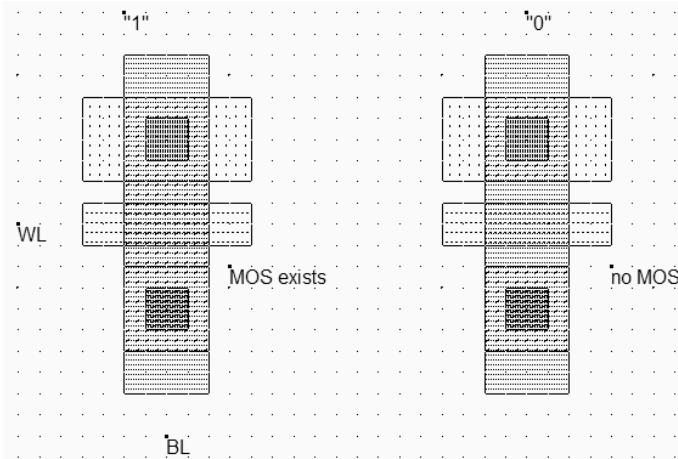


Figure 10-70 The basic patterns of the ROM memory (Rom.MSK)

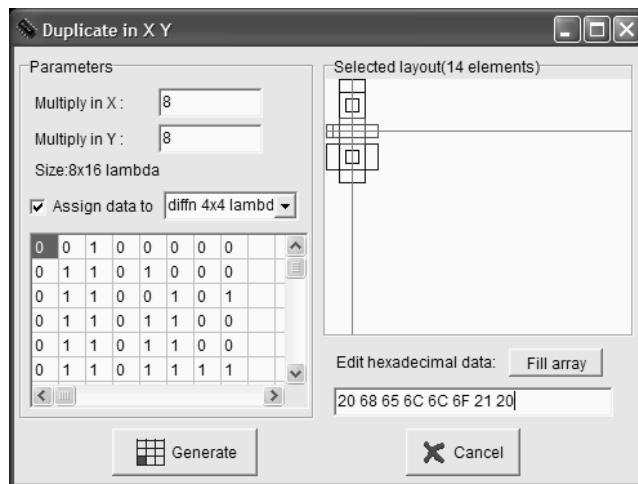


Figure 10-71 Programming the ROM memory during the duplication phase (Rom.MSK)

The duplication in X and Y can be programmed according to a binary information, through the menu shown in figure 10-71 (**Edit → Duplicate X,Y** in Microwind). The first step consists in giving the desired X and Y size, 8 in this example. Secondly, we choose the layout box that will be affected by the logic information. In our case, we select the n-diffusion box used for the channel. The selected layout box is marked by the cross in the layout window. Finally, we enter the desired information in hexadecimal format, and we click **Fill Array** to transform the string of data into elementary binary information. After a click on **Generate**, the regular ROM layout appears as shown in figure 10-72, which contains the binary version of the string "Hello!".

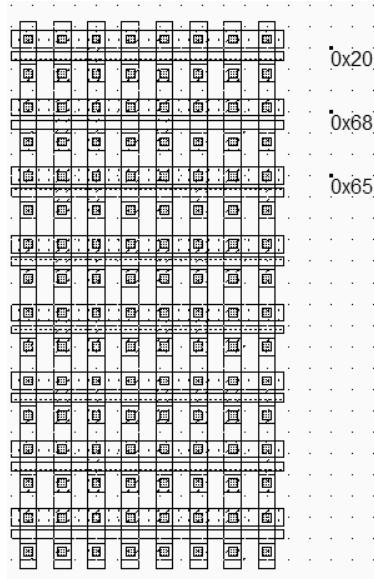


Figure 10-72 The ROM memory stores the string "Hello !" in binary format (Rom.MSK)

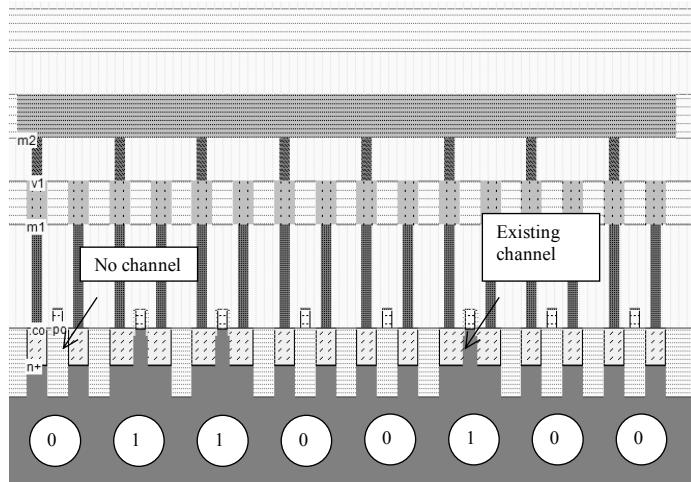


Figure 10-73 The ROM memory cross-section (Rom.MSK)

The cross-section (Figure 10-73) reveals the diffusion programming which creates or not the path to ground. If no channel is fabricated, the memory cell is equal to a zero. When the channel exists, the memory cell is equal to a one.

## 10.6 EEPROM memory

### Double-Gate MOS

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 10-74, 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.

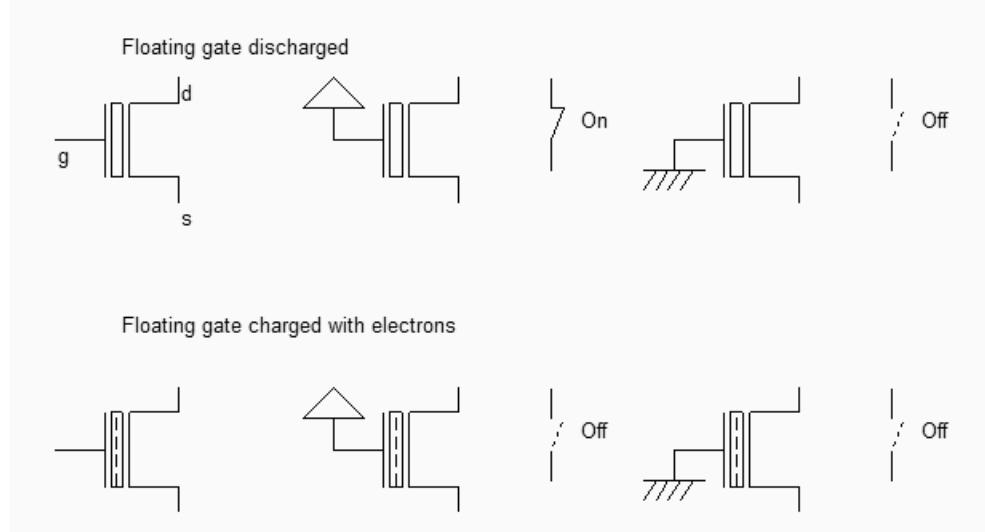


Fig. 10-74: The two states of the double gate MOS (EepromExplain.SCH)

When charges are trapped in the floating polysilicon layer (Figure 10-74, 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 repealing 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.

### Double-gate MOS Layout

The double gate MOS layout is shown in figure 10-75. 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 on at the top. The cross-section of figure 10-76 reveals the stacked *poly/poly2* structure, with a thin oxide in between.

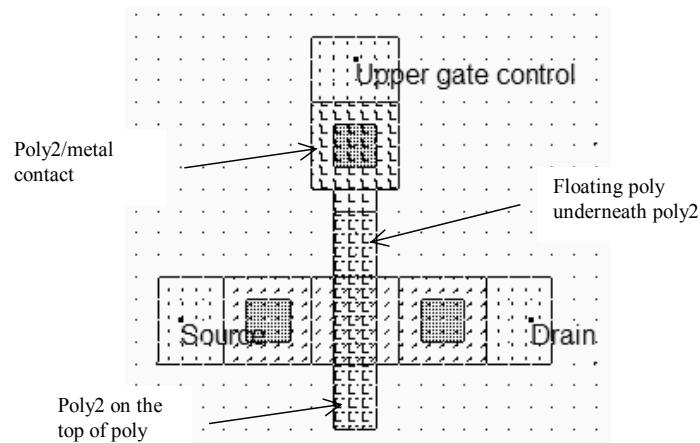
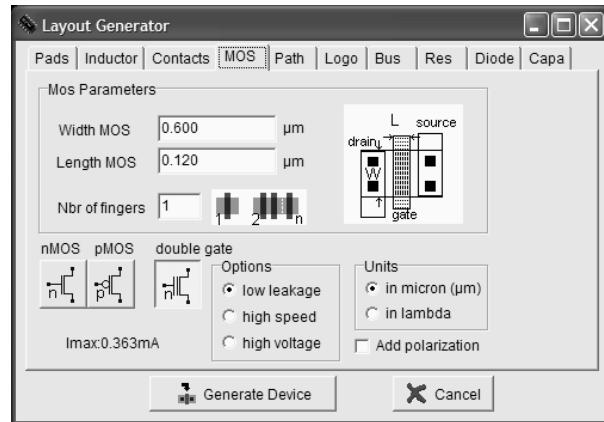


Fig. 10-75: The double gate MOS generated by Microwind (Eeprom.MSK)

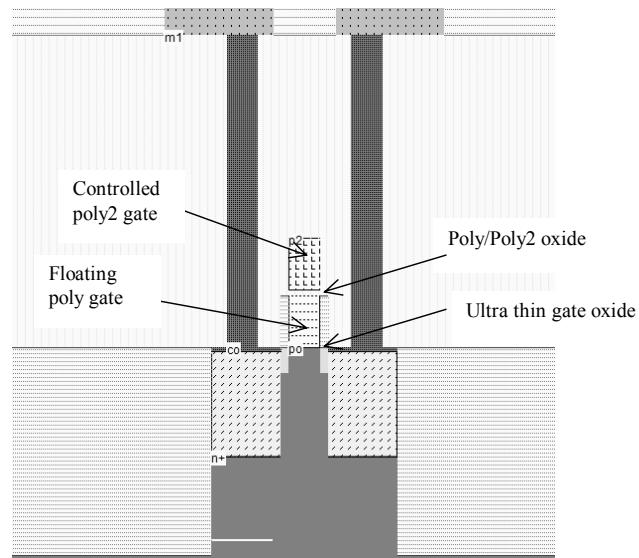


Fig. 10-76: Cross-section of the double gate MOS (Eeprom.MSK)

### 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 10-78). 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.

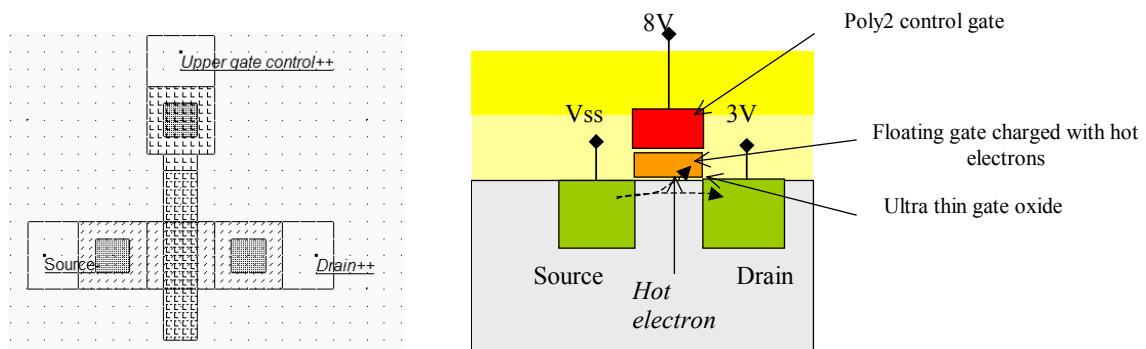


Fig. 10-78: The floating gate is charged with hot electrons thanks to a tunneling effect through the ultra-thin gate oxide (EepromCharge.MSK)

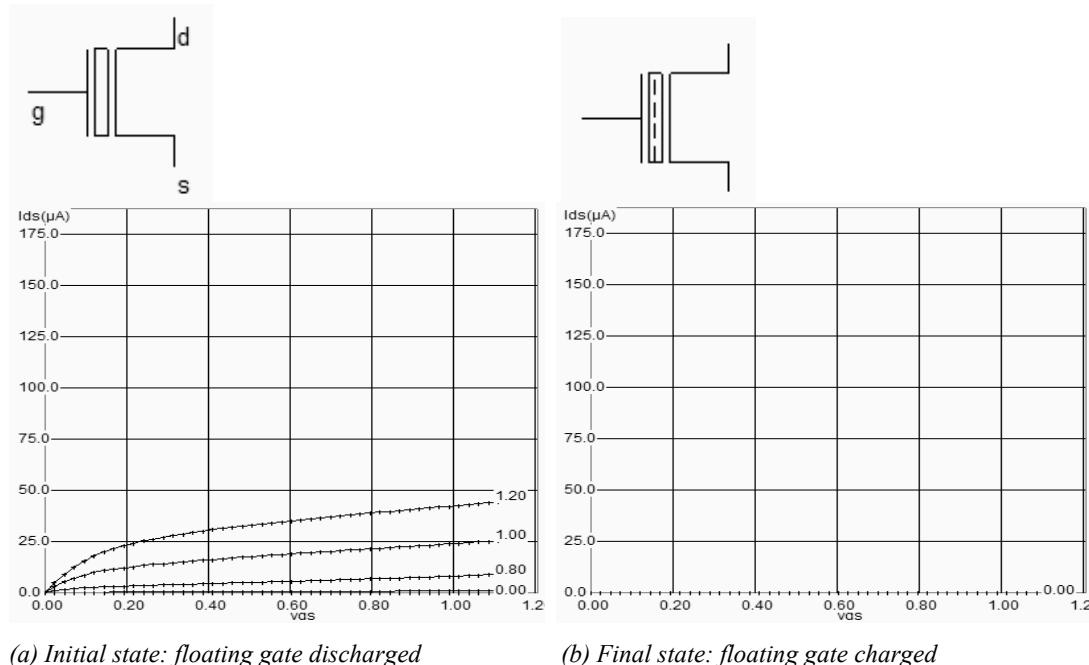


Fig. 10-79: Double-gate MOS characteristics without (a) and with charges (EepromCharge.MSK)

At initialization (Figure 10-79-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 Id/Vd characteristics (Figure 10-79-b).

### 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 (Figure 10-80).

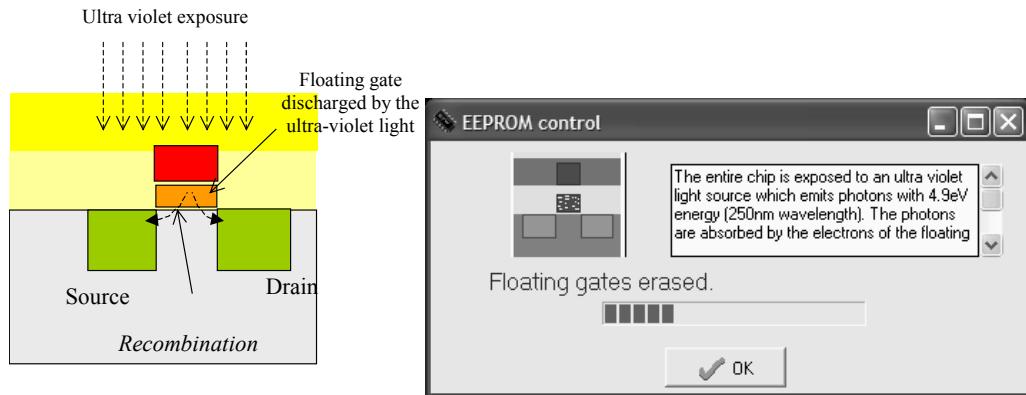


Fig. 10-80: The floating gate may be discharged by U.V light exposure (EepromCharge.MSK)

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. At the end of the simulation (Which takes 10 seconds instead of 20 minutes in reality!), all floating gates of the layout are discharged. Alternatively, the charge contained in the floating gate can be accessed individually using the command **Simulate→Mos characteristics**. On the right lower corner of the device characteristics, a cursor named **Charge** appears, representing the amount of electrons stored in the floating gate. Changing the cursor position (Which corresponds in figure 10-81 to the minimum charge of electrons) modifies dynamically the MOS characteristics.

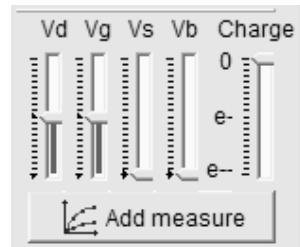


Fig. 10-81. Access to the double gate electron charge (EepromCharge.MSK)

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 10-82).

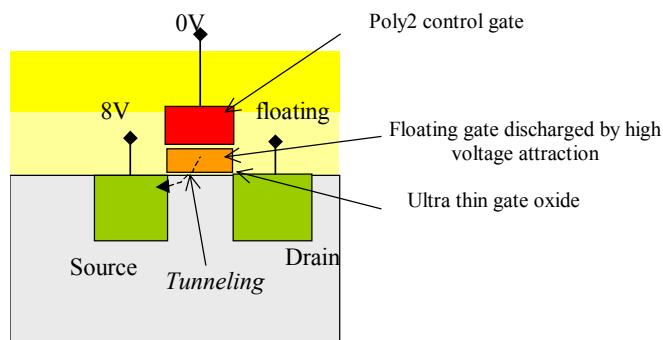


Fig. 10-82. Discharging the double gate MOS device (EepromDischarge.MSK)

### EEPROM Array

EEPROM memories can be programmed electrically bit-by-bit and erased electrically bit-by-bit. EEPROM memory cells are based on one double-poly MOS devices, used for storage, and one n-channel MOS device for row selection (WL), as shown in figure 10-83.

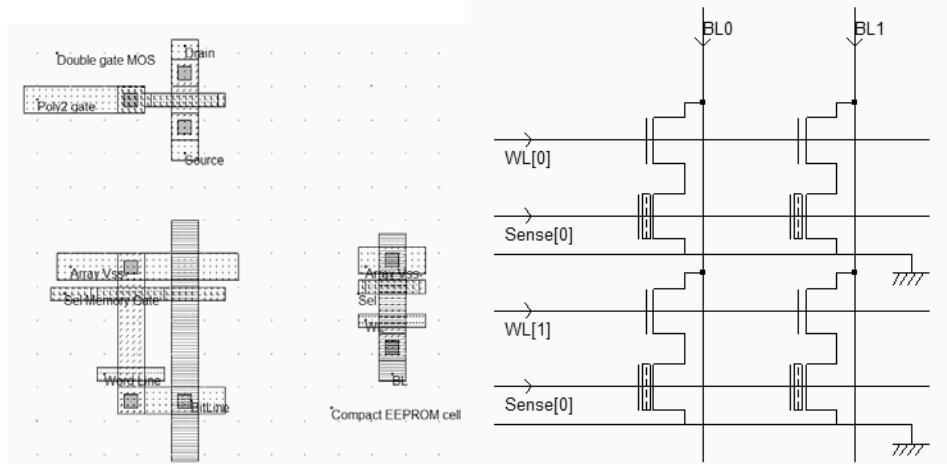


Fig. 10-83. Building an EEPROM memory cell (Eeprom.MSK)

The design requires two MOS devices per memory point, a memory transistor and a select transistor. This results in a large cell size, because of the two devices in series. A compact 2x2 arrangement is proposed in figure 10-83. The two devices are placed as close as possible (The usual design rule is 3 lambda), and the bit line and ground contacts are shared with other cells.

The basic structure for reading the EEPROM information is described in the schematic diagram of figure 10-84. 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 10-84 left, the floating gate has no charge, so *BL* is tied to ground after the precharge, meaning that *DataOut* is 1. 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.

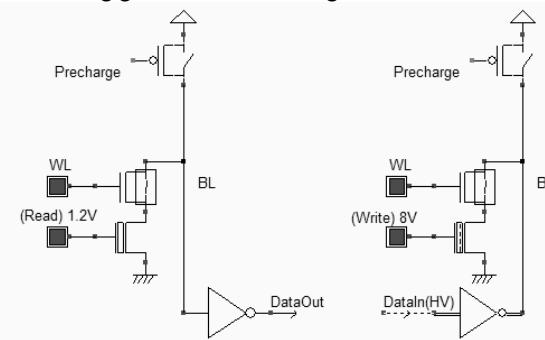


Fig. 10-84 Reading and writing in the EEPROM (Eeprom.MSK)

The 8x8 bit array using this basic EEPROM memory cell is reported in figure 10-84. The need for two devices per memory cell means a large memory array, and a relatively low level of integration.

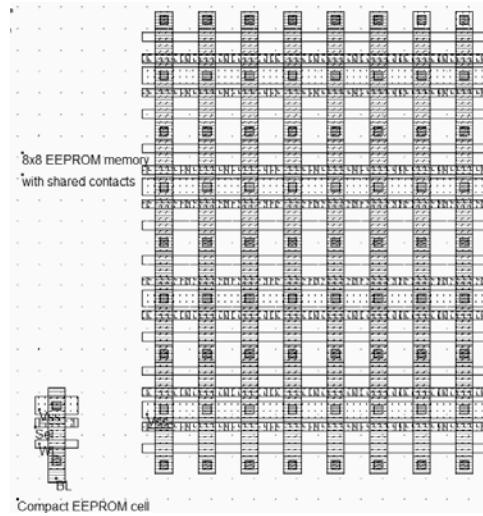


Fig. 10-84. The EEPROM memory array (Eeprom8x8.MSK)

## 10.7 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 10-85). 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.

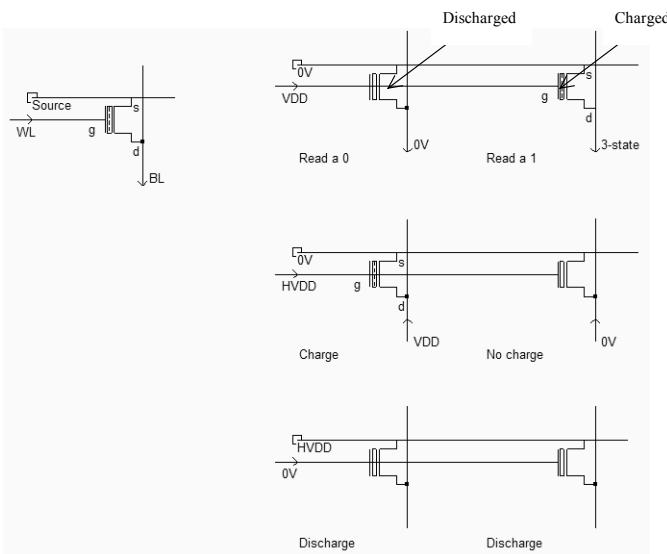


Fig. 10-85. The flash memory point and the principles for charge/discharge (FlashMemory.SCH)

The main characteristics of the Flash memory are given in figure 10-85. Assuming that the floating gate may be charged or discharged, the reading operation consists in applying a VDD voltage on the control gate, and a ground on the source (WL). The bit line drops to 0 if the gate is discharged, or remains in high impedance if the gate is charged. The charge is selective as it depends on the applied information on the vertical bit line: a VDD value provokes a charge injection, while a VSS value disables hot electron effect. The charge effect requires a high voltage HVDD on the control gates. Finally, the discharge is common to all double-gate MOS devices as soon as a high voltage HVDD is applied to the source. This is the main difference with EEPROM cells, where the high voltage was applied or not to the double-gate device, depending on the pass transistor.

### Flash Memory layout

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.

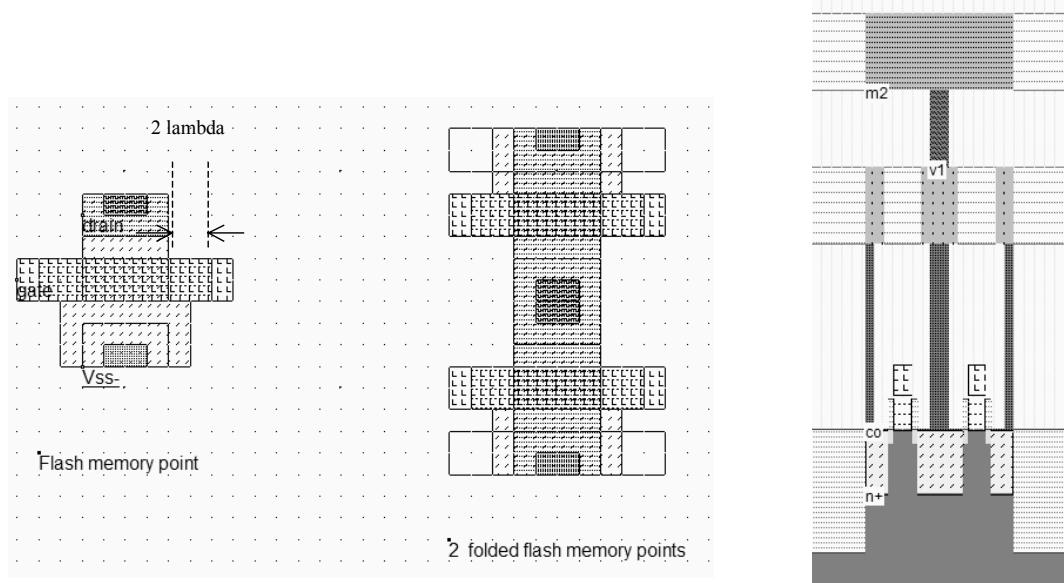


Fig. 10-86. The flash memory point and the associated cross-section (Flash8x8.MSK)

It is not an uncommon practice to violate usual design rules, in order to achieve more compact layout. In the case of figure 10-86, the poly extension is reduced from 3 lambda to 2 lambda. The cell density is increased, but the probability of an erroneous MOS behavior is also increased. Such a compromise between integration and reliability is justified by yield analysis with and without complying to design rules. This kind of information is confidentially kept within the IC company. An example of 8x8 bit Flash memory array is shown in figure 10-87.

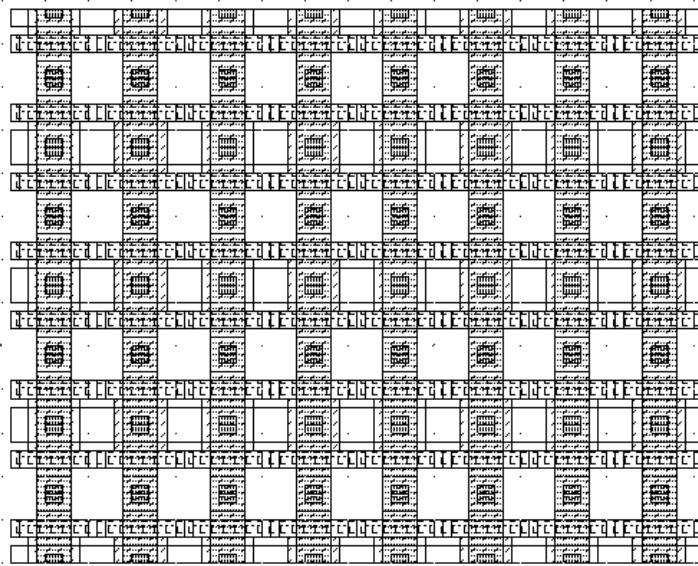
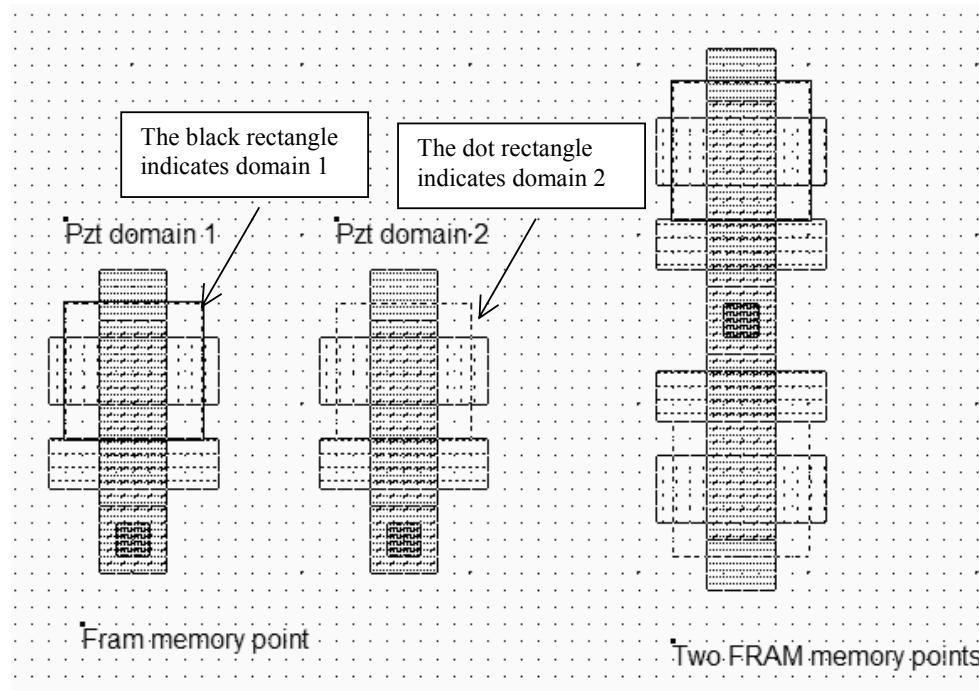


Fig. 10-87. The flash memory bank consisting of 8x8 memory cells (Flash8x8.MSK)

## 10.8 Ferro-electric RAM

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. 10-88. The two domains of the FRAM memory (FramCell.MSK)*

The 2D cross-section (Figure 10-89) 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.

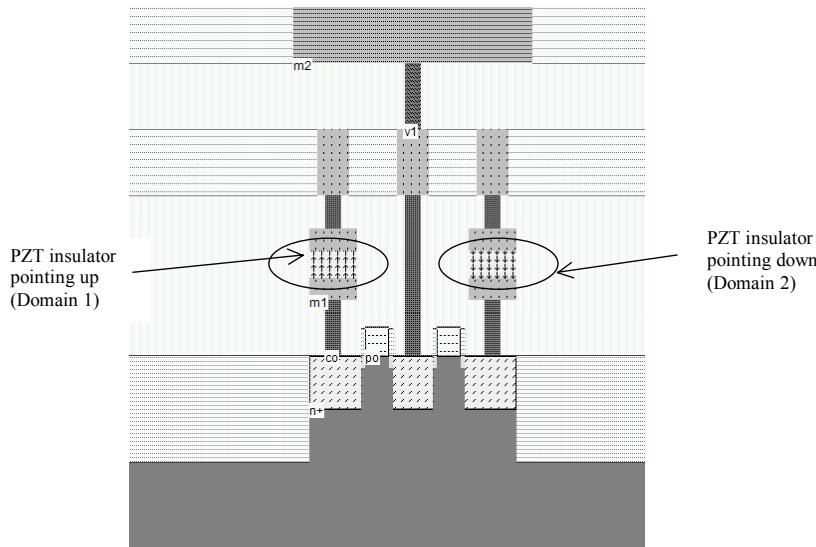


Fig. 10-89. The two domains of the FRAM memory (FramCell.MSK)

We describe here the  $\text{PbTiO}_3$  material, which is a ferroelectric insulator with a dielectric constant higher than 100, and has two stable positions, called *domains*. In the first domain, the molecular dipoles point up, in the second domain, the molecular dipoles point down. This property is handled by Microwind using arrows in the insulator material, as shown in figure 10-89. The  $\text{PbTiO}_3$  insulator is in between two metal electrodes, fabricated in Iridium.

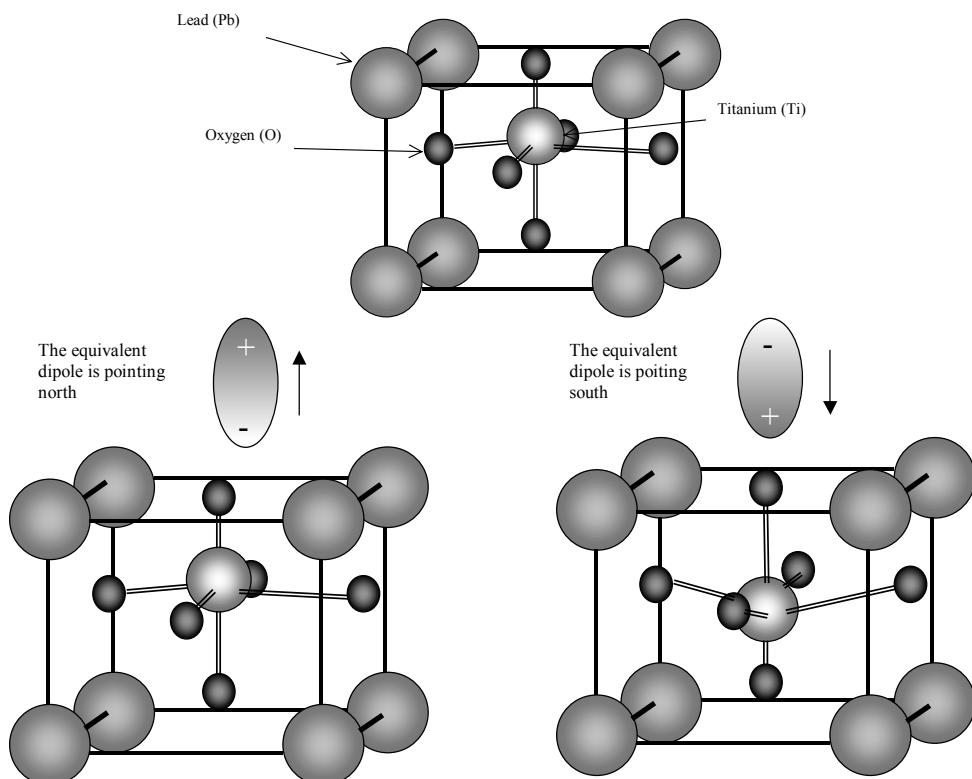


Fig. 10-90. The two domains of the structure which change the orientation of the equivalent dipole

The PbTiO<sub>3</sub> 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 (See the table of elements in chapter 1), with oxygen atoms at its ends, shared with neighbors. The two stable states of the molecule are shown in figure 10-90. 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).

The PZT capacitor behavior is usually represented by the hysteresis curve shown in figure 10-91. 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.

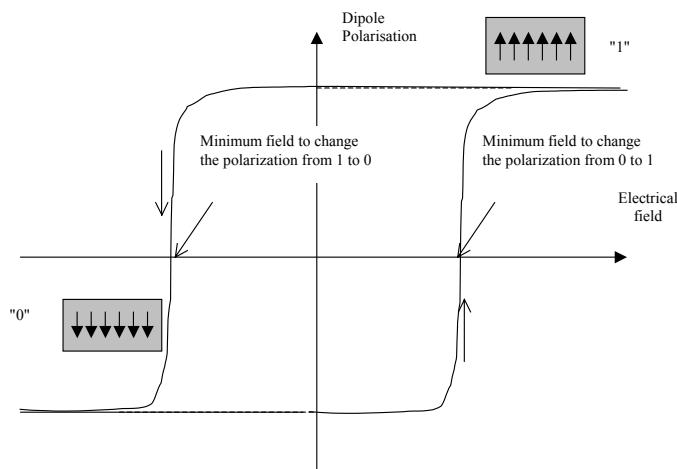


Fig. 10-91. The hysteresis curve of the PZT insulator

Consequently, the write cycle simply consists, for a 1, in applying a large positive step on which orients the dipoles north, and for a zero in applying a negative voltage step, which orients the dipoles south (Figure 10-92).

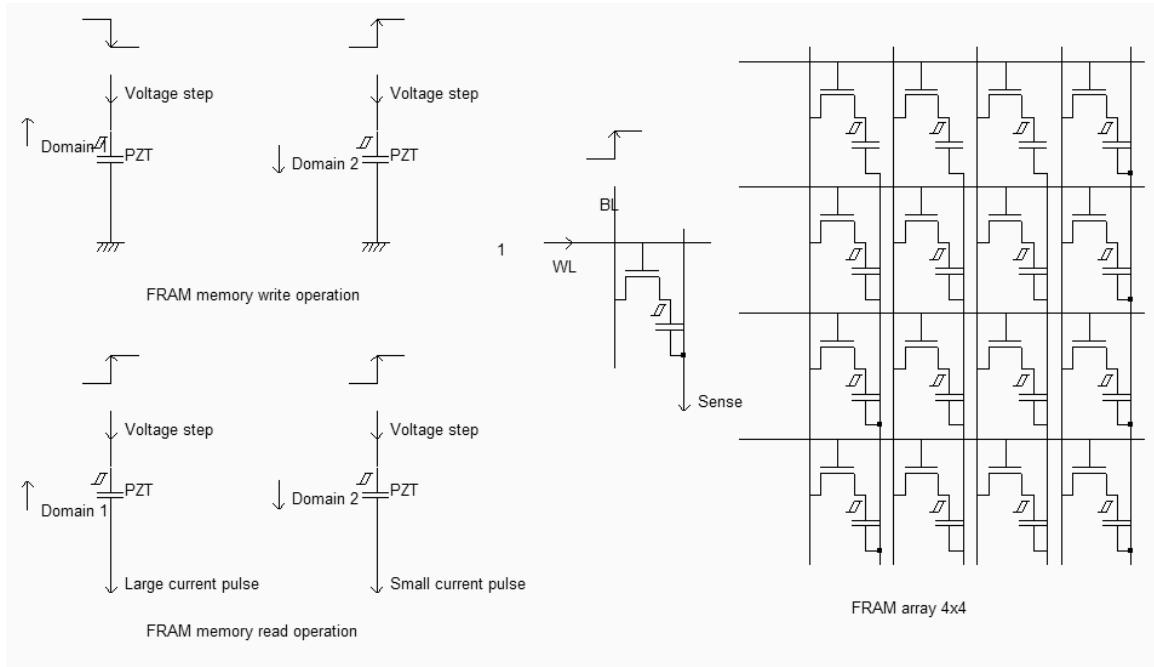


Fig. 10-92. The FRAM circuit principles and architecture (Fram4x4.SCH)

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 (Figure 10-92). 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.

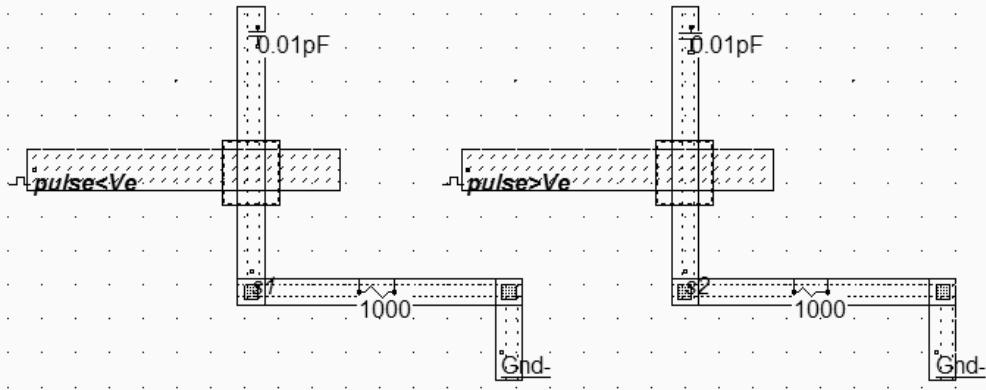


Fig. 10-93. Layout of the two PZT capacitor with initial domains oriented down (FramSimu.MSK)

In figure 10-93, two PZT capacitors are designed at the interface of metal1 and n-diffusion. The programming of the domain may be done by a double click in the option layer, and a tick in the "Embedded FRAM" option. Up-to-down and down-to-up domains are accessible.

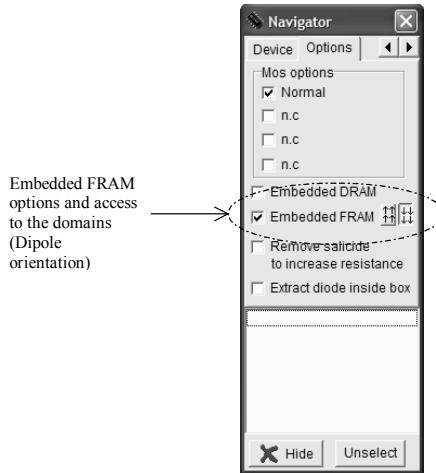


Fig. 10-94. Manual access to the domains of the ferroelectric material used in the FRAM (FramSimu.MSK)

The PZT capacitor domains are oriented down at initialization. A sufficient electrical field, created by a high voltage difference between the diffusion and the metal gives the conditions for domain change (Figure 10-95).

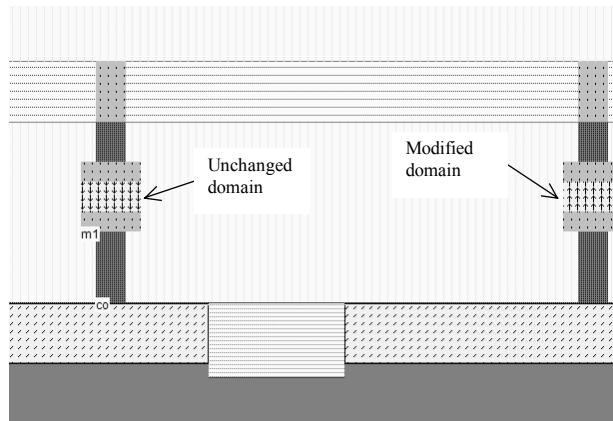


Fig. 10-95. A sufficient voltage provokes the domain change on the PZT capacitor situated on the right side (FramSimu.MSK)

The simulation must be performed using the option **With crosstalk**, which is user-accessible in the simulation menu. Consequently, the PZT capacitor coupling is taken into account. At a sufficient voltage difference  $V_e$  between the two plates ( $V_e$  is 600mV in this case), the domain is changed. This is the case on the right side as the voltage difference reaches almost 1V, but it is not the case on the left side, where the diffusion rises to a lower voltage. The final state is shown in the 2D cross-section of figure 10-96.

The ferroelectric RAM has the potential of becoming the ideal non-volatile memory for a wide range of applications, as it only requires two additional masks (Instead of 4-6 for embedded DRAM with stacked capacitor). The FRAM module has a cell size two to three times smaller than the static RAM memory cell, with the advantage of a zero standby power consumption. However, the read/write cycles are not unlimited, due to fatigue in the PZT material. Heavy read/write cycles could still be supported by SRAM memories.

## 10.9 Classification

A summary of CMOS embedded memory performances is given in table 10-2. The typical memory bank capacity gives an advantage to the ROM, EPROM, EEPROM and Flash memories, because of a small cell area. The reading and writing performances vary very significantly, as well as the retention of data. Dynamic RAM (DRAM) are slow but compact. Static RAM (SRAM) are fast but large. Reading the information from a passive capacitor, such as in DRAM, is much slower than reading the information from the active inverter-based memory such as in SRAM. In contrast, a single trench of stacked capacitor requires much less silicon surface than a 2-inverter memory structure, at the cost of 8 supplementary process steps. The FLASH memories combine a small area, an acceptable reading cycle and interesting non-volatile capabilities, at the price of a slow writing process ( $1\mu s$ ). Promising performances are achieved by ferroelectric RAM (FRAM) which are the most advanced of non-volatile challengers. FRAM have endurance writing/erasing cycles comparable to the best memories, with fast reading and writing cycles, and require only two additional process masks.

Memory type	Typical Capacity	Cell area	Reading	Writing	Cycles	Retention	Process complexity	High voltage
ROM	32Mb	Very small	Medium	Impossible	0	No limit	0	no
EPROM	16Mb	Very small	Slow	Extremely slow	1-10	>30 YEARS	3	yes
E2PROM	1Mb	Large	Slow	Very slow	$1E5 - 1E7$	>10 YEARS	4	no
FLASH	16Mb	Very small	Medium	Very slow	$1E4 - 1E5$	>10 YEARS	4	yes
FRAM	4Mb	Small	Fast	Fast	$1E12 - 1E15$	>10 YEARS	2	No
eDRAM	32Mb	Small	Slow	Fast	$>1E15$	Volatile, needs to refresh	8	no
SRAM	4Mb	Large	Very fast	Very fast	$>1E15$	Volatile	0	No

Table 10-2: A classification of embedded memories according to their performances

## 10.10 Interfacing

The signals used in old-style RAM memory circuits are shown in figure 10-96. The memory is enabled by *Chip Enable* signal (*CE*), usually active low. When *CE* is high, the memory is in stand-by mode, usually consuming the minimum possible power. When active (*CE=0*), the memory is controlled by the signals *Read/write (WE)*, *Row Address Selection (RAS)* and *Column Address selection (CAS)*. The output is enabled on the n-bit Data bus thanks to *Output Enable (OE)*.

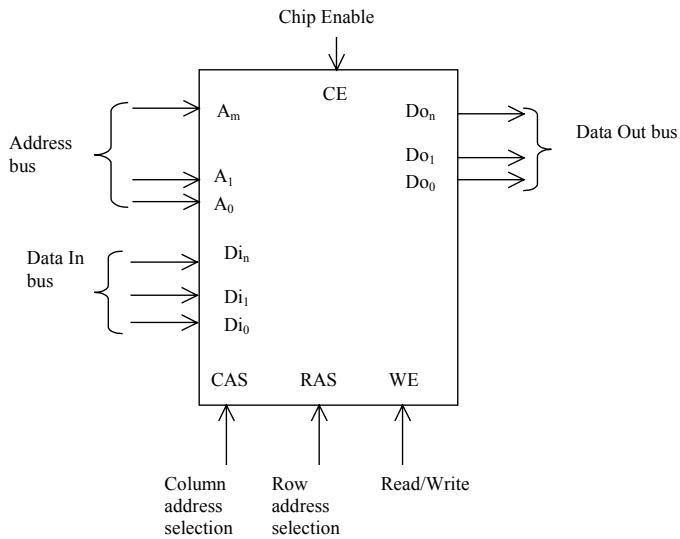


Figure 10-96: Typical RAM interfacing

#### Read & Write Cycles

The operating conditions of RAM memories involve a specific timing of control signal and data, as shown in figure 10-97. We consider here an address bus where the row and column address information is available with a time domain multiplexing. The row address selection (RAS) and Column address selection (CAS) provide the demultiplexor signals for the address bus. The total read cycle duration is  $t_{RC}$ . The delay between the active edge of the row address selection (RAS) and the valid data out is called the row access cycle  $t_{RAC}$ .

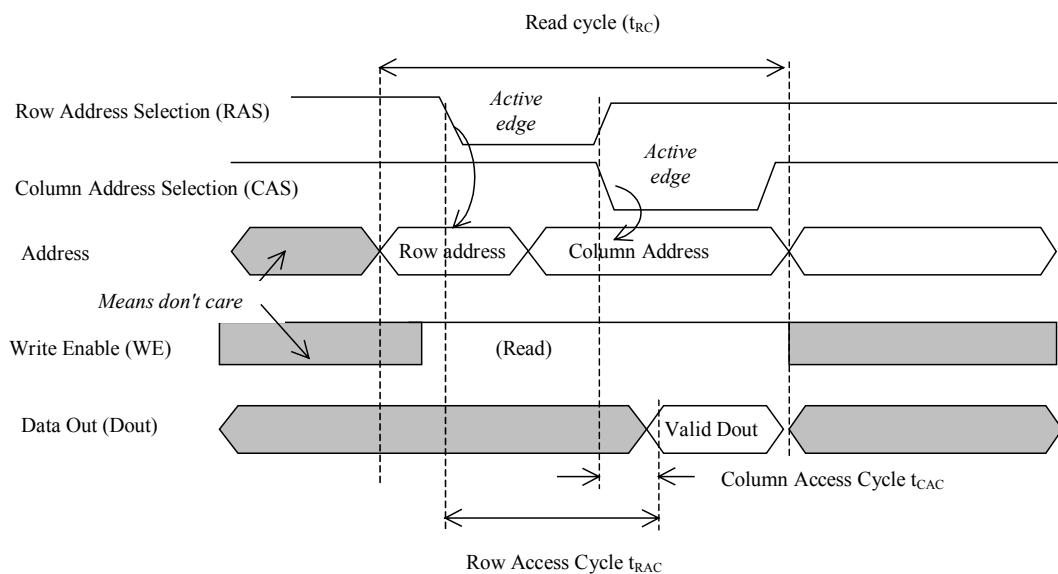


Figure 10-97: RAM timing parameters (Read cycle)

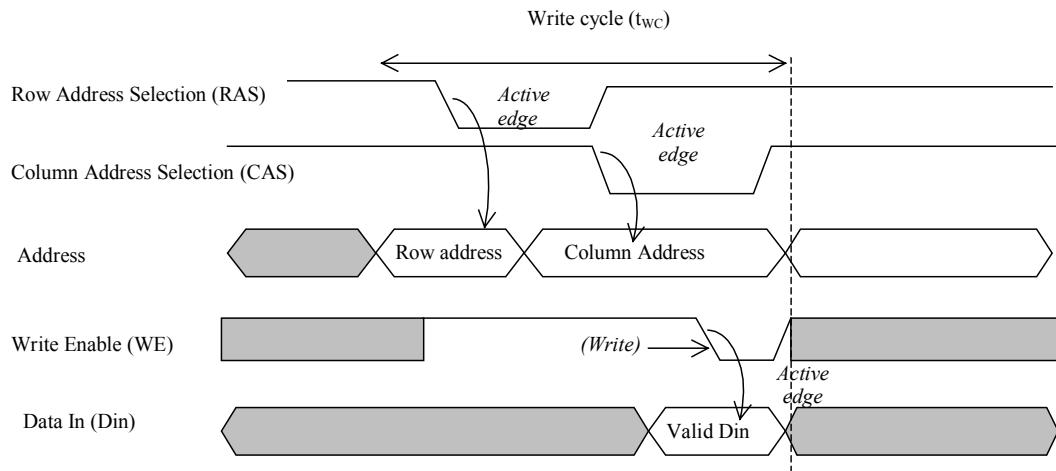


Figure 10-98: RAM timing parameters (Write cycle)

The total write cycle duration is  $t_{WC}$  (Figure 10-98). The switching performances differ considerably whether the memory is a stand-alone integrated circuit or an embedded block. Typically the read and write cycle time is 10-20ns for a stand-alone product, but around 2-5 ns for an embedded memory. This very important gain in performance is due to the short and low-capacitance interconnection between the processor and the memory, with very simple and fast block interfacing, as compared to the connection between one die and an other die through the packaging and printed circuit board.

### Synchronous RAM

The synchronous RAM differs from the conventional dynamic RAM in two main points [Sharma]: all inputs and outputs 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 10-99. 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 10-99. 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.

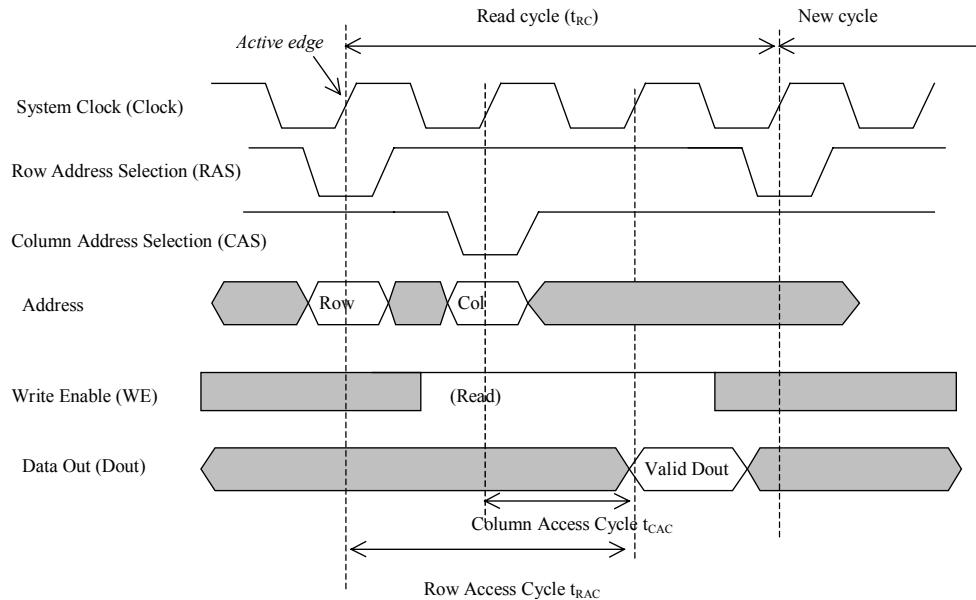


Figure 10-99: Synchronous RAM timing diagram

### Double Data Rate

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 in series 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 10-100. Notice that *Data In* and *Data Out* work almost independently.

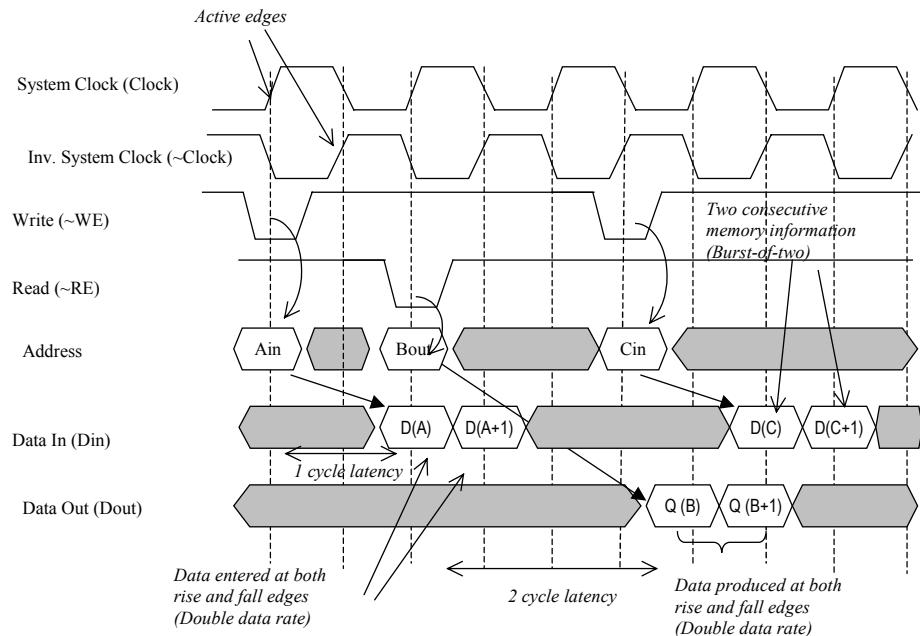


Figure 10-100: Illustration of the double data rate, and the use of burst-of-two memory

## 10.11 Conclusion

This chapter has focused on memories, which are a very important part of modern integrated circuits. The static memory has been described first, with an illustration of design challenges and dangers in the case of five-transistor architecture. The 6-transistor cell has been presented together with some layout optimization techniques to achieve the most compact memory design. The column and row selection circuits have been rapidly described, as well as the sense amplifiers used to speed up the read cycle. The principles and technological challenges related to the dynamic RAM have been introduced, with focus on the embedded stacked capacitor. The ROM memory has also been introduced. Extensive details on the EEPROM memory and its FLASH derivative have been provided in this chapter. Finally, we have introduced the principles and layout implementation of the ferro-electric RAM memory (FRAM), and concluded by a description of the asynchronous and synchronous memory interfaces.

## REFERENCES

- [Sharma] A. K. Sharma "Semiconductor Memories", IEEE Press, 1997, ISBN 0-7803-1000-4
- [Kalter] Kalter Y. "Embedded DRAM, technological opportunities and challenges", IEEE spectrum, April 99, pp56-64
- [Haraszti] Tegze P. Haraszti "CMOS memory Circuits", Kluwer Academic Publishers, 2001, ISBN 0-7923-7950-0
- [Geppert] Linda Geppert "The New Indelible Memories", IEEE spectrum, March 2003, page 49
- [Song] Yoon-Jong Song "Ferroelectric Thin Films for High desnsity Non vonlatile Memories", PhD dissertation, 1998, Digital Library and Archives, <http://scholar.lib.vt.edu/theses>

## EXERCISES

### Exercise 10-1

Compare the leakage current on a DRAM cell for the following technologies : 0.35 $\mu$ m, 0.12 $\mu$ m and 90nm.

### Exercise 10-2

Given a 4x4 EEPROM memory array, create the chronograms to write the words 0001, 0010, 0100 and 1000, and then to read these values.

### Exercise 10-3

The dual-port RAM is used as a memory shared by two blocs. The first bloc is the microprocessor unit which addresses the contents of the memory for write and read operations. The second bloc is usually a video memory or a slave processor which can only access the memory for reading. The basic dual-port cell is proposed in figure 10-101. Design and implement a 4x4 dual port RAM and perform the asynchronous reading of the DRAM contents by the slave memory circuit, while writing on the master memory circuit.

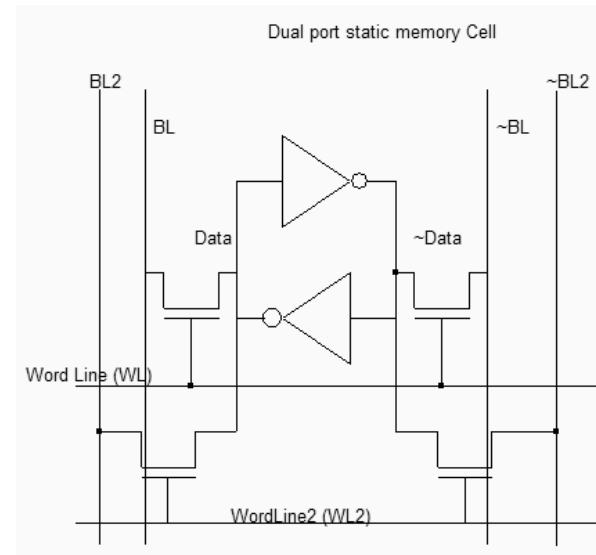


Figure 10-100: Illustration of the dual port SRAM cell

#### Exercise 10-4

Modify the ROM array to write the word "Welcome".

#### Exercise 10-5

Build a 4x4 MRAM array with complete read/write circuits.

# 11 Analog Cells

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

## 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  $\Omega/\square$ . The default value polysilicon resistance per square is  $10\Omega$ , which is quite small, but rises to  $200\Omega$  if the salicide material is removed (Figure 11-1).

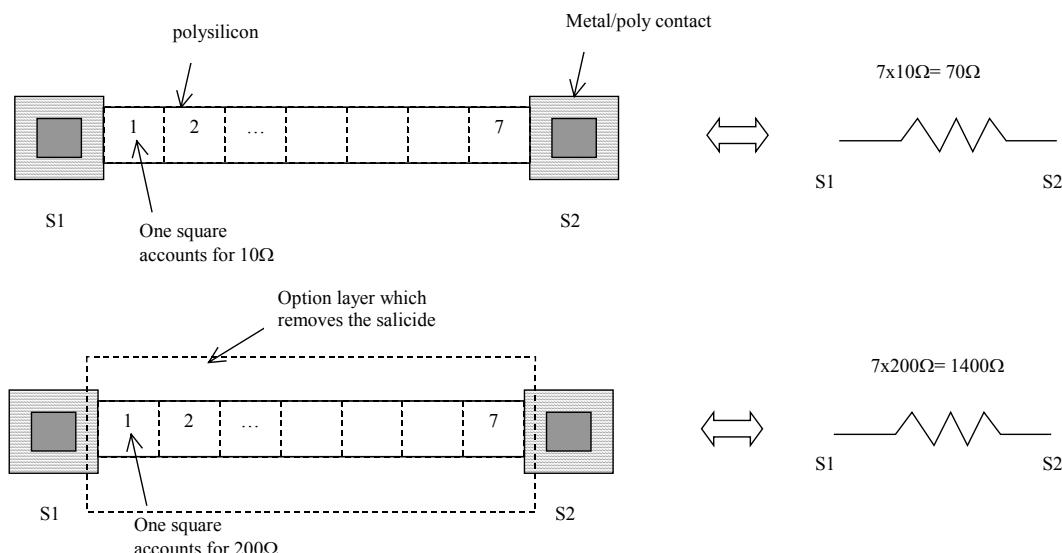


Figure 11-1 : The polysilicon resistance with unsalicide option

In the cross-section shown in figure 11-2, the salicide <Gloss> material deposited on the upper interface between polysilicon and the oxide creates a metal path for current that reduced 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.

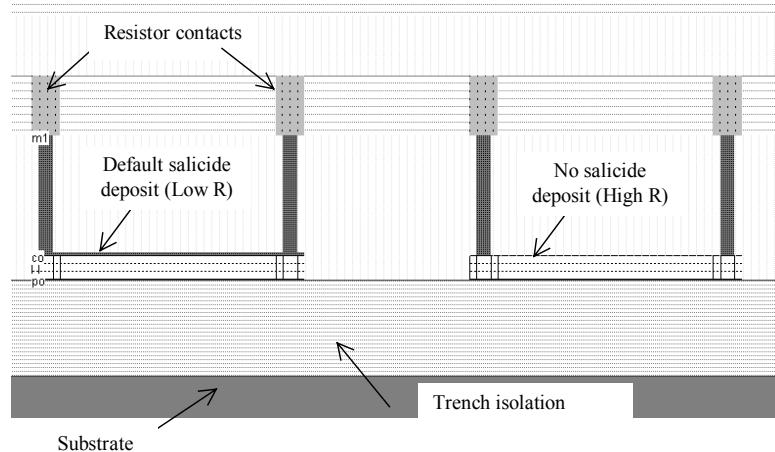


Figure 11-2 : Removing the salicide material to increase the sheet resistance (ResPoly.MSK)

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 11-3, the default resistance is  $76\Omega$ , and the unsalicide resistance rises to  $760\Omega$ .

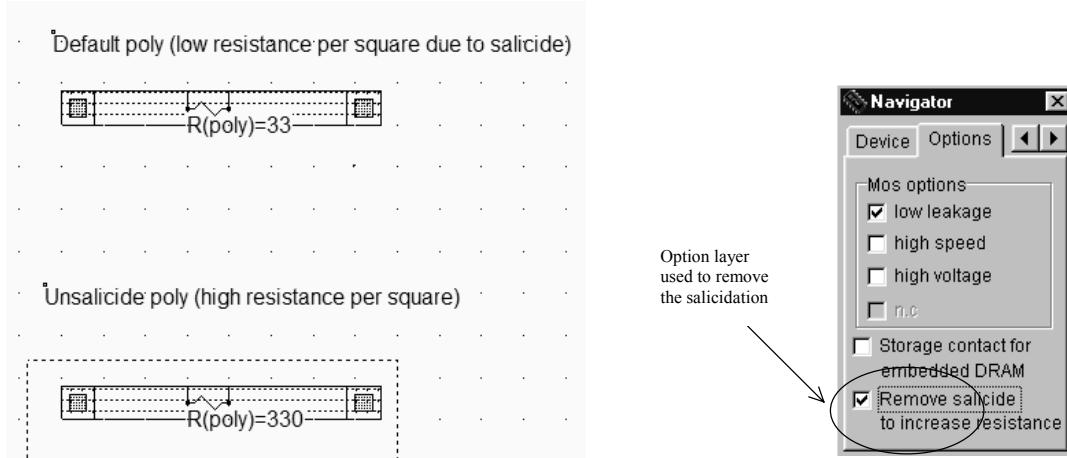


Figure 11-3 : Removing the salicide material thanks to an option layer

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. As illustrated in figure 11-4, when V1 goes below 0V, the P-substrate/N+ diode is turned on and a path is created to the ground. Remember that the P-substrate is usually considered as a common ground reference. The diffusion resistor is used in input/output protection devices which are described in chapter 13.

The command **Help → Design Rules** gives access to the square resistance and unsalicide square resistance of all materials, as reported in figure 11-4.

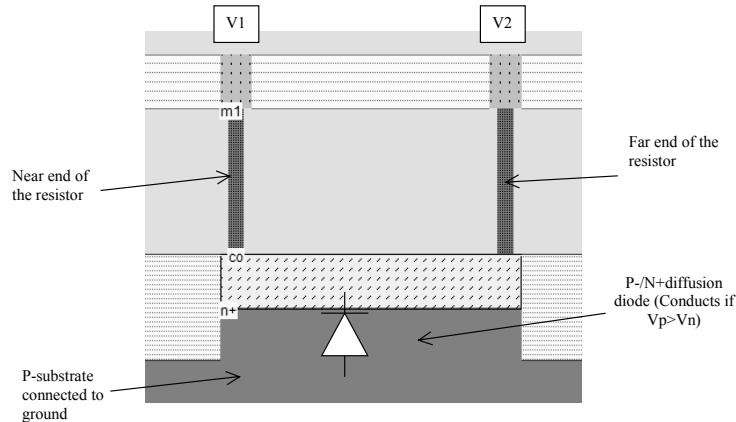


Figure 11-4 : The diffusion resistor combines a resistance effect and a diode (ResDiff.MSK)

Design rules for CMOS 0.12µm - 6 Metal											
Design rules and electrical parameters											
Layer	Width	Spacing	Surface	Surf capa	Lin capa	Ctk capa	Res	Unsalicid	Thickn	Height	Permit
via3	2	4	0				2.00/via		0.50	3.30	4.00
metal3	3	4	16	160.00		30.00	0.05/sq	1.00/sq	0.40	2.90	3.10
via2	2	4	0				2.00/via		0.50	2.40	4.00
metal2	3	4	16	180.00		30.00	0.05/sq	1.00/sq	0.40	2.00	3.10
via	2	4	0				2.00/via		0.50	1.50	4.00
metal	3	4	16	200.00		30.00	0.05/sq	1.00/sq	0.40	1.10	3.10
poly	2	3	16	Default resistance per square			4.00/sq	40.00/sq	0.20	Resistance if unsalicide	
poly2	2	2	8				4.00/sq	40.00/sq	0.20		
contact	2	4	0				20.00/via		1.10	1.00	4.00
diffn	4	4	16	350.00	100.00		25.00/sq	250.00/sq	0.40	0.00	4.00
diffp	4	4	16	300.00	100.00		30.00/sq	300.00/sq	0.40	0.00	4.00
nwell	10	11	144	250.00			120.00/sq		1.00	0.00	4.00

Figure 11-5 : Resistance parameters in CMOS 0.12µm

## Process Variations

The process variations have a strong impact on the physical value of the resistor. Most processes specify square resistance within +/-25% [Hastings]. This means that the value of the resistor is linked to a statistical distribution, usually between a min-max range, and not an exact value. In reality, the spread of resistance is usually less than 10% within a single integrated circuit. However, two different integrated circuits may produce a significantly different resistance distribution. In figure 11-6, the average resistance  $R1$  measured on a test chip n°1 is 5% higher than the target resistance  $R_{typ}$ , and the average resistance  $R2$  on a test chip n°2 is 10% lower than the typical value  $R_{typ}$ . In his process specifications, the integrated circuit manufacturer only warranties that the measured resistance will not be larger than +/-25% the typical value  $R_{typ}$ .

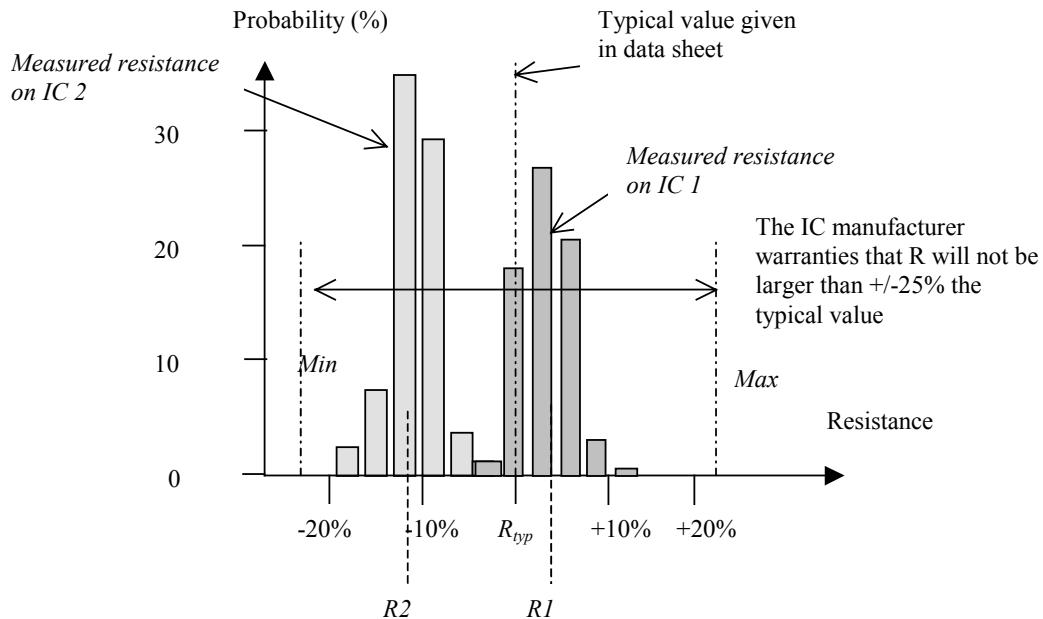


Figure 11-6: The spread in resistance value and the typ/min/max specifications

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 11-7 left). Polysilicon resistors are rarely designed with the minimum 2 lambda width, but rather 4 or 6 lambda, so that the impact of the width variations is smaller. But the equivalent resistance is smaller, meaning less silicon efficiency. As illustrated in figure 11-7, a variation  $\Delta W$  of 0.2 $\lambda$  on both edges results in a 20% variation of the resistance on a 2 $\lambda$  width resistor, but only a 10% variation for a larger resistor designed with a width of 4 $\lambda$ .

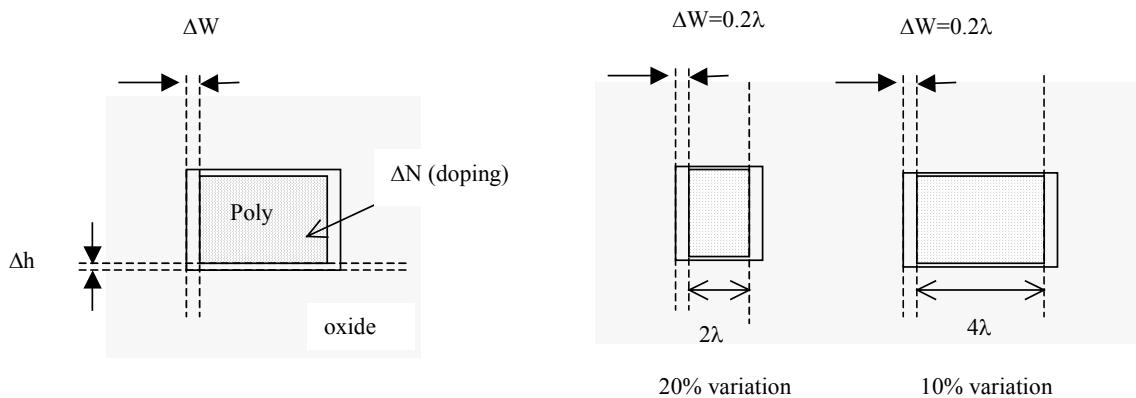


Figure 11-7 : Resistance variations with the process

## Resistor design

There exist efficient techniques to reduce the resistance variation within the same chip. In figure 11-8, 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 (Figure 11-8) 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.

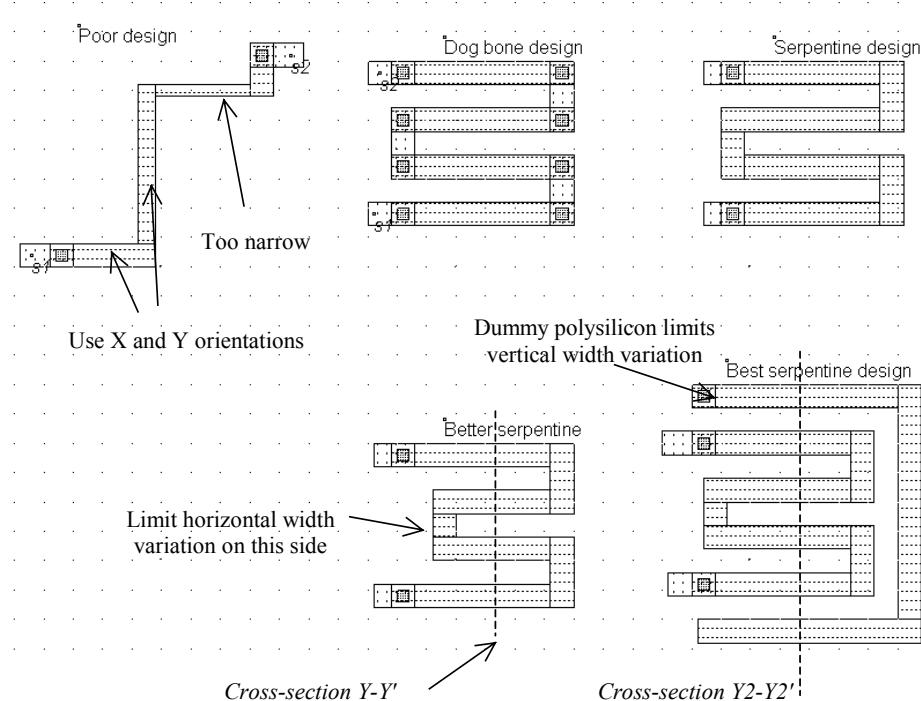


Figure 11-8 : Resistance design (ResPoly.MSK)

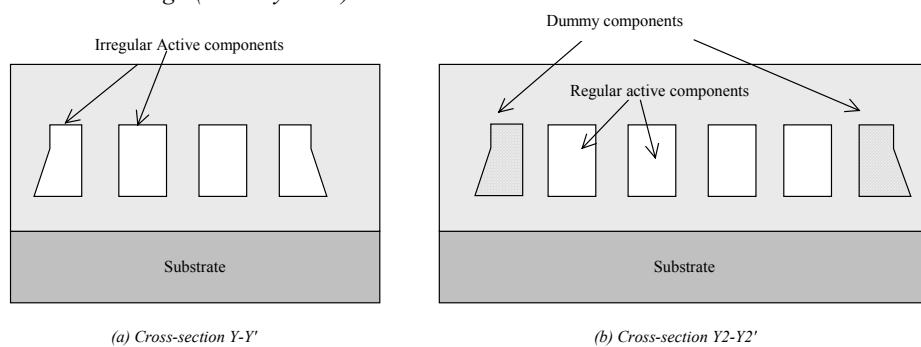


Figure 11-9: 2D aspect of the circuit without and with dummy components

The use of dummy devices is an efficient technique to avoid irregularities in resistor elements. In the case of a polysilicon serpentine without dummy devices, the cross-section Y-Y' in figure 11-9 exhibits important irregularities between bones. This results in an important process dependence and resistance variation. Now, if dummy components are inserted at the bottom and top of the design (Cross-section Y2-Y2'), the irregularities impact the dummy bones, but not the active parts of the resistor, which is much less impacted by the process variations. Typically, the resistance variation between two identical designs within the same chip is around 5% without dummy devices, and less than 1% with dummy devices.

## 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. We describe in this paragraph the diode capacitor, the MOS capacitor, the poly-poly2 and inter-metal capacitor. Some of these capacitors exist between an active node and a fixed voltage.

### 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 0.12 $\mu$ m, the capacitance is around 300aF/ $\mu$ m<sup>2</sup> (1 atto-Farad is equal to 10<sup>-18</sup> Farad).

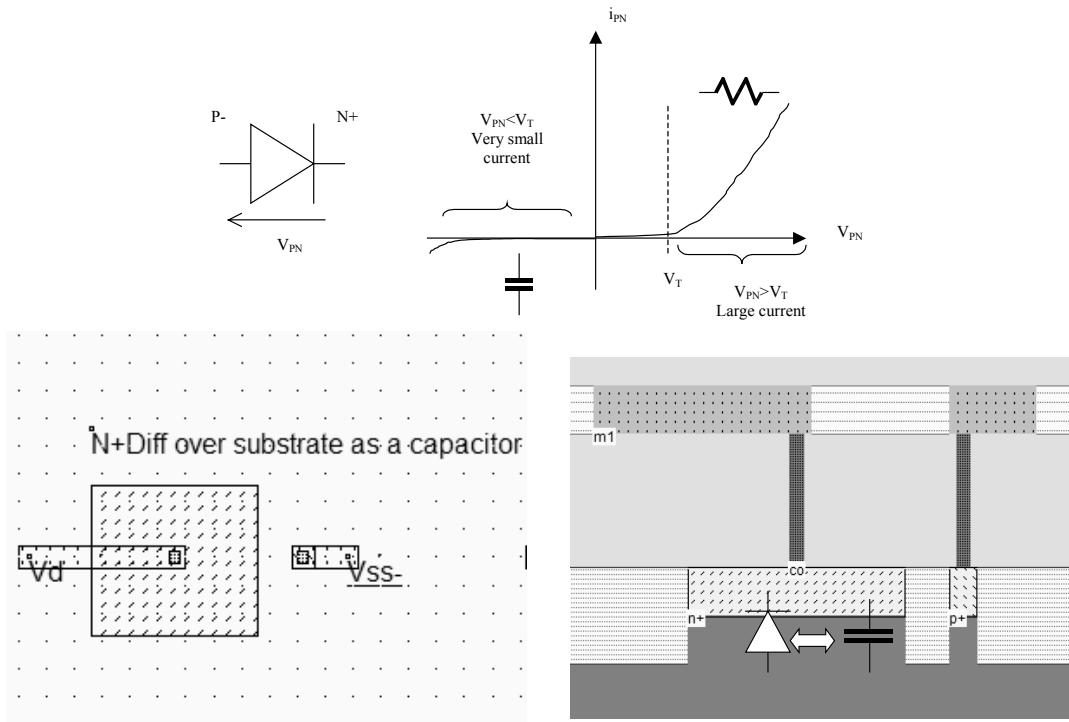


Figure 11-10: The diffusion over substrate as a non-linear capacitor (Capa.MSK)

Let us recall briefly the diode operation. A strong current flows from the P to the N region when the voltage difference is significantly higher than the threshold voltage  $V_T$ . For example, the P-/N+ diode is in such conditions when  $V_P$  is significantly higher than  $V_N + V_T$ , where  $V_T$  is approximately 0.3V in 0.12 $\mu\text{m}$ . The current is limited by the serial resistance of the diode, which is the order of 100 ohm to 1000ohm, depending on the surface of the diode (Figure 11-10). A very small current flows between the P and the N region as long as  $V_P < V_N + V_T$ . In such conditions, the diode may be considered as a capacitance. The order of the parasitic current is between the nano-ampere and the pico-ampere ( $10^{-9}$  to  $10^{-12}$  A). As the substrate is grounded,  $V_N$  is always higher than  $V_P$  meaning that the N+/P- combination is equivalent to a capacitor.

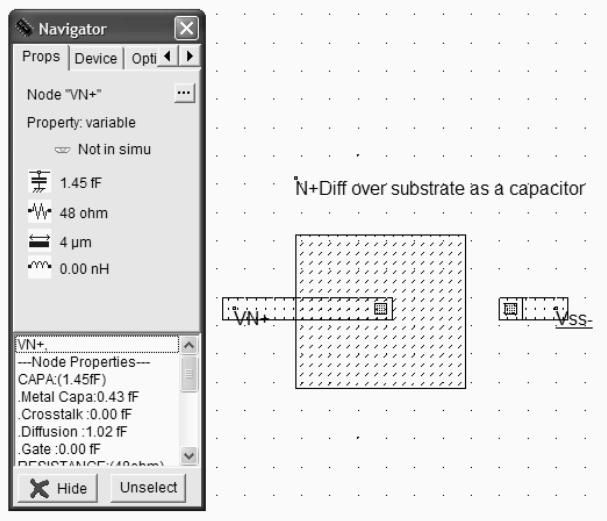


Figure 11-11: Extraction of the diffusion capacitance (Capa.MSK)

The capacitance may be extracted by a double click in the N+ diffusion area, or by the icon **View Electrical Node**. The N+diffusion has an equivalent parasitic capacitance of around 1fF, as extracted in figure 11-11. Notice that the value of this capacitance is an average value, computed for  $V_N$  equal to VDD/2. This capacitor suffers two main drawbacks: first, the capacitance is small as compared to the silicon area. Secondly, the capacitance depends significantly on the value of  $V_N$ , with a non-linear law.

The typical variation of the capacitance with the diffusion voltage  $V_N$  is given in figure 11-12. The capacitance per  $\mu\text{m}^2$  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.

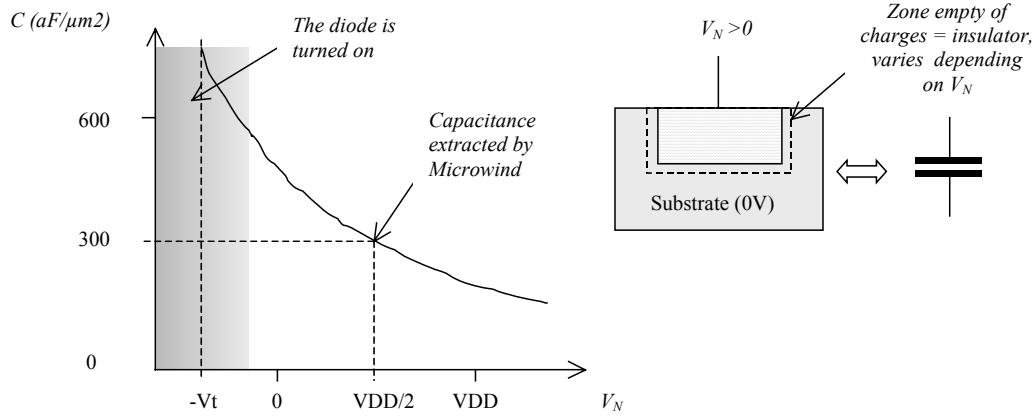


Figure 11-12: The diffusion capacitance varies with the polarization voltage

### 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 11-1. There is also a thicker oxide used for high voltage MOS devices, called dual-oxide (5nm or 50Å in CMOS 0.12μm), for which equation 11-2 is applicable. The parameter in the design rules used to configure the gate oxide capacitor is CPOOxide.

$$C_{\text{thinox}} = \frac{\epsilon_0 \epsilon_r}{e} = \frac{8.85 e^{-12} \times 3.9}{2.0 \times 10^{-9}} = 17 e^{-3} F/m^2 = 17 e^{-15} F/\mu m^2 = 17 fF/\mu m^2 \quad (\text{Eq. 11-1})$$

where

e= gate oxide thickness (m)

$\epsilon_0$ =vacuum permittivity (F/m)

$\epsilon_r$ =relative permittivity (no unit)

$$C_{\text{dualox}} = \frac{\epsilon_0 \epsilon_r}{e^2} = \frac{8.85 e^{-12} \times 3.9}{5.0 \times 10^{-9}} = 6.8 fF/\mu m^2 \quad (\text{Eq. 11-2})$$

where

e<sub>2</sub>= dual-oxide thickness (m)

The design of a gate capacitor using a large MOS device is shown in figure 11-13, 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.

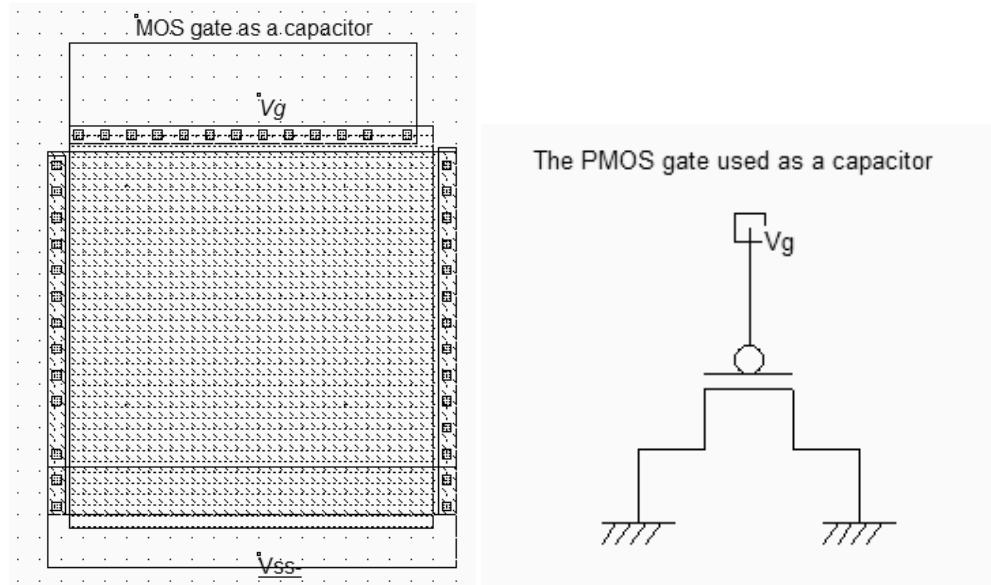


Figure 11-13: Generating an efficient capacitor based on a MOS device with a very large length and width (CapaPoly.MSK)

Using the ultra-thin gate increases the capacitance, however the risk of oxide damage due to overstress is significantly increased. The dielectric strength of the SiO<sub>2</sub> oxide is the critical field above which the electric field damages the dielectric material. The dielectric strength  $E_{crit}$  ranges from 4 to 10MV/cm, depending on the fabrication technique [Hasting]. The critical voltage  $V_{crit}$  above which the oxide is damaged is approximated by equation 11-3. Be careful with the unusual units: the oxide thickness is directly expressed in nanometers and the dielectric strength in MV/cm.

$$V_{crit} = 0,1 \cdot e \cdot E_{crit} \quad (\text{Eq. 11-3})$$

where

$e$ = equivalent thickness (nm)

$E_{crit}$ = dielectric strength (MV/cm)

A SiO<sub>2</sub> gate dielectric of 2nm leads to a critical voltage ranging between 0.8V (Dry oxide growth) and 2V (High quality gate oxide deposit). Long term reliability requires the supply voltage to keep below half of this critical voltage. This is why VDD is around 1V, and a voltage stress significantly above VDD may lead to the gate oxide destruction. The gate oxide used in high voltage MOS devices is fitted with the user requirements: to handle 3.3V, a reliable value for the gate oxide is 7nm. To handle 5V, the minimum gate oxide rises to 10nm.

### 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  $1,7\text{fF}/\mu\text{m}^2$  (equation 11-3).

In Microwind, the command **Edit → Generate → Capacitor** gives access to a specific menu for generating capacitor (Figure 11-14). The parameter in the design rules used to configure the poly-poly2 capacitor is CP2PO.

$$C_{PolyPoly2} = \frac{\epsilon_0 \epsilon_r}{e_{pp}} = \frac{8.85 e^{-12} \times 3.9}{20 \times 10^{-9}} = 1700 aF/\mu m^2 \quad (\text{Eq. 11-1})$$

where

$e_{pp}$ = distance between poly1 and poly2 (m)

$\epsilon_0$ =vacuum permittivity (F/m)

$\epsilon_r$ =relative permittivity (no unit)

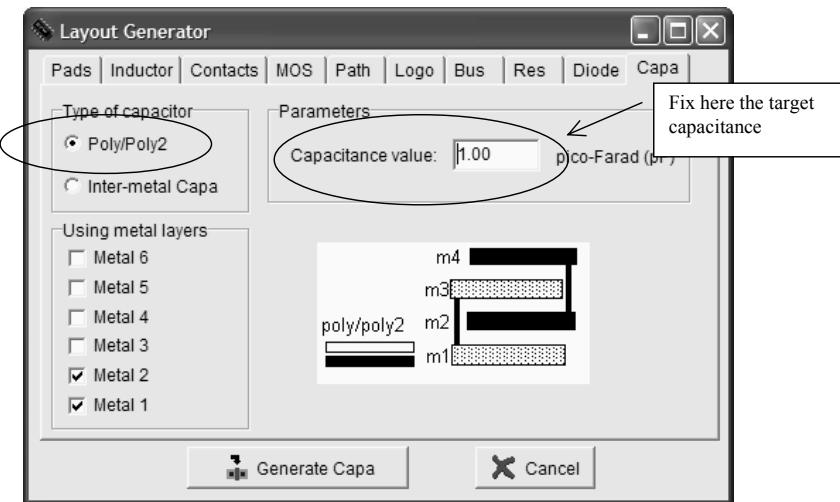


Figure 11-14: The generator menu handles the design of poly/poly2 capacitor and inter-metal capacitors

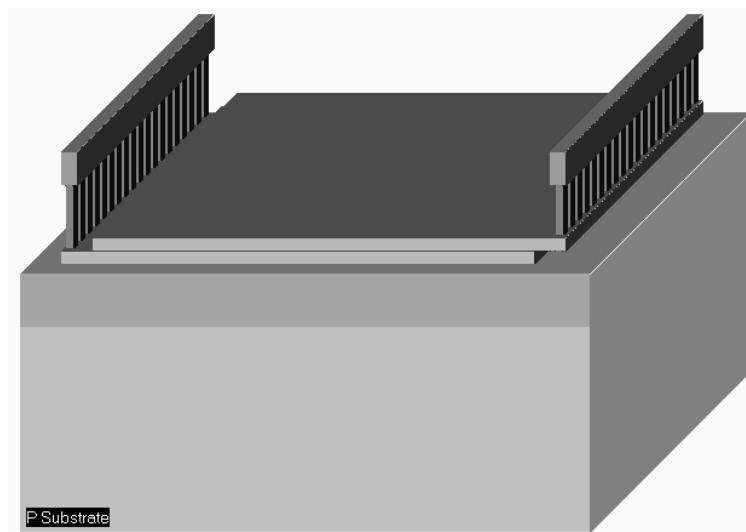


Figure 11-14: The 3D aspect of the poly-poly2 capacitor (CapaPolyPoly2.MSK)

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 0.12 $\mu$ m process. The contacts are placed on both sides of the capacitor, between poly and metal on the left, between poly2 and metal on the right (Figure 11-14). The gate oxide is not used here because of its low breakdown voltage. A dual-oxide (5nm) would also suffer from voltage overstress that may occur in many analog designs, such as power amplifiers in radio-frequency (See chapter 12). Moreover, a thick oxide suffers from less process variations, which ensures a better control of the final capacitance.

### High precision Poly-Poly2 Capacitor

As process variations mainly affect the peripheral aspect of the layers, square geometry performs better than rectangular geometry. The optimum dimensions lie between 10x10 $\mu$ m and 50x50 $\mu$ m [Hastings]. If larger sizes are used, gradient effects affect the quality of the oxide which is no more uniform in the whole dielectric surface. Consequently, the capacitor should be split into 50x50 $\mu$ m units. Also, no device or diffusion region should be designed next to the capacitor. It is highly recommended to shield the capacitor area by using a guard ring of contacts which limit the substrate noise that may couple to the lower capacitor plate.

Also, the high impedance node should be connected to the upper plate of the metal, which is more isolated from the substrate and lateral noise. Finally, dummy capacitors can also be placed on the layout, for high precision matching. An example of high precision capacitor using 4 units of 4pF each is shown in figure 11-15. Dummy capacitors do not have the same size as capacitor units for a reason of silicon area saving. However, the spacing between the dummy capacitor and the active capacitor is preserved. The dummy devices serve as electrostatic shielding.

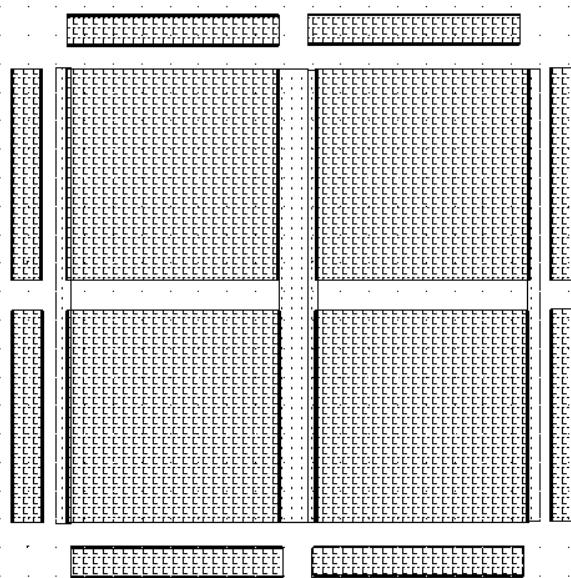


Figure 11-15: The layout of a 16pF poly-poly2 capacitor (CapaPoly15pF.MSK) using four units and dummy capacitor

## Inter-Metal Capacitor

The multiplication of metal layers create lateral and vertical capacitance effects of rising importance. Although the inter-metal oxide is 10 to 50 times thicker than the ultra-thin gate oxide, the spared silicon area in upper metal layers may be used for small size capacitance, which might be attractive for compensation or local decoupling.

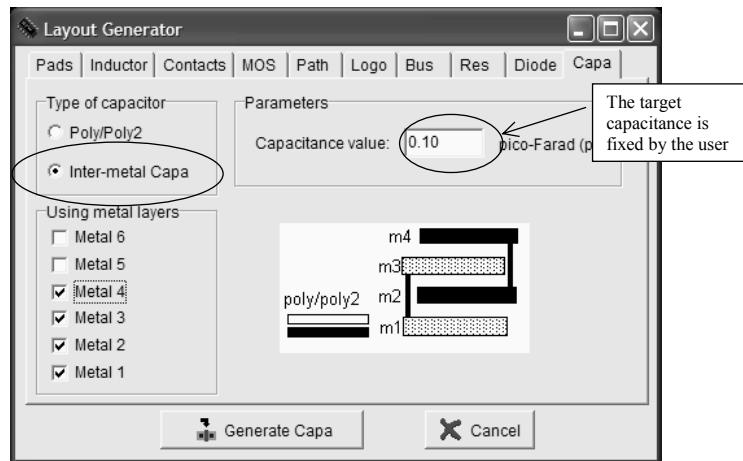


Figure 11-16: Menu for generating an inter-metal capacitor

The menu for generating the inter-metal capacitor is the same as for the poly/poly2 capacitor, except that the type of capacitor is changed (Figure 11-16). 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 11-16, a sandwich of metal1, metal2, metal3 and metal 4 is selected, for a target value of 100fF. The comparative aspect of the poly/poly2 capacitor and inter-metal capacitor is given in figure 11-17 for an identical capacitance value of 100fF. We confirm the poor efficiency of inter-metal capacitor due to the thick oxide, in comparison with the area-effective poly/poly2 structure.

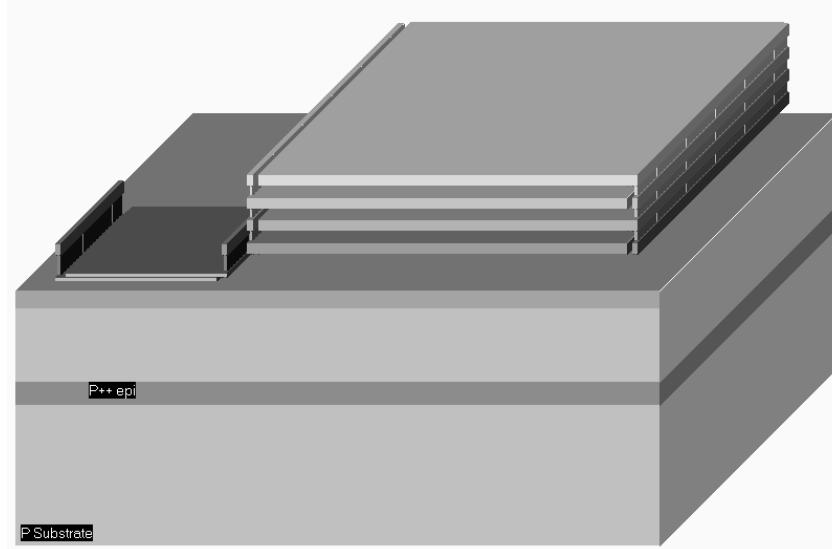
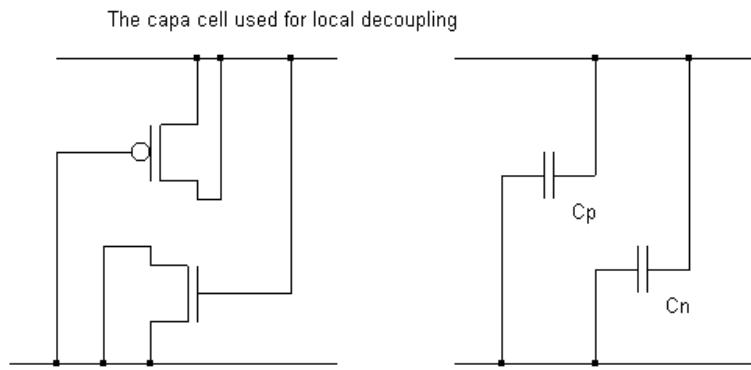


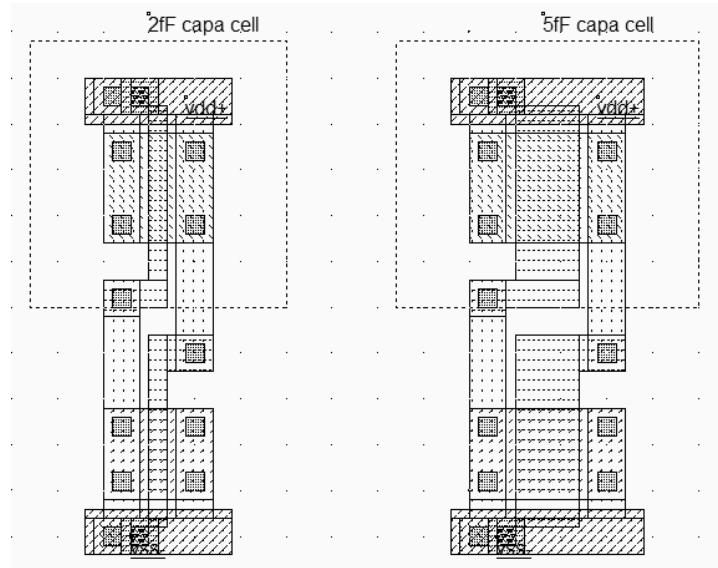
Figure 11-17: Generating an inter-metal capacitor (CapaPolyMetalComp.MSK)

## Capacitor Cell

Notice that a capacitor cell also exist in most logic cell libraries. This cell is inserted regularly in the design to add voluntary capacitance between the power rails VDD and VSS. This capacitor acts as a noise decoupling and reduces the external parasitic noise provoked by logic gate switching. The main drawback of this gate oxide capacitor is its low breakdown voltage, and the non-negligible possibility of gate-oxide defect which may result in a permanent conductive path between the supply rails. An implementation of the capacitor cell in a silicon area identical to the basic inverter is proposed in figure 11-19 left. Using a larger cell area, the gate area can be enlarged, which raises the equivalent decoupling capacitance very rapidly.



*Figure 11-18: Principles and equivalent diagram of the capacitor cell, inserted in the logic circuit core for improved noise decoupling (CapaCell.MSK)*



*Figure 11-19: Two implementations of the capacitor cell (2fF, 5fF) (CapaCell.MSK)*

### 3. The MOS device for analog design

The MOS has been used in previous chapters mainly as a switch. The most important parameters are the maximum available current  $I_{on}$  current and the parasitic leakage current  $I_{off}$ . The  $I_{on}$  current corresponds to a maximum  $V_{gs}$  and maximum  $V_{ds}$  (Upper right point in figure 11-20). In the case of analog design, the MOS do not operate only in cut-off or saturation regime. It also operates in the so-called quadratic zone, that appears in the static characteristics shown in figure 11-20. In that case,  $V_{ds}$  is rarely very large, as well as  $V_{gs}$ . The MOS device operates in an intermediate regime which is attractive for most analog applications.

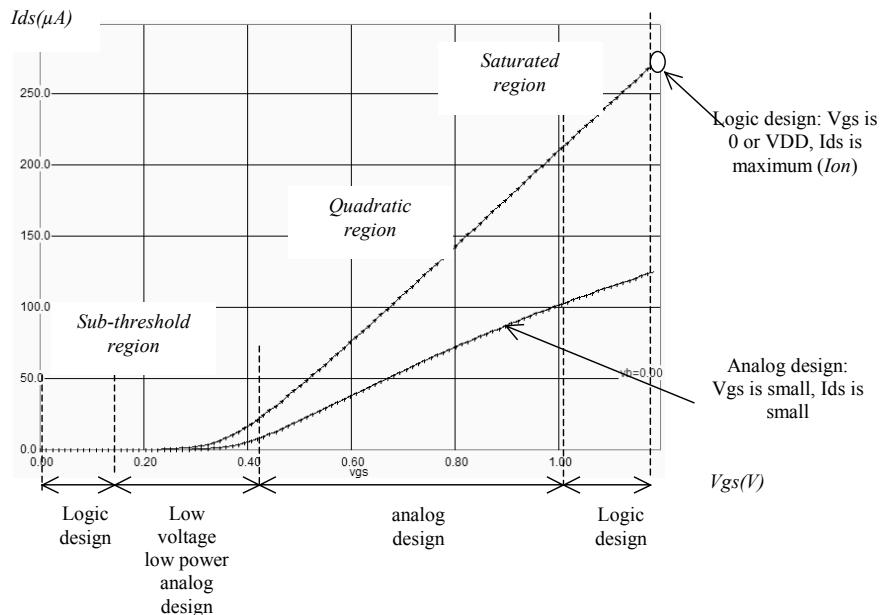


Figure 11-20: In analog design, the MOS device also operates in quadratic zone

#### Using Bsim4

The MOS model "level 3" is reasonably accurate in the case of logic circuits. When dealing with analog circuits, significant mismatch are observed between MOS level 3 and the advanced model BSIM4. This is because the equations of the model in level 3 do not account for many second order parameters, that have a very important impact when running in small voltages and currents. We recommend the use of the model BSIM4 in all analog cells described in this chapter. Select BSIM4 in the menu of the command **Simulate → Using Model → BSIM4**. An alternative consists in selecting the model BSIM4 in the list proposed in the simulation parameters (**Simulate → Simulation parameters**).

You may automatically select BSIM4 by adding a text in the layout that starts with "BSIM4". In the layout example shown in figure 11-21, we added a text "BSIM4", which forces the simulator to use the BSIM4 model instead of the default MOS level 3. This text appears for example at the left lower part of the layout of the transmission gate, described in figure 11-22.

## Analog switch

The analog switch is able to transfer or interrupt an analog signal. It can be constructed using the pass gate described in chapter 2 which used one n-channel MOS and one p-channel MOS in parallel. The transmission gate lets an analog signal flow if  $en=1$  and  $\sim en=0$ . In that case both the n-channel and p-channel devices are on.

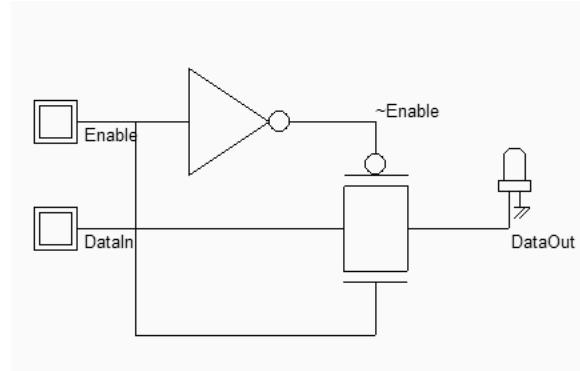


Figure 11-22: Schematic diagram of the analog switch (TGATE.Sch)

The layout of the analog switch is shown in figure 11-23. The inverter is situated on the left, the transmission gate on the right. A sinusoidal wave with a frequency of 2GHz is assigned to *DataIn*. The sinusoidal property may be found in the palette of Microwind2, near the clock and pulse properties. With a zero on *Enable* (And a 1 on  $\sim Enable$ ), the switch is off, and no signal is transferred (Figure 11-24). When *Enable* is asserted, the sinusoidal wave appears nearly identical to the output.

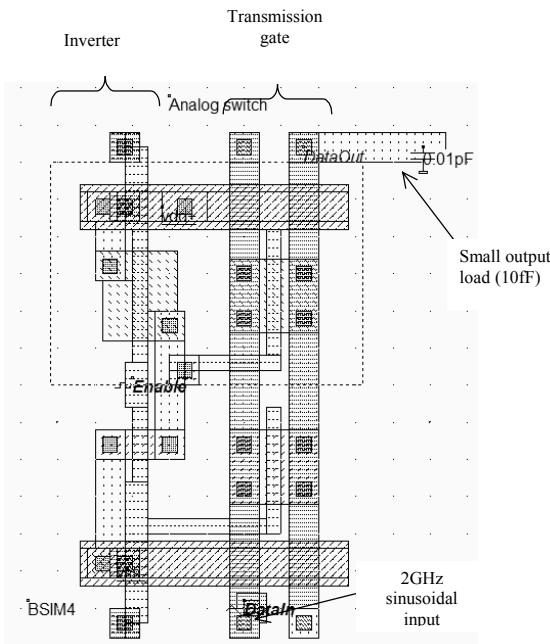


Figure 11-23. Simulation of the analog switch (AnalogSwitch.MSK)

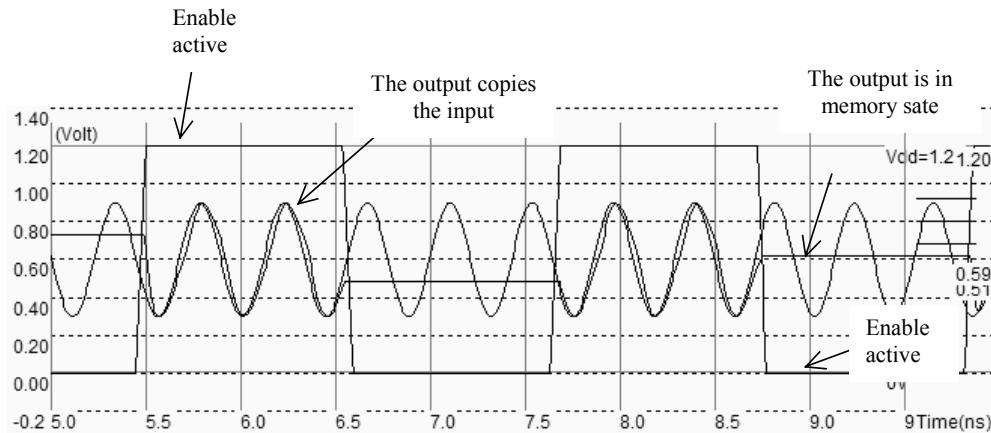


Figure 11-24. Simulation of the transmission gate (AnalogSwitch.MSK)

As the output is loaded with a 10fF virtual capacitance, the output is slightly distorted. This is due to the non linear resistance variation of the transmission gate with the input voltage. Consequently, the transmission gate reacts faster for a low voltage or a high voltage, and slower for an intermediate voltage where n-channel and p-channel devices have less current capabilities, even if both conduct at the same time.

#### 4. Diode-connected MOS

The schematic diagram of the diode-connected MOS is proposed in figure 11-25. 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.

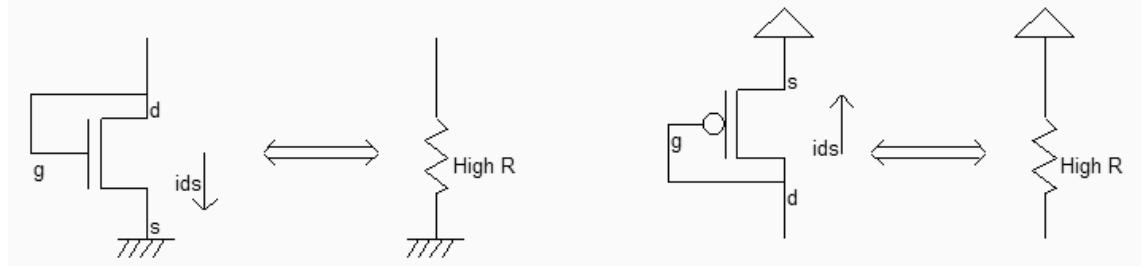


Figure 11-25 : Schematic diagram of the MOS connected as a diode (MosRes.SCH)

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  $W=0.24\mu m$  and  $L=2.4\mu m$ . This sizing corresponds to a long channel, featuring a very high equivalent resistance.

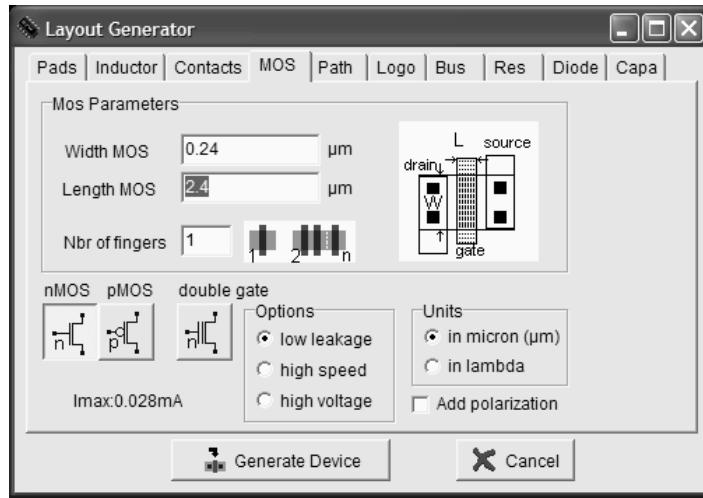


Figure 11-26 : Using the MOS generator to create a n-channel MOS with a large length and small width.

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 11-27.

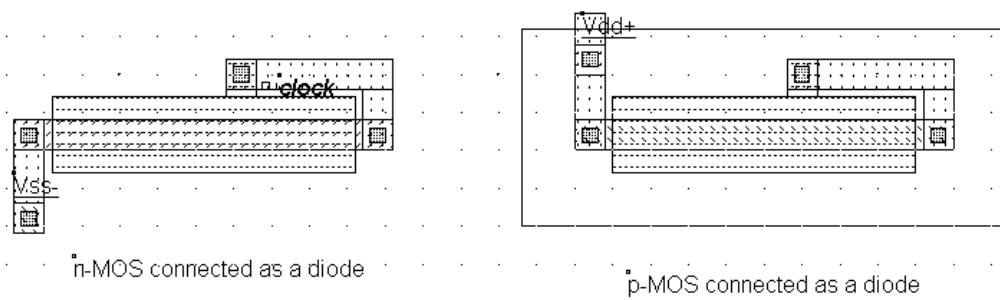


Figure 11-27 : Schematic diagram of the MOS connected as a diode (ResMos.MSK)

Now, click **Simulation on Layout**. In a small window, the MOS characteristics are drawn, with the functional point drawn as a color dot (Figure 11-28). It can be seen that the  $I/V$  characteristics correspond to a diode. The resistance is the invert value of the slope in the  $Id/Vd$  characteristics. For  $Vds$  larger than 0.6V, the resistance is almost constant. As the current  $Ids$  increases of  $10\mu A$  in 0.4V, the resistance can be estimated around  $40K\Omega$ . 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.

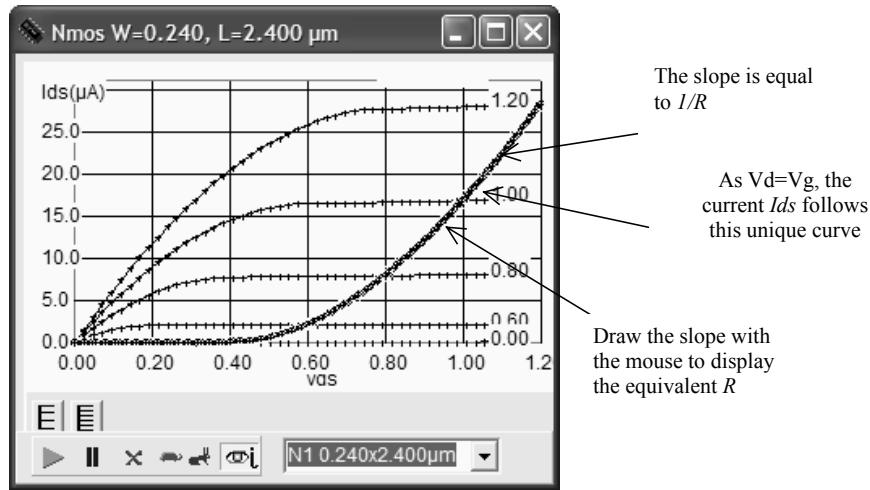


Figure 11-28 : Using the Simulation on Layout to follow the characteristics of the diode-connected MOS (ResMos.MSK)

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  $100\text{ k}\Omega$  in a very small silicon area. The same resistance can be drawn in poly but would require a much larger area. The resistance per square of an unsalicide polysilicon serpentine is approximately 40 ohm. In figure 11-29, a polysilicon resistance of  $30\text{ k}\Omega$  is drawn close to the MOS device with a  $30\text{ k}\Omega$  resistance. The advantage of using a MOS resistance rather than a polysilicon resistance is obvious in terms of silicon area.

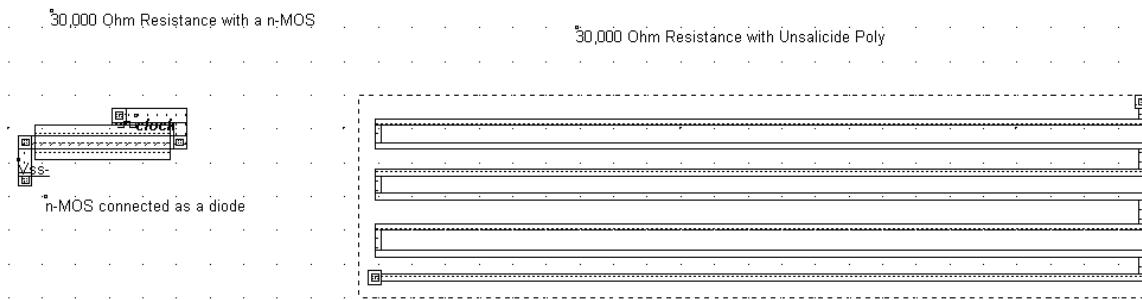


Figure 11-29: A MOS device resistance compared to the same resistance in Poly(ResMos.MSK)

## 5. Voltage Reference

The voltage reference is usually derived from a voltage divider made from resistance. The output voltage  $V_{ref}$  is defined by equation 11-4.

$$V_{ref} = \frac{R_N}{R_N + R_P} V_{DD} \quad (\text{Eq. 11-4})$$

with

$V_{DD}$ =power supply voltage (1.2V in  $0.12\mu m$ )

$R_N$ =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 11-30. The most common design uses one p-channel MOS and one n-channel MOS connected as diodes.

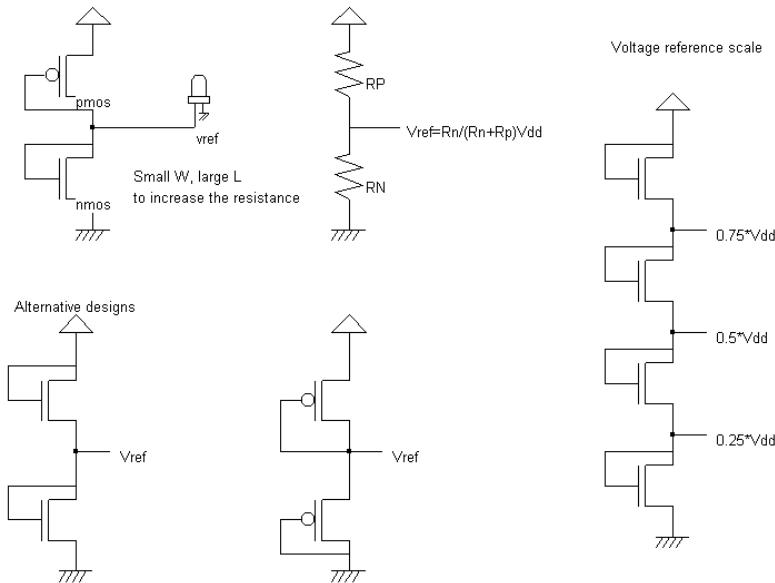


Figure 11-30 : Voltage reference using PMOS and NMOS devices as large resistance

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.

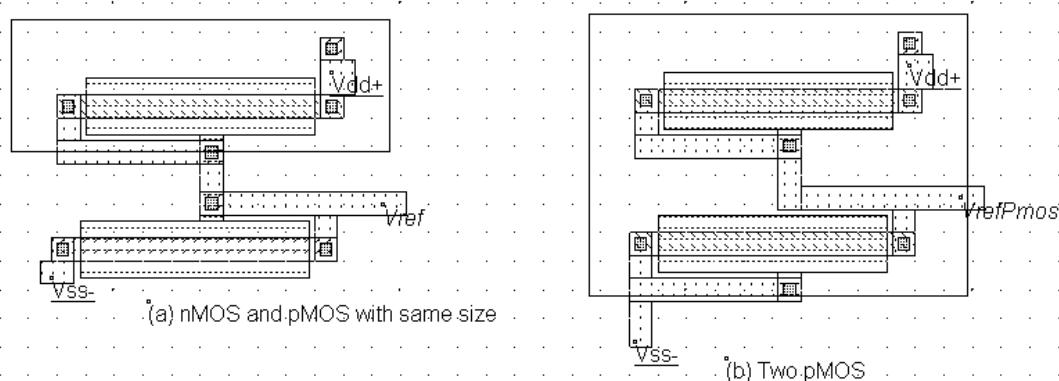


Figure 11-31 : Voltage reference circuits (a) with one nMOS and one pMOS (b) with two pMOS ( $V_{ref,MSK}$ )

In the layout of figure 11-31, 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 11-32. 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 (Figure 11-33). 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.

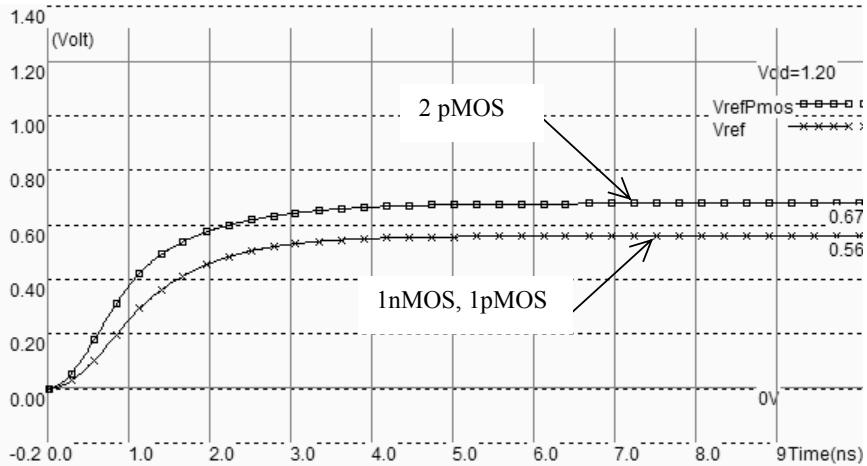


Figure 11-32 : Simulation of the two voltage reference circuits (Vref.MSK) using BSIM4 model

Substrate polarization

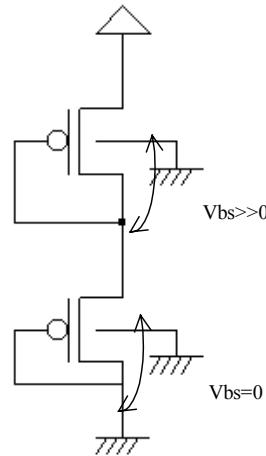


Figure 11-33 : The polarization of the two pMOS is not identical due to the substrate effect (Vref.SCH)

The value of the voltage reference  $V_{ref}$  versus the size of the n-channel and p-channel MOS is quite difficult to calculate as the resistance of the channel is highly non-linear. In [Baker], the formulation is deduced from the equations of model 1:

$$V_{ref} = \frac{V_{DD} - V_{tp} + \sqrt{\frac{\beta_N}{\beta_P}} V_m}{\sqrt{\frac{\beta_N}{\beta_P}} + 1} \quad (\text{Eqn. 11-5})$$

with

$$\beta_N = \mu_N \frac{W_N}{L_N} \text{ and } \beta_P = \mu_P \frac{W_P}{L_P}$$

where

$\mu_N$ =mobility of electrons ( $600 \text{ cm}^2/\text{V.s}$ )

$\mu_P$ =mobility of holes ( $250 \text{ cm}^2/\text{V.s}$ )

$W_N$ = nMOS width ( $\mu\text{m}$ )

$L_N$ =nMOS length ( $\mu\text{m}$ )

$W_P$ =pMOS width ( $\mu\text{m}$ )

$L_P$  = pMOS length ( $\mu\text{m}$ )

### Multiple voltage reference

Not more than three MOS devices can be connected in series to produce intermediate voltage references. The limiting factor is the threshold voltage. When trying to simulate a series of more than 3 MOS connected as diodes, the operating regime keeps in sub-threshold mode, which is attractive for very low power consumption, but introduces important setup delays and creates very weak voltage references. When more than 3 reference voltages are needed, the network is built using resistance, as shown in figure 11-34. The static power consumption is  $100\mu\text{W}$ , which is due to the DC current flowing through the resistors between VDD and VSS. Larger resistance would decrease this static power, down to the specified user requirement.

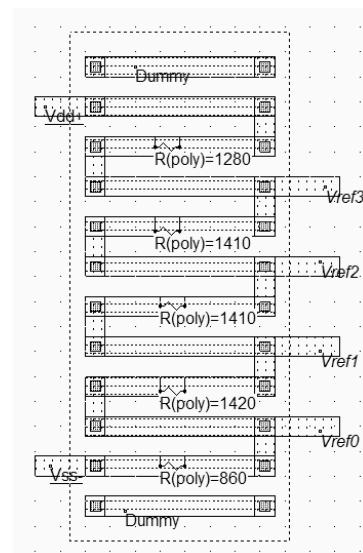
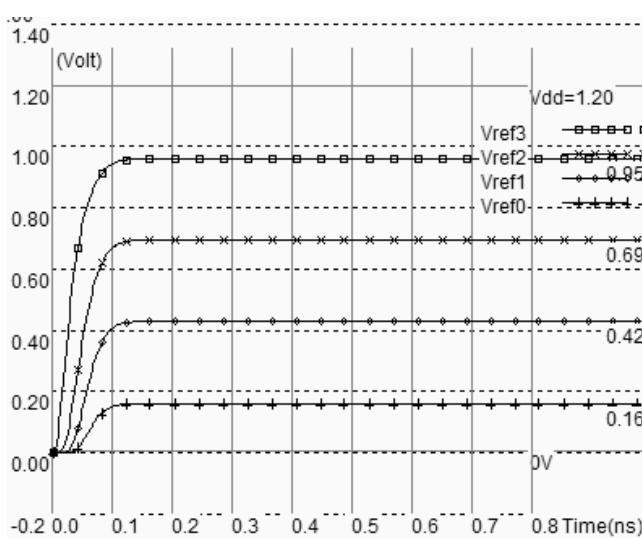


Figure 11-34 : Resistor scale (VrefMul.MSK)

## Shielding

The voltage reference  $V_{ref}$  created by our circuits is very weak, in the sense that the current which flows in the MOS branch is small. In other words, the  $V_{ref}$  signal is highly resistive, we say also "high impedance". This means that a parasitic signal that couples with the  $V_{ref}$  connection may induce some noise, for example by proximity effect and capacitance coupling  $C_x$ , as shown in figure 11-35.

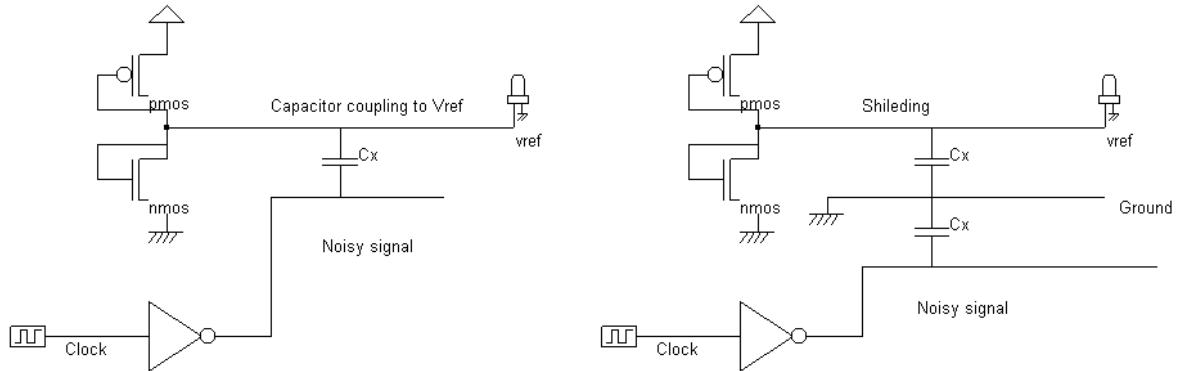


Figure 11-35 : Without shielding, the  $V_{ref}$  voltage may be altered by noisy signals routed close to its interconnect ( $V_{refNoise}.SCH$ )

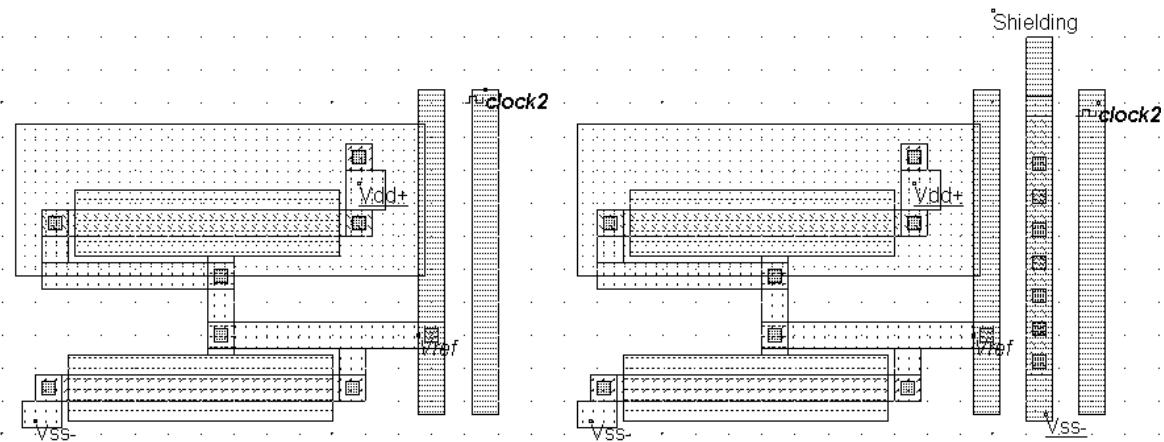


Figure 11-36 : Unshielded voltage reference (left) and shielded voltage reference (right) ( $V_{refNoise}.MSK$ )

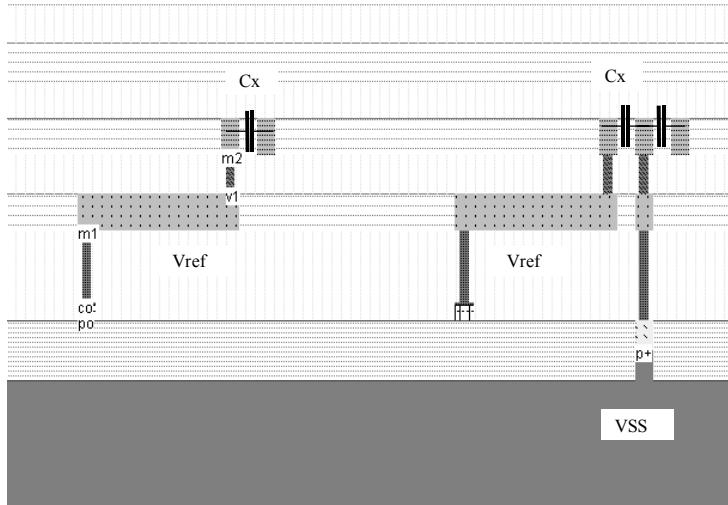


Figure 11-37 : 2D cross-section of the unshielded (Left) and shielded voltage reference (VrefNoise.MSK)

Adding a metal shielding acts as a noise barrier and protects *Vref* from the clock coupling (Figure 11-36). The 2D cross-section shows the two structures: on the left side, the output signal *Vref* has a parasitic coupling capacitance *Cx* connected to the noise signal. When the noisy signal switches, the voltage reference is altered. On the right structure, a barrier made of a p-type diffusion, metal and metal2 create an electrostatic screen. The coupling still exists, but *Cx* is now connected to a cold signal, meaning that the ground capacitance is increased, and the noise is eliminated.

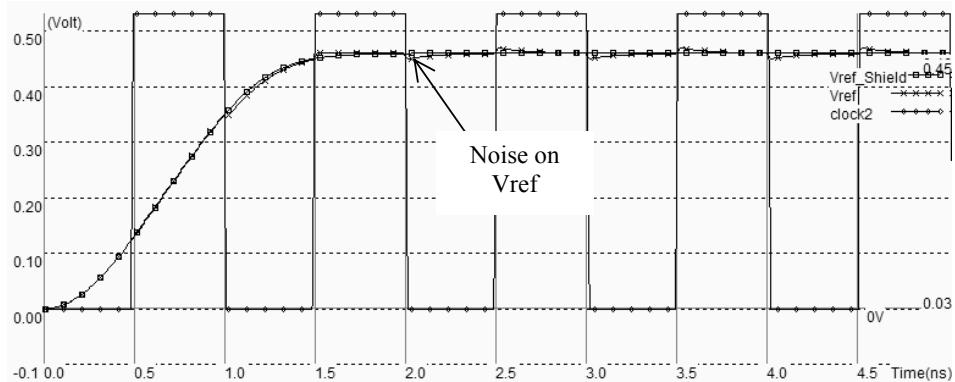


Figure 11-38 : Simulation of the unshielded and shielded *Vref* showing a small crosstalk noise (VrefNoise.MSK)

The effect of the shielding is demonstrated in the simulation shown in figure 11-38. In Microwind, the crosstalk effect due to lateral coupling is extracted but not simulated by default. The crosstalk simulation is activated by the command **Simulate → With crosstalk**. Although the coupling capacitor still exists, the shielding is connected to ground and prevents the clock from interfering with the *Vref\_shield* signal which remains flat.

## Influence of Temperature

You may change the temperature (**Simulate → Simulate Options**) and see how the voltage reference is altered by temperature. The circuit based on one nMOS and one pMOS, already presented in figure 11-30, is not much influenced by temperature. The temperature coefficient (TC), to a first order, is almost equal to zero. A similar result is obtained using model 3 or BSIM4. This means that a stable on-chip voltage reference around VDD/2 is quite simple to achieve.

The design of a reference voltage VDD/3 lead to unbalanced resistivity, which has a direct impact on the temperature coefficient. This time, the voltage is no more a reference voltage as it only coincides with VDD/3 at room temperature of 25°C. At high temperature, the voltage reference is much too low. The simulation results are summarized in figure 11-39. The parametric analysis has been used to compute the voltage  $V_{ref}$  iteratively at the end of a 5ns simulation, with increased temperature from -40 to 120°C.

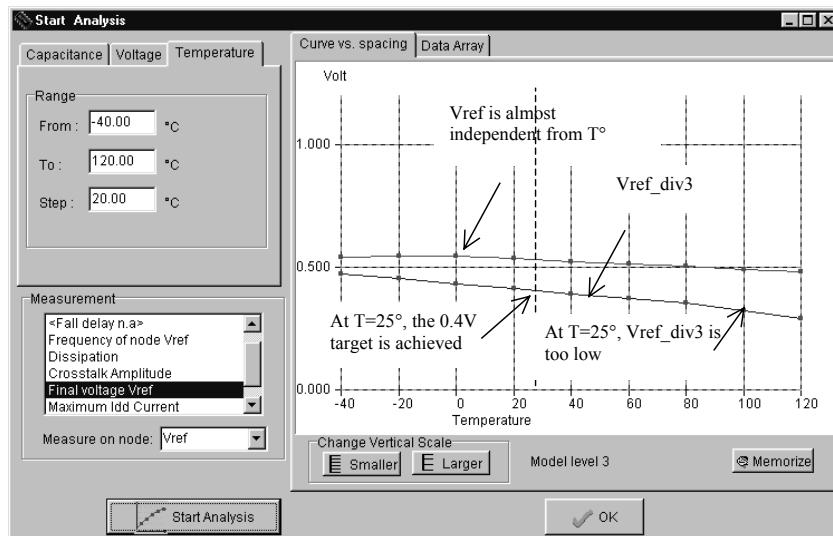


Figure 11-39 : Simulation of the influence of temperature on the reference voltage (Vref.MSK)

## 6. 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 11-40. 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 11-40 by a resistor  $R_{load}$ .

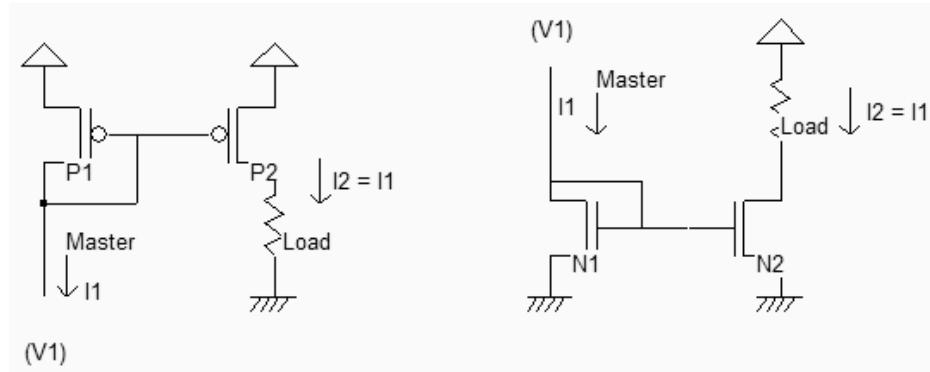


Figure 11-40: Current mirror principles in nMOS and pMOS version (CurrentMirror.SCH)

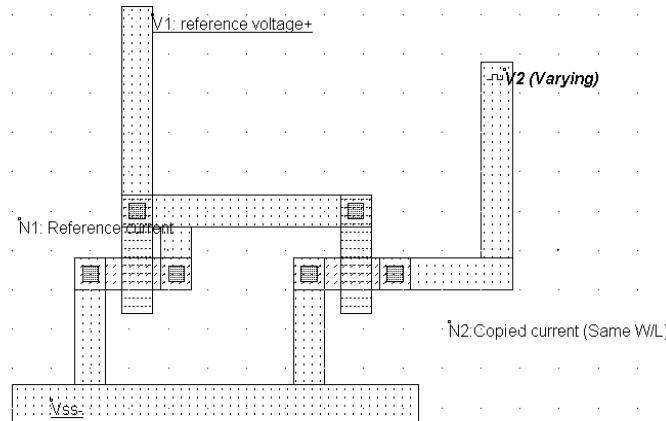


Figure 11-41: Layout of an n-channel current mirror with identical size (CurrentMirror.MSK)

The illustration of the current mirror behavior is proposed in the case of two identical N-channel MOS (Figure 11-41). 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  $N2$ .

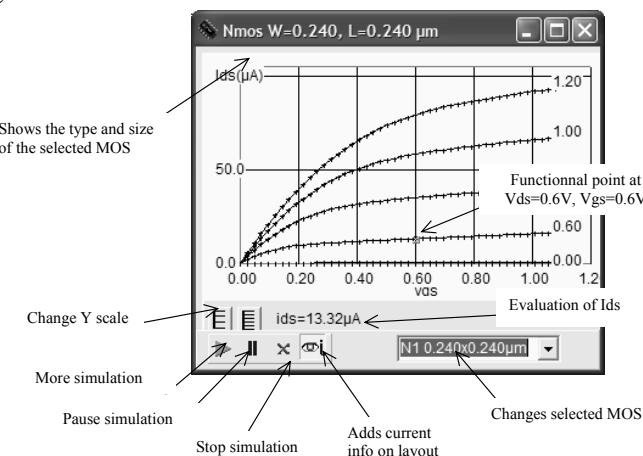


Figure 11-42: The nMOS  $N1$  has a fixed current around  $12\mu A$  flowing between drain and source (CurrentMirror.MSK)

Concerning  $N1$ , the gate and drain voltage is fixed to 0.6V, which corresponds to a constant current of around  $13\mu A$ , as shown in figure 11-42. The voltage  $V2$  (Figure 11-43) varies thanks to a clock. We observe that the current  $I2$  is almost equal to  $13\mu A$ , independently of  $V2$ , except when  $V2$  is lower than 0.2V. More precisely, the variation of  $I2$  is between  $12$  and  $16\mu A$  when  $V2$  varies from 0.2 to 1.2V.

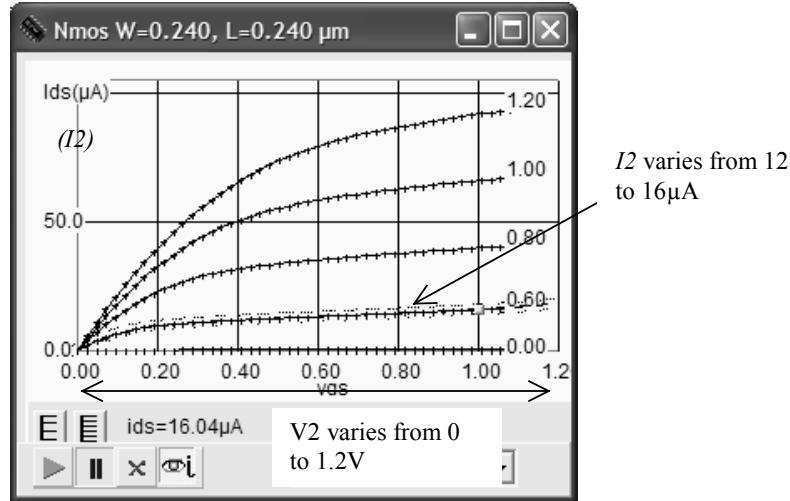


Figure 11-43: Illustration of the nMOS current mirror principles (CurrentMirror.MSK)

### Improving the Current Mirror

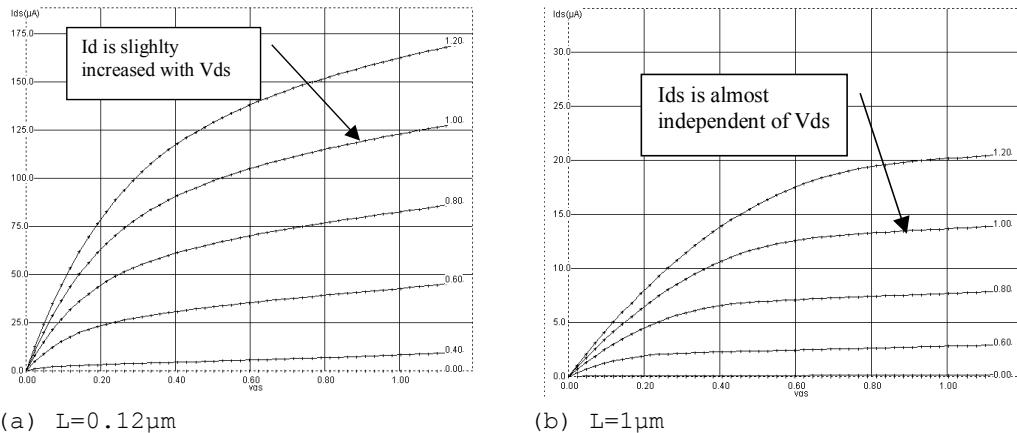


Figure 11-44: Long channel MOS are preferred for high performance current mirrors

As the basic principle of the current copy is the assumption that  $Id$  is independent of  $Vds$ , the long channel MOS (Fig 11-44 right) is a better candidate than the short channel MOS (Fig 11-44 left). Although the short channel MOS works faster, the long channel MOS is preferred for its higher precision when copying currents.

## 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  $Ids$  is almost independent of  $Vds$ .

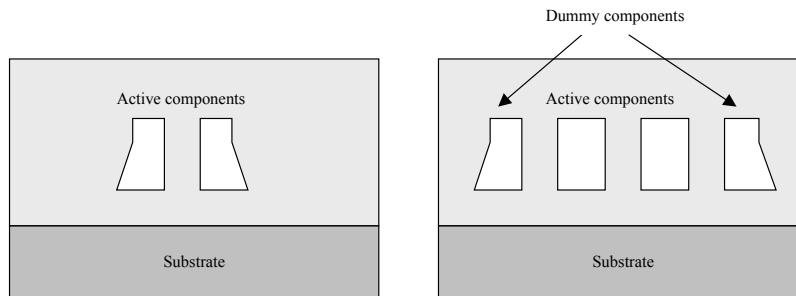


Figure 11-45: 2D aspect of the circuit without and with dummy components

- 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 11-45).
- 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.

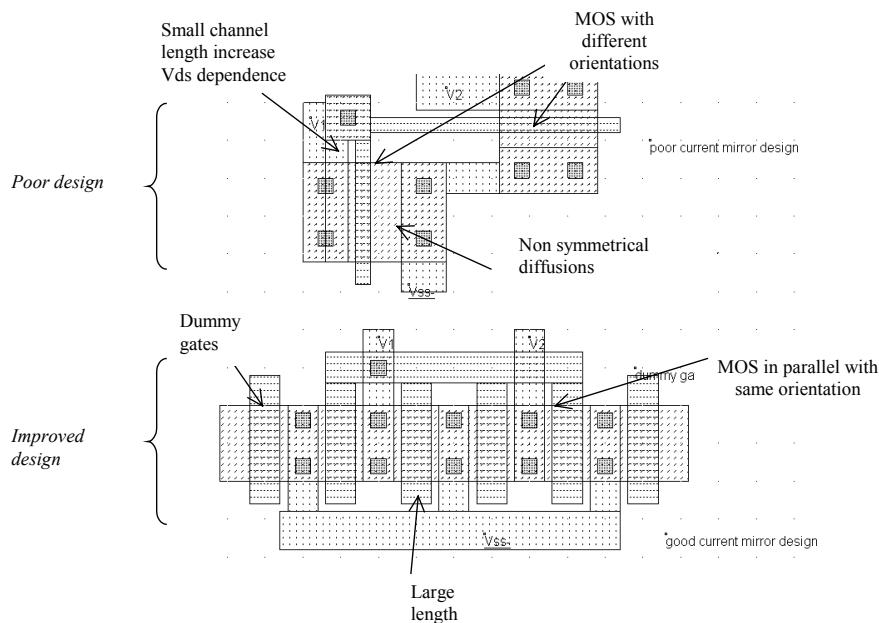


Figure 11-46: Design of high performance current mirrors (MirrorMatch.MSK)

A synthesis of these recommendations is proposed in the two designs of figure 11-46. The current mirror situated at the top of the figure cumulates design weaknesses: short channel length, non-symmetrical drain and source design, and different orientations for the devices. The design at the bottom realizes the same current copying function, and complies with the most important rules for a good current copying: dummy components, same orientation, symmetrical design, and MOS devices with large length.

### Current Multiplier

If the ratio W/L of the *Slave* (Transistor P2) is 10 times the ratio of the *Master*, the current  $I_2$  on the right branch is 10 times the current  $I_1$  on the left branch. This is illustrated by the schematic diagram in the left of figure 11-47. In the case of the PMOS current mirror, the ratio W/L of the *Slave* (P2) is 5 times the ratio of the *Master*, the current  $I_2$  is 5 times the current  $I_1$ .

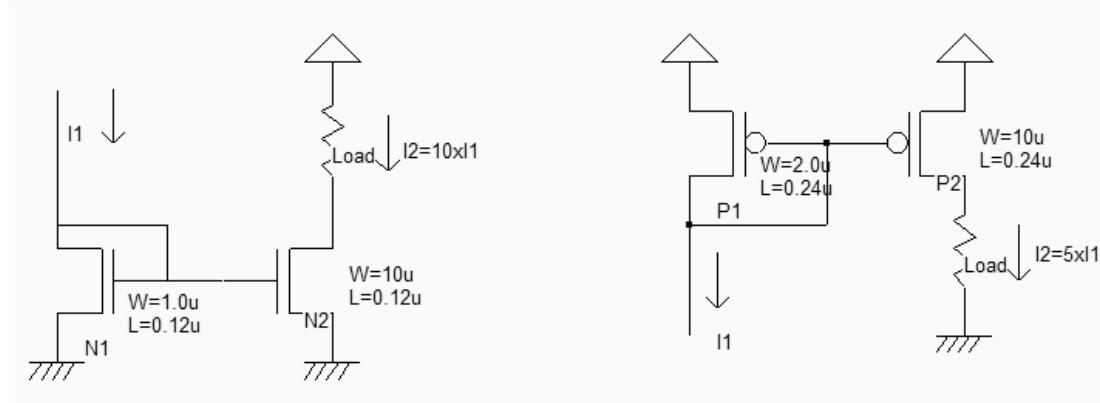


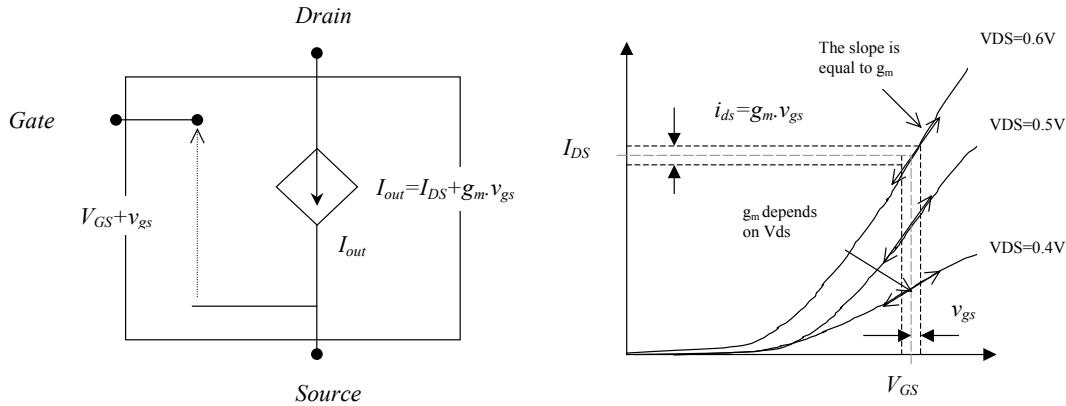
Figure 11-47: Multiplying currents by changing the size of the MOS (MirrorMatch.MSK)

## 7. The MOS transconductance

In its most simple form, the MOS can be represented as a current generator controlled by the voltage, or Voltage-Controlled Current Source (VCCS). The schematic diagram of the VCCS is given in figure 11-6. We add to  $V_{GS}$  a small sinusoidal input  $v_{gs}$  which provokes a small variation of current  $i_{ds}$  to the static current  $I_{DS}$ . For small variations of  $v_{gs}$ , the link between the variation of current  $i_{ds}$  and the variation of voltage  $v_{gs}$  can be approximated by (equ. 11-6).

$$i_{ds} = g_m v_{gs} \quad (\text{Equ. 11-6})$$

The transconductance  $gm$  has the dimension of the ratio of current to voltage, that is the invert of the resistance, and its definition is given in equation 11-7.

Figure 11-47 : The MOS transconductance  $g_m$ 

$$g_m \equiv \frac{\partial I_{DS}}{\partial V_{GS}} \quad (\text{equ. 11-7})$$

The derivation of the transconductance  $g_m$  from the MOS equations in level 1 leads to a quite simple expression reported in equations 11-8 (Linear region) and 11-9 (saturation region).

$$\text{Linear region } g_m = \frac{\partial(\beta((V_{gs} - vt).V_{ds} - \frac{(V_{ds})^2}{2}))}{\partial V_{GS}} = \beta(V_{ds}) \quad (\text{Equ. 11-8})$$

$$\text{Saturation region } g_m = \frac{\partial(\beta(V_{gs} - vt)^2)}{\partial V_{GS}} = \frac{\beta(V_{gs} - vt)}{2} \quad (\text{Equ. 11-9})$$

with

$$\beta = U_0 \frac{\epsilon_0 \epsilon_r}{TOX} \cdot \frac{W}{L}$$

where

$V_{gs}$ =voltage between gate and source (V)

$V_t$ = threshold voltage (V)

$W$ =transistor width ( $\mu\text{m}$ )

$L$ =transistor length ( $\mu\text{m}$ )

$U_0$ =electron mobility ( $\text{m}/\text{V}^2$ )

$TOX$ =gate oxide (m)

For deep submicron technology, more accurate expressions of the transconductance  $g_m$  are proposed such as those proposed in [Janssens]. The most important point to remember is the dependence (Linear in a first approximation) of  $g_m$  with the width and  $V_{ds}$ .

## 8. Single Stage 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. Such a circuit may be found at the input stage and the output stage of all telecommunication devices such as mobile phones. The input stage amplifier increases the amplitude of the captured signal from around 0.1-1mV to 10-100mV for further processing, while the output stage amplifier delivers a high voltage on the antenna to emit a significant power (Figure 11-48).

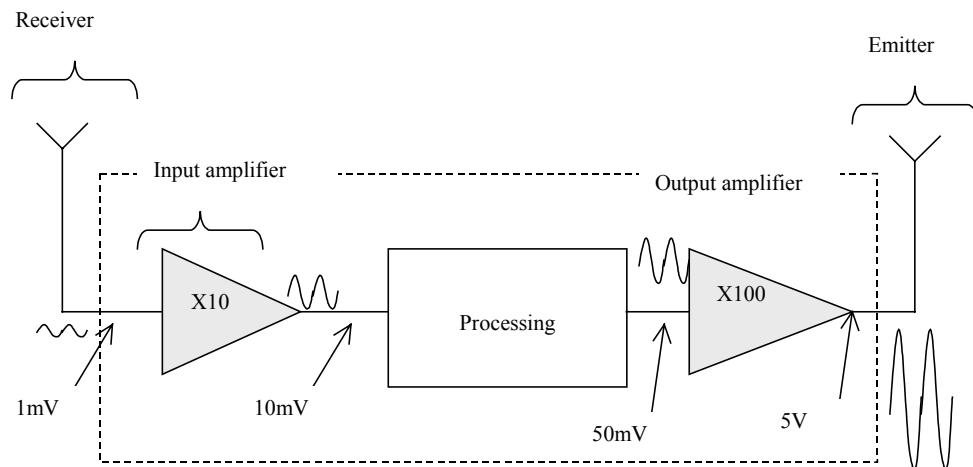


Figure 11-48: Example of amplifier circuits used in mobile devices

The single stage amplifier may consist of a MOS device (we choose here an 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 11-49). The pMOS which replaces the passive load is called an active resistance.

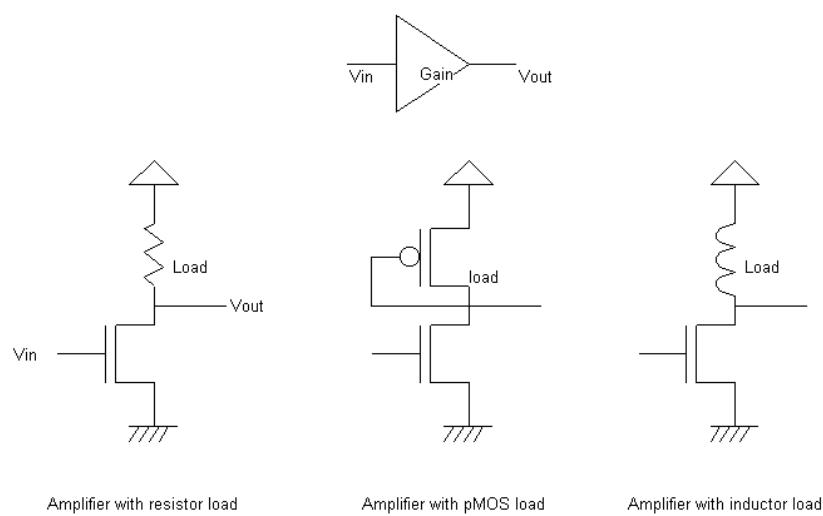


Figure 11-49: Single stage amplifier design with MOS devices (AmpliSingle.SCH)

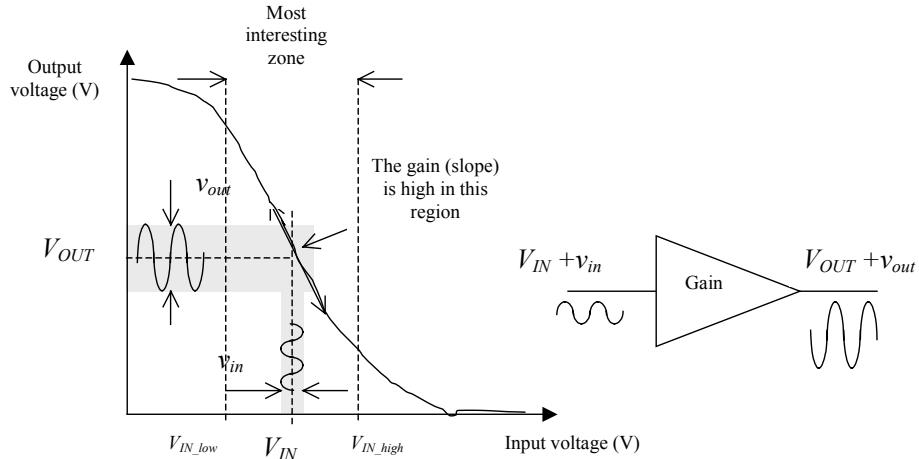


Figure 11-50: The amplifier has a high gain at a certain input range, where a small input signal  $v_{in}$  is amplified to a large signal  $v_{out}$ .

The single stage amplifier characteristics between  $V_{IN}$  and  $V_{OUT}$  have a general shape shown in figure 11-50. 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 11-10.

$$i_{ds} = g_m v_{gs} \quad (\text{Eq. 11-10})$$

Consequently, the gain of the amplifier for small signals can be expressed by equation 11-11.

$$\text{Gain} = \frac{v_{in}}{v_{out}} = \frac{-i_{ds} \frac{1}{g_{mp}}}{i_{ds} \frac{1}{g_{mn}}} = -\frac{g_{mn}}{g_{mp}} \quad (\text{Eq. 11-11})$$

In other words, the gain of the amplifier is high if  $g_{mp}$  is low, which is equivalent to a high pMOS pass resistance. The sign minus in equation 11-11 illustrates the fact that an increase of  $v_{in}$  corresponds to a decrease of  $v_{out}$ . The diode-connected p-channel MOS creates a high resistance when the channel width is minimum and the channel length is very large. Such a design means a high amplifier gain. In figure 11-51, an 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 is expected is a 500mV sinusoidal wave ( $v_{out}$ ) with a certain DC offset ( $V_{OUT}$ ).

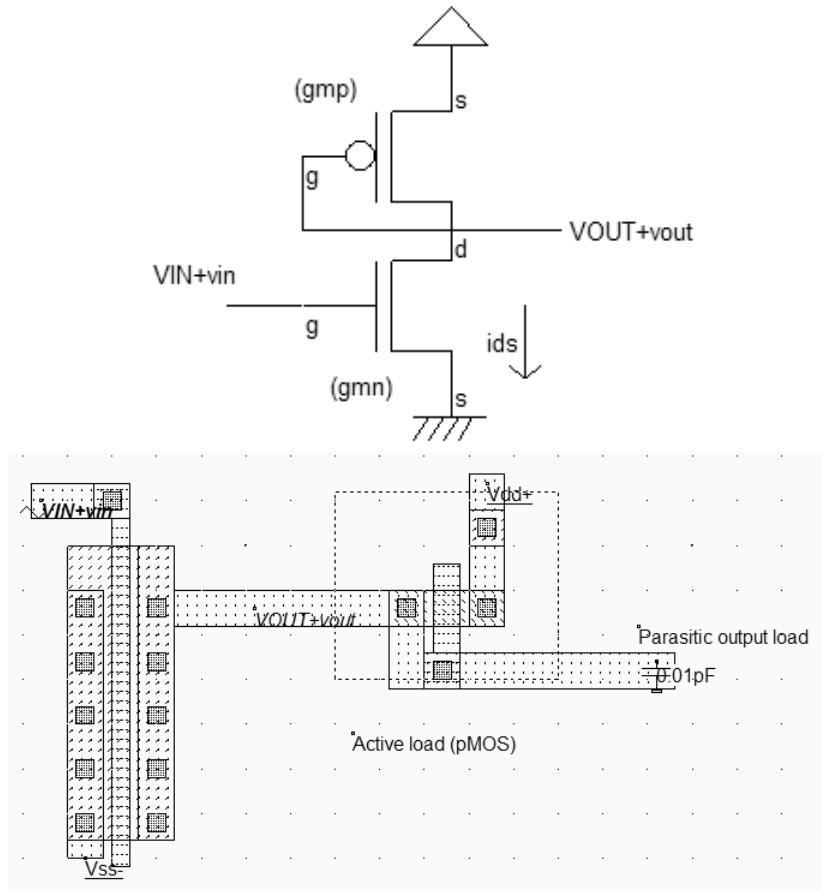


Figure 11-51: Single stage amplifier layout with a pMOS as a load resistor (AmpliSingle.MSK)

The time-domain simulation of the amplifier with a 1GHz sinusoidal input exhibits very poor performances. The gain is almost 0 and the output is very low, close to ground. This is because the offset  $V_{IN}$  has been fixed to a default value of  $VDD/2$  (0.6V) which does not correspond to the region where the circuit provides a high gain. We are probably higher than  $V_{IN\_high}$ .

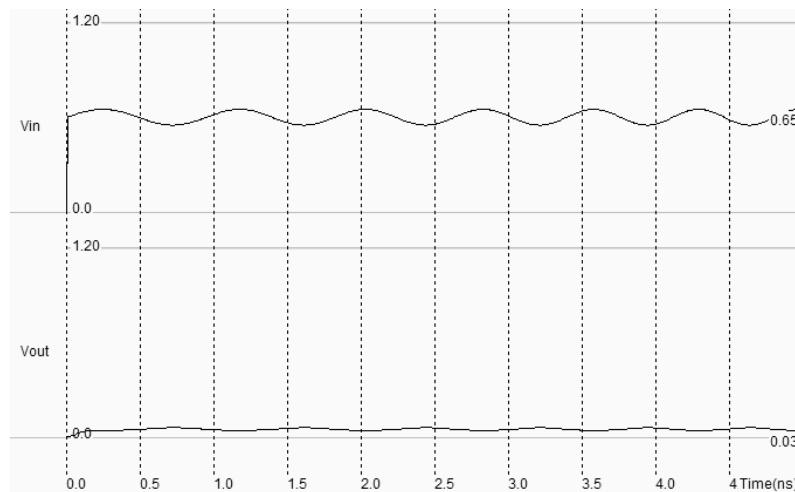


Figure 11-52: Simulation of the amplifier response to a 50mV input with a 0.6V offset (AmpliSingle.MSK)

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 11-53). 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.

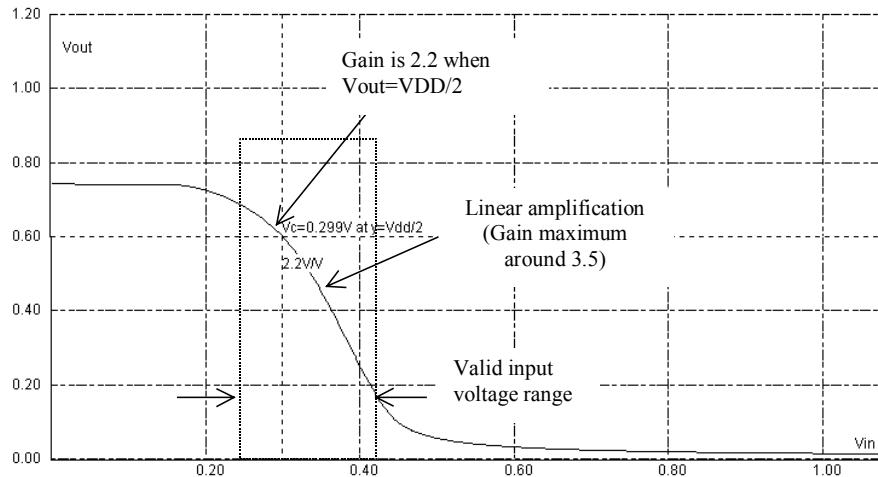


Figure 11-53: Single stage amplifier static response showing the valid input voltage range.

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 11-54, 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.

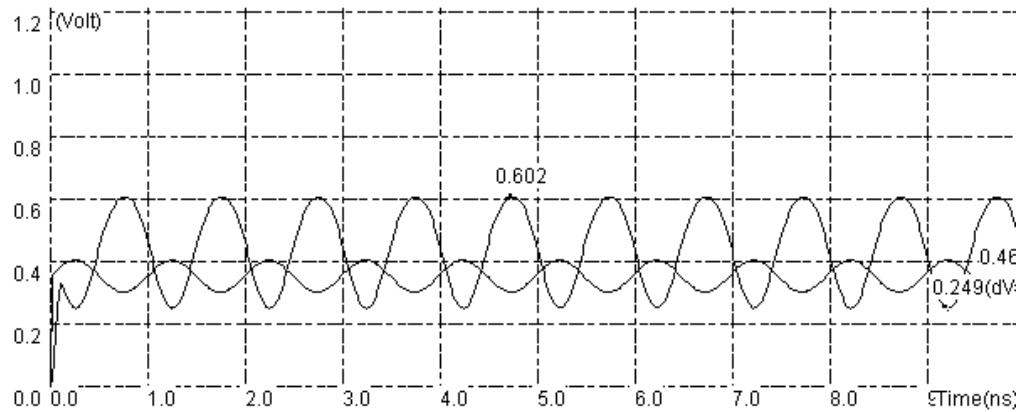


Figure 11-54: Single Stage amplifier with correct polarization  $V_{IN}=0.35V$

To increase the gain, the ratio between the active load resistance and the n-channel MOS resistance should be increased. In the layout proposed in figure 11-55, three amplifiers are implemented: one with a pMOS load (layout with output *sinus1*), the second with high resistance pMOS (Layout with output *sinus2*), and the third with a very high resistor symbol (20Kohm).

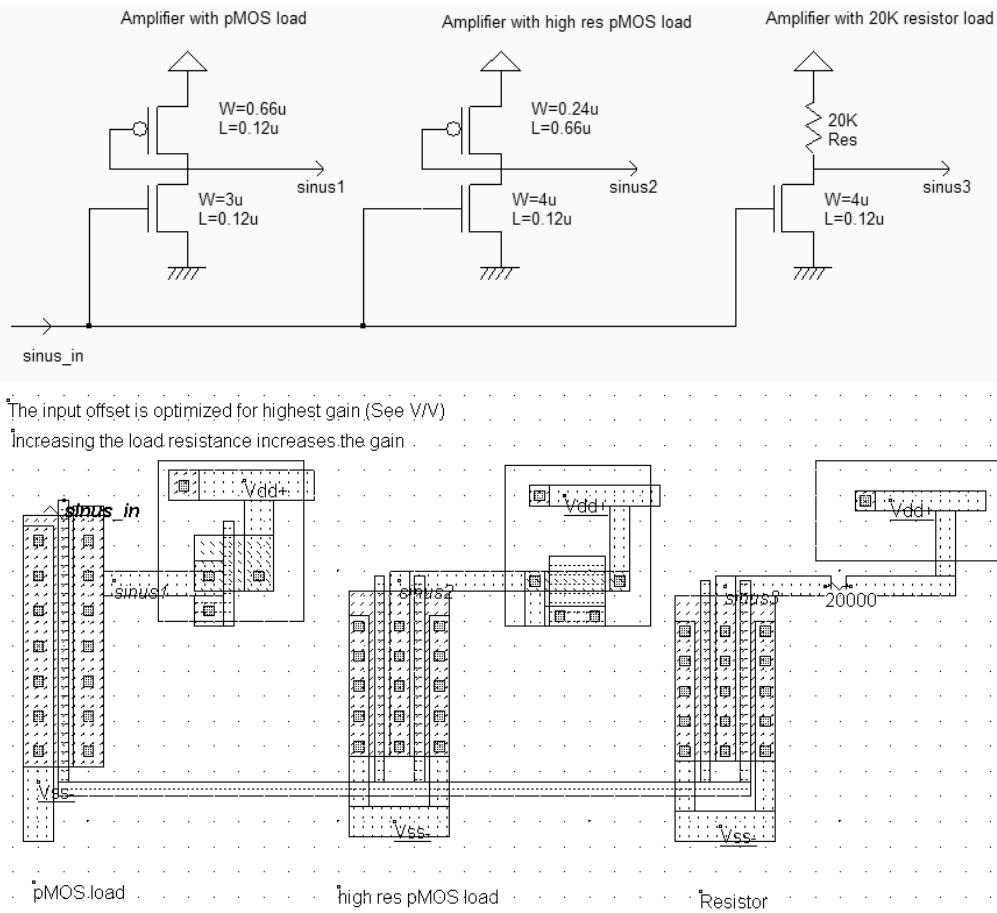


Figure 11-55: Single Stage amplifier with correct polarization  $V_{IN}=0.35V$  (AmpliSingle2.MSK)

The gain for  $sinus2$  is increased to 4.5, as observed in the static simulation (Figure 11-56), with sharper characteristics, but the input voltage range that features amplification is significantly reduced. A further increase of the pMOS resistance does not increase the gain. The gain saturates to around 8. If we replace the pMOS device by a resistor symbol, we also observe that the gain is limited to around 9.0.

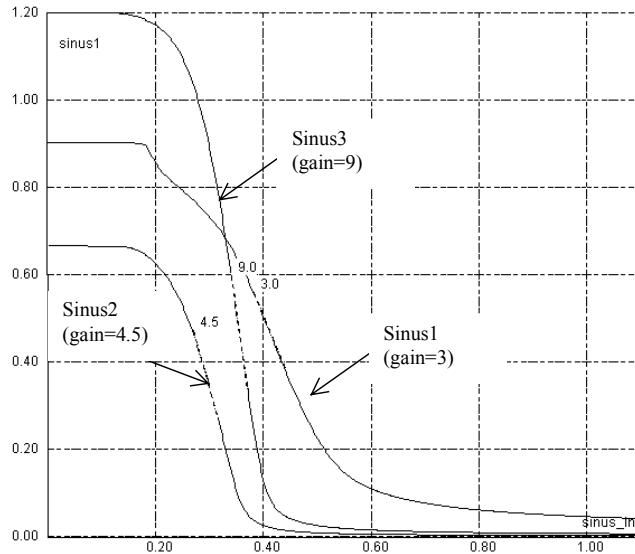


Figure 11-56: The active load sizing acts on the gain, but reduces the input voltage range for amplification (AmpliSingle2.MSK)

### Transit and Cutt-Off frequency

The transit frequency  $f_t$  is a parameter well representative of the "speed" of the MOS device. It corresponds to the frequency at which the current  $i_{ds}$  starts being lower than the current flowing through the gate  $i_{gs}$ . The  $i_{gs}$  current is due to the charge and discharge of the capacitor  $C_{gs}$ . The  $i_{ds}$  current is the main amplifier current that flows between the drain and the source. Thus,  $f_t$  is the frequency for which the current gain of the MOS device is unity. Based on the equations of level 1, an analytical approximation of  $f_t$  is reported below [Baker].

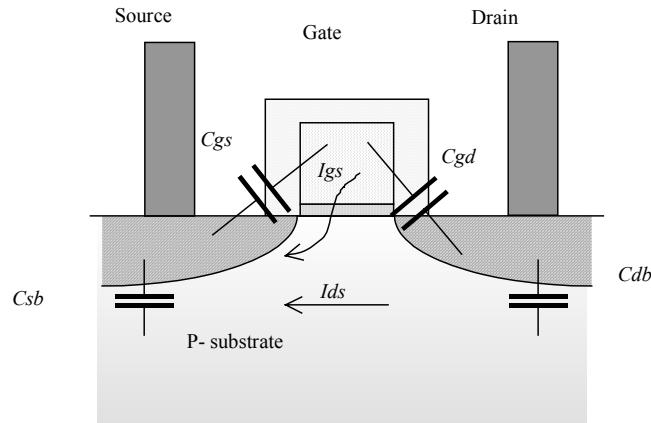


Figure 11-57: Cross-section of the MOS device showing the main parasitic capacitors

$$f_t = \frac{\epsilon_0 \cdot \sigma \cdot \mu_n \cdot W}{2\pi \cdot TOX \cdot L \cdot C_{gs}} (V_{gs} - V_t) \quad (\text{Equ. 11-12})$$

with  $C_{gs} \approx \frac{2}{3}(WLC_{ox})$

with

$\mu_n$  = electron mobility ( $\text{m} \cdot \text{V}^{-2}$ )

W = channel width ( $\mu\text{m}$ )

L = channel length ( $\mu\text{m}$ )

$C_{gs}$  = gate to source capacitance ( $\text{F}$ )

$V_{gs}$  = gate to source voltage (V)

$V_t$  = threshold voltage (V)

Replacing the value of  $C_{gs}$  in the expression of  $f_t$ , the transit frequency becomes independent of the channel width (Equation 11-13). For an n-channel MOS device in  $0.12\mu\text{m}$ , the measured value of  $f_t$  is around 50GHz, for  $V_{gs}$  around  $V_{dd}/2$ . The transit frequency  $f_t$  is an important metric of technology performances for very high frequency circuit design.

$$f_t = \frac{\epsilon_0 \epsilon_r \cdot \mu_n}{\frac{4}{3} \pi^2 \cdot \text{TOX} \cdot L^2 \cdot C_{ox}} (V_{gs} - V_t) \quad (\text{Equ. 11-13})$$

A similar parameter, the cut-off frequency, is the frequency at which the gain starts decreasing. We consider  $G_0$  as the gain for low frequency input signals. The cut-off frequency is usually defined as the frequency when the gain is decreased by 1.4, according to equation 11-14.

$$f_{cut-off} = f(Gain = \frac{G_0}{\sqrt{2}}) \quad (\text{Equ 11-14})$$

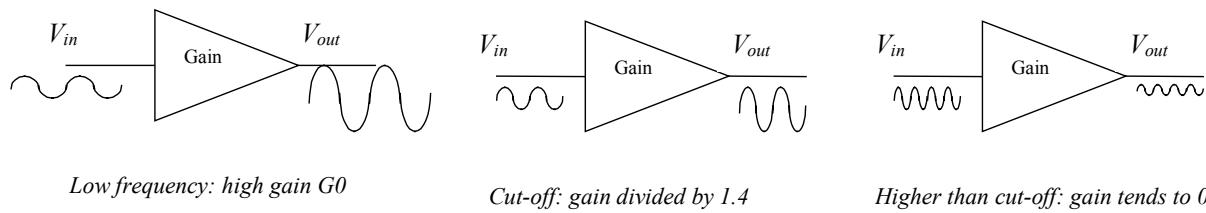


Figure 11-57: The cut-off frequency corresponds to the input frequency where the gain starts to decrease

Microwind does not perform frequency analysis. However, we can create a sinusoidal wave with an increasing frequency, thanks to the parameter **Increase f** of the sinus property assigned to  $V_{in}$  which is fixed in figure 11-58 to 0.2. At each period, the frequency is increased by 20%.

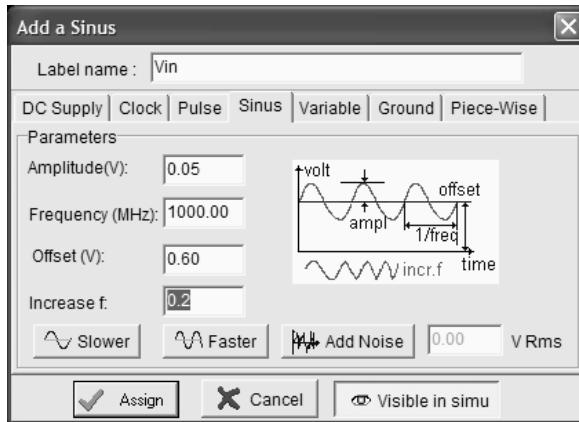


Figure 11-58: Time-domain simulation with a sinusoidal wave with frequency sweep (AmpliSingle.MSK)

The best simulation mode is **Frequency vs. time**. The upper screen displays the input frequency while the lower screen shows the amplitude of the input and output signals (Figure 11-59). There is no precise tool to locate the cut-off frequency. However, we observe a significant decrease of the gain from 2 GHz onwards, when the output is loaded with a 10fF parasitic capacitance. The parasitic capacitance of the output node has a direct impact on the cut-off frequency.

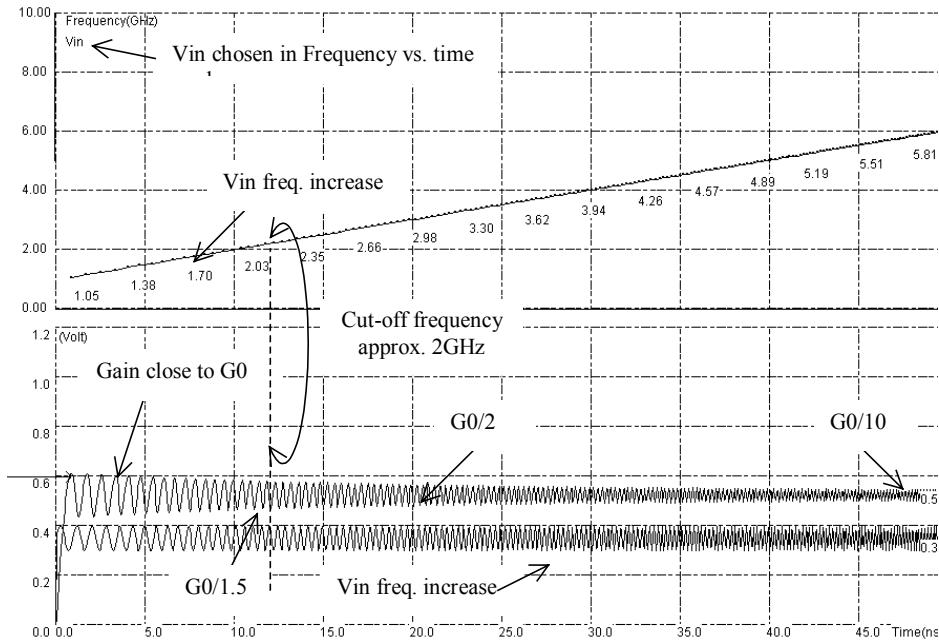


Figure 11-59: Extracting the cut-off frequency from sweep sinusoidal input response (AmpliSingle.MSK)

### The Inverter as an Amplifier?

Could the logic CMOS inverter act as an amplifier? Theoretically yes, as the static characteristics of the CMOS inverter are very much like the static response of the basic amplifier described earlier. Using the mode **Voltage vs. Voltage**, we find a gain of 10 for the basic inverter.

To operate in the amplifier zone, we should inject a signal around VDD/2, otherwise there is no chance of taking advantage of the high amplification. The commutation point varies according to the ratio between the nMOS and pMOS size, as illustrated in figure 11-60 where the static characteristics of three inverters are compared. The BSIM4 model is mandatory for reliable simulations.

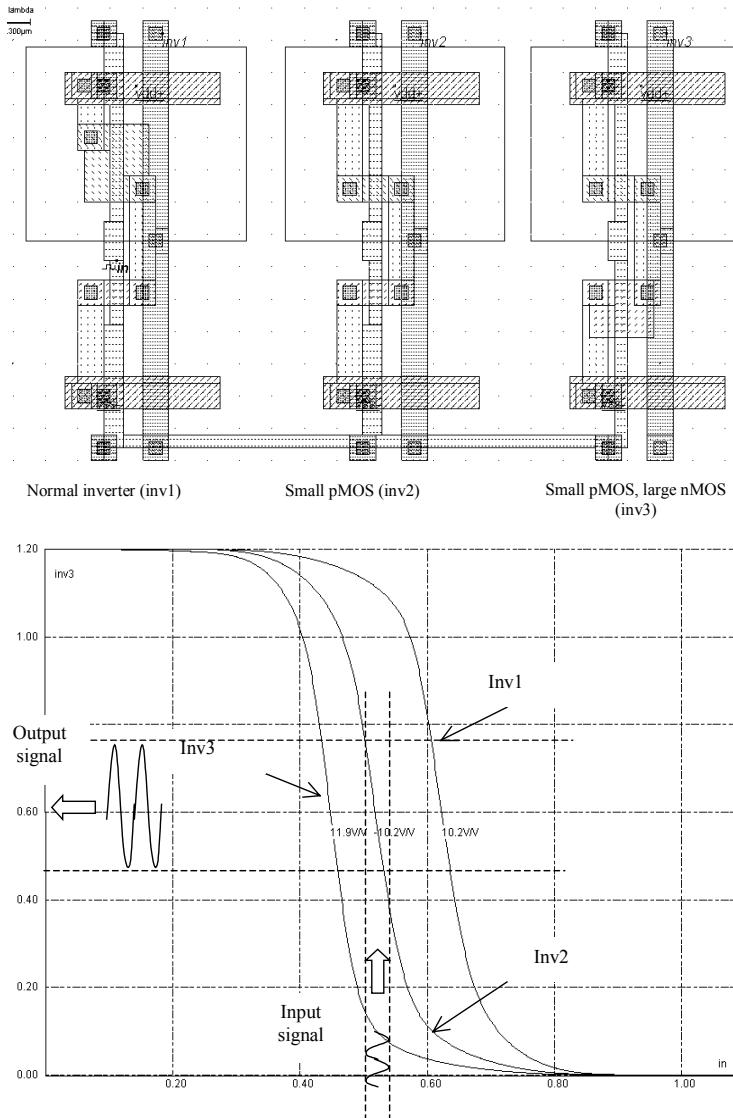


Figure 11-60: Static characteristics of the CMOS inverter in 0.12 $\mu$ m (invAmpli.MSK)

In 0.35 $\mu$ m CMOS technology, the static characteristics of the three inverters exhibit a much higher gain (Figure 11-61). Use the command **File → Select Foundry** and choose **cmos035.RUL** to switch to the 0.35 $\mu$ m technology. Again the BSIM4 model is forced thanks to the label 'BSIM4' in the layout, for accurate results. The measured gain is around 15. As the process parameters are not well controlled, the commutation point of the inverter may fluctuate over a significant range, depending on the location of the die on the wafer, or even on the die itself. As a consequence, placing the input voltage in the exact region of amplification is difficult and requires a specific control circuitry.

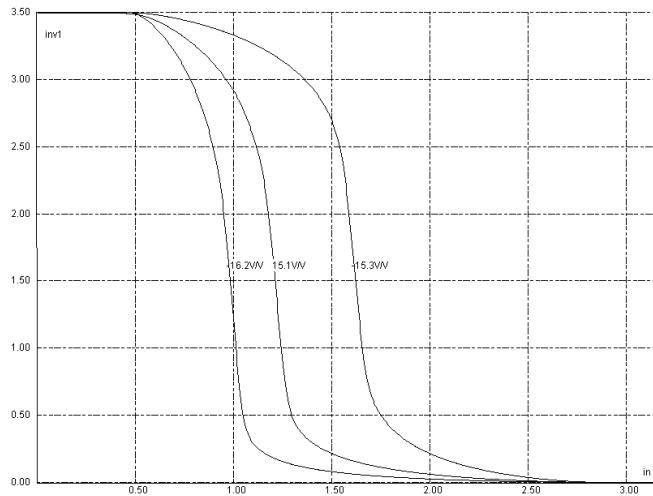


Figure 11-61: CMOS inverter characteristics in  $0.35\mu\text{m}$  exhibit higher gain (invAmpli.MSK)

High gain amplifiers are preferably built from two medium gain stages of amplifiers, rather than one very high gain stage (Figure 11-62). The constraints for the input voltage range are easier to handle in the case of two stage amplifiers. Also, voltages higher than the logic voltage supply are used to increase the voltage range of the circuit. In  $0.12\mu\text{m}$  technology, the majority of analog amplifiers is based on dual-oxide MOS devices, operating at 2.5V.

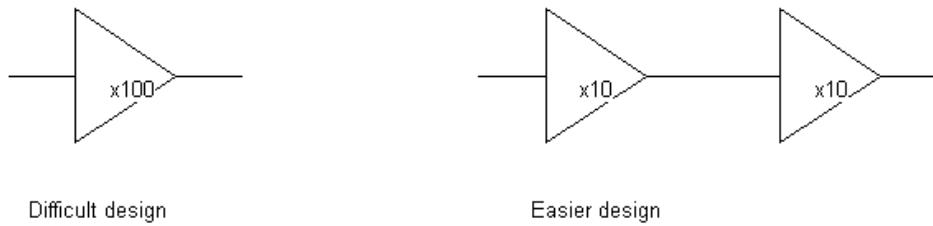


Figure 11-62: High gain is usually achieved by two stages of amplifiers

## 9. 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 11-13). 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.

$$V_{out} = K(V_p - V_m) \quad (\text{Equ. 11-13})$$

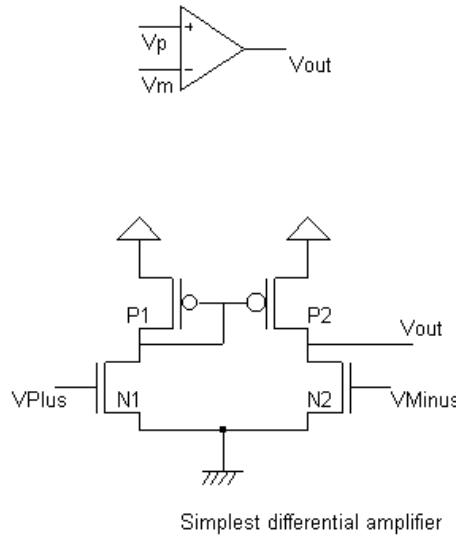
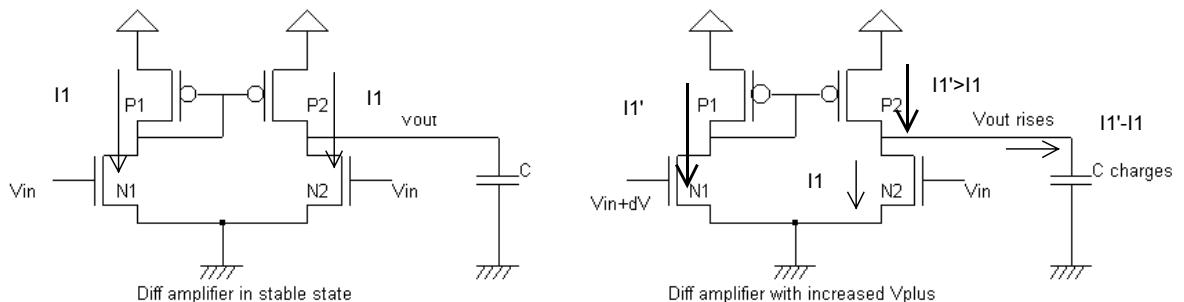


Figure 11-63: Symbol and schematic diagram of the differential amplifier

The usual symbol for the differential amplifier is given at the top of figure 11-63, with its most simple MOS implementation. The differential amplifier principles are illustrated in figure 11-64. 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.



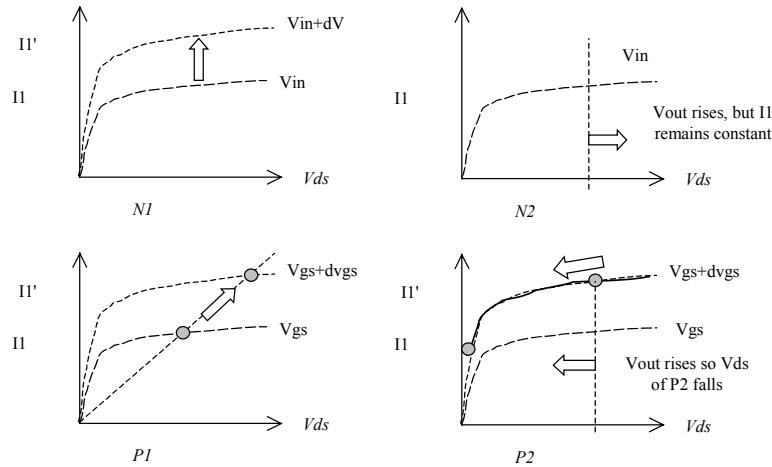


Figure 11-64: The differential amplifier at work (AmpliDiff.SCH)

The process will end when  $V_{out}$  is high enough so that  $P2$  is no more a good current mirror, which means that  $II'$  is finally decreased to  $II$ . The key idea is that a small variation  $dV$  of the input voltage is transformed into a huge variation of the output voltage  $V_{out}$ , which is the definition of a high gain amplifier.

### A Poor Design

A direct translation of the differential amplifier into layout is performed as illustrated in figure 11-65. The differential pair is built from short channel nMOS devices. Their size is kept identical, and drawn with the same orientation, to minimize the offset generated by the transistor mismatch. In the simulation, it can be seen that a 50mV voltage difference between  $V_p$  and  $V_m$  is amplified by the circuit. However, the output is very far from saturation. The gain of the circuit is very small. The main reasons are the use of very short channel MOS devices for which the current mirror performances are quite poor, and the high voltage difference between the drain and source of the differential pair which forces the devices to operate at a low performance regime, with several saturation effects.

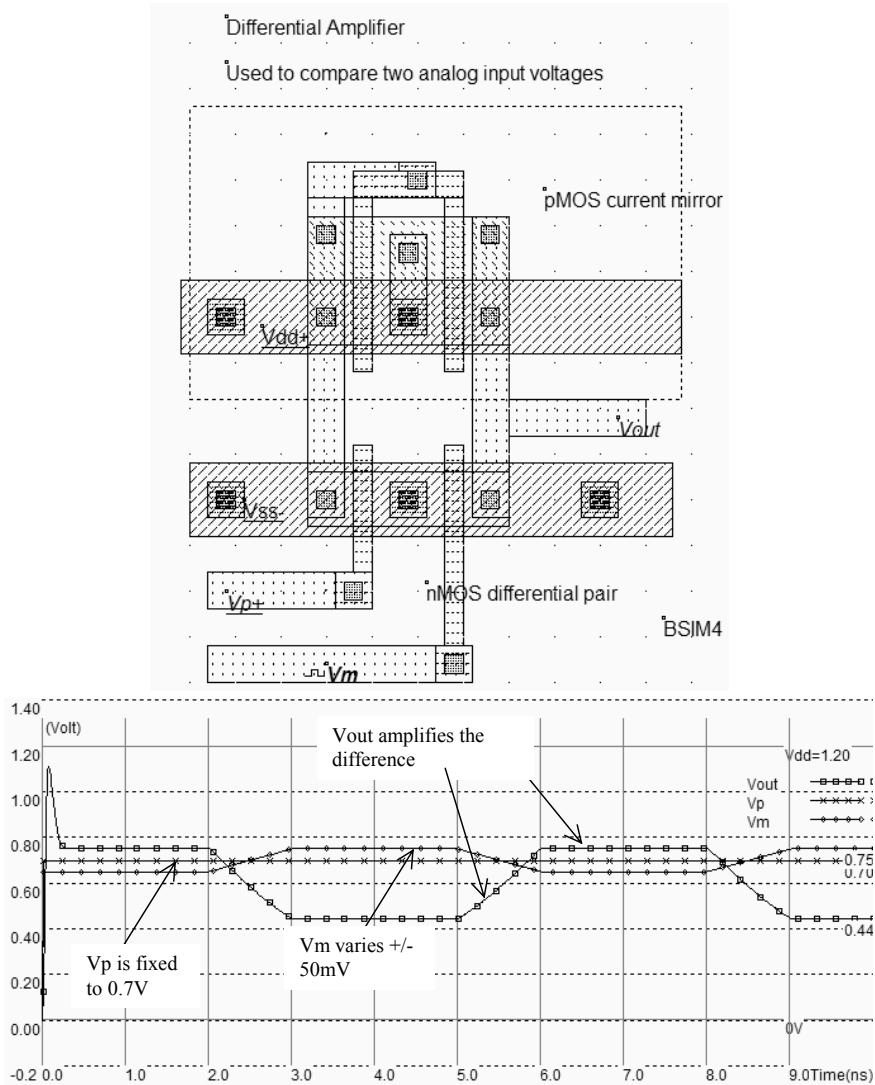


Figure 11-65: Layout and transient simulation of the differential amplifier (AmpliDiff.MSK)

The mismatch between Level 3 and BSIM4 models is quite important in this particular circuit. This is why it is highly recommended to use BSIM4 to simulate properly the performances of analog circuits. Remember that a convenient way of forcing Microwind to use BSIM4 instead of the default Model 3 is to add a text in the layout starting with "BSIM4".

### Measure the Gain

The gain of the differential amplifier is the  $K$  factor appearing in the equation 11-13. This equation is only valid for small differences between  $V_p$  and  $V_m$ , otherwise  $V_{out}$  saturates near  $V_{SS}$  or near  $V_{DD}$ .

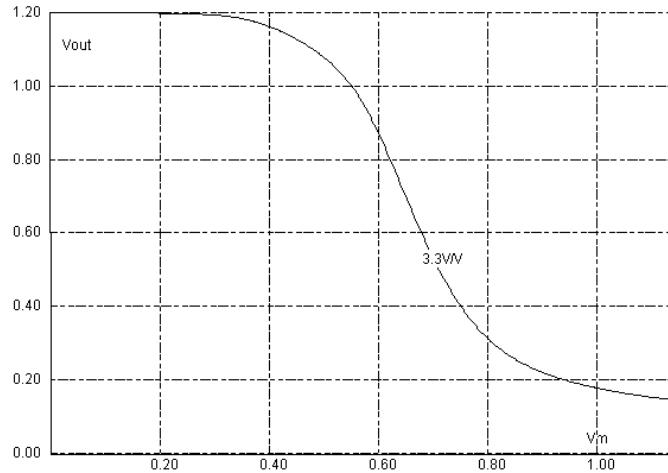


Figure 11-66: Computing the gain of the differential amplifier (AmpliDiff.MSK)

The gain can be computed by Microwind in the Voltage vs. Voltage simulation mode, by selecting the item **Slope** appearing in the menu **Evaluate**. Once the static characteristics of the differential amplifier are obtained, the gain is extracted at the crossing of  $VDD/2$ . The gain is very low :  $3.3V/V$  (Figure 11-66).

### Improving the Amplifier

The first action consists in the use of long channel MOS device which suffers less channel modulation effects. This is proposed in the layout shown in figure 11-67. The second action consists in inserting an nMOS device between the differential pair and the ground. The gate voltage  $Vbias$  controls the amount of current that can flow on the two branches. This pass transistor permits the differential pair to operate at lower  $Vds$ , which means better analog performances and less saturation effects.

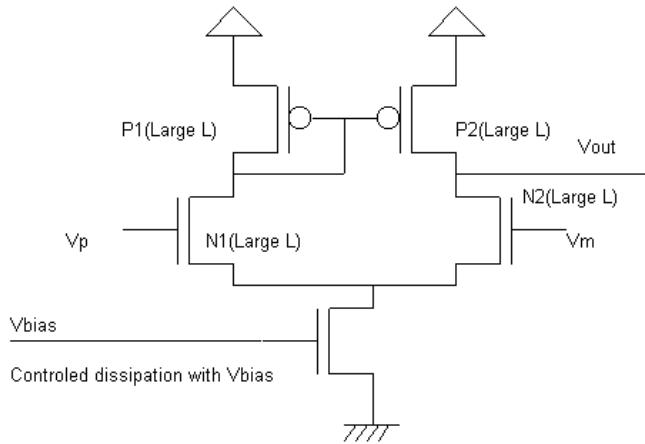


Figure 11-67: An improved differential amplifier (AmpliDiff.SCH)

## Find the input Range

The best way to measure the input range is to connect the differential amplifier as a follower, that is  $V_{out}$  connected to  $V_m$ , as shown in figure 11-68. 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  $V_m$ . The new differential amplifier layout is shown in figure 11-68. 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.

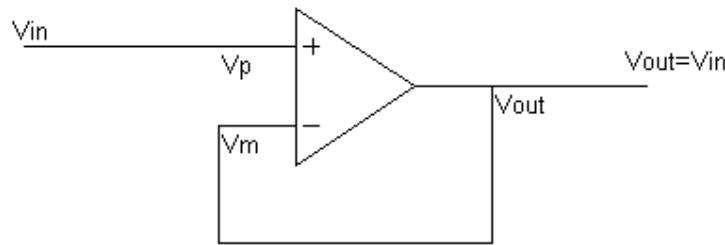


Figure 11-68: The connection between  $V_{out}$  and  $V_m$  creates a follower (AmpliDiff2.MSK)

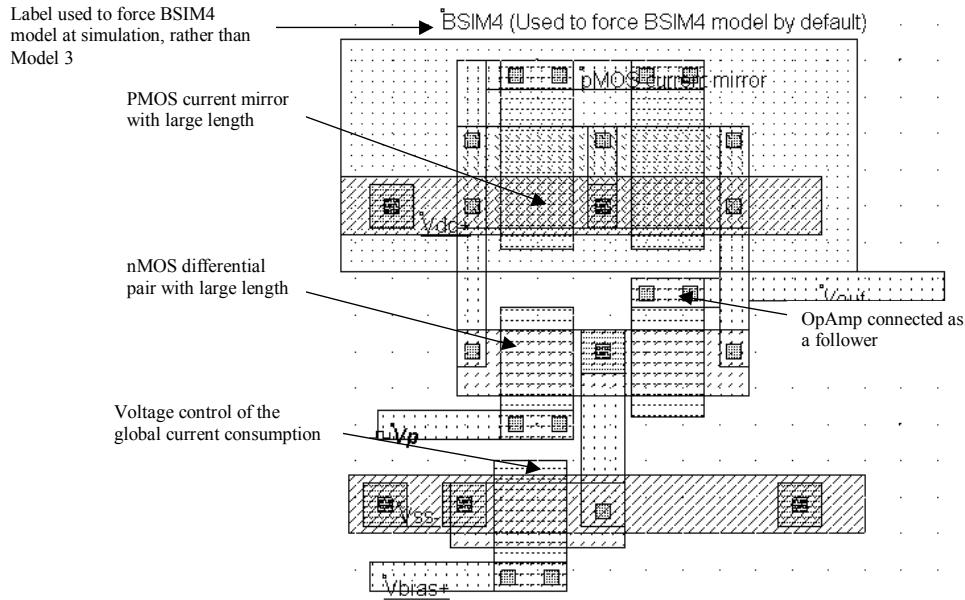


Figure 11-69: The layout corresponding to the improved differential amplifier (AmpliDiffLargeLength.SCH)

One convenient way of simulating 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 11-70, 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.

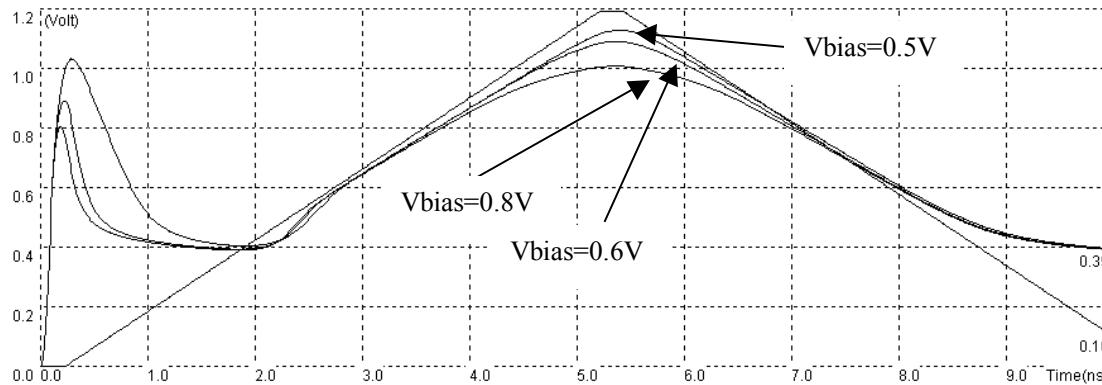


Figure 11-70: Effect of  $V_{bias}$  on the differential amplifier performance (AmpliDiffVbias.MSK)

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.

The differential amplifier may be constructed using nMOS devices for the current mirror and pMOS devices for the differential pair. This circuit, proposed in figure 11-71, features a symmetrical behavior, that is a good follower performance for low voltages and an intrinsic limitation near  $VDD-0.4V$ .

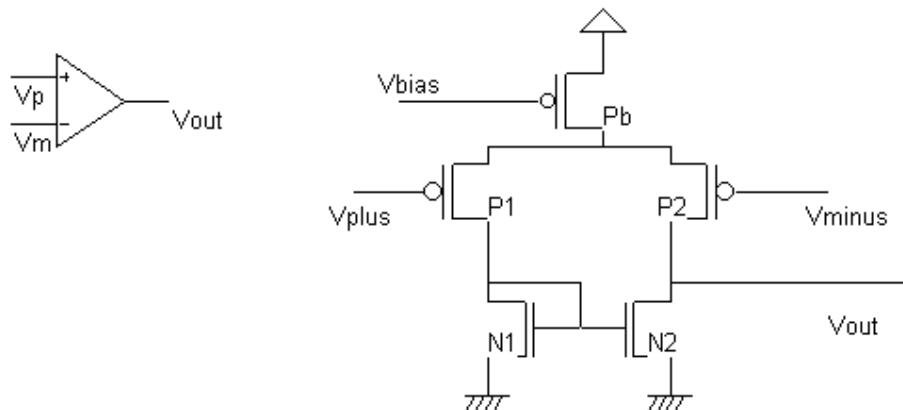


Figure 11-71: A differential amplifier based on a pMOS differential pair and an nMOS current mirror

### Double Differential Amplifier

The double differential amplifier is built using the two previous differential amplifiers connected to a common output. This circuit is valid because the output stage works alternatively: one for the high voltages, one for the low voltages. The result, shown in figure 11-72, is quite correct: the follower copies the input with a reduced error, from 0.1V to 1.1V, that is 100mV close to the supply voltage. This amplifier is close to a rail-to-rail operational amplifier that may be found in most CMOS analog cell libraries.

Still the layout is incomplete: neither dummy devices nor proper isolation circuits have been placed. The devices could also be arranged in a more compact way, to decrease the silicon area. Finally, to further decrease the channel length modulation effect, MOS devices with larger length could be used. The speed performances could be kept identical with an increased width, at the price of a larger silicon area.

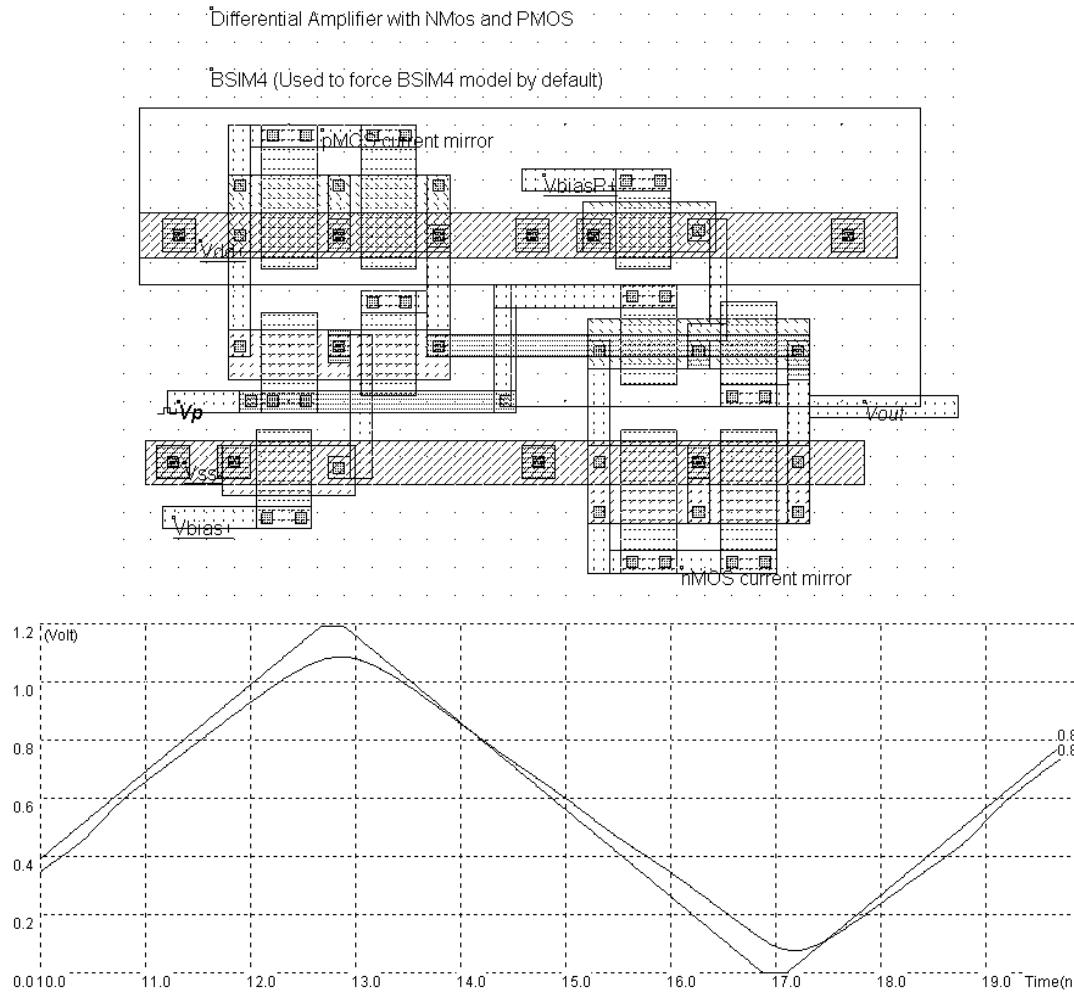


Figure 11-73: Two differential amplifiers with symmetrical structure to enhance performances (AmpliDiffNP.MSK)

### 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 11-74, 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  $N_1$  and  $N_2$ . This time, two stages of current mirrors are used:  $P_1$ ,  $P_2$  and  $P_3$ ,  $P_O$ .

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

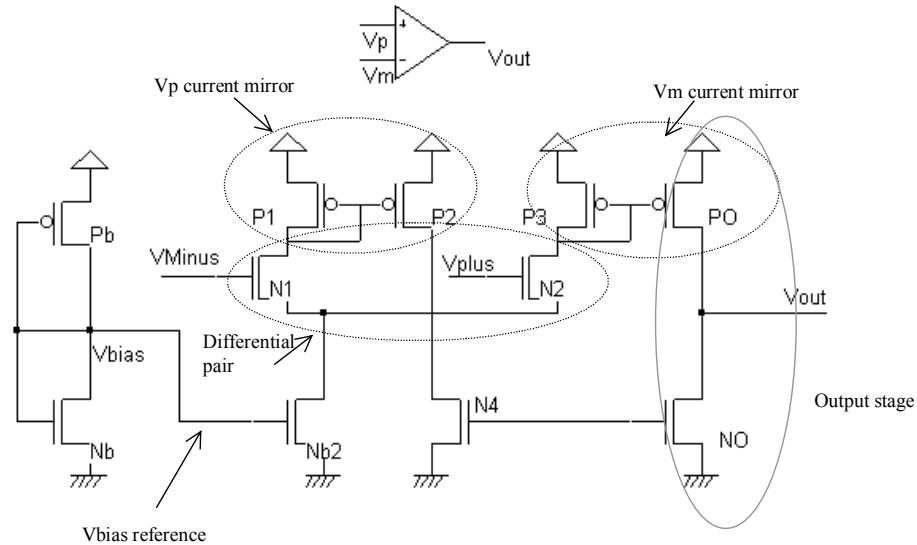
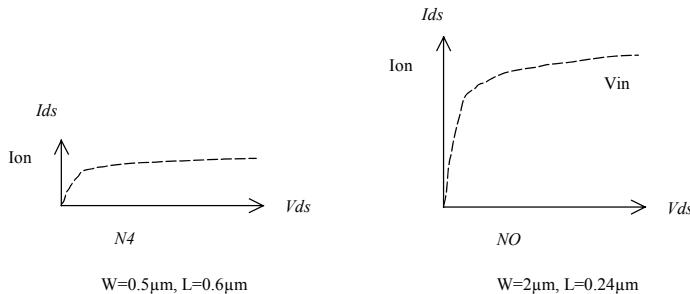


Figure 11-74: Push-pull operational amplifier (AmpliPushPull.SCH)

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 11-75 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.



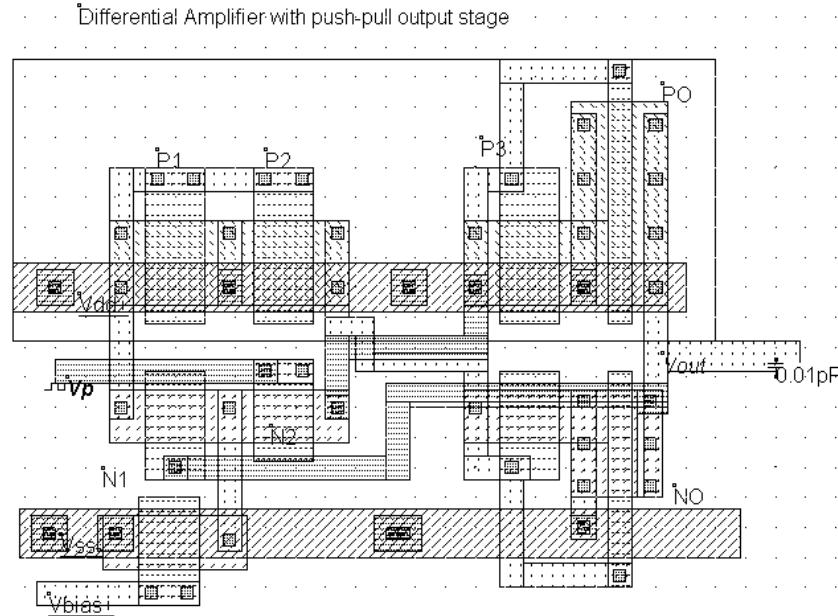


Figure 11-75: Push-pull operational amplifier (AmpliDiffPushPull.MSK)

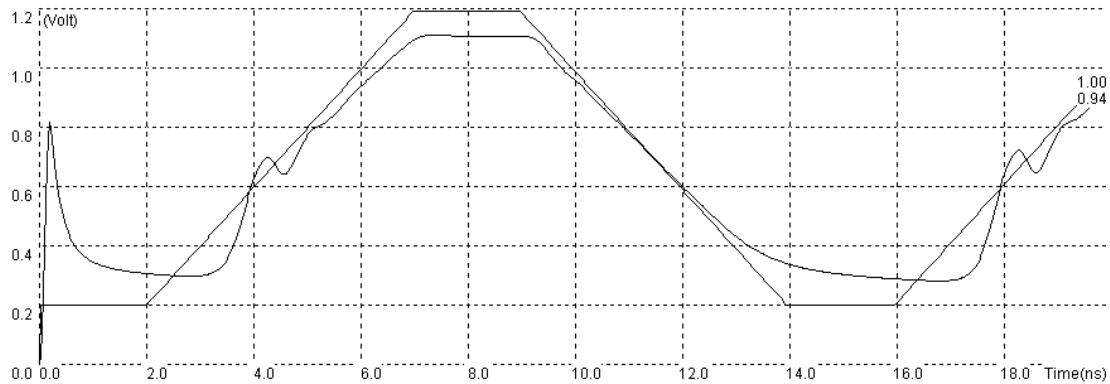


Figure 11-76: Simulation of the push-pull operational amplifier (AmpliDiffPushPull.MSK)

The transient simulation (Figure 11-76) 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 is not dangerous in itself. However, it signifies that the output stage is too strong compared to its charge. If you use the **Voltage vs. Voltage** simulation mode to get the transfer characteristics  $V_{out}/V_+$ , you may see the consequence of the oscillation effect : the simulator hardly converges to a stable result. Increasing the precision does not improve the design significantly (Figure 11-77).

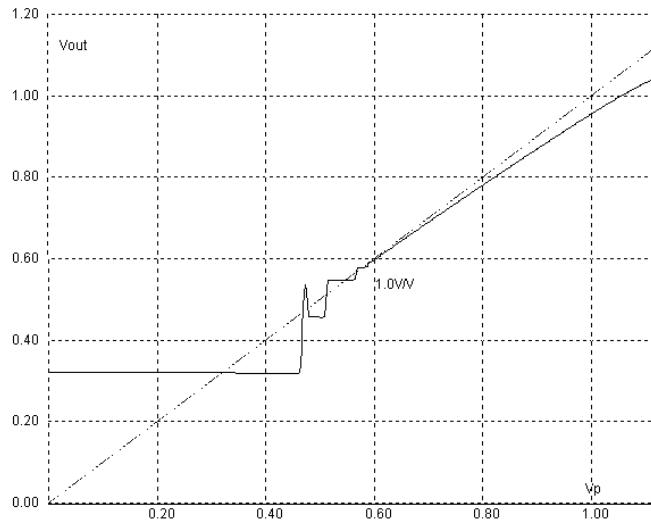


Figure 11-77: The oscillation of the push-pull operational amplifier is also observed when trying to obtain the transfer characteristics of the circuit (AmpliDiffPushPull.MSK)

## 10. Wide Range Amplifier

An other popular operational amplifier design is shown in figure 11-78. The amplifier is built using a classical voltage comparator and a power output stage similar to the push-pull circuit. However, the pMOS device  $PO$  has a constant gate voltage, thus acting as a current generator, while the nMOS device  $NO$  is controlled as seen before.

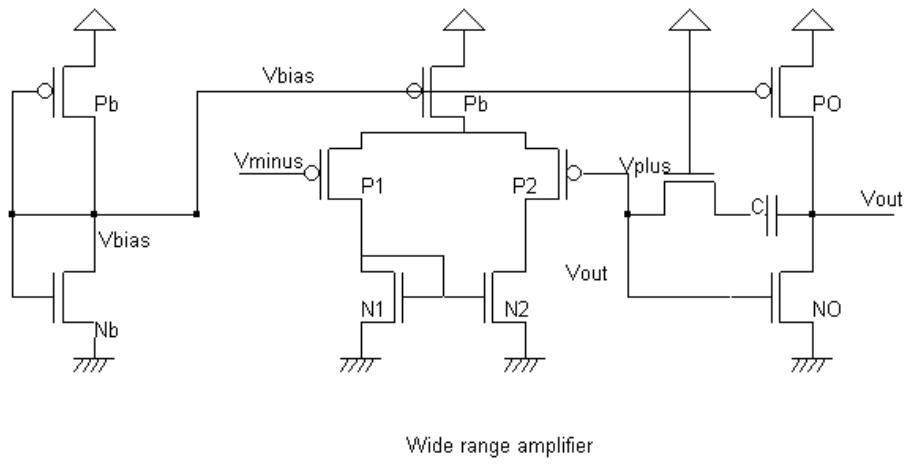


Figure 11-78. Schematic diagram of the wide range amplifier

The circuit shown in figure 11-79 has been implemented in a  $0.35\mu\text{m}$  CMOS process. The corresponding layout is reported in figure 11-80. Not all design rules for a high quality analog design have been observed at that time:

- The orientation of the upper pMOS is not identical. This may impact the quality of the mirror (Design warning 1)
- The differential pairs use channels with minimum length. This increases strongly the second order effects, the offset and non-linearities (Design warning 2)

- No dummy device has been used to improve the quality of the differential pair response (Design warning 3)
- The arrangement is far from being optimal. A lot of silicon area remains unused (Design warning 4)

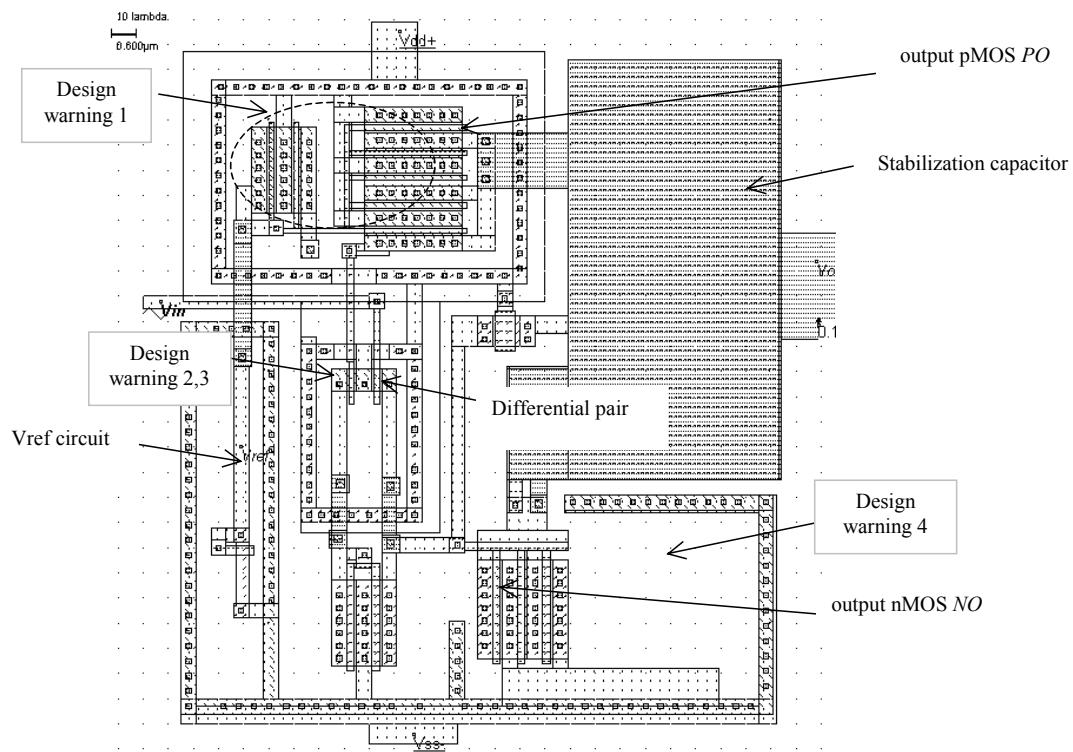


Figure 11-79. Implementation of the wide range amplifier using a  $0.35\mu\text{m}$  CMOS technology (AmpliWide.MSK)

To minimize the parasitic offset, the critical MOS devices ( $N1, N2, P1$  and  $P2$ ) should be divided into sub-elements  $N1a, N1b$ , etc.. and placed in a centroid geometry as illustrated in figure 11-80. Dummy elements are also added around the amplifier. The effects of process non-uniformity are efficiently compensated and parasitic electrical effects are consequently reduced [Haraszti].

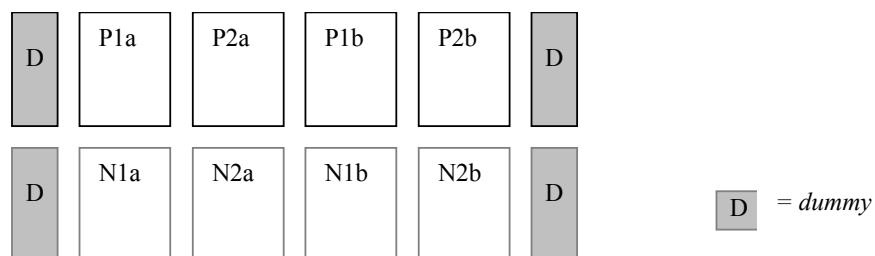


Figure 11-80: Design efforts to limit the impact of process variations on the sense amplifier

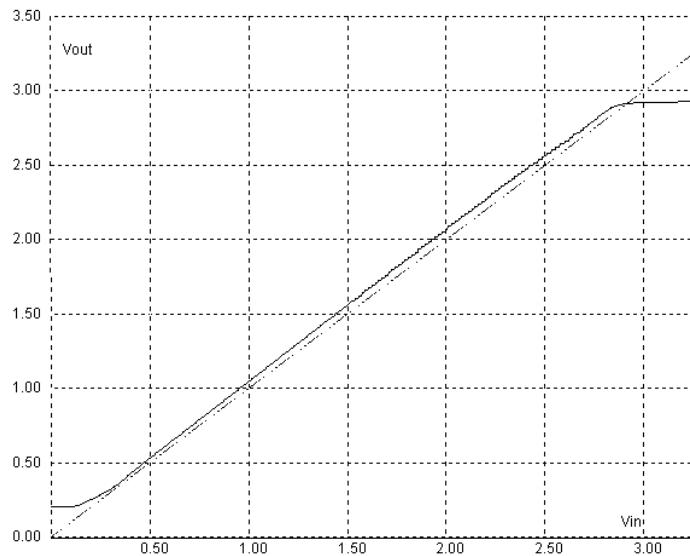
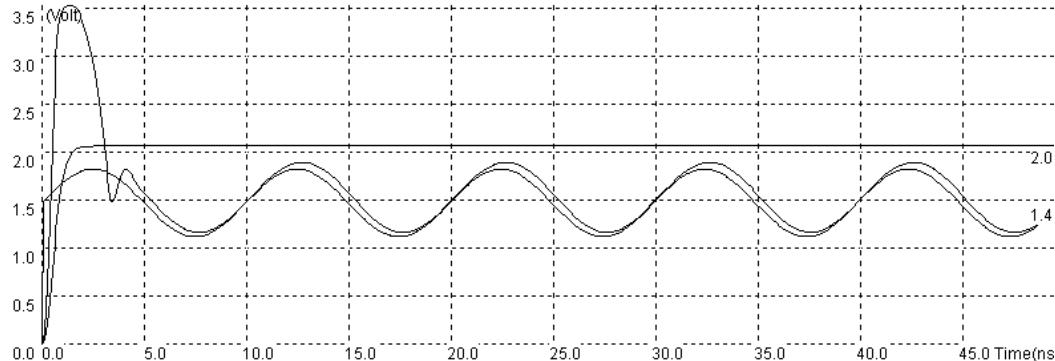
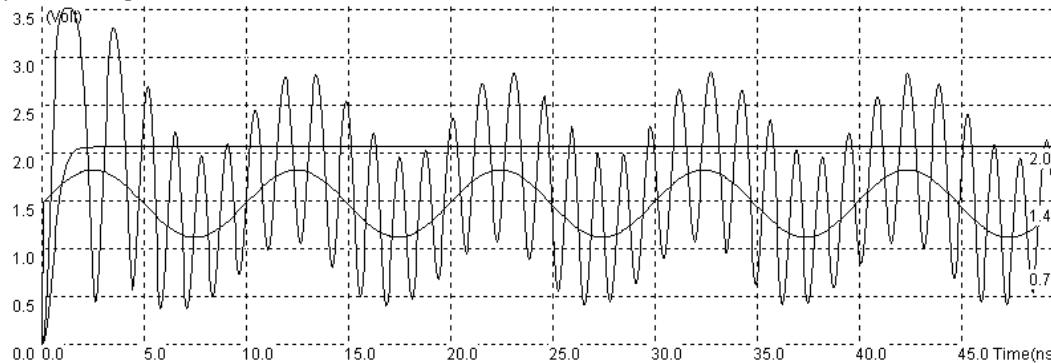


Figure 11-79. Simulation of the wide range amplifier in  $0.35\mu\text{m}$  CMOS technology (AmpliWide.MSK)

The simulation of the wide range amplifier is reported in figure 11-79. It matches very nicely with real case measurements made on a CMOS test chip fabricated in  $0.35\mu\text{m}$ , which included the circuit without any modification, except of course the addition of supply rails and input/output pads.



(a) With feedback capacitor



(b) Without feedback capacitor

Figure 11-80: The stabilization circuit is required to ensure a correct follower response and avoid instability (AmpliWide.MSK)

It is important to point out that the compensation capacitor has a very important role. When the stabilization circuit is active, we notice almost no ringing effect except at the early stages when the circuit is turned on (Figure 11-80-a). In contrast, if we delete one connection in the stabilization circuit, an enormous ringing effect is superimposed to the output voltage (Figure 11-80-b).

## 11. On-chip Voltage Regulator

In deep sub-micron technology, the use of very thin gate oxide implies a low supply voltage. This supply decrease is mainly due to the increased risk of damaging the oxide that separates the gate from the drain and source regions, when high voltage differences are present. In contrast, the input/output interface of the integrated circuit must meet standard requirements in terms of voltage, basically 5V or 3.3 V supply. This means that a specific circuit must be designed to generate a low voltage source internally from an external high voltage supply.

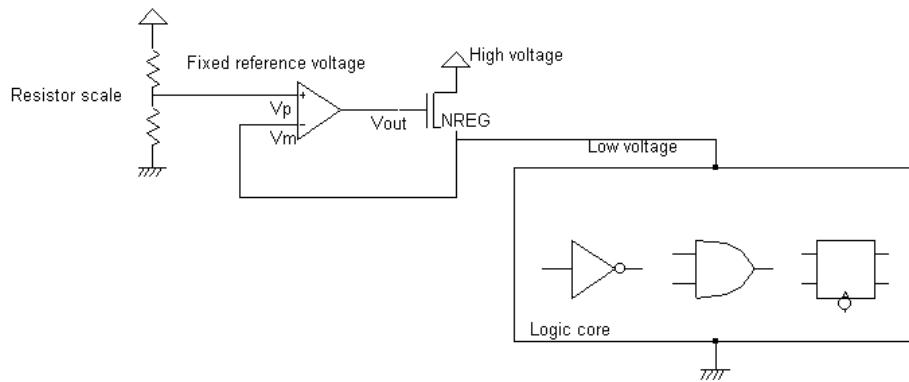


Figure 11-81: Principles of an on-chip voltage regulator based on an operational amplifier

A circuit that realizes the voltage shift is proposed in figure 11-81. The basic idea consists in using an operational amplifier to control the gate of an n-channel MOS device. When the logic core switches, the core voltage connected to  $V_m$  is lowered. Consequently the operational amplifier tends to increase  $V_{out}$ , which reduces the NREG device resistance, and increases the voltage until the reference voltage is attained. The negative feedback creates a stable feedback loop to recover from the voltage drops during the switching of circuits.

From a design point of view, the NREG device must be designed very large, even when the supplied logic circuit is small (Figure 11-82). If the equivalent width of the NREG is too small, the gate voltage saturates and the feedback is inefficient. In the case of figure 11-82, the initial design using a width of 100 lambda did not work properly. Using a 200 lambda MOS device, the regulation is effective.

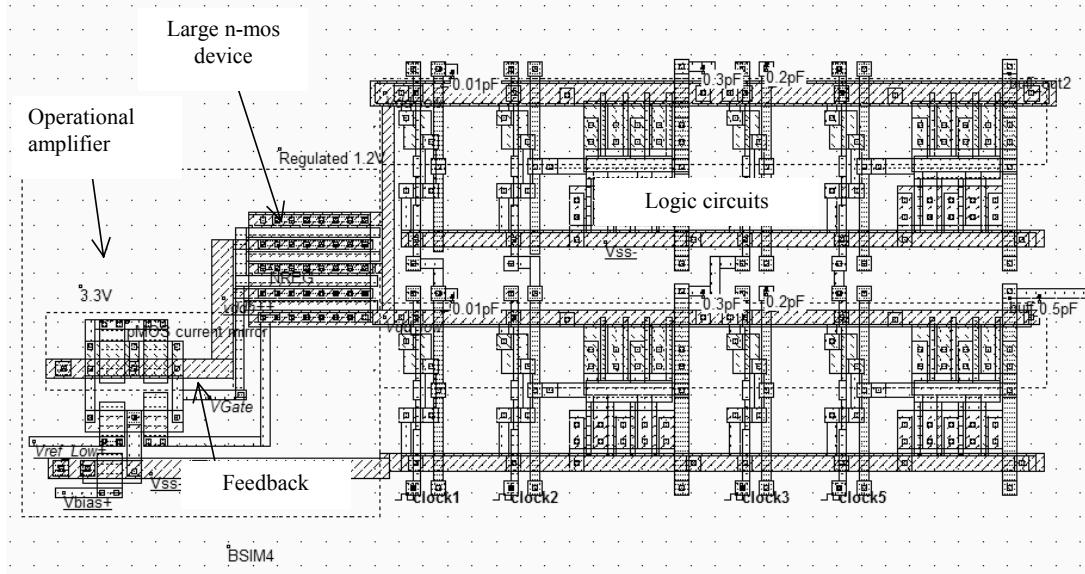


Figure 11-82 : Implementing a small on-chip voltage regulator(Vreg.MSK)

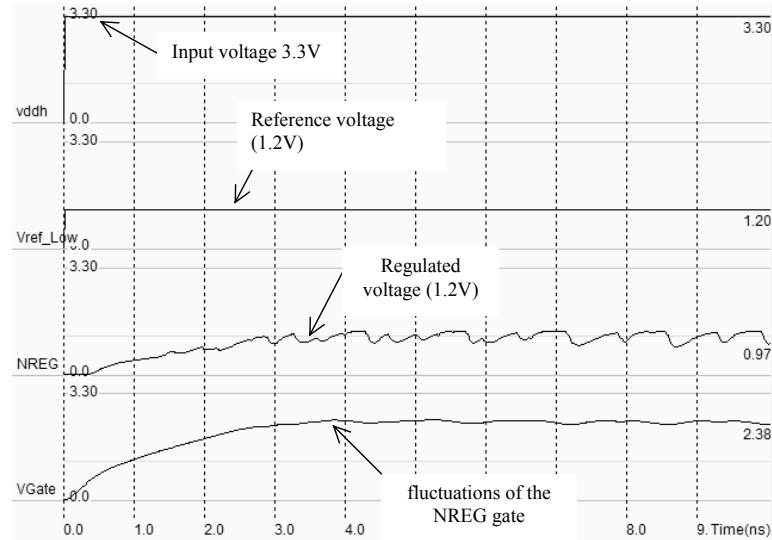


Figure 11-83 : The on-chip voltage regulator at work during multiple transitions of the logic core (Vreg.MSK)

In the example shown in figure 11-83, the voltage  $V_{ref\_Low}$  is fixed to approximately 1.2V. The high voltage supply is 3.3V, which requires high voltage MOS devices for the amplifier and the regulator device  $NREG$ . When a strong current flows due to concurrent switching in the core, the regulator reacts quite rapidly. In the case of very large logic core, the size of the pass transistor is increased to very impressive values : the combined width may be larger than a millimeter. Such a giant MOS device is obtained by placing MOS gates in parallel. A design example is shown in figure 11-84, which has a width of 1mm, with an  $I_{on}$  current around 600mA.

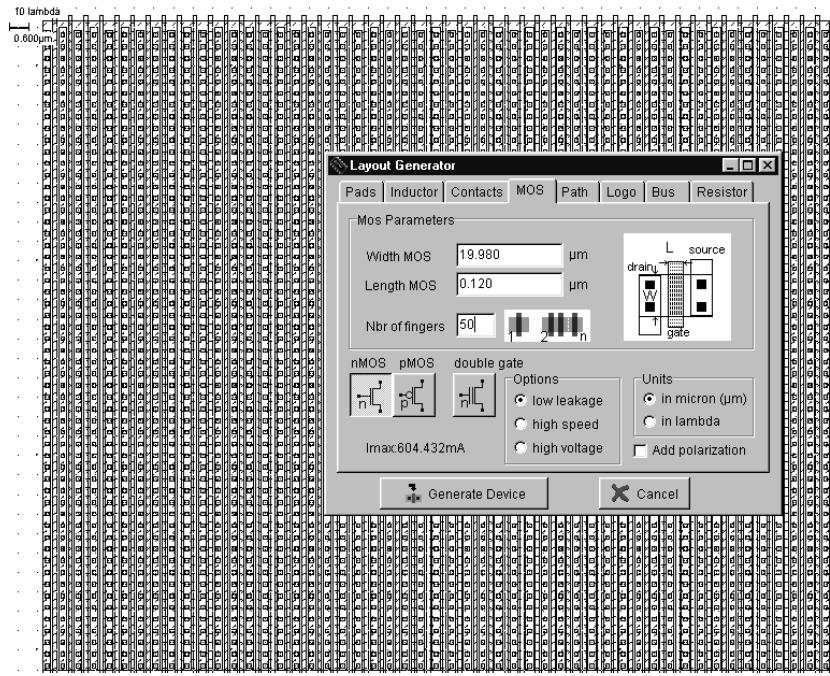


Figure 11-84 : Giant MOS built from MOS devices in parallel, to regulate very high density core circuits  
(VregBigMos.MSK)

## 12. Noise

The random motion of electrons in conductors create an unwanted signal called noise. The two major sources of noise in CMOS circuits are the resistors and MOS devices. The thermal noise is a very important parasitic effect in resistor. The parasitic voltage due to thermal noise is shown in figure 11-85. Without thermal noise in resistors, the voltage should be constant. Taking into account the thermal noise, a small random fluctuation is observed.

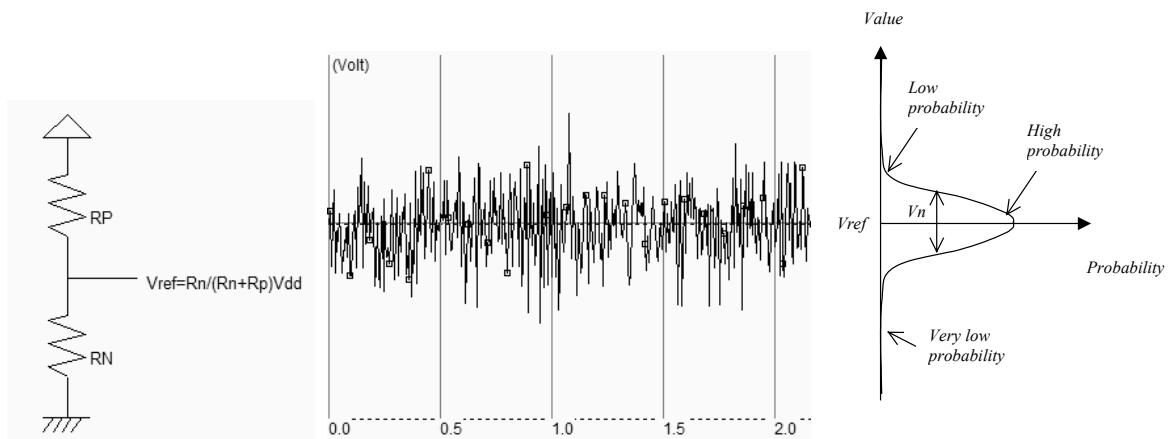


Figure 11-85 : Thermal noise created by large resistances

The voltage oscillates around the desired voltage with a Gaussian distribution. The amplitude of the noise is linearly proportional to temperature and to the value of the resistor. The noise is almost independent of the nature of the resistor material and the frequency.

The signal is not periodic due to its random nature. However, its properties are well predictable when we consider a long time and average parameters. It can be seen in a spectrum analyzer that the energy is homogenously spread over the whole range of measurable frequencies (Typically 1KHz, 10GHz). The thermal noise is considered as a "white noise" by analogy to light where all colors added together create a white color. As the thermal noise is Gaussian distributed, it is called a "Gaussian white noise". Its spectral power density (SPD <gloss>) can be approximated by the following formulation.

$$SPD \approx 4kTR \quad (\text{Equ. 11-14})$$

where

$k$ =Boltzmann constant =  $1.38 \cdot 10^{-23} \text{ J/K}$

$T$ =temperature ( $^{\circ}\text{K}$ )

$R$ =resistance value (Ohm)

An example of measured noise power density is given in figure 11-86. The thermal noise appears as predicted by the formulation 11-14, at a level approaching  $10^{-17} \text{ V}^2/\text{Hz}$ . A new noise source is also found at low frequencies. The noise is called  $1/f$  noise as it is proportional to  $1/f$ . We also observe other important noise around 100MHz, which corresponds to the FM radio emission. Any discrete device with dimensions larger than some millimeter acts as an antenna than captures a small portion of radio-frequency signals. The same effect is observed in mobile phone bands (900MHz, 1800MHz).

The MOS device has significant serial access resistance on the drain, source, gate and channels. Consequently, the MOS device is also an important source of noise. There is no noise model in Microwind MOS devices. But the noise analysis may be performed in SPICE simulations, as the noise sources and parameters are extensively described [WinSpice].

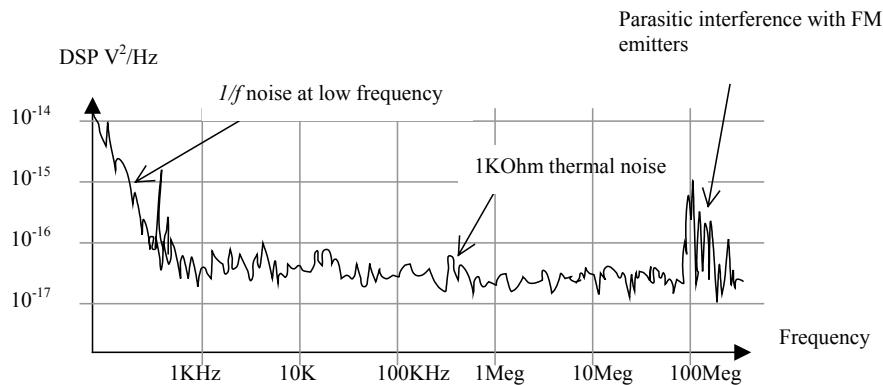


Figure 11-86 : Measured thermal noise on a 10K resistor

Low noise design [Ref book low noise] refers to specific techniques which try to limit the noise effect and its possible consequences on analog signal quality. The most efficient techniques consists in replacing resistors by inductors, by minimizing the values of resistors when such components are absolutely required, and by using MOS devices with large channel length, which have better noise performances than short channel devices.

## 13. Conclusion

Several aspects of analog design have been described in this chapter. First we detailed the implementation of resistor and capacitor elements. Secondly, we focused on several analog properties of the MOS device, as an analog switch and a high value resistance. We created voltage references, and analyzed the impact of the device size, temperature and shielding. In a third part, the current mirror was presented and simulated. Some guidelines for device matching were illustrated. The transient and frequency characteristics of the single stage amplifier were also studied, and several designs for the differential amplifier were reviewed, including a rail-to-rail amplifier. The gain and input range were characterized for each design. An example of fabricated wide-range amplifier was proposed. Finally, we detailed an on-chip voltage regulator, with a focus on very large MOS devices.

### Exercises

#### Exercise 11.1

Considering the switch N1 off and switch N2 on, what is the value of the inverter output and input? When N1 is turned on and N2 is turned off, what is the value of the output if the input rises from  $V_{in}$  to  $V_{in}+dV$ ? We assume an inverter slope with a maximum gain  $G$  of -5.

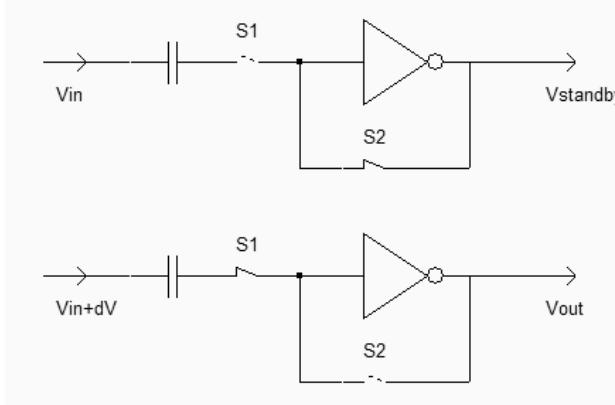


Figure 11-86 : Exercises 11-1

#### Exercise 11.2

The cascode current mirror [Gregorian p59] 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.

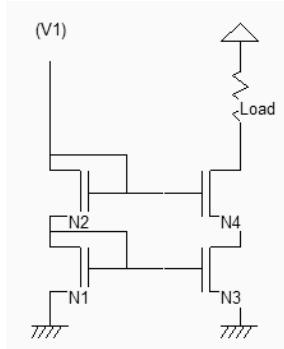


Figure 11-87 : The cascode current mirror

### Exercise 11.3

Let us consider the amplifier of figure 11-88 [Gregorian p108].

- What is the typical value for  $V_o$ ? What is the impact of a change in the voltage  $V_o$ ?
- If the width of  $P_2$   $W_{P_2}$  is  $10 \times W_{P_1}$ , and  $P_3$  is  $50 \times W_{P_1}$ , what is the value of the current flowing through  $P_2$  and  $P_3$ , compared to  $I_1$  crossing  $P_1$ ?
- Where are the  $V_-$  and  $V_+$  inputs located?
- What is the role of  $P_4$  and  $P_5$ ?
- What is the equivalent function for  $N_1$  and  $N_2$ ?
- Locate the two stage output driver.
- Design the operational amplifier and extract the gain.

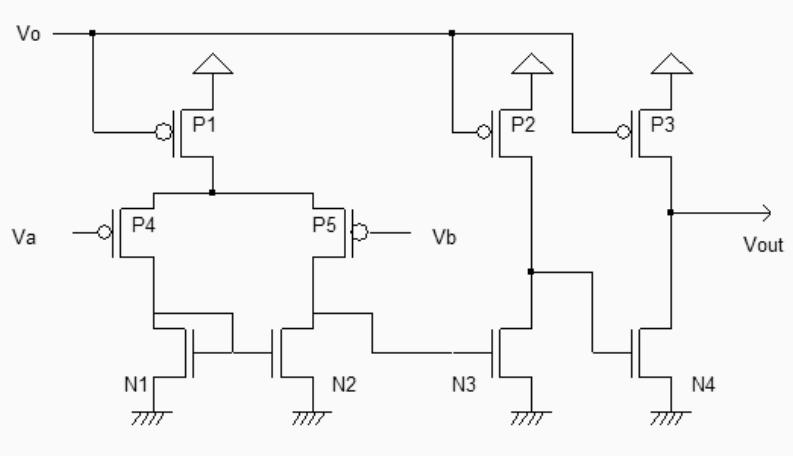


Figure 11-88 : An operational amplifier

[Gregorian p108] has several advantages over the simple current mirror. The output

### References

- [Goyal] Goyal R. "High Frequency Analog Integrated Circuit Design", Wiley, 1995, ISBN 0-471-53043-3
- [Hastings] Alan Hastings "The art of Analog Layout", Prentice Hall, 2001, ISBN 0-13-087061-7
- [Gregorian] tbd
- [WinSpice] tbd



# 12 Radio-Frequency Circuits

Wireless communication systems require specific radio-frequency integrated circuits, which means optimum performances. The radio-frequency integrated circuits have to deal with traditional requirements such as low power consumption or high speed , but also with low process variation influence, power efficiency, linearity, low temperature influence, and low noise sensitivity. This chapter describes the general context of radio-frequency circuit design, integrated LC resonators, power amplifiers, high performance oscillators and frequency up/down converters.

## 1. Target Radio-Frequencies

Application	GSM	DECT	UMTS	Bluetooth	IEEE 820.11a	IEEE 820.11b
Description	Mobile phone 1 <sup>st</sup> generation	Mobile phone 2nd generation	Mobile phone 3rd generation	Wireless network	Very high rate wireless networking	High rate wireless networking
Frequency (MHz)	890–915	1880–1900	1910–2200	2450	5200	2450
Data rate	12Kb/s	100Kb/s	0.1–2Mb/s	<Etienne>	6–18Mb/s	1–5Mb/s
Output Power	1–2 Watts	100mW	1 watt	100mW	0.1–1 Watt	0.1–1Watt

Table 12-1 : Main applications using radio-frequency Ics

Modern radio frequency equipments operate at frequency ranges officially called ultra-high frequencies (UHF) ranging from 300MHz to 3GHz, and super high frequencies (SHF) ranging from 3GHz to 30GHz. The “HF” bandwidth designates the bandwidth 3-30MHz. Mobiles phones and wireless networking have been the driving applications of radio-frequency integrated circuits, as described in figure 12-1.

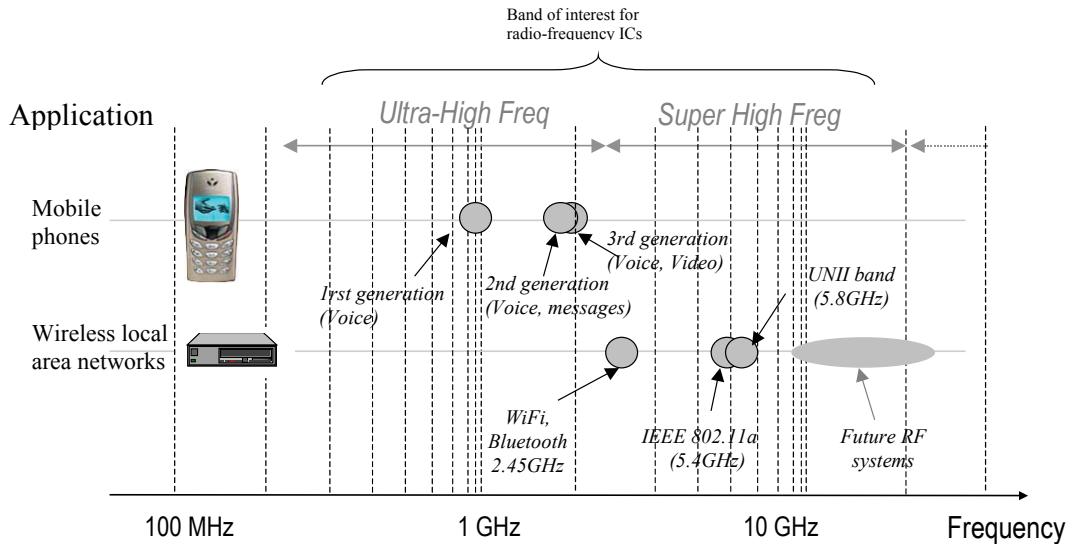


Figure 12-1. Some key radio-frequency applications

The general diagram of a mobile phone (Also called Universal Mobile Telecommunication System UMTS <Gloss>) is given in figure 12-2. The circuits detailed in this chapter refer mainly to the oscillators (VCO), the amplifiers and the filters.

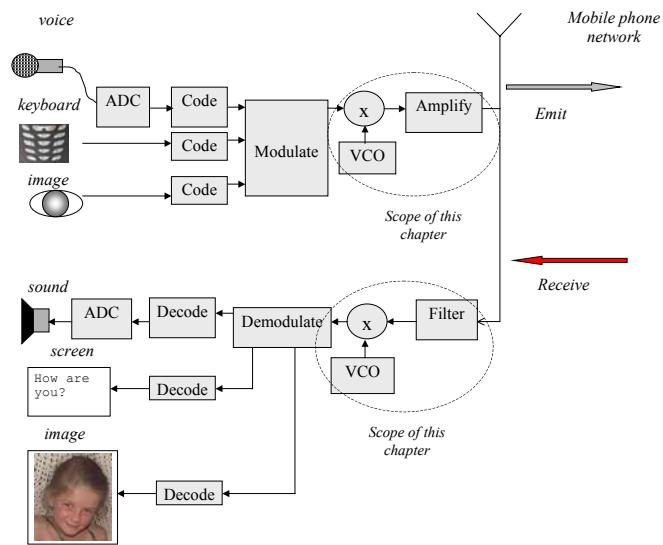


Figure 12-2. Generic diagram of the mobile phone structure

## 2. Inductor

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 12-3).

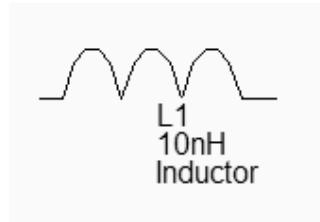


Figure 12-3. The inductance symbol

The layout of an on-chip inductor is typically a square spiral, since standard CMOS processes constrain all angles to be 90° (Figure 12-4). When possible, a polygon spiral using 45° tracks is used to increase the electrical performances of the inductor.

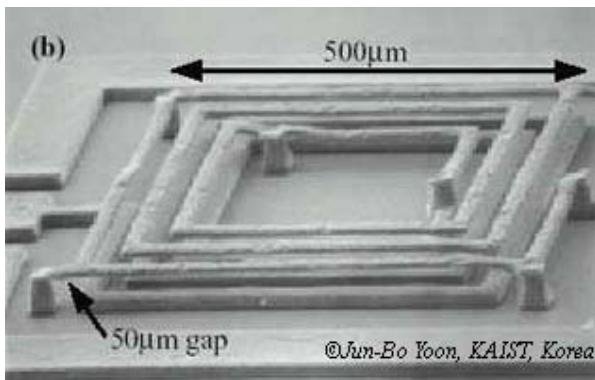


Figure 12-4. An integrated inductor

There exist a huge number of inductance calculation techniques, as detailed in the review from [Thompson]. A very interesting discussion about square planar spiral inductor may be found in [Lee]. The inductance formula used in Microwind (Equation 12-1) is one of the most widely known approximation, proposed as early as 1928 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  $L=11.6\text{nH}$ .

$$L = 37.5 \mu_0 \cdot \frac{n^2 \cdot a^2}{(22.r - 14.a)} \quad (\text{Eqn. 12-1})$$

with

$$r = n.(w + s)$$

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

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 12-5. Consequently, the Q factor is approximately given by equation 12-2.

$$Q = \frac{\sqrt{\frac{L1}{(C1 + C2)}}}{R1} \quad (\text{Eqn. 12-2})$$

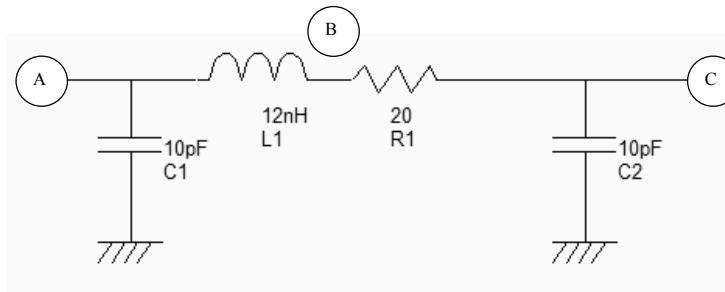


Figure 12-5. The equivalent model of the 12nH default coil and the approximation of the quality factor Q

### 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 12-6. Notice the virtual inductance (L1) and resistance (R1) symbols placed in the layout. These symbols indicate to the extraction that three separate electrical nodes are requested (A,B and C), with a serial inductor 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. Only the capacitance (C1, C2) would be properly extracted.

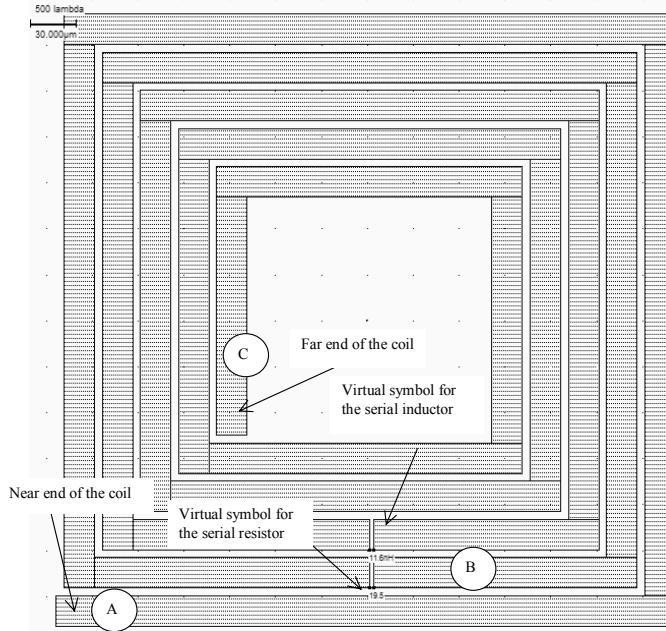


Figure 12-6. The inductor generated by default (inductor12nH.MSK)

### Inductor Impedance

On-chip inductance has a typical value ranging from 1 to 100nH, which give an equivalent impedance between 10 and 1000 ohm, within the radio-frequency range 300MHz-3GHz (Figure 12-7), by applying the formulation of the impedance versus frequency.

$$Z_L = jL\omega \quad (\text{Equ. 12-3})$$

At frequencies lower than 100Hz, discrete off-chip are used because of the high inductor values (From 1to 100 $\mu$ H) to keep the impedance between 10 and 1000 Ohm. Such high inductances cannot be integrated in a reasonable silicon area. Around 1GHz, a 10nH on-chip inductor matches the standard 50 $\Omega$  impedance of most input and output stages in very high frequency applications.

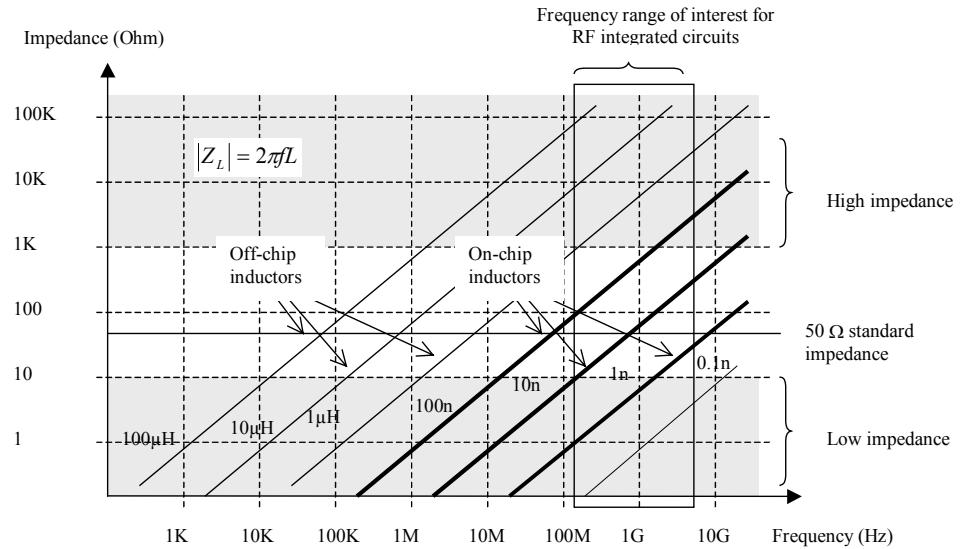
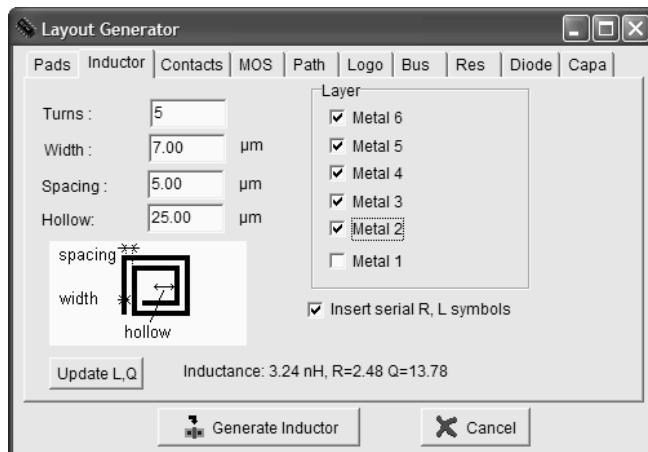


Figure 12-7. The inductor impedance versus frequency

### High Quality Inductor

A high quality factor  $Q$  is attractive because it permits high voltage gain, and high selectivity in frequency domain. The usual value for  $Q$  is between 3 and 30. The main limiting factors for  $Q$  are the serial resistance  $R_I$  of the wire and the substrate coupling capacitors  $C_1$  and  $C_2$ . From equation 12-2, it clearly appears that  $R_I, C_1$  and  $C_2$  should be kept as low as possible to increase  $Q$ . There are several ways to improve the coil quality factor. The first one consists in using the upper metal layer (metal 6 in  $0.12\mu\text{m}$ ) which features a smaller sheet resistance together with a smaller capacitance. Unfortunately, the quality factor is only increased to 2.

A significant improvement consists in using metal layers in parallel (Figure 12-8). The selection of metal2, metal3, up to metal6 reduces the parasitic resistance of  $R_I$  by a significant factor, while the capacitance of  $C_1$  and  $C_2$  is not changed significantly. The result is a quality factor near 13, for a  $3\text{nH}$  inductor. Even when the conductor width is increased to further reduce  $R_I$ , or if the number of turns and the coil shape are changed, the maximum  $Q$  is almost invariably below 20.



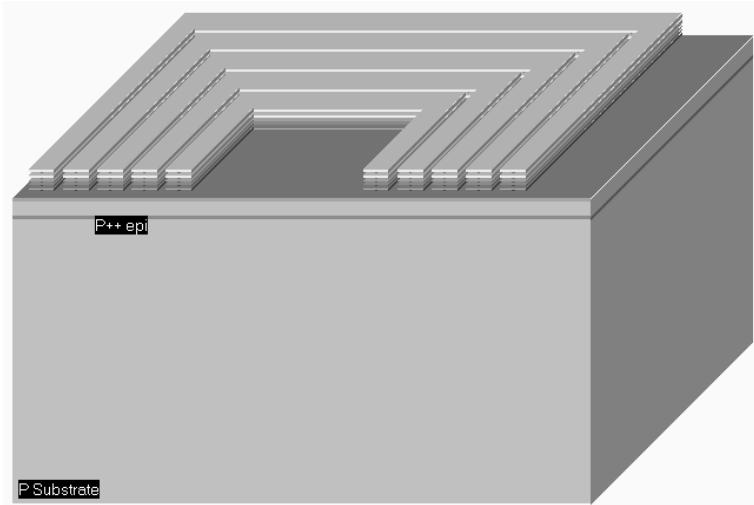


Figure 12-8: A 3D view of a high Q inductor using metal layers in parallel (Inductor3nHighQ.MSK)

### 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 12-9 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  $R1$ . At very high frequencies, the inductor is an open circuit, the capacitor a short circuit (Figure 12-9 right). Consequently, the link between  $C$  and  $A$  tends towards an open circuit.

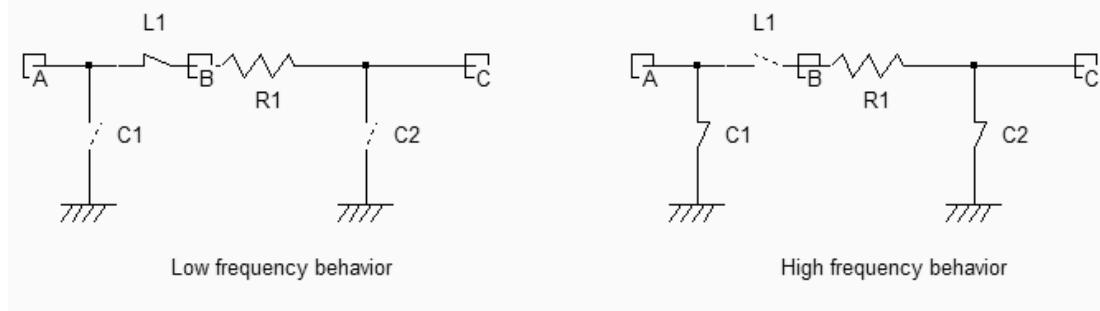


Figure 12-9. The behavior of a RLC circuit at low and high frequencies (Inductor.SCH)

At a very specific frequency the LC circuit features a resonance effect. The theoretical formulation of this frequency is given by equation 12-4.

$$f_r = \frac{1}{2\pi\sqrt{L_1(C_1+C_2)}} \quad (\text{Equ. 12-4})$$

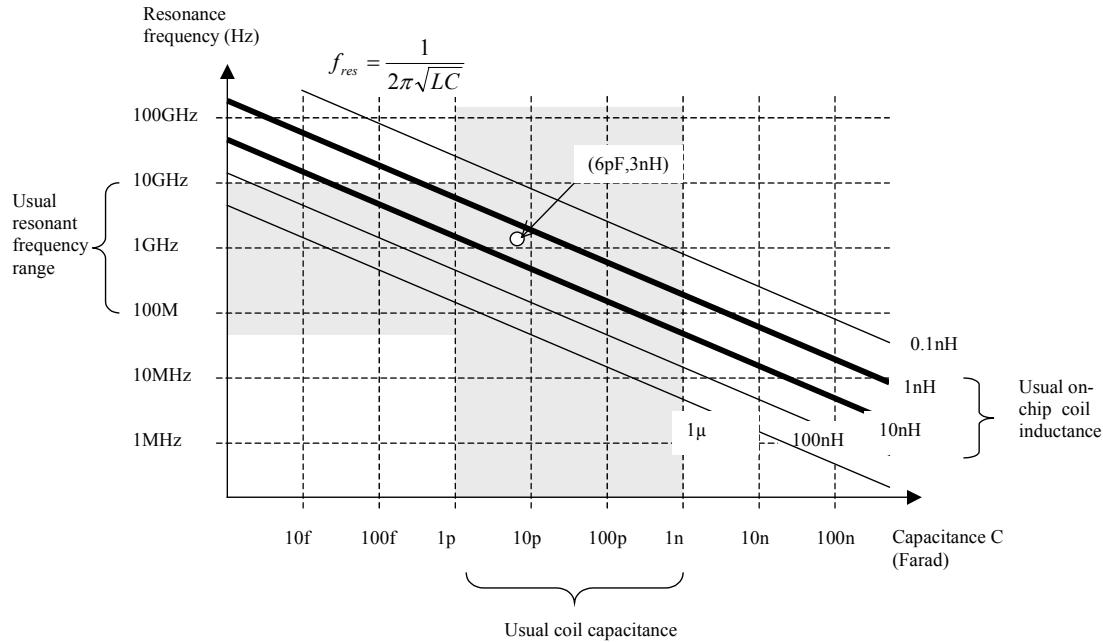


Figure 12-10. The resonance frequency depending on the capacitance and inductance

The variation of the resonant frequency with the capacitor and inductor is proposed in figure 12-8. On-chip coil inductances are within the range of 1 to 100nH. As the capacitance may vary from 1pF to 1nF, the range of the resonant frequency is around 100MHz to 10GHz, which includes most of the radio-frequency designs.

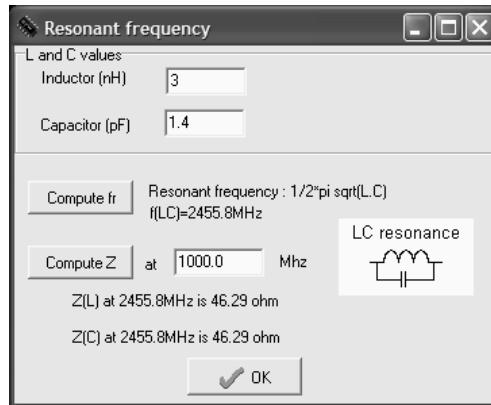


Figure 12-9. Microwind can compute the resonance frequency corresponding to user-defined L and C values

In the **Analysis** menu, the command **Resonance Frequency** includes a resonant frequency calculator, as shown in figure 12-9. For a given value of inductor and capacitor (3nH and 1.4pF in this example), the resonant frequency is directly computed in mega-hertz (MHz). For a target frequency of 2.45GHz, and a given inductor value of 3nH, we must choose a capacitor close to 1.4pF.

### Simulation of the Coil

In the case of  $Ll=3nH$  (Design corresponding to figure 12-6), the total capacitor is around 7pF. From the abacus given in figure 12-8, we obtain a resonant frequency around 1GHz. 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.

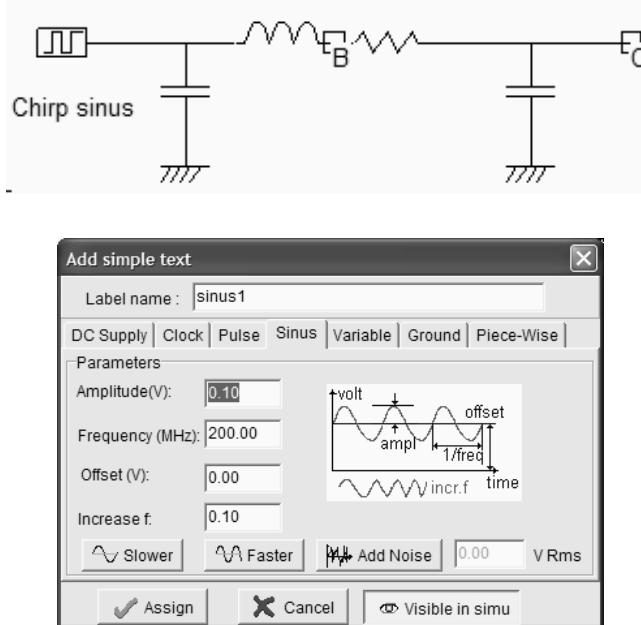


Figure 12-10. Using a sinusoidal waveform with increased frequency (Inductor3nHighQ.MSK)

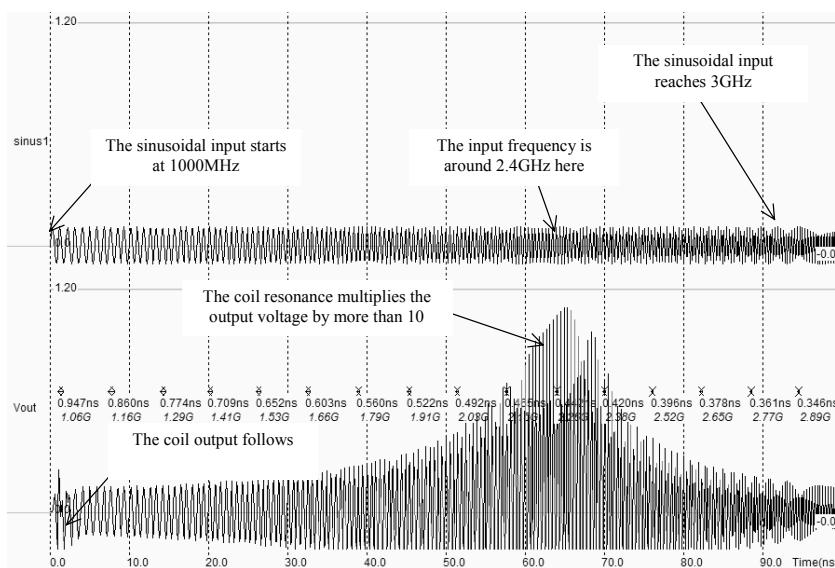
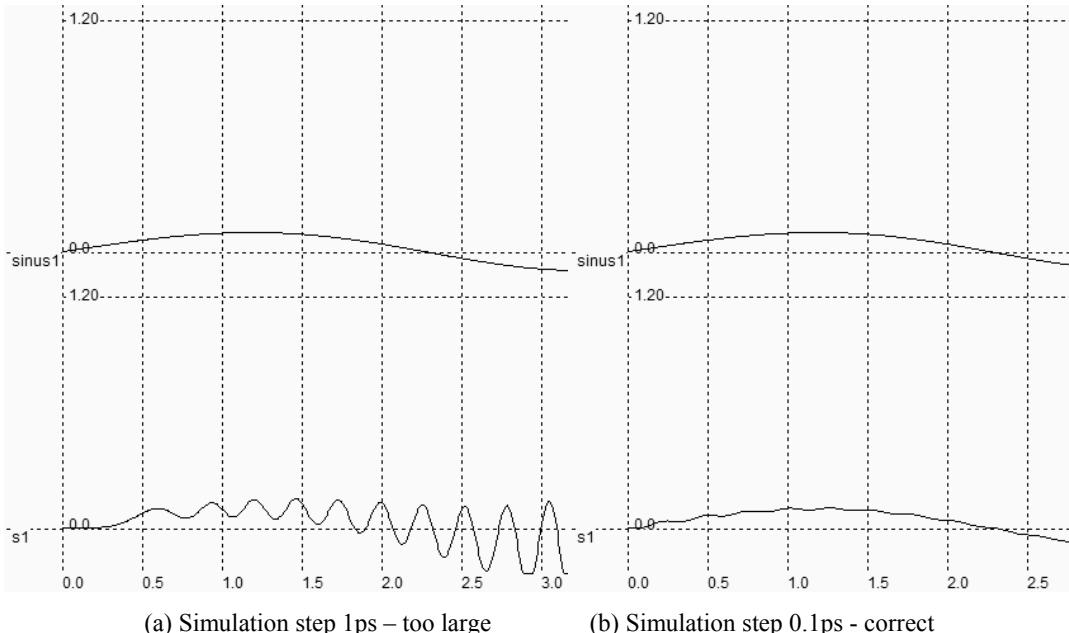


Figure 12-11. The behavior of a RLC circuit near resonance (Inductor3nHighQ.MSK)

The frequency corresponding to the resonance is around 2.4GHz, as predicted by the theoretical formulation. However, some mismatch between the prediction and the simulation may appear: first of all, the sinusoidal generator forces node  $A$  to a given voltage, which inhibits the role of capacitor  $C1$ . The resonance is only based on  $L1$ ,  $R1$  and  $C2$ , which shifts the frequency to higher levels. Secondly, the simulation of the inductor effect requires a significant amount of computation, with a high precision, otherwise the simulation becomes unstable.

In  $0.12\mu\text{m}$ , the simulation step is fixed to  $0.3\text{ps}$ , which is a good compromise between accuracy and speed. However, when dealing with an inductor, this step should be reduced. If we increase the step to  $1\text{ps}$  (Figure 12-12-a), an important parasitic instability effect appears and the output tends to oscillate. With a small simulation step ( $0.1\text{pS}$  in the case of figure 12-12-b), the simulation converges but the computation is significantly slowed down.



*Figure 12-12: The numerical instability appears in the inductor simulation when using a large integration interval (1ps) which is removed when lowering this interval to 0.1ps.*

### 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. A numerical or analog information is processed at low frequency, and converted to an appropriate sinusoidal waveform combined with a modulation. The high frequency converter transforms the low frequency signal  $f_{low}$  to a high frequency signal  $f_{high}$ . The shape of  $f_{high}$  is identical to  $f_{low}$  except that the frequency is one or two orders of magnitude higher. Details about this circuit are provided later in this chapter. The amplitude of  $f_{high}$  is usually small (10-100mV). A power amplifier is required to multiply the amplitude of the signal in order to transmit enough power to the emitting antenna.

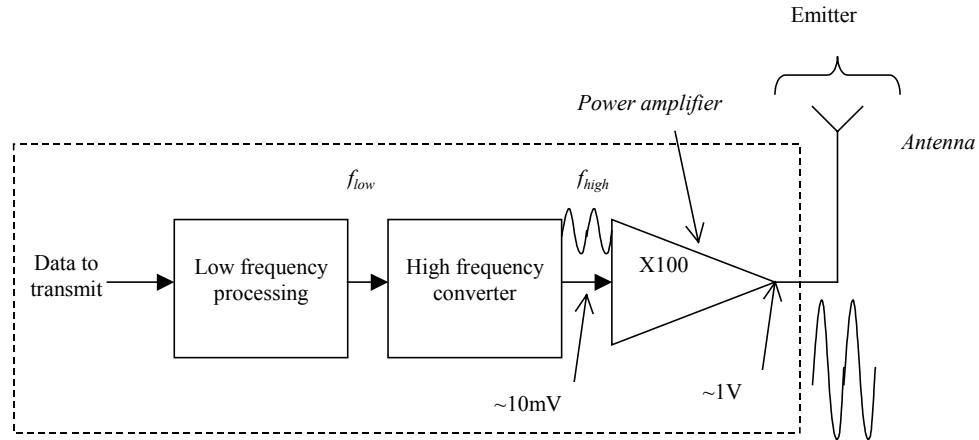


Figure 11-13: The power amplifier in a typical radio-frequency system

### Antenna Model

We can consider an antenna as a load that, in the ideal case, will be a pure resistance. The antenna resistance  $R_a$  accounts for the power absorbed by the antenna. This power is mainly radiated by the antenna. Most mobile phone antennas are resonant monopoles [Macnamara] for which the antenna resistance  $R_a$  varies from  $20 \Omega$  (Ground plane width  $w=0$ ) to  $40 \Omega$  (Infinite ground plane width). The monopole radiates mainly on X and Y directions (Figure 12-14). The length of the antenna is often chosen close to  $\lambda/4$ , where  $\lambda$  is the wavelength of the emitted signal. That length corresponds to the first maximum in the sinusoidal wave. From an electrical point of view, we can consider the antenna as a pure resistive load. The value of  $50\text{ ohm}$  is commonly used for  $R_a$  in simulations, as most equipments are "50 ohm adapted".

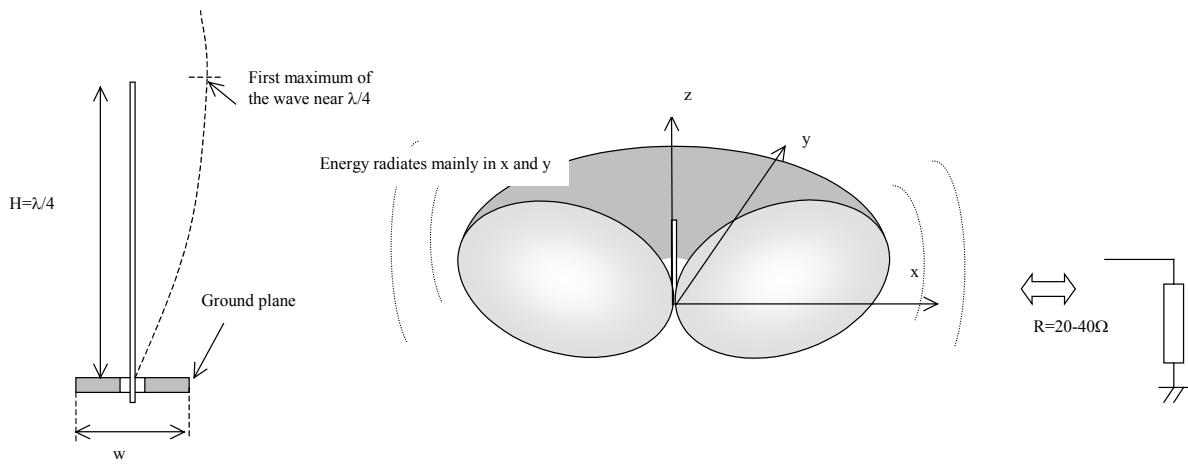


Figure 12-14. In first approximation, the antenna can be approximated as a load resistance around  $30\Omega$ .

## Power Amplifier Principles

Most CMOS power amplifiers are based on a single MOS device, loaded with a “Radio-Frequency Choke” inductor  $L_{RFC}$ , as shown in figure 12-16. 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  $C_L$ . The formulation for resonance is given below.

$$f_{resonance} = \frac{1}{2\pi\sqrt{L_{RFC}C_L}} \quad (\text{Eq. 12-5})$$

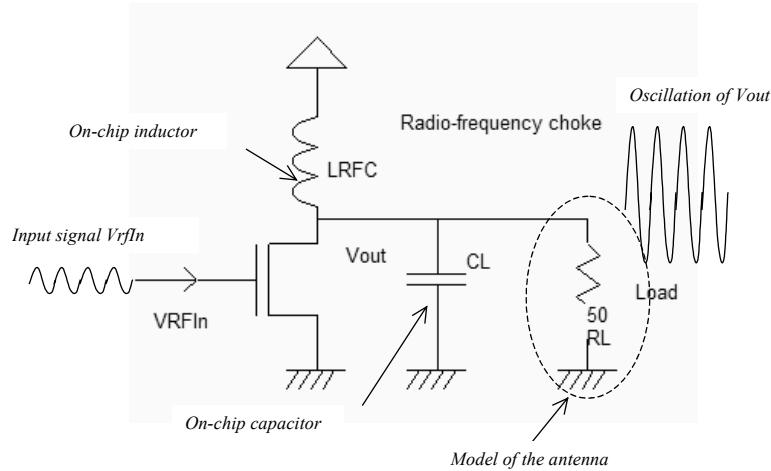


Figure 12-16: The basic diagram of a power amplifier (PowerAmp.SCH)

For example, a power amplifier designed for Bluetooth operation should resonate around 2.4GHz. If we assume that the inductance has a value of 3nH, the corresponding capacitor is around 1.5pF.

## Power Amplifier MOS

The MOS devices used in power amplifier designs have very huge current capabilities to be able to deliver strong power on the load. This leads to very unusual constraints on the width of the transistor so that devices with a width larger than 1000μm are commonly implemented [Hella]. The radio frequency choke inductor has a resonant effect which induces an important voltage swing of node  $Vout$ . Consequently, high voltage MOS devices are used to handle large overvoltages. A MOS device with a very large width is not drawn directly, but is obtained by connecting medium size MOS devices in parallel. In Microwind, we generate multiple-finger MOS devices easily, thanks to the MOS generator command (Figure 12-17). The high voltage option is selected, and the number of fingers is fixed to 10.

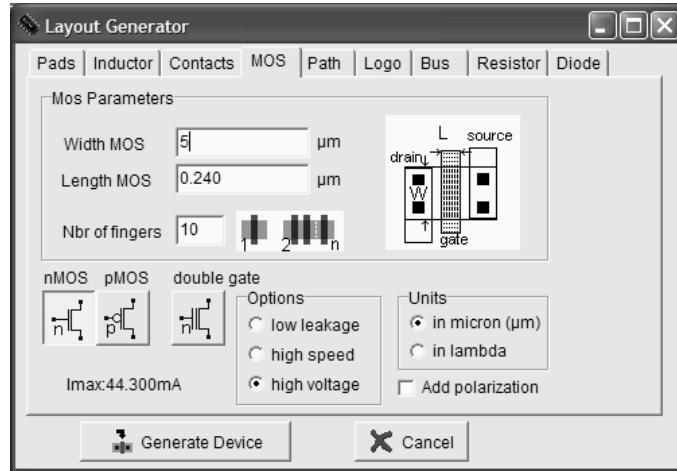


Figure 12-17. Generating a transistor with large current capabilities (PowerAmplifier.MSK)

The layout generated by Microwind is completed by adding a polarization ring to VSS, and metal2 contacts to the gate (Signal *VRF\_In*) and the drain (Signal *Vout*). The result is shown in figure 12-18. The maximum current is close to 40mA. A convenient way of generating the polarization ring consists in using the Path generator command, and in selecting the option **Metal and p-diffusion**. Then the location for the polarization contacts must be drawn in order to complete the ring.

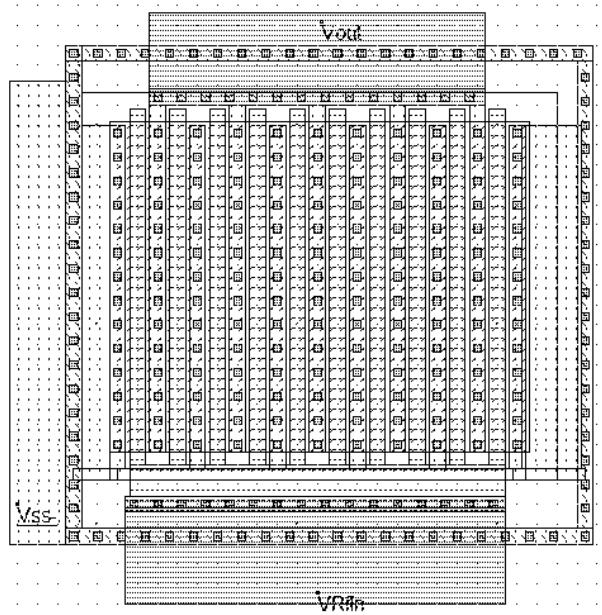


Figure 12-18. The layout of the power MOS also includes a polarization ring, and the contacts to metal2 connections to *VRF\_in* and *VOut* (PowerAmplifier.MSK)

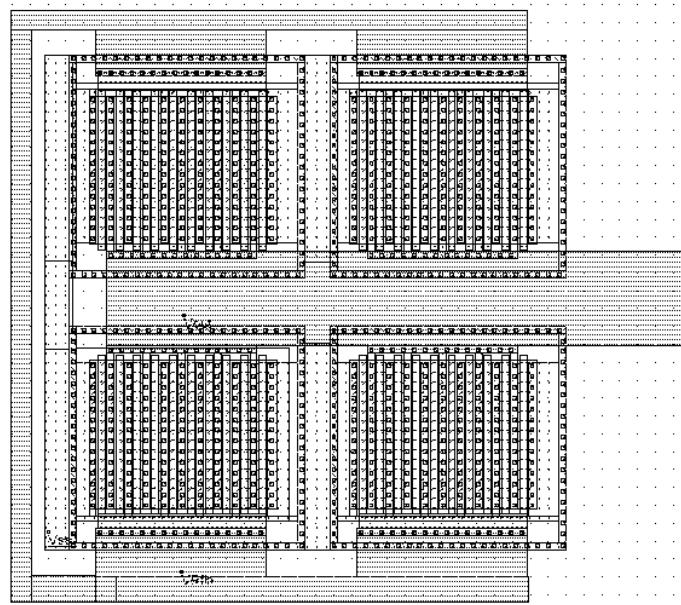


Figure 12-19: The layout of a 160mA power MOS using four large MOS in parallel (PowerAmplifierBig.MSK)

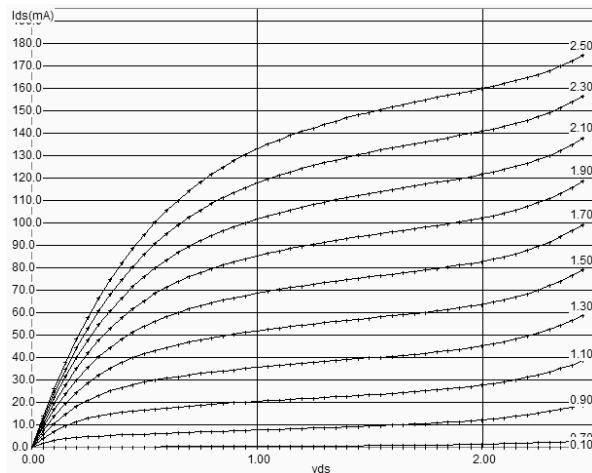


Figure 12-20. The static characteristics of the 160mA power MOS (PowerAmplifierBig.MSK)

An example of 160mA power device is shown in figure 12-19. Four devices are connected in parallel. The output node drives a large current and must be designed as wide as possible, with a short connection to the output pad to limit the serial resistance and parasitic capacitor to ground. The ESD protection is removed in some cases to enhance the power amplifier performances [Hella]. In the characteristics  $I_d/V_d$ , the maximum  $I_{on}$  current is close to 170mA (Figure 12-20). The ground connection also drives a strong current and must be carefully connected to the ground supply.

## Power Amplifier Efficiency

One of the most important characteristics of the power amplifier is the power efficiency, also called "drain efficiency" [Lee]. The definition of drain efficiency is given by equation 12-6.

The power efficiency is a ratio between the power delivered to the load and the supply power. The power efficiency (PE) is usually given in %. Typically, power efficiency  $PE$  ranges from 25 to 50%. The PE of CMOS power amplifiers is usually close to 30%. Higher efficiency is obtained with bipolar or GaAs semiconductors.

$$PE = \eta = \frac{P_{RF\_out}}{P_{DC}} \quad (\text{Equ 12-6})$$

Where

$P_{RF\_out}$  is the RF output power (in watt)

$P_{DC}$  is the total power delivered from the supply (in watt)

We may evaluate the power efficiency of the power amplifier with Microwind, using the following simulation procedure. The power amplifier is designed with a virtual load ( $RL=50$  ohm in the case of figure 12-21). Notice that the connection of the  $RL$  virtual load is unusual: one end of the resistor is connected to VDD rather than VSS. Connecting  $RL$  to ground would add a very important standby DC current, flowing through  $RL$  even without RF input. In reality, the  $RL$  resistor represents the antenna radiation resistance which has no direct path to ground. To avoid the parasitic DC contribution, we connect one end of  $RL$  to VDD.

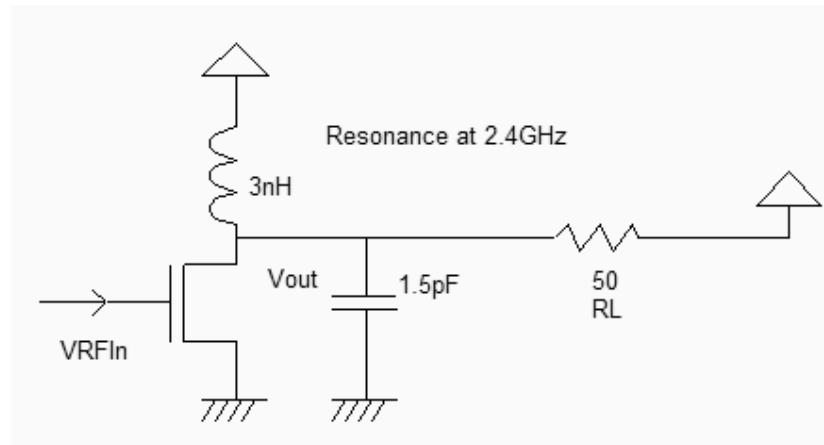


Figure 12-21. The evaluation of the power amplifier efficiency (PowerAmp.SCH)

The layout corresponding to the power amplifier is shown in figure 12-22. The inductor is virtual, as well as the 50ohm load. By default, the power  $P_{DC}$  is computed at each simulation and appears at the right lower corner of the simulation window of Microwind. In the simulation window corresponding to the mode **Current And Voltage vs. time**, we select the current flowing in R(50ohm). At the end of the simulation (Figure 12-23), the evaluation of the power efficiency is also displayed.

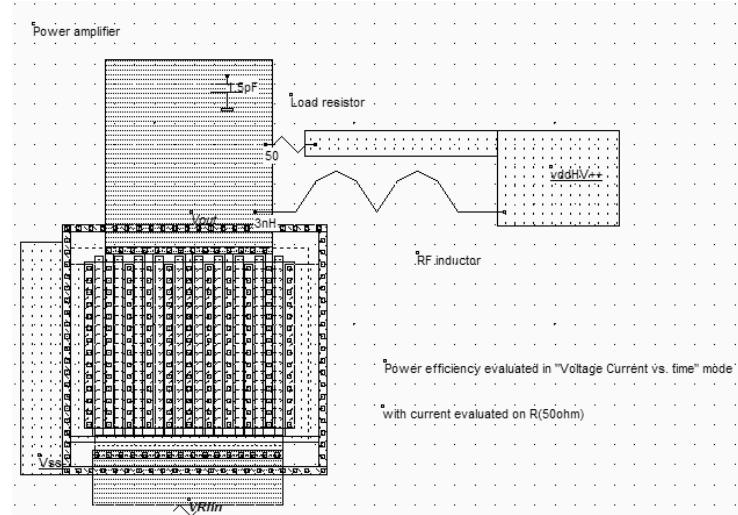


Figure 12-22. The evaluation of the power amplifier efficiency (PowerAmplifierEfficiency.MSK)

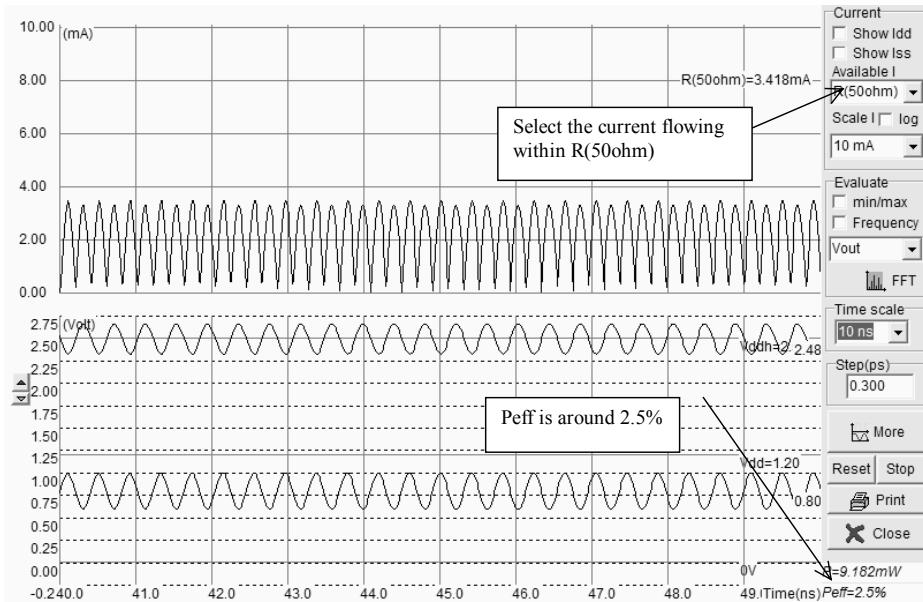


Figure 12-23. The evaluation of the power amplifier efficiency is accessible in the " Voltage and Current vs. Time " mode, by selecting the virtual load (PowerAmplifierEfficiency.MSK)

From the simulation of the simple power amplifier, we obtain a power efficiency of 2.5%, which is extremely low (Figure 12-23). In other words, 97.5% of the supply energy is dissipated and lost in the circuit, with only 2.5% delivered to the load. There are several techniques to improve the power efficiency: increasing the MOS size, modifying the amplitude of the input sinusoidal wave, and modifying the DC offset of the input sinusoidal wave.

An other metric for the power amplifier efficiency is the power added efficiency or PAE [Hella]. The PAE is very similar to equation 12-6. It includes the input power  $P_{RF\_in}$  as given in equation 12-7. Microwind does not evaluate directly this parameter.

$$PAE = \left( \frac{P_{RF\_out} - P_{RF\_in}}{P_{DC}} \right) \quad (\text{Equ 12-7})$$

Where

$P_{RF\_out}$  is the RF output power (in watt)

$P_{RF\_in}$  is the RF input power (in watt)

$P_{DC}$  is the total power delivered from the supply (in watt)

### Class A Power Amplifier

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 12-24. The power MOS is designed very big to improve the power efficiency.

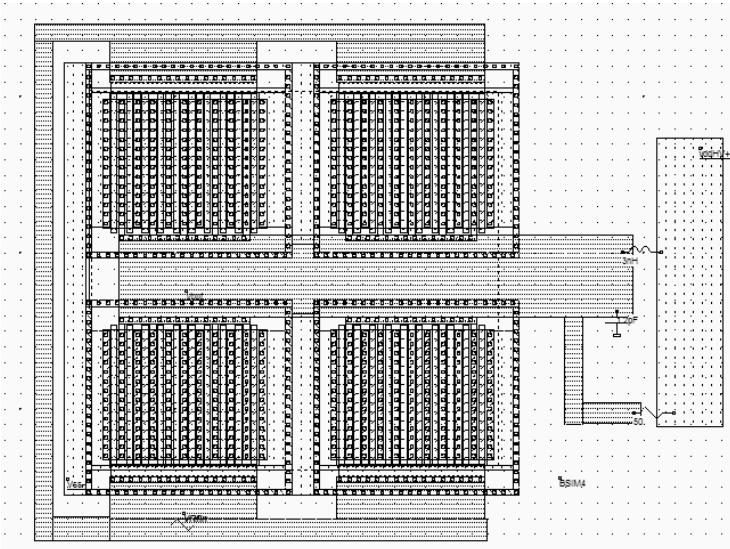


Figure 12-24. The class A amplifier design with a very large MOS device (PowerAmplifierClassA.MSK)

The sinusoidal input offset is 1.3V, the amplitude is 0.4V. The power MOS functional point trajectory is plotted in figure 12-25 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.

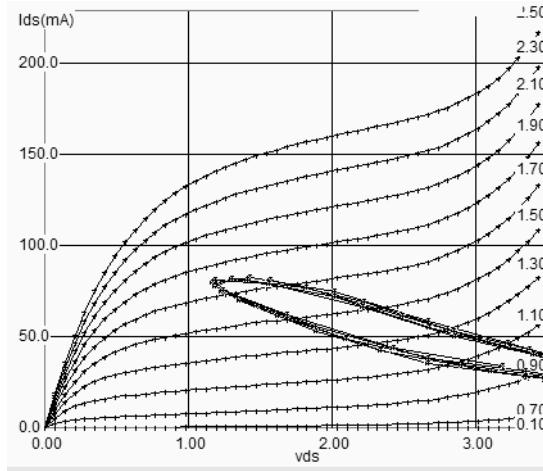


Figure 12-25. The class A amplifier has a sinusoidal input (PowerAmplifierClassA.MSK)

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 12-26.

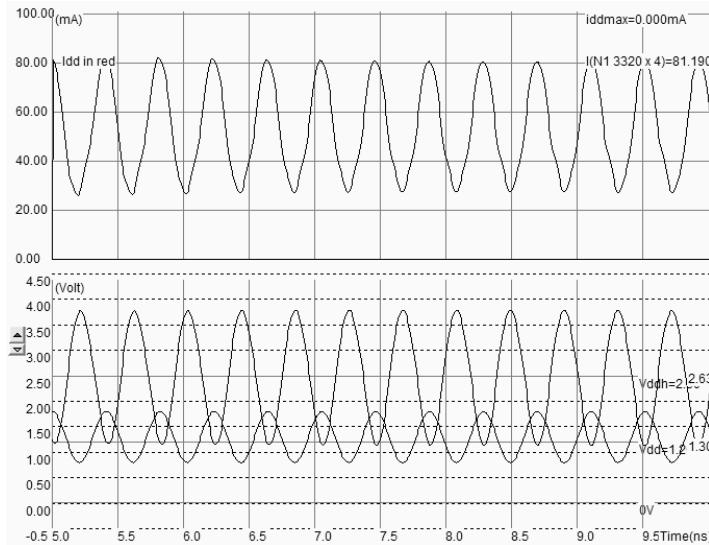


Figure 12-26. The Class A Amplifier simulation (PowerAmplifierClassA.MSK)

### Class B Amplifier

In class B, the MOS device only conducts for half a cycle. The monitoring of the current flowing in the power MOS shows a peak of current over the first half of the input period. Over the other half, the power MOS is off, and the LC resonator transmits the power to the 50 ohm load. The power efficiency rises to 20%. The main drawback is the severe distortion of the output voltage, which was much less visible with the class A polarization. The intermediate class, called AB, corresponds to a conduction between half the cycle and the full cycle.

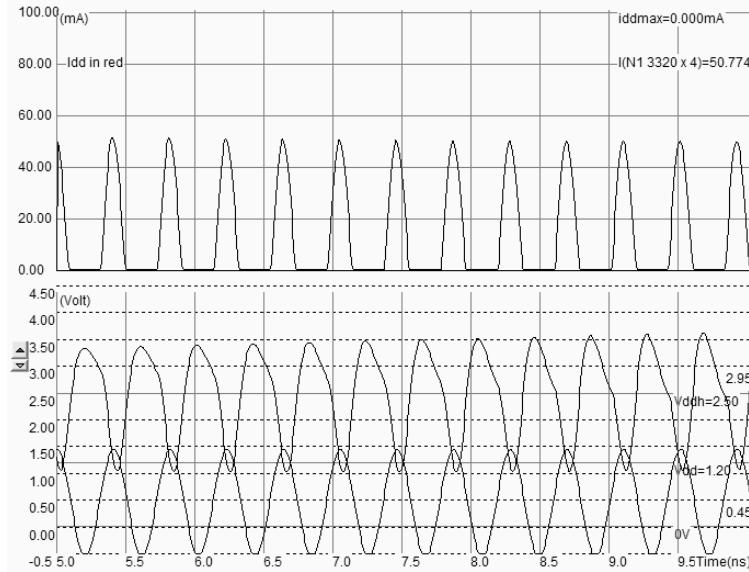


Figure 12-27. The class distinction for the power amplifier is linked to the DC value of the input signal

An evaluation of the spectral contents of the output node may be performed by the Fourier transform, with a plot in logarithmic scale. The Fourier transform is accessible on the simulation menu, through the button **FFT**. The fast Fourier transform translates the voltage waveform of the selected node into an evaluation of the energy of the signal versus its frequency. The energy plot shown in figure 12-28 reveals a peak near 2.5GHz. A noticeable energy is found on the second harmonic ( $2xf_0=4900\text{MHz}$ ) and third harmonics ( $3xf_0$ ). This is the consequence of a non-linear amplifier device.

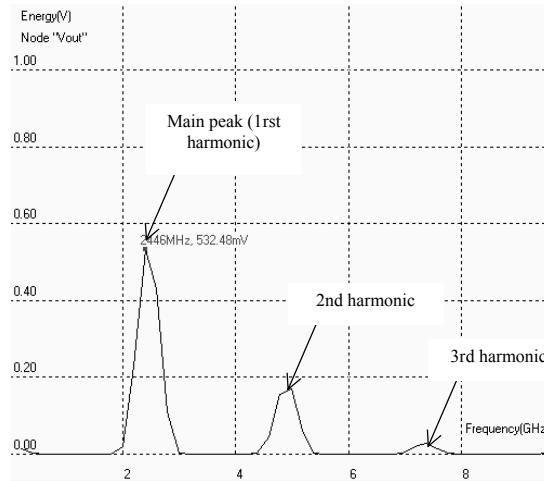


Figure 12-28: The class B amplifier is less linear than the class A amplifier (PowerAmplifierClassB.MSK)

### Other classes

In class C, the conduction occurs for less than half the cycle. The increase of efficiency obtained by reducing the conduction period is achieved at the expense of a reduced output power delivered to the load. The class E amplifier schematic diagram is shown in figure 12-29.

A band pass filter (LHF, CHF) is added to the output stage and fitted to the *VRFIn* input frequency. The effect of this passive circuit is to decrease the amplitude of the parasitic harmonics due to the non-linear nature of the amplifier, and to pass the desired frequency contribution. In some particular cases, the 3<sup>rd</sup> or even 5<sup>th</sup> harmonic is the desired one (Such as is 77GHz automotive radars for example, where the amplifier also serves as a frequency shifter). The power stage is coupled to the resonator through a coupling capacitor *Cc*. The role of *Cc* is to transfer the energy to the load, without any DC path between the supply and the load. The MOS drain can reach very high values when the switch is OFF. Consequently, a high breakdown voltage transistor is required. The theoretical efficiency of class E amplifier is higher than 50%.

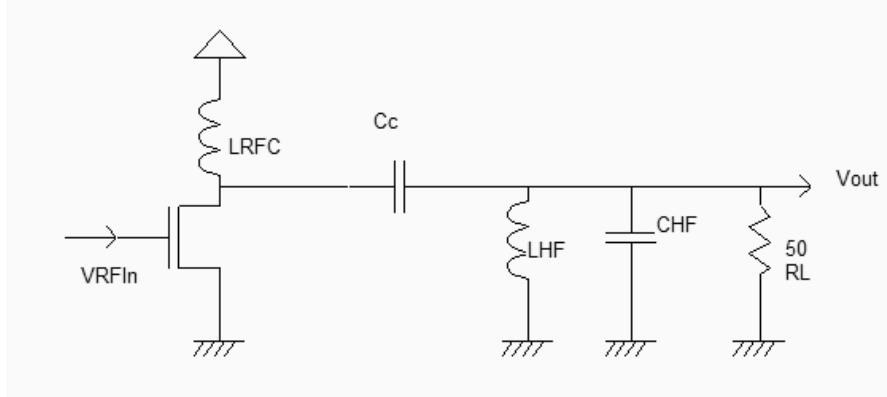


Figure 12-29: Class E amplifier (PowerAmpl.SCH)

### Self Heating

Self heating refers to the temperature rise that can occur in power devices, due to excessive heat energy accumulated before being dissipated through the substrate, the package and ultimately through the air. The thermal time constant is the order of one micro-second. Simulations usually consider a typical temperature of 25°C. This is realistic in the case of low power dissipation (Some milli-watts). In the case of hundreds of milliwatts, the simulation should take into account a significant temperature rise near the device. For example, a temperature of 80°C is commonly considered in medium power devices (Below 1W). In some cases, the IC may operate up to 250°C. In Microwind, the operating temperature may be changed in the menu **Simulator Parameters** of the simulation menu. In the window shown in figure 12-30, the temperature is fixed to 85°C.

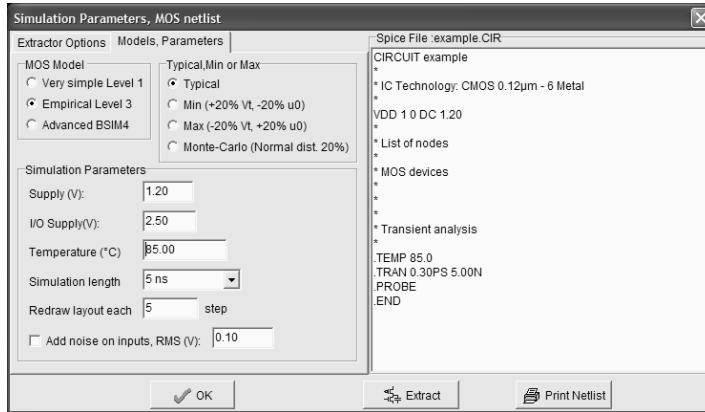


Figure 12-30: Setting up a high temperature for analog simulation

## 4. Oscillators

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 the main clocks of processors.

### Ring Oscillator

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. Usually, one inverter in the chain is replaced by a NAND gate to enable the oscillation (Figure 12-31).

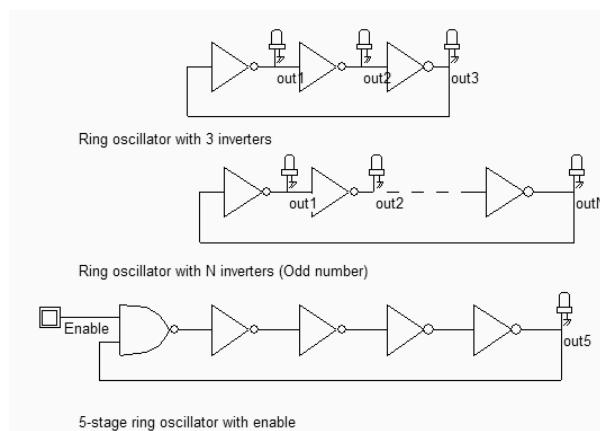


Figure 12-31: A ring oscillator is based on an odd number of inverters (Inv3.SCH)

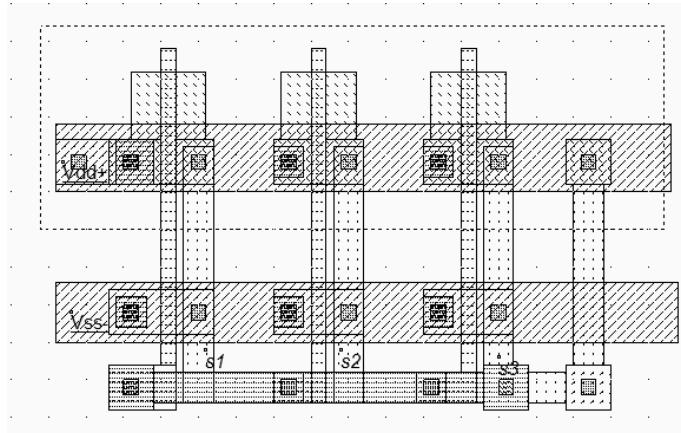


Figure 12-32: The implementation of a 3-inverter oscillator (Inv3.MSK)

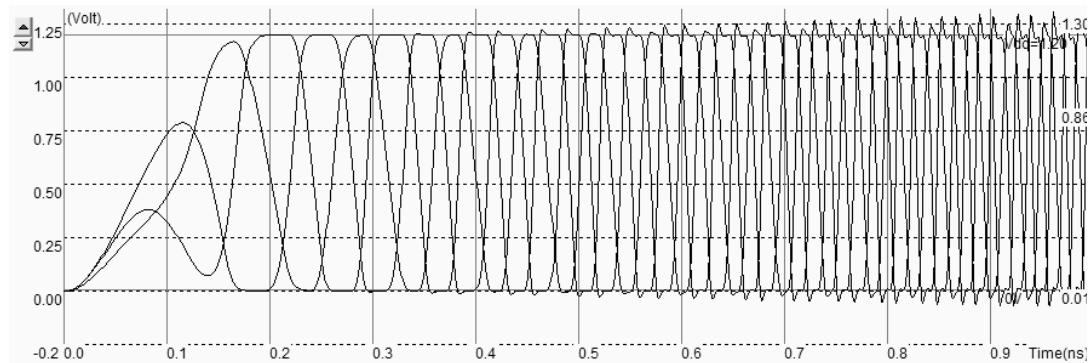


Figure 12-33: The simulation of the 3-inverter ring oscillator (Inv3.MSK)

The 3-inverter ring-oscillator layout is shown in figure 12-32. The right-most inverter output is connect the left-most inverter input by a metal bridge to create the desired feedback. Notice that no clock is assigned in this layout as the oscillation appears naturally, because of an intrinsic instability. The simulation of figure 12-33 shows the "warm-up" of the inverter circuit followed by a stable frequency oscillation.

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. 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.

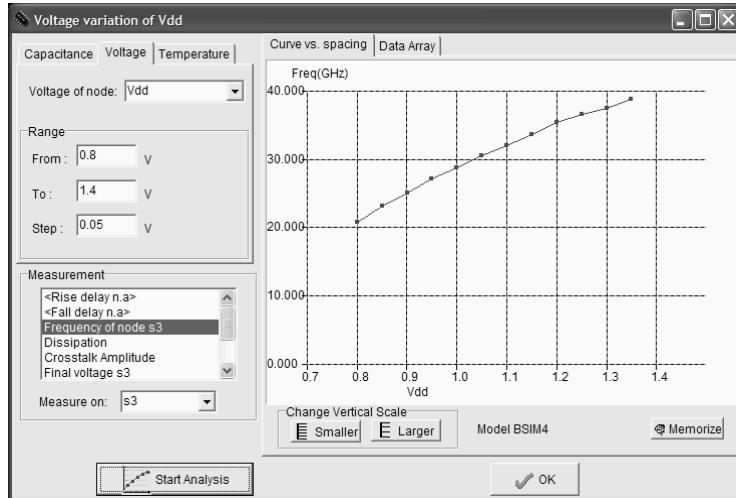


Figure 12-34: The oscillator frequency variation with the power supply (Inv3.MSK)

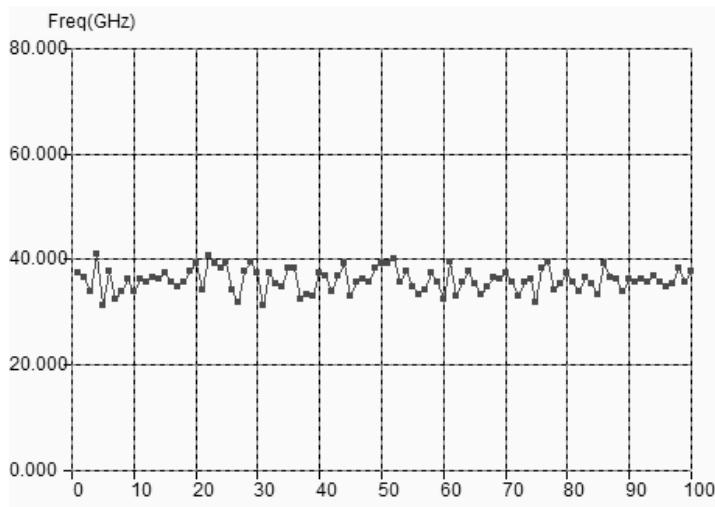


Figure 12-35: The process variations also have a direct impact on the switching frequency (Inv3.MSK)

The oscillation frequency of the ring oscillator is neither stable, nor controllable, and even not precisely predictable, as it is based on the switching characteristics of logic gates which may fluctuate +/-20%. A Monte Carlo analysis is performed in figure 12-35, to observe the technology variation influence on the oscillator frequency. The basic principle of this analysis is to sort a set of technological parameters in a random way, and to conduct the complete analog simulation for each random set. Each point in the X axis corresponds to one simulation, with a specific set of parameters.

There is no correlation between adjacent points, because of the random nature of all the different condition. We observe again the significant fluctuation of the oscillator frequency. As a conclusion, ring oscillators have poor performances, and may only be used in low performance clocking systems, or for a dynamic characterization of the technology. The design of several ring oscillators on CMOS test chips was also experienced to tune Microwind simulations with real-case ring oscillator measurements, and a good correlation between measured and simulated oscillator frequency was obtained.

### Random simulation

In Microwind, the threshold and mobility parameters are varying with a *Normal* distribution <Gloss>, with a typical variation of 10%. The normal distribution of the threshold voltage  $V_t$  corresponds to a density of probability following the equation 12-8. The aspect of  $f$  versus  $V_t$  is given in figure 12-36.

$$f_{V_t} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(V_t-V_{t0})^2}{2\sigma^2}} \quad (\text{Equ. 12-8})$$

where

$f$  is the density of probability for a given value of the threshold voltage  $V_t$  (0 to 1)

$\sigma=0.1$  (Equivalent to 10% typical fluctuation of the parameter)

$V_{t0}$ =typical threshold voltage (0.4V)

$V_t$ = threshold value (V)

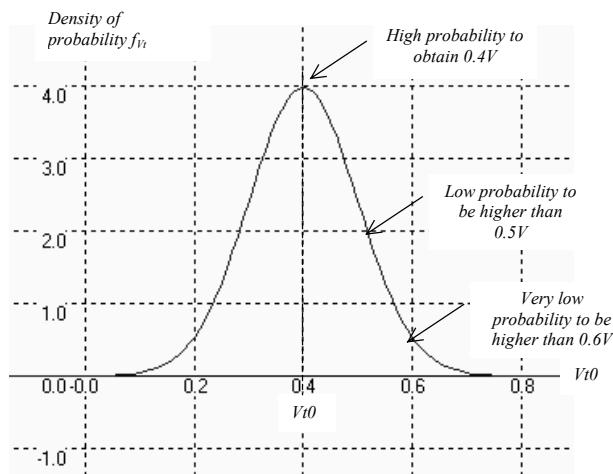


Figure 12-36: The normal distribution of  $V_t$ , with a typical variation of 10%

### LC oscillator

The LC oscillator proposed in this paragraph 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 12-37, the inductor  $L_1$  resonates with the capacitor  $C_1$  connected to S1, combined with  $C_2$  connected to S2.

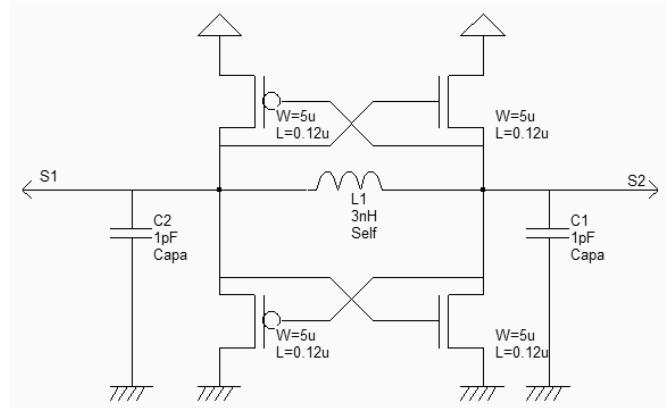


Figure 12-37: A differential oscillator using an inductor and companion capacitor (OscillatorDiff.SCH)

The layout implementation is performed using a 3nH virtual inductor and two 1pF capacitors (Figure 12-38). Notice 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 12-39) 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. The measured frequency approaches 3.75GHz with a 3nH inductor  $L1$  of and 1pF capacitors  $C1$  and  $C2$ .

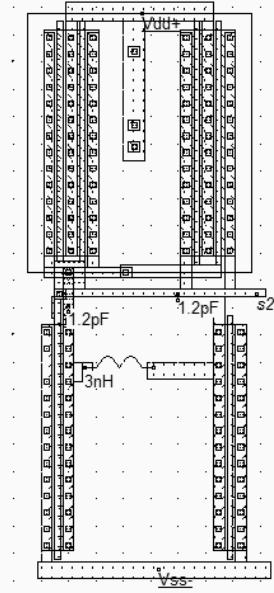


Figure 12-38: A differential oscillator using 3nH inductor (OscillatorDiff.MSK)

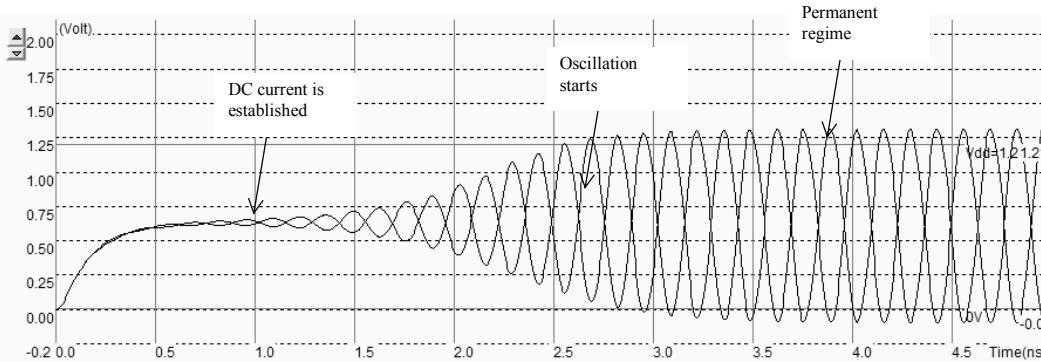


Figure 12-39: Simulation of the differential oscillator (OscillatorDiff.MSK)

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 12-40). 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.

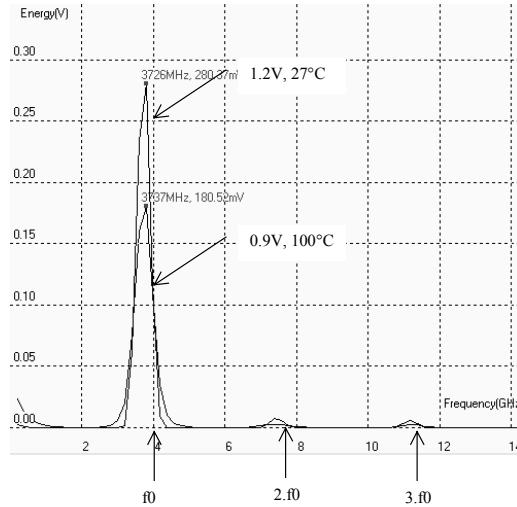


Figure 12-40: The frequency spectrum of the oscillator (OscillatorDiff.MSK)

For example, we may investigate the effect of VDD on the resonating frequency by lowering manually VDD from 1.2V down to 0.9V in the menu **Simulate→Simulation parameters**. The result is a significant increase of the warm-up phase, while the final oscillation frequency remains unchanged. A parametric analysis on VDD, from 0.7 to 1.4V, confirms that the LC oscillator performs much better than the ring-inverter oscillator, as it turns out to be almost immune to supply voltage fluctuations.

Unfortunately, the inductance of an on-chip coil is not perfectly constant, as the material resistance, conductor width and oxide thickness may vary by several %. The capacitance of a poly/poly2 structure, used for implementing the passive capacitor, may also vary due to the process fluctuation impact on the inter-layer oxide. The temperature also has an influence on the capacitance value [Reference].

In Microwind, the Monte-carlo simulation mode also impacts the value of all virtual elements in a similar way as for the threshold voltage and the mobility: before the simulation starts, the  $L$  and  $C$  values are assigned a value that fluctuates by +/-10% with a normal distribution around the user-defined impedance. The result is a significant variation of the oscillator frequency with the process parameters (Figure 12-41).

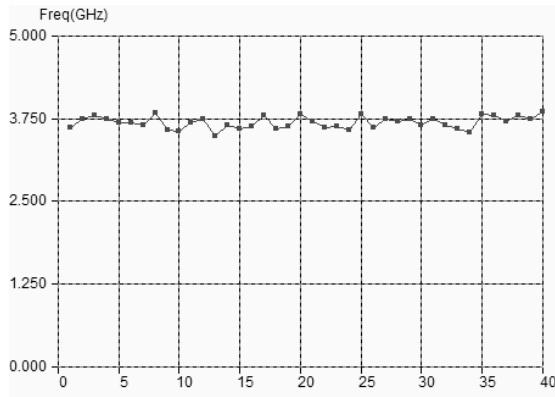


Figure 12-41: The frequency of the LC oscillator varies with the process parameters, mainly due to the capacitor and inductor process dependence (OscillatorDiff.MSK)

It can be concluded that a predictable and stable frequency oscillation is very hard to obtain on-chip, without any external high precision component. In radio-frequency applications, a base frequency is always delivered by a Quartz, which is the best discrete device to create an almost perfect oscillation circuit.

## 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 12-42 [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.

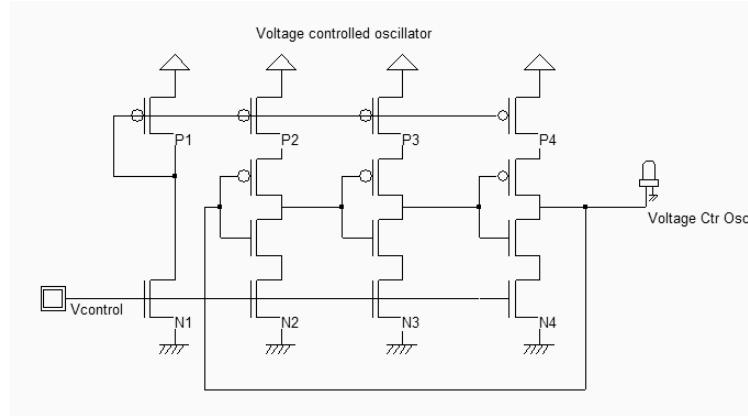


Figure 12-42: Schematic diagram of a voltage controlled oscillator (VCOMos.SCH)

The implementation of the current-starved VCO for a 5-inverter chain is given in figure 12-43. 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.

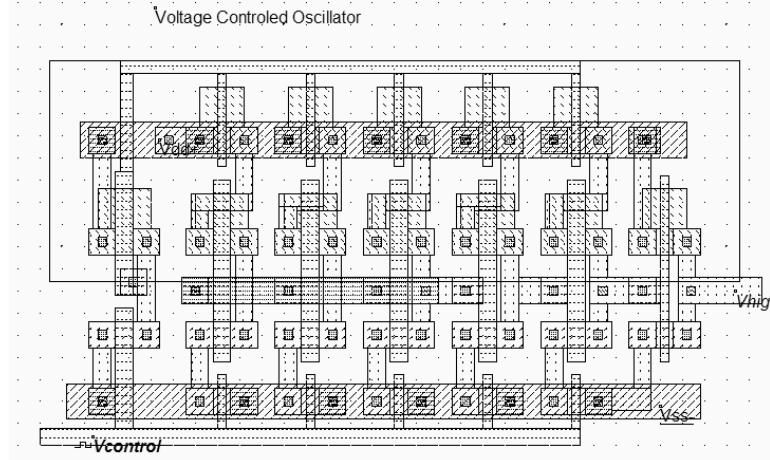


Figure 12-43. A VCO implementation using 5 chained inverters (VCO.MSK)

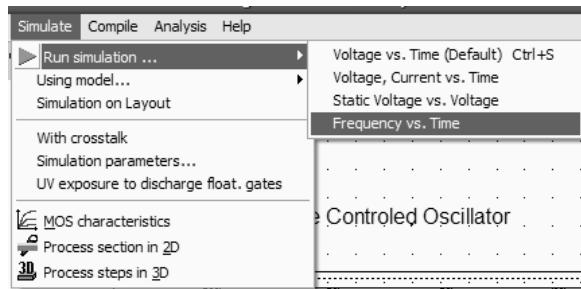


Figure 12-44. The access to Frequency vs. time simulation mode

The VCO circuit frequency variation with *Vcontrol* is accessed by using the layout shown in figure 12-43. A convenient simulation mode is directly accessible (figure 12-44), to display the frequency variations versus time together with the voltage variations. The frequency is evaluated on the selected node, which is the output node *Vhigh* 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.

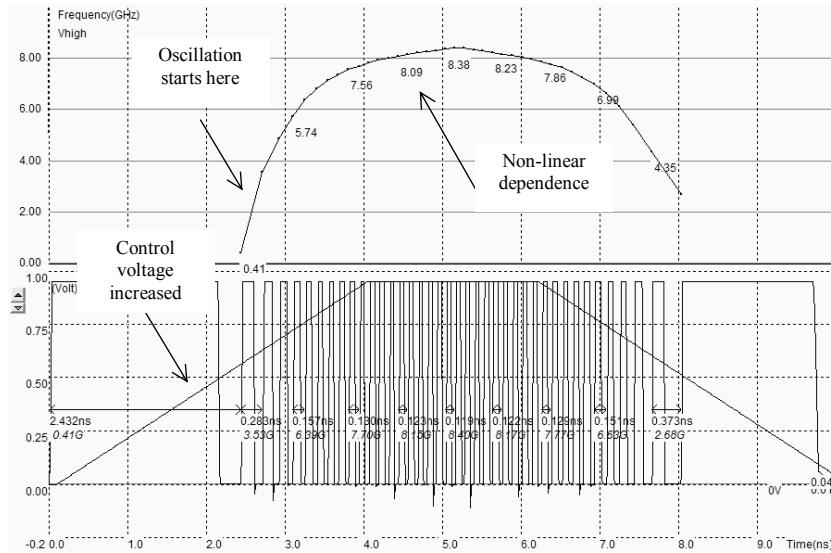


Figure 12-45. The frequency variations versus the control voltage show a non-linear dependence (VCO.MSK)

### High Performance VCO

A voltage controlled oscillator with good linearity is shown in figure 12-46. This circuit has been implemented in several test-chips with successful results in 0.8, 0.35 down to  $0.18\mu m$  technologies. The principle of this VCO is a delay cell with linear delay dependence on the control voltage [Bendhia]. The delay cell consists of a p-channel MOS in series, controlled by  $V_{control}$ , and a pull-down n-channel MOS, controlled by  $V_{plage}$ . The delay dependence on  $V_{control}$  is almost linear for the fall edge. The key point is to design an inverter just after the delay cell with a very low commutation point  $V_c$ . The rise edge is almost unchanged. To delay both the rise and fall edge of the oscillator, two delay cells are connected, as shown in the schematic diagram.

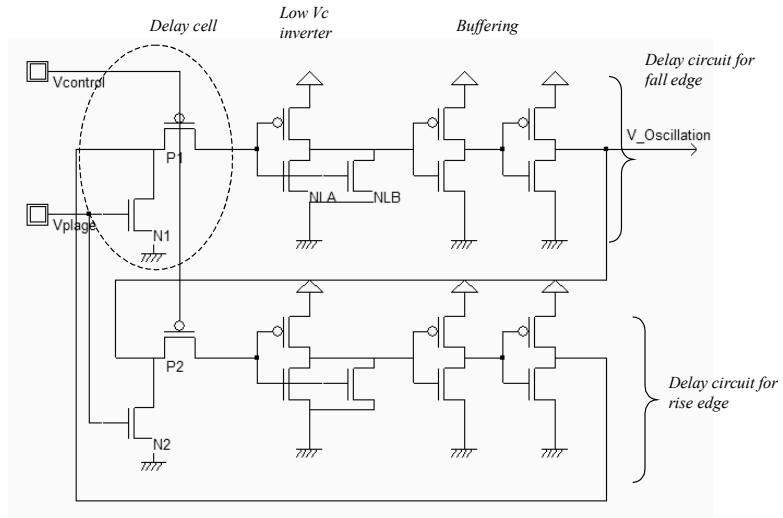


Figure 12-46: The layout implementation of a high performance VCO circuit (VCOLinear.MSK)

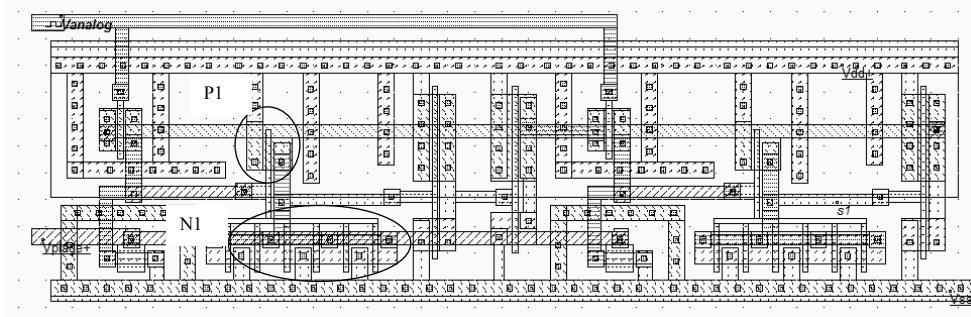


Figure 12-47: The layout implementation of a high performance VCO circuit (VCOLinear.MSK)

The layout of the VCO is a little unusual due to the needs for a very low commutation point for the inverter situated immediately after the delay cells. This is done by implementing a large n-channel MOS (N1 in figure 12-47) with high drive capabilities and a tiny p-channel MOS with low drive capabilities (P1 in figure 12-47).

The simulation of a high performance VCO circuit is given in figure 12-48. A quasi-linear dependence of the oscillating frequency on the input voltage control is observed within the range 0..0.6V. Although not displayed in the simulation, the voltage of  $V_{plage}$  has a strong influence on the oscillating frequency range. A high value of  $V_{plage}$  (Close to VDD) corresponds to a high frequency oscillation, while a low value (Close to the threshold voltage  $V_t$ ) corresponds to a low frequency oscillation.

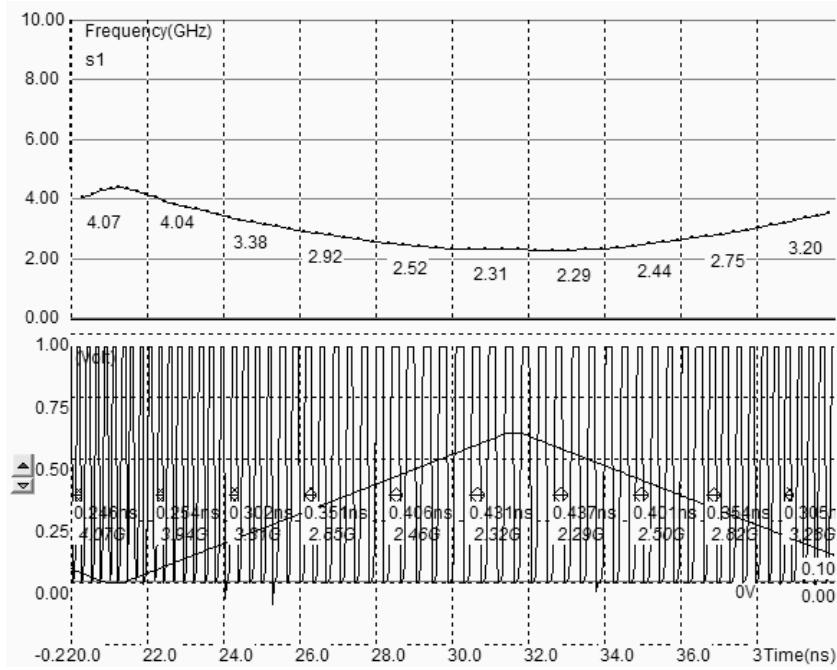


Figure 12-48: The simulation of a high performance VCO circuit (VCOLinear.MSK)

The main drawback of this type of oscillator is the great influence of temperature and VDD supply on the stability of the oscillation. If we change the temperature, the device current changes, and consequently the oscillation frequency is modified. Such oscillators are rarely used for high stability frequency generators.

## 5. Phase-Lock Loop

The phase-lock-loop (PLL<Gloss>) is commonly used in microprocessors to generate a clock at high frequency ( $F_{out}=2\text{GHz}$  for example) from an external clock at low frequency ( $F_{ref} = 100\text{MHz}$  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 12-49). The PLL may also be found in frequency demodulation circuits, to transform a frequency varying waveform into a voltage.

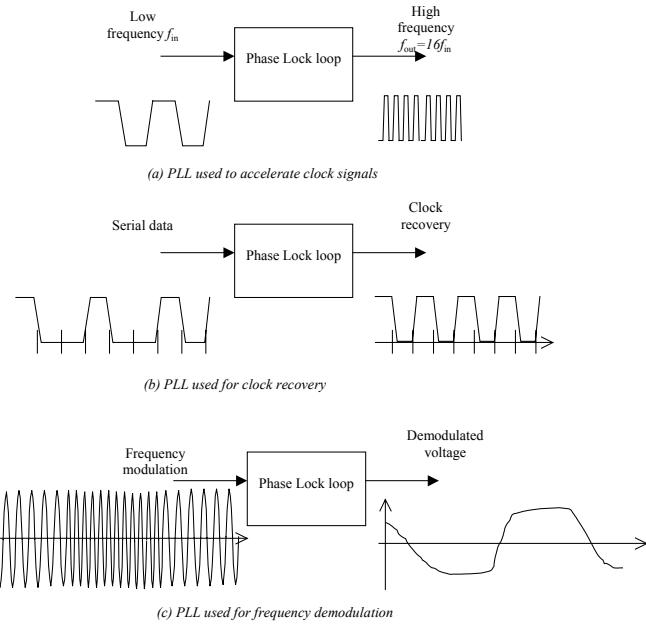


Figure 12-50. Principles of phase lock loops

The PLL uses a high frequency oscillator with varying speed, a counter, a phase detector and a filter (figure 12-51). 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 into variation of the analog signal  $V_c$ . This signal changes the VCO frequency which is divided by the counter and changes  $clkDiv$  according to  $f_{in}$ .

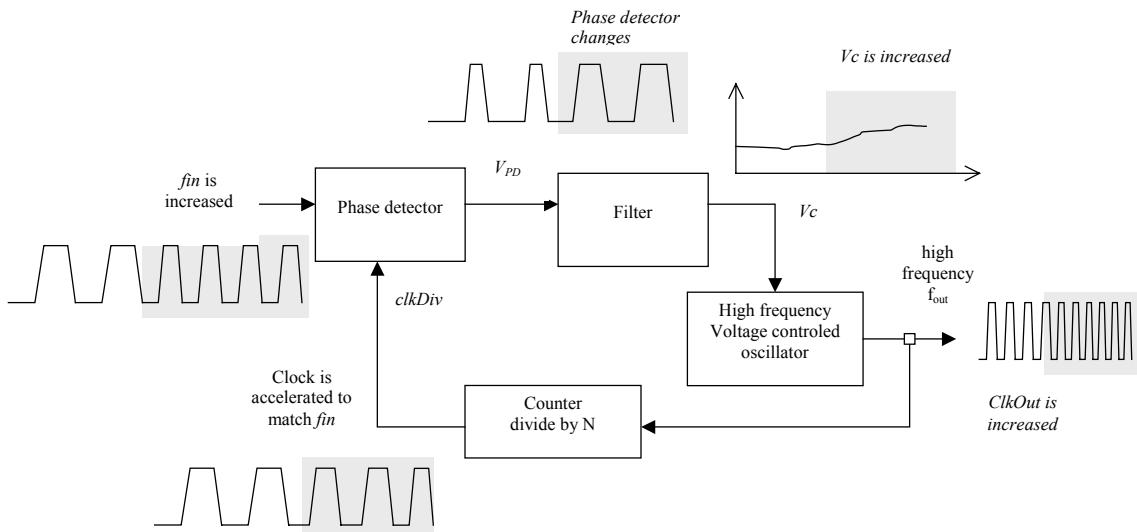


Figure 12-51. Principles of phase lock loops

## 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  $90^\circ$  or  $\pi/2$ ). For other angles, the output is no more regular. In figure 12-52, two clocks with slightly different periods are used in Dsch2 to illustrate the phase detection.

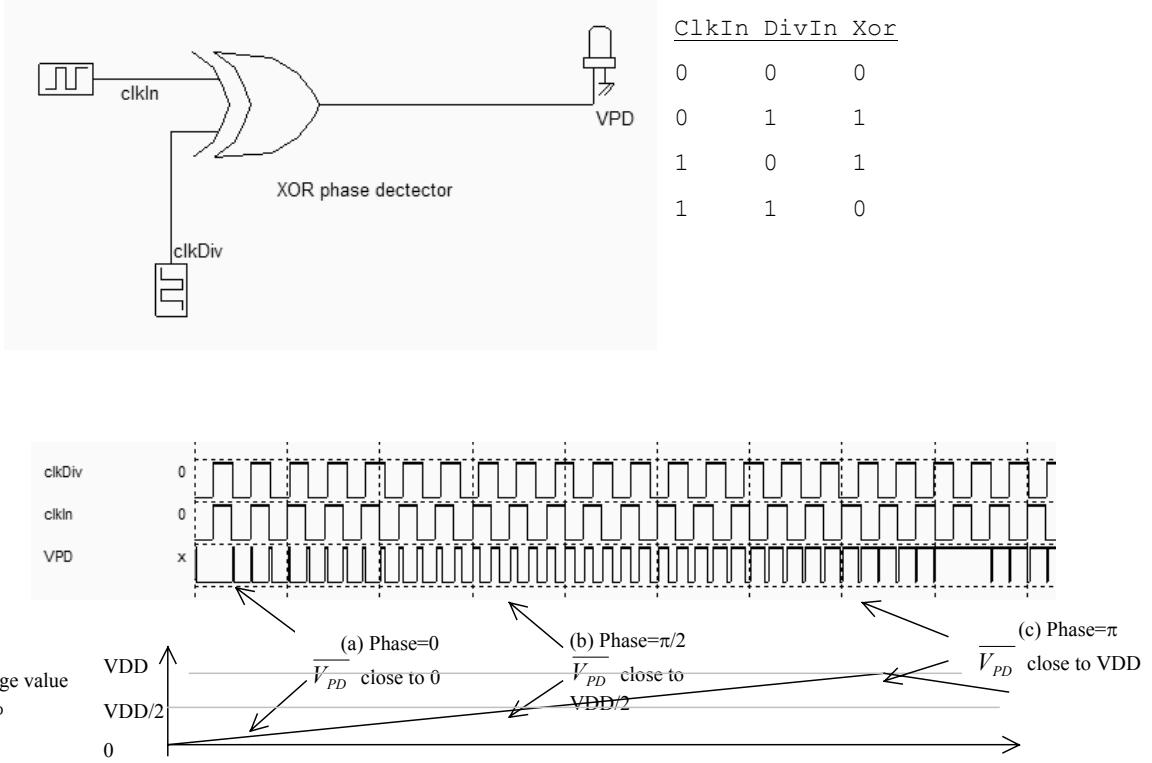


Figure 12-52. The XOR phase detector at work (PhaseDetectXor.SCH)

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  $VDD$ . Consequently,  $\overline{V_{PD}}$  and the phase difference are linked by expression 12-10. For example, when  $\Delta\phi=\pi/2$ ,  $\overline{V_{PD}}$  is  $VDD/2$ .

$$\overline{V_{PD}} = \frac{VDD \cdot \Delta\phi}{\pi} \quad (\text{Equ. 12-10})$$

The gain of the phase detector is the ratio between  $\overline{V_{PD}}$  and  $\Delta\phi$ . The gain is often written as  $K_{PD}$ , with an expression derived from equation 12-10, which is valid for  $\Delta\phi$  between 0 and  $\pi$ , as drawn in figure 12-53.

$$K_{PD} = \frac{VDD}{\pi} \quad (\text{Equ. 12-11})$$

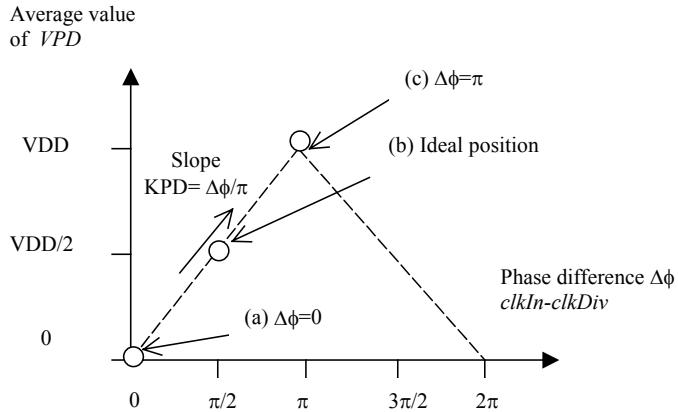


Figure 12-53. The XOR phase detector at work (PhaseDetectXor.SCH)

When the phase difference is larger than  $\pi$ , the slope sign is negative until  $2\pi$ . When locked, the phase difference should be close to  $\pi/2$ .

## 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  $R_{on}C$  delay creates a low-pass filter. Figure 12-54 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  $V_c$ .

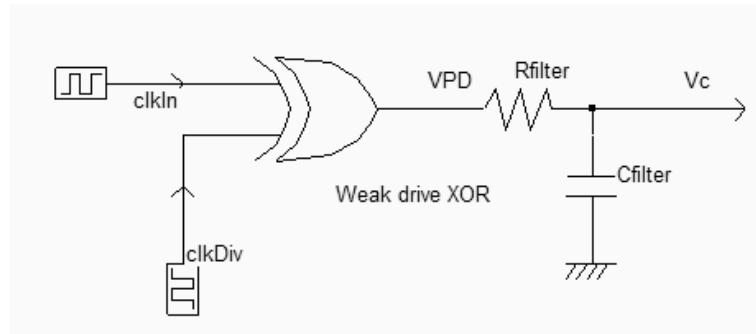


Figure 12-54. Large load capacitance and weak XOR output stage to act as a filter (phaseDetectAndFilter.SCH)

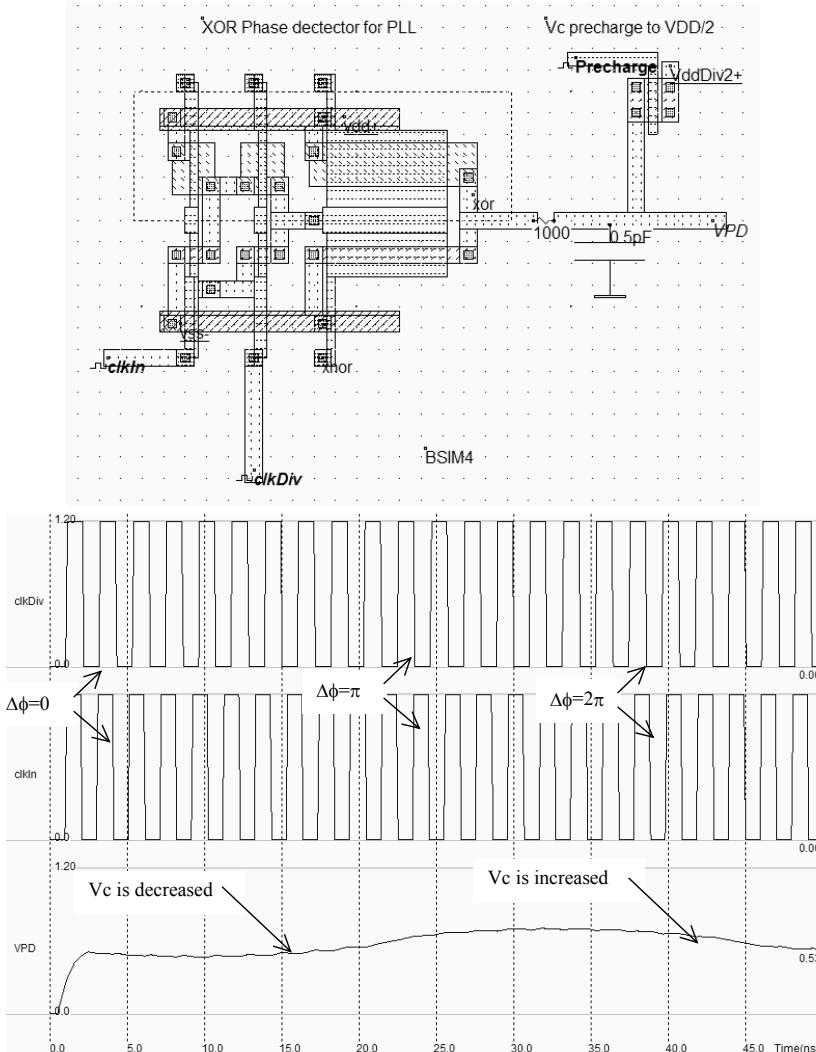


Fig. 12-55. Response of the phase detector to slightly different input clocks (phaseDetect.MSK)

In the figure above, the filtered version of the XOR gate output  $VPD$  is shown. It can be seen that  $VPD$  is 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. The XOR gate layout has been modified so that the output voltage  $Vc$  is very close to  $VDD/2$  when one input is fixed to ground and the other input is a regular clock.

### 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 12-56).

- 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 (figure 12-56).
- When the control voltage  $V_c$  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.

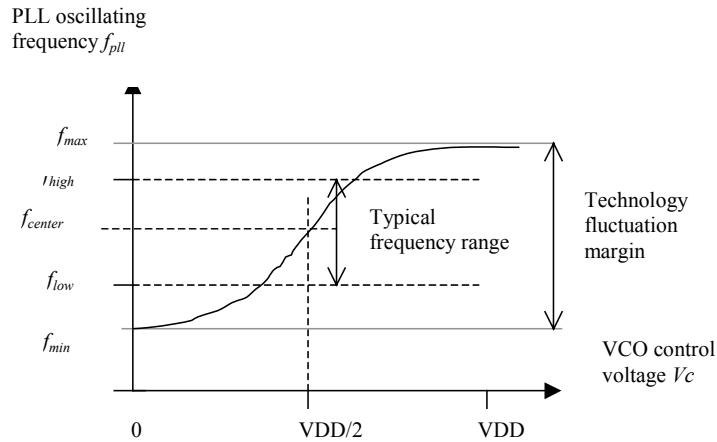


Figure 12-56: Requirements for the VCO used in the PLL

The current starved oscillator can be used as a VCO for the phase lock loop, with a modification of its voltage control circuit so that the center frequency is 2450MHz at  $V_c=VDD/2$ , and the frequency range does not exceed 2800MHz and does not drop lower than 1800MHz. The modification consists in providing a permanent current path through  $Rvdd2$  to  $VDD/2$  (Figure 12-57), which helps keeping  $V_c$  around  $VDD/2$ . When  $VPD$  is  $VDD$ ,  $V_c$  is increased and the VCO frequency is close to  $f_{max}$ . When  $VPD$  is 0,  $V_c$  is lowered and the VCO frequency is close to  $f_{min}$ .

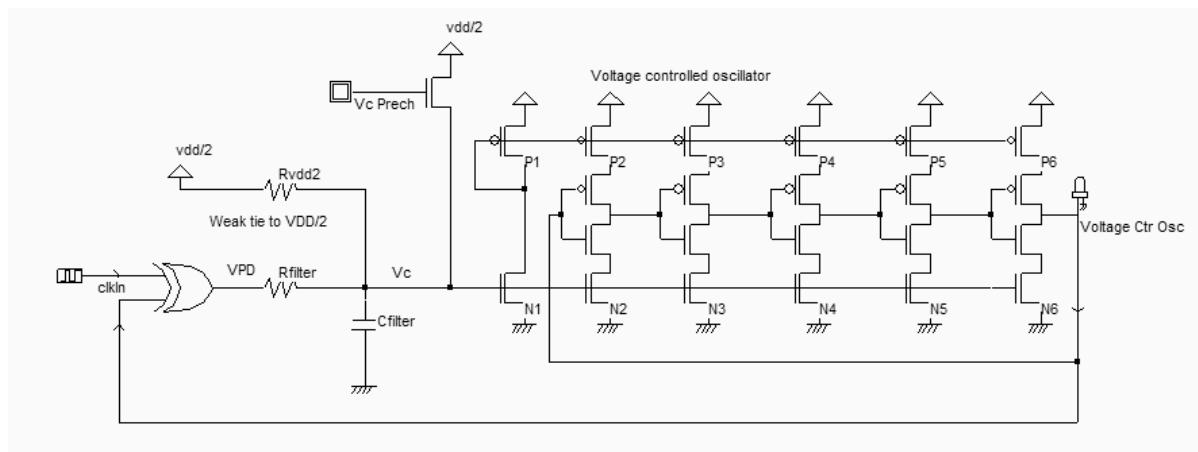


Figure 12-57: Connecting the current-starved VCO to the phase detector (Pl.SCH)

A second important sub-circuit added in the PLL is the precharge to VDD/2. The nMOS device controlled by  $V_{c\_Prech}$  helps the big capacitor  $C_{filter}$  to reach VDD/2 during the first nanoseconds. This precharge circuit speeds up the locking of the PLL.

### Complete Phase Lock Loop

The implementation of the PLL shown in figure 12-58 is a direct copy of the schematic diagram of figure 12-57. Notice that the resistor  $R_{filter}$  (1000Ohm) and  $R_{vdd2}$  (5000 Ohm) have been implemented using virtual elements and not physical resistance. The same can be said for the capacitor  $C_{filter}$  (0.3pF). However, these resistance and capacitance are easy to integrate on-chip.

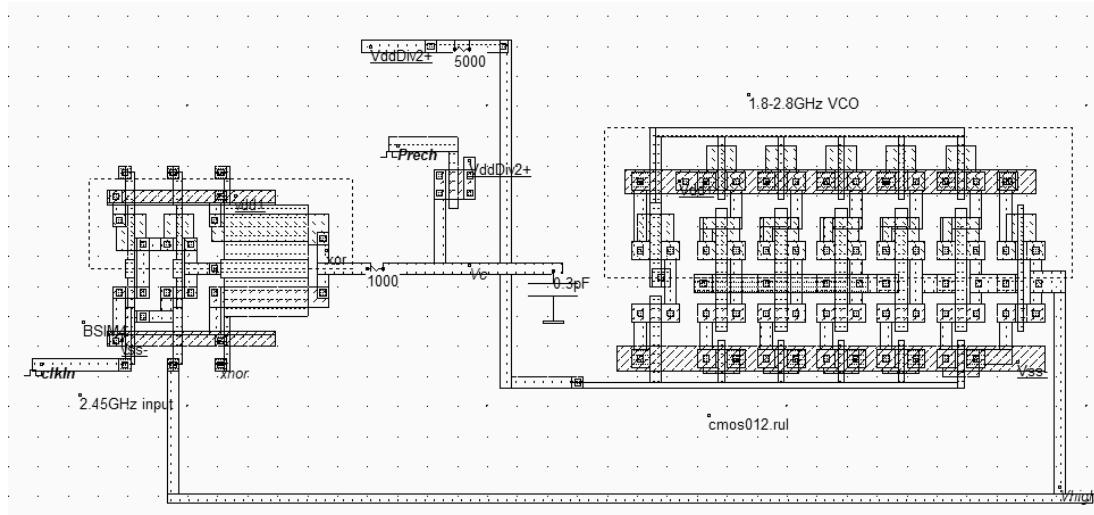


Figure 12-58: Connecting the current-starved VCO to the phase detector (VCOPll.SCH)

The input frequency is fixed to 2.44GHz. During the initialization phase (Simulation of figure 12-59), the precharge is active, which pushes rapidly the voltage of  $V_c$  around VDD/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.

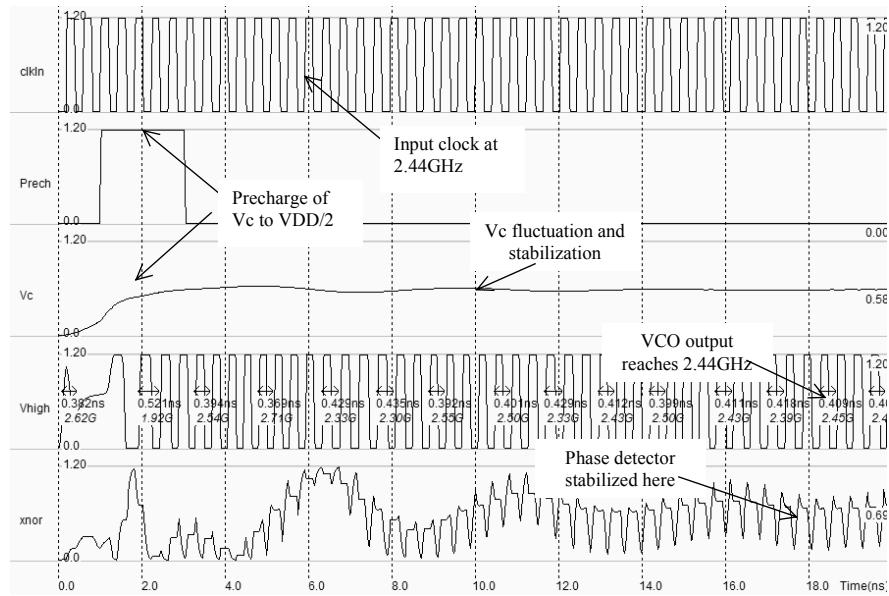


Figure 12-58: Simulation of the PLL showing the locking time (VCOPll.SCH)

### Frequency demodulation

The PLL may be used to transform a frequency into a voltage. When clocks with a small frequency difference are applied serially to the input of the PLL thanks to a multiplexor, the  $V_c$  voltage changes accordingly. A fast clock leads to a high  $V_c$  voltage, a slow clock to a low  $V_c$  voltage.

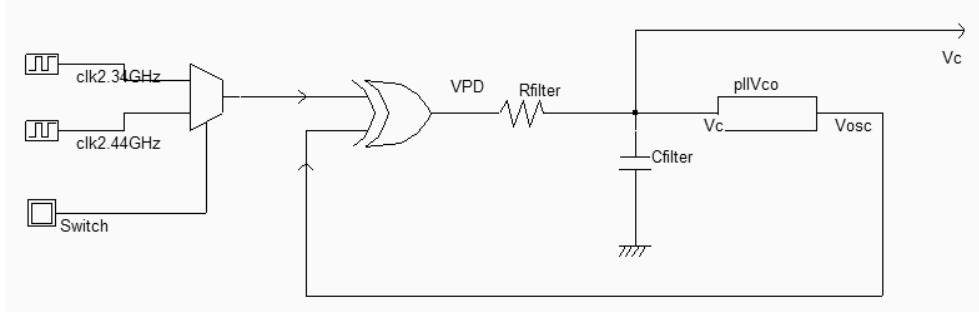


Figure 12-59: Using the PLL for frequency demodulation (Pllfm.SCH)

Multiplexing clocks is done by a CMOS multiplexor, with a switching from one clock to the next every each 25ns. The simulation shows that  $V_c$  is decreased when the input frequency is decreased. Over a certain range of input frequencies, the circuit is able to convert a frequency into a voltage in a linear way, which is the base of frequency demodulation.

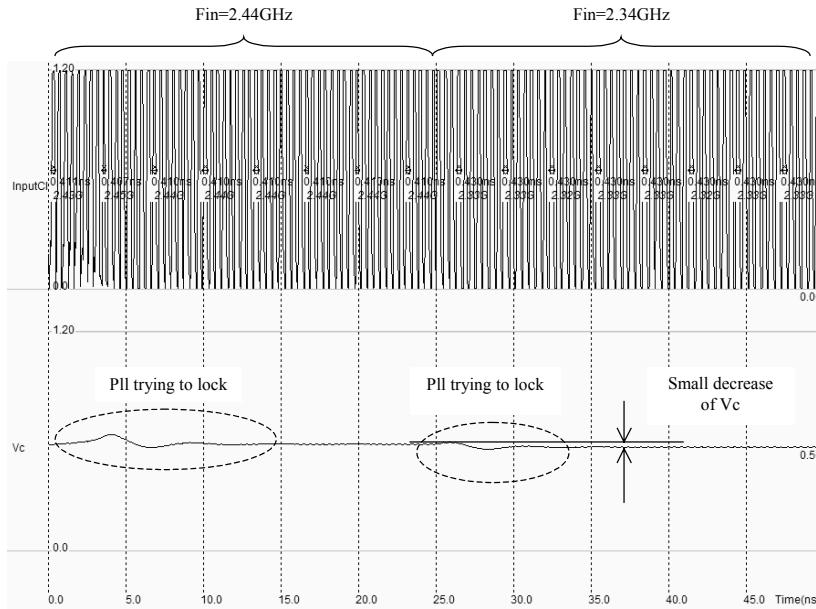


Figure 12-60:Frequency demodulation using the PLL(PllFm.MSK)

### Frequency synthesis

One very important application of PLL in micro-processors and controllers consists in generating a fast on-chip clock from a slow external clock, usually fixed by a quartz. The fast clock signal is synthesized by the VCO and its stability is controlled by the PLL. The new feature is a clock divider circuit on the path of the feedback loop, as shown in figure 12-61. The fast clock is divided by N which can be programmed by the user. For example, a 100MHz external clock may be used to create a 500,600,700 or 800MHz internal clock. In that case, N is 5,6,7 or 8. The VCO should cover the range 500-800MHz with sufficient margin due to process variations.

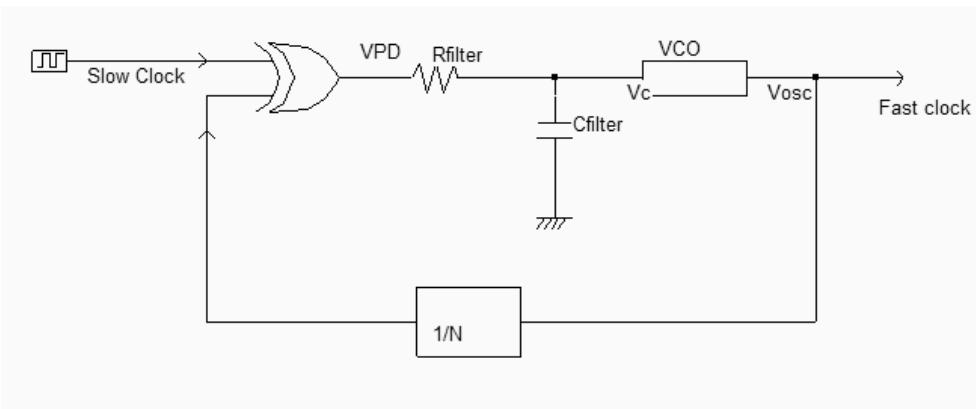


Figure 12-61:Frequency synthesis to generate a fast clock (PllDigital.SCH)

A schematic diagram for the frequency divider by N is given in figure 12-62. The number N is fixed on the keyboard and the circuit performs the division of the input clock *Clock1* by N, with a result appearing on *ClkOut*. In the logic simulation, once the *Reset* is inactive, the number N is fixed on the keyboard.

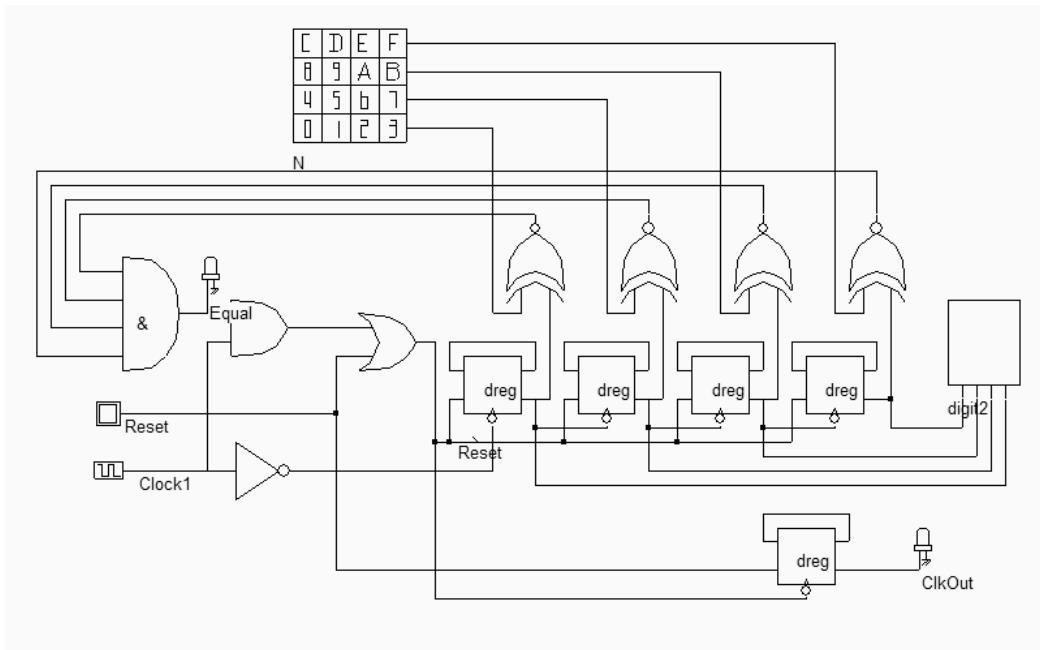


Figure 12-62 Programmable clock divider for the Digital Phase lock loop (PllDivn.SCH)

When the asynchronous counter attains the desired value  $N$ , an *Equal* pulse appears in the loop thanks to the XNOR comparators and the AND gate, which provokes an asynchronous *Reset* and restarts the counter.

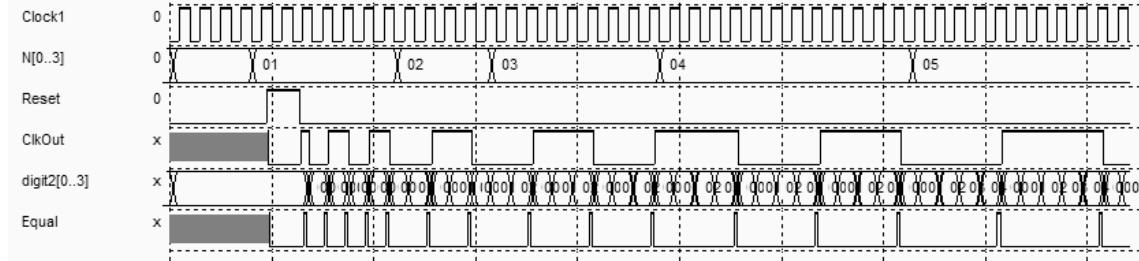


Figure 12-63 Simulation of the clock divider for various values of  $N$  (PllDivn.SCH)

An implementation of the frequency synthesizer into layout is proposed for  $N$  fixed to 8. The reference clock is 100MHz, the VCO target clock is 800MHz. A three stage clock divider is implemented on the layout to divide the VCO clock by 8 before entering the phase comparator. The VCO transistor sizing has been rearranged to produce a 800MHz oscillation around  $VDD/2$ , which eases the locking. Furthermore, the filter capacitance has been increased to 2pF to avoid instability in the VCO.

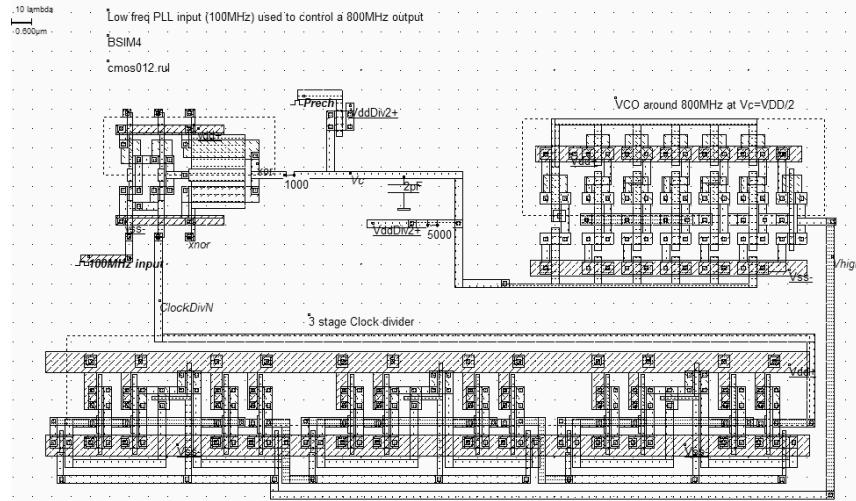


Figure 12-64: A 100MHz external reference clock used to control a 800MHz on-chip clock (PlIDigital.MSK)

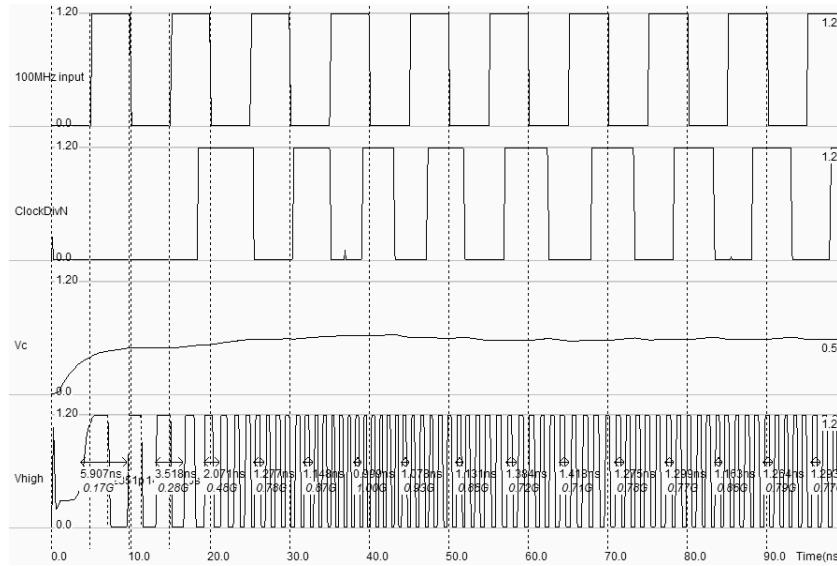


Figure 12-65: Simulation of the 800MHz PLL controlled by a 100MHz external clock (PlIDigital.MSK)

In the simulation of figure 12-65, the first 20ns correspond to the initialization phase. The VCO is warmed up and the clock divider starts to produce the signal *ClockDivN*, which is equal to *Vhigh* divided by 8. Around 80ns are required to lock the VCO to the desired frequency ( $100 \times 8 = 800\text{MHz}$ ). What we observe in the output signal is a phenomenon called jitter: the output frequency is not stable as *Vc* fluctuates around 0.6V. The VCO output exhibits a spread of frequency around 800MHz.

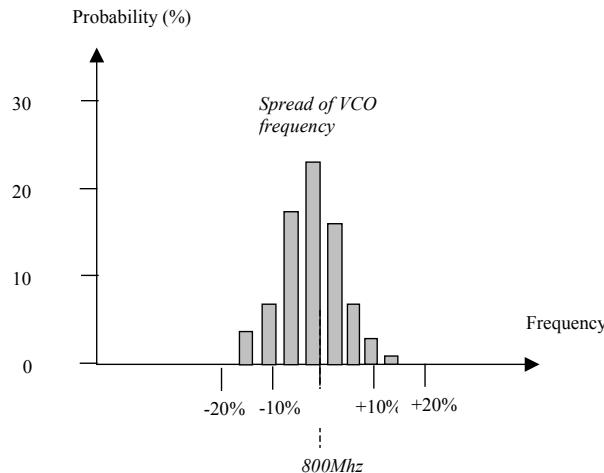


Figure 12-66: The VCO frequency is jittering around 800MHz (PLLDigital.MSK))

The requirements in terms of jitter are very severe in most PLL designs. For example, the PLL used in mobile phones should produce a very stable frequency around 800MHz with less than 100KHz jittering. In some cases however, a voluntary jitter is added to the PLL, which creates a small fluctuation of the synthesized clock. This technique is found in some clock synthesis circuits in micro-controllers, to transform the perfect synchronous clocking into a slightly asynchronous clocking, which lowers the peaks of parasitic interferences.

Several techniques exist to lower the jitter, that are described in [Baker]:

- Lowering the gain of the VCO. The effect of  $V_c$  is less important than the VCO frequency. The drawback is the reduction in frequencies lock range.
- Increasing  $R_{filter}$  and  $C_{filter}$ . The effect of  $VPD$  change is less important on the VCO control. The drawback is a larger locking time, and a larger silicon area required to implement the capacitance. A very high value for the on-chip resistance is not recommended as it generates a parasitic noise proportional to the resistance.
- Reducing the gain of the phase detector. The sizing of the XOR gate may be changed to produce less current. However, the MOS parasitic noise is increased, the design is more sensitive to supply and substrate noise, which tends to cancel the benefits of a lower phase gain.

## 6. Frequency Converter

### Principles

In many situations for radio frequency emitters and receivers, there is a need for shifting an input waveform into a lower or higher frequency waveform. From an emission point of view, most of the signal processing is done within the range 10-100MHz. However, the emission bandwidth may be significantly higher (900MHz, 1.8GHz for mobile, 2.4, 5GHz for wireless local area network).

A direct generation of the desired signal at such a high frequency would consume too much power. A low power frequency translator circuit is preferred. In the case of figure 12-67, the frequency converter shifts the original signal (Say 100MHz) to the desired emission frequency 900MHz.

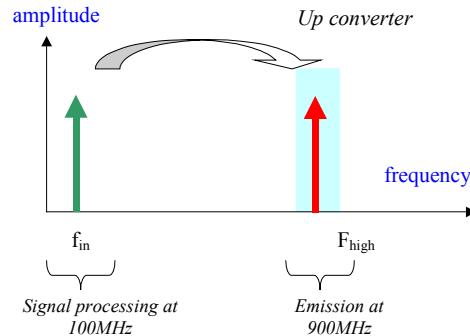


Figure 12-67: the principles for frequency conversion

The operation which translates a high frequency signal into a low frequency signal is called down conversion. In frequency domain, it consists in shifting a high frequency information contained in frequency  $f_{in}$  to a lower frequency  $f_{low}$ , as illustrated in figure 12-68. The information contained in the original signal  $f_{in}$  (Which may include an amplitude, frequency or phase variation) is preserved in the resulting signal  $f_{out}$ .

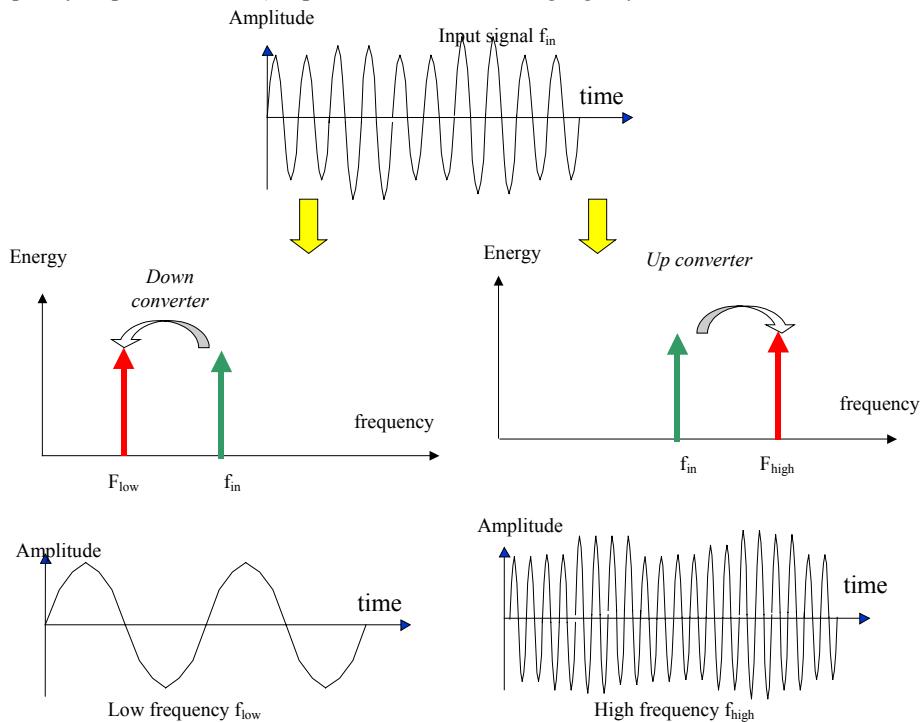


Figure 12-68: the principles for frequency conversion

### Adding sinusoidal waves

Adding sinusoidal waves is very easy. A simple circuit containing 3 resistor produces the addition of two sinusoidal waves, as shown in figure 12-12. The formulation is easily demonstrated using the superposition theorem.

$$V_{out} = \frac{1}{3} [\cos \omega_1 t + \cos \omega_2 t] \quad (\text{Eqn. 12-12})$$

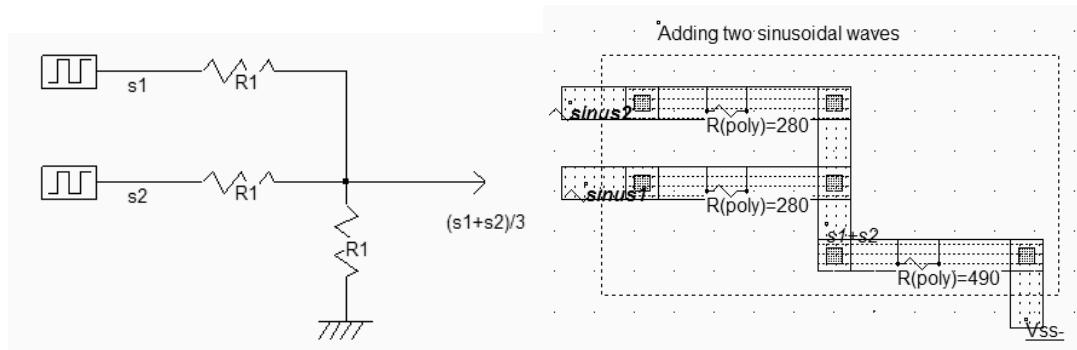


Figure 12-69: Adding sinusoidal waves is easy, a set of 3 resistors is sufficient to build the sum (AddSinus.MSK)

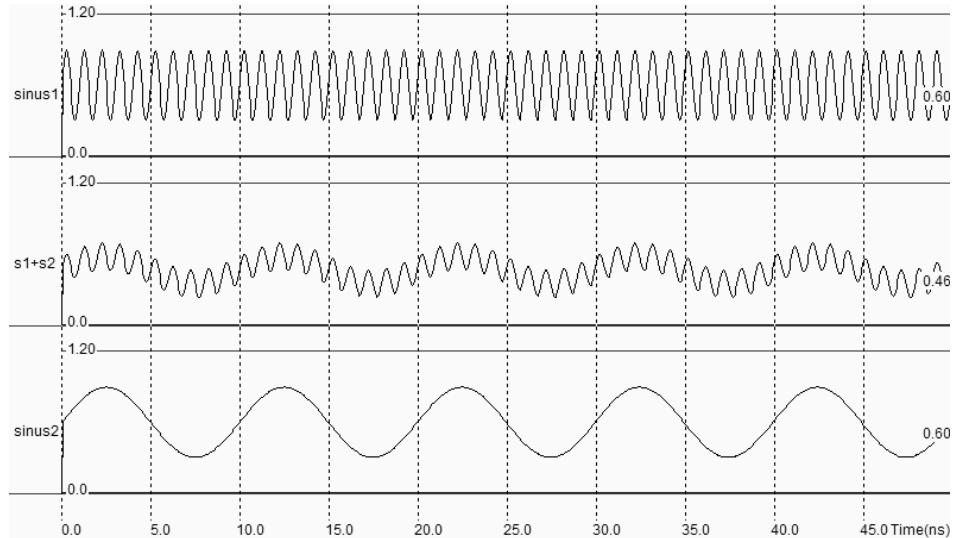


Figure 12-70: The simulation of the sinusoidal wave adder (AddSinus.MSK)

The Fourier transform of the signal  $s1+s2$  reveals two harmonics (Figure 12-71), one at the frequency of signal 1, the other at the frequency of signal 2, as the formulation 12-12 suggested. Clearly, no frequency shift may be obtained using sinusoidal addition.

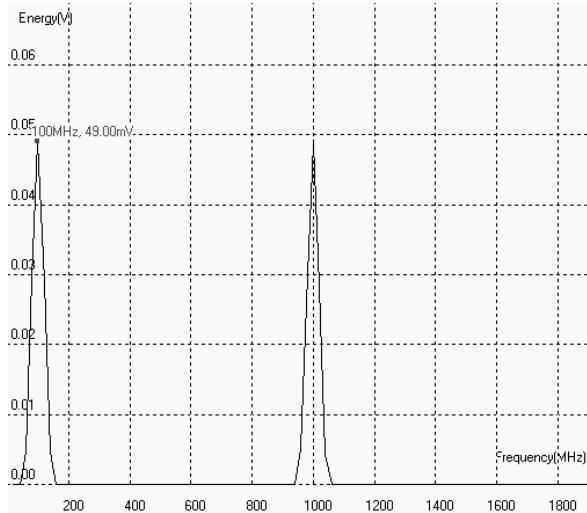


Figure 12-71: The Fourier transform of  $s_1 + s_2$  reveals two harmonics, one at 100MHz the other at 1GHz  
(AddSinus.MSK)

### Multiplying sinusoidal waves

At the core of up/down frequency conversion is the multiplication of two sinusoidal waves in the time domain [Lee]. The result of that multiplication is the generation of two new frequencies: one at the sum of frequency, one for the difference.

$$\sin(\omega_1 t) \cdot \sin(\omega_2 t) = \frac{1}{2} [\sin(\omega_1 - \omega_2)t - \sin(\omega_1 + \omega_2)t] \quad (\text{Equ. 12-13})$$

where

$$\omega_1 = 2\pi f_1$$

$$\omega_2 = 2\pi f_2$$

$f_1$  = frequency of signal 1 (Hz)

$f_2$  = frequency of signal 2 (Hz)

If we consider a low frequency  $f_{in}$ , and a high frequency  $f_{osc}$  and only consider absolute values, the multiplication of these two sinusoidal signals creates two new sinusoidal contributions: one at  $f_{osc} - f_{in}$ , one at  $f_{osc} + f_{in}$  (Figure 12-72). Using an LC resonant circuit, we only keep the desired frequency contribution. In the case of figure 12-72, the L and C values are tuned to highlight the  $f_{osc} + f_{in}$  contribution which fits with the emission bandwidth. The LC resonator also serves as a filter of undesired harmonics, such as  $f_{osc} - f_{in}$  and  $f_{osc}$ .

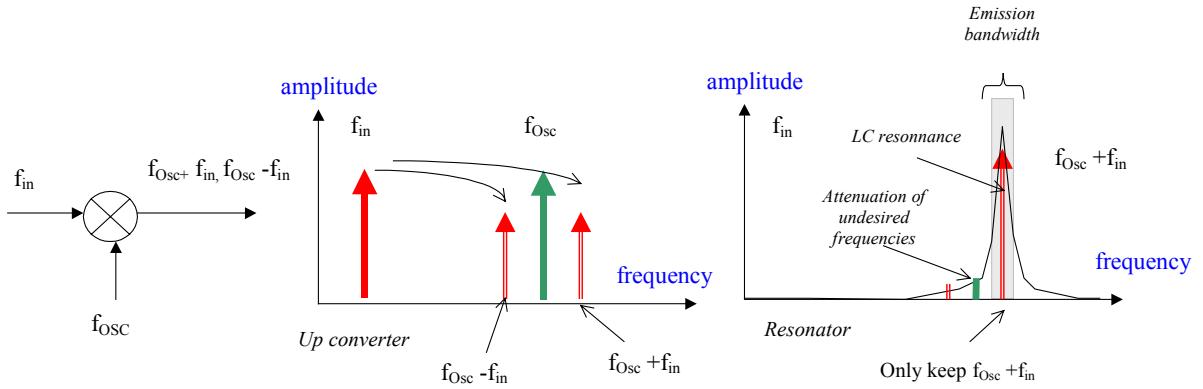


Figure 12-72: The multiplication of two frequencies creates new frequency components

### Using a MOS for Sinus Multiplication

The process for multiplying signals with CMOS devices is far from being simple. The nMOS and pMOS are non-linear devices. The best example is the long channel nMOS which gives approximately a square law dependence between  $V_{gs}-V_t$  and  $I_{ds}$ , as illustrated in figure 12-73. A linear device would give a linear dependence between  $I_{ds}$  and  $V_{gs}$ , which is almost the case for short-channel devices. See chapter 3 for more details about device modeling.

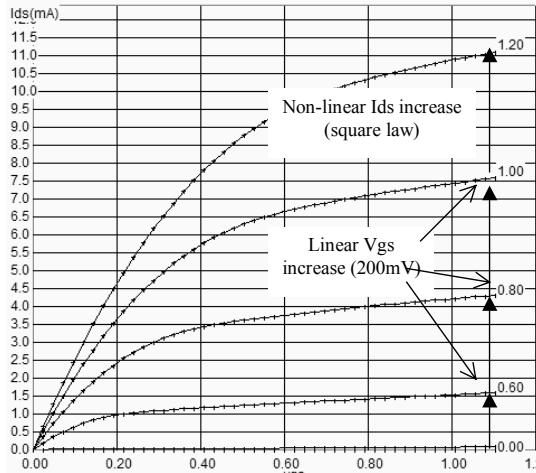


Figure 12-73: The long-channel MOS characteristics exhibit a square dependence of  $I_{ds}$  vs.  $V_{gs}$  (MixerMos.MSK)

The idea is as follows (Figure 12-74): the two sinusoidal inputs  $f_{in}$  and  $f_{osc}$  are added on the gate  $V_{gs}$ . The current  $I_{ds}$  is a non linear function of  $V_{gs}$ . The static characteristics of the device ( $W=50\mu m$ ,  $L=0.5\mu m$ ) show a "quadratic" dependence: each  $V_{gs}$  step induces a square increase of  $I_{ds}$ . This can be simply written as:

$$I_{DS} \approx k.(V_{GS} - V_t)^2 \quad (\text{Equ. 12-14})$$

where

$k$  depends on the design and technology

$V_t$  is the threshold voltage (Around 0.35V)

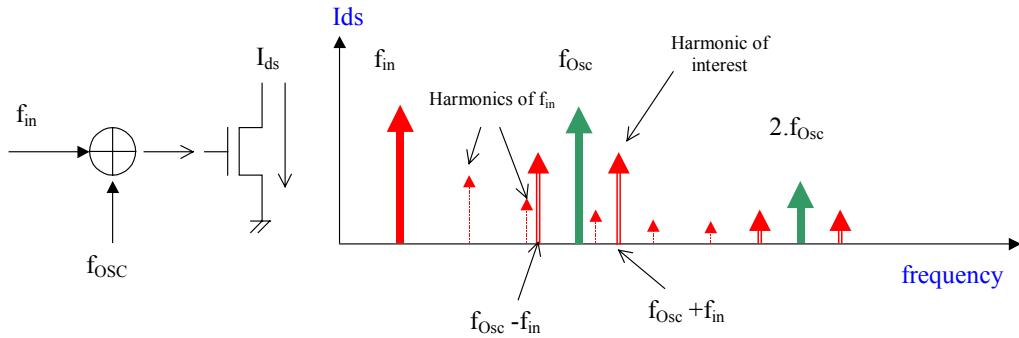


Figure 12-74: The  $I_{ds}$  current exhibits several harmonics, including the desired high frequency  $f_{osc}+f_{in}$  (MixerMos.MSK)

If  $V_{gs}$  is a sum of sinusoidal waveforms, as we did in the previous section, the current may be written as:

$$I_{DS} \approx k [V_{bias} + v_{in} \cdot \sin(\omega_{in}t) + v_{osc} \sin(\omega_{osc}t) - Vt]^2 \quad (\text{Equ. 12-15})$$

$$I_{DS} \approx I_{DS0} + k_1 [v_{in} \cdot v_{osc} (\sin \omega_{osc}t \cdot \sin \omega_{in}t)] \quad (\text{Equ. 12-16})$$

$$I_{DS} \approx I_{DS0} + \frac{k_1}{2} [v_{in} \cdot v_{osc} \sin(\omega_{osc} + \omega_{in})t - \sin(\omega_{osc} - \omega_{in})t] \quad (\text{Equ. 12-17})$$

The most important result beyond this approximation is that the input signal and the oscillator signal are indeed multiplied and create the desired harmonics. In other words, passing a sum of sinusoidal waveforms into a non-linear device creates several harmonics, from which  $f_{in}+f_{osc}$  and  $f_{in}-f_{osc}$  are the most important. The desired harmonic is underlined in equation 12-17 by rearranging the product of sinus into a sum of sinus. The term  $I_{DS0}$  also contains the original input signal, the oscillator signal and all their respective harmonics too, which lead to a quite complex output. A band-pass filter is mandatory to eliminate undesired harmonics and amplify the desired signal. The circuit is called a single-balanced mixer.

## Layout Implementation

The n-channel MOS implemented in the mixer layout must have a large length to eliminate short channel effects and exhibit a square law dependence between  $V_{gs}$  and  $I_{ds}$ . This is the case of MOS devices with a length larger than 0.5μm. A resistance load is mandatory to perform amplification. The resistor is matched to the  $R_{on}$  resistance of the nMOS device. The input is the sum of two sinusoidal components, through a resistor bridge (Figure 12-75).

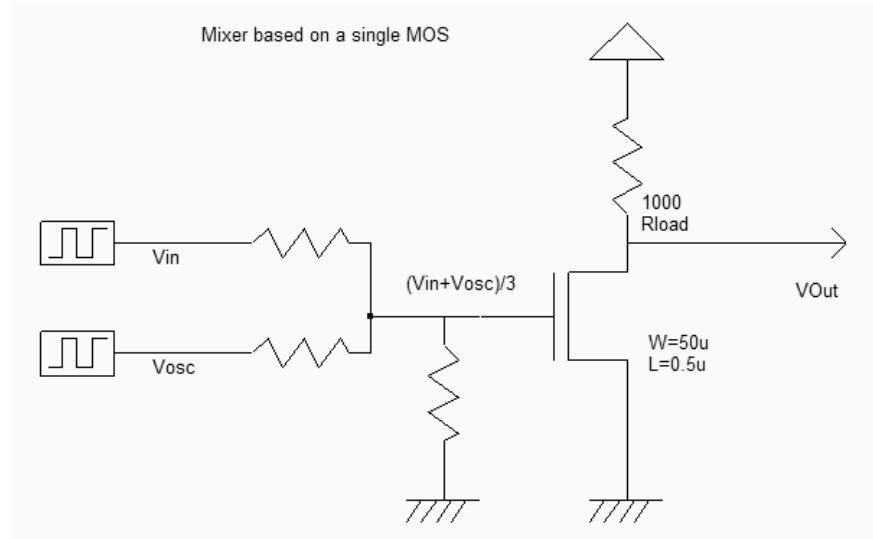


Figure 12-75: Building a single-balanced mixer with an n-channel MOS device (MixerMos.SCH)

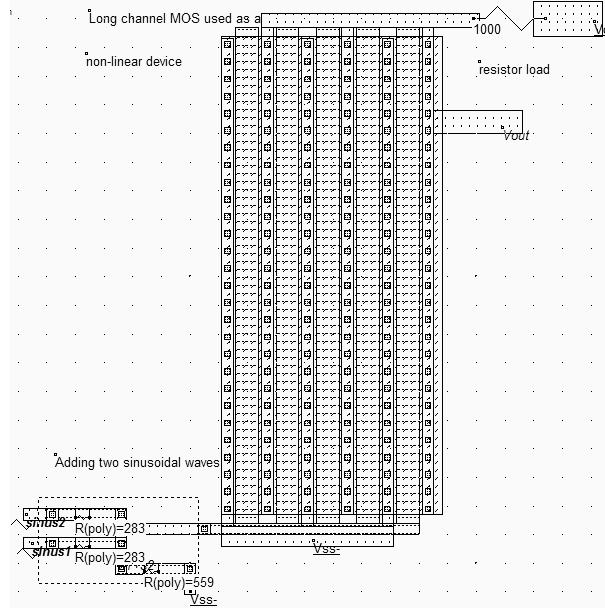


Figure 12-76: Design of a mixer using a large width, large length nMOS device, with a sum of sinusoidal waves at the input (MixerMos.MSK)

Notice the unusual aspect of the MOS device in the layout shown in figure 12-76. Five gates are connected in parallel, which is equivalent to one single MOS with the sum of channel widths, but at the same time, the length is enlarged to obtain a sufficient quadratic dependence between voltage and currents which is the main origin of harmonics . According to the theory, the time-domain simulation of the mixer reveals that the signal  $V_{out}$  has a very complex aspect (Figure 12-77).

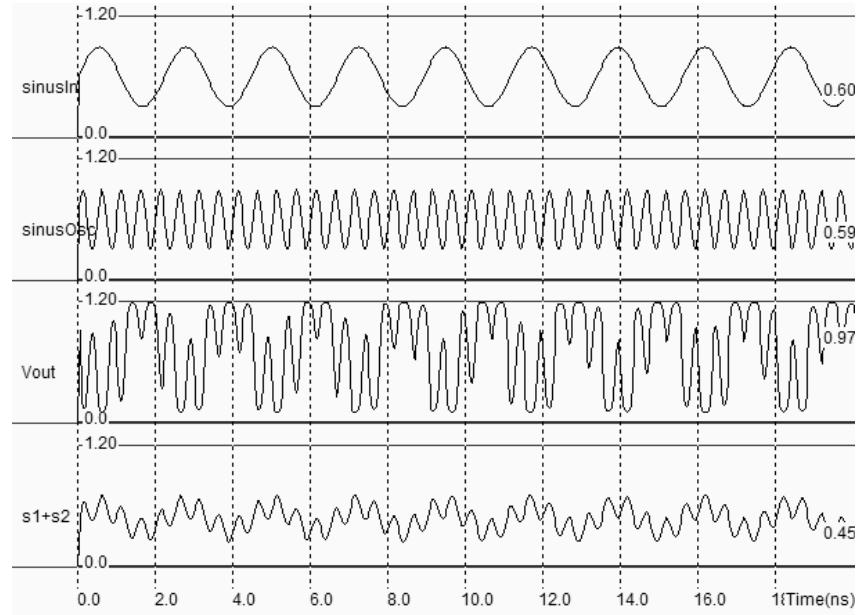


Figure 12-77: Simulation of the mixer with a 450Mhz and 2Ghz added inputs (MixerMos.MSK)

The Fourier Transform is obtained by a click on "FFT" in the simulation window (Figure 12-78). The 450MHz input signal, the 2GHz oscillator signal, as well as the harmonics and products are present in the spectrum. The only desired harmonic is the 2.45GHz contribution, corresponding to  $f_{in} + f_{osc}$ .

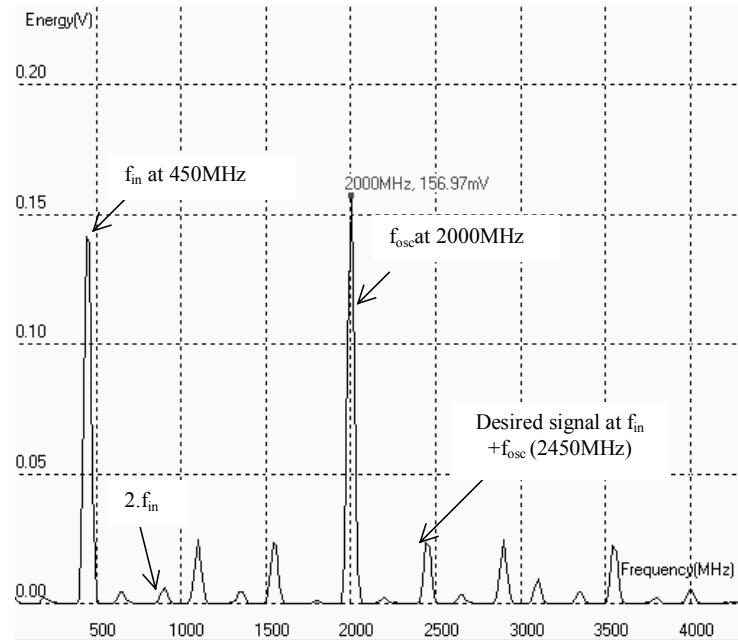


Figure 12-78: The output voltage includes  $f_{in}$ ,  $f_{osc}$  and their corresponding harmonics. The desired signal is at  $f_{osc}+f_{in}$  (MixerMos.MSK)

### Mixer with LC resonator

The mixer shown in figure 12-79 has two important features: the serial resistor is replaced by an inductor  $L_{HF}$  of 3nH, and a capacitor  $C_{HF}=1.2\text{pF}$  is added to the output. The LC resonator formed by the inductor  $L_{HF}$  and the capacitor  $C_{HF}$  matches the target frequency 2.45GHz (Use the resonant frequency evaluator in the **Analysis** menu to confirm). The serial resistor  $RL$  accounts for the finite quality of the inductor, and corresponds to the long metal wire resistance of the physical inductor. Removing this serial resistor would create overestimated oscillations, possibly numerical instability, and the results could not be exploited.

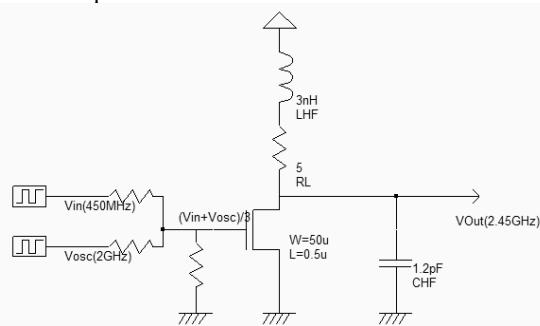


Figure 12-79: The schematic diagram of a mixer with a LC resonator tuned to 2.45GHz

The mixer implementation has not been completed in the layout of figure 12-80. For simplicity's sake we used virtual L and C rather than a physical inductor. The 3nH inductor is placed in series with a parasitic resistance which accounts for the physical serial resistance of the on-chip inductor, and limits the LC resonance effect. The capacitor 1.2pF is also virtual, and is placed near  $V_{out}$ .

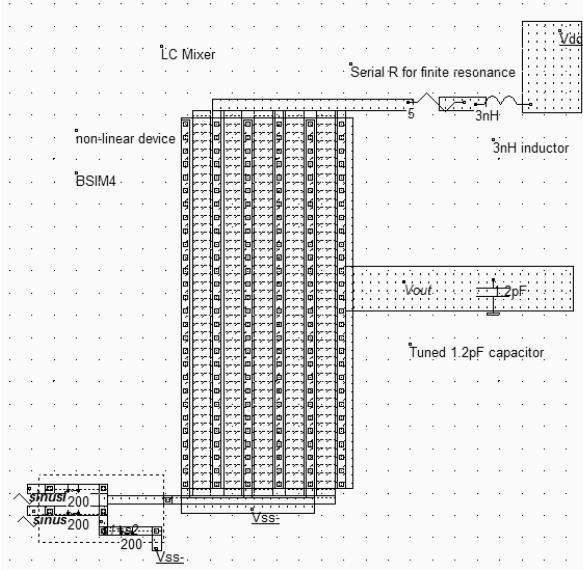


Figure 12-80: The mixer with a tuned LC resonator targeted to 2.45GHz (MixerLC.MSK)

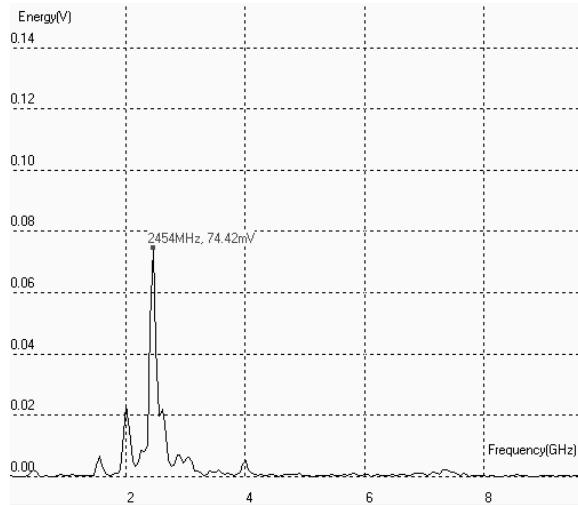


Figure 12-81: The Fourier transform of  $V_{out}$  shows a main contribution at the desired frequency and residues of other harmonics (MixerLC.MSK)

The Fourier transform of the time-domain simulation is proposed in figure 12-81, and corresponds to the output node  $V_{out}$ . The desired signal at 2.45GHz appears much more clearly than in figure 12-78, because of the pass-band passive resonator centered around that frequency. Unfortunately, the selectivity of the LC circuit is not high enough to erase the oscillator frequency at 2GHz. Residues of other harmonics also appear in the Fourier transform: 1.6GHz, 4GHz.

An increase of the input frequency  $f_{in}$  is translated into a corresponding increase of the output frequency. For example, a slow increase in  $f_{in}$  shifts the main peak to the right in a proportional way. Also, an increase of the amplitude of  $f_{in}$  induces a corresponding increase of the 2.45GHz harmonic. This property is illustrated in figure 12-82 by adding a regular increase of the sinusoidal input (Parameter **Increase f**).

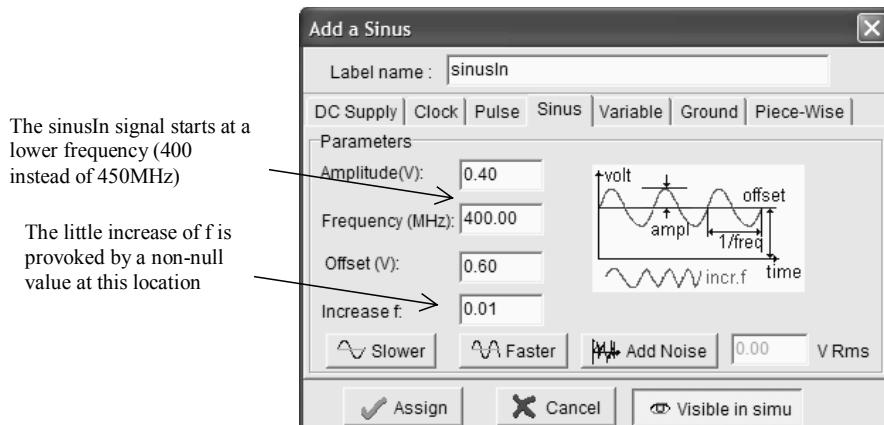


Figure 12-82: The input SinusIn starts from 400MHz and slowly rises to 500MHz (MixerLC.MSK)

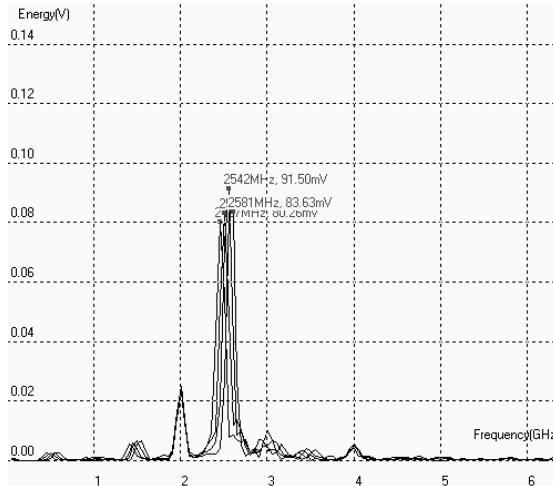


Figure 12-83: A little increase of the input frequency is translated into an increase of the main harmonic (MixerLC.MSK)

The evolution of the FFT of  $V_{out}$  shows a shift in the peak resonance, due to the fact that the input sinusoidal wave has also shifted toward high frequencies. This illustrates an important property of mixers that are conservative in terms of amplitude and frequency variations, except that the output frequency is situated at a fixed distance of the the input frequency.

### Double-balanced Mixer

The main drawback of the mixer output provided by the LC mixer is the important amount of parasitic signals added to the desired signal. The undesired signals 2.55GHz ( $f_{osc}-f_{in}$ ), 2GHz ( $f_{osc}$ ), 2.9GHz ( $f_{osc}+2.f_{in}$ ), 4GHz (2. $f_{osc}$ ), appear in the spectrum and should be eliminated. A very brilliant idea would consist in creating two signals where all harmonics would be in opposite phase except the desired harmonics which would be in phase. Adding these two signals would create a miraculous signal with  $f_{osc}+f_{in}$  and  $f_{osc}-f_{in}$ .

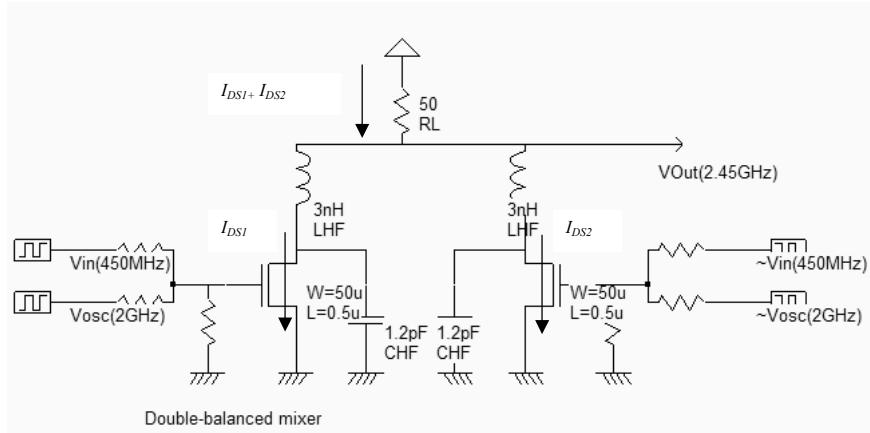


Figure 12-84: Implementation of the double-balanced mixer (MixerDoubleLC.SCH)

A circuit that realizes this function is proposed in figure 12-84. The signals  $vin$  and  $Vosc$  are combined as seen previously, in the left branch of the mixer, on the gate of the n-MOS device. The current that flows through the nMOS device situated on the left is  $Ids1$ , which can be approximated by equation 12-18. In the right branch of the mixer, the signals  $\sim vin$  and  $\sim Vosc$ , representing the same signals as  $vin$  and  $Vosc$  but with an opposite phase, are combined on the gate of the second n-MOS device. The current that flows on the right nMOS device is  $Ids2$  which can be approximated by equation 12-19.

$$I_{DS1} \approx k[V_{bias} + v_{in} \cdot \sin(\omega_{in}t) + v_{osc} \sin(\omega_{osc}t) - Vt]^2 \quad (\text{Equ. 12-18})$$

$$I_{DS2} \approx k[V_{bias} - v_{in} \cdot \sin(\omega_{in}t) - v_{osc} \sin(\omega_{osc}t) - Vt]^2 \quad (\text{Equ. 12-19})$$

Developing equations 12-18 and 12-19, the sum can be arranged as:

$$I_{DS1} + I_{DS2} \approx k[I_{DS0} + 2v_{in} \cdot \sin(\omega_{osc} + \omega_{in})t + 2v_{in} \cdot \sin(\omega_{osc} - \omega_{in})t] \quad (\text{Equ. 12-20})$$

The remarkable point that can be seen in equation 12-20 is that the sum of currents  $I_{ds1}+I_{ds2}$  that flows in the 50ohm load resistor  $RL$  mainly includes a constant value  $I_{ds0}$  and the mixer products at frequencies  $f_{osc}+f_{in}$  and  $f_{osc}-f_{in}$ , which was exactly the goal of the mixer.

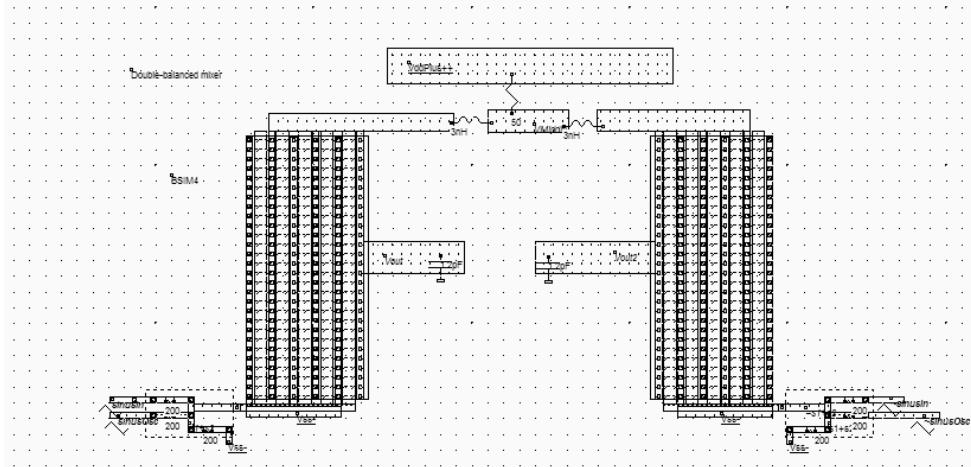


Figure 12-85: Layout of the double-balanced mixer (MixerDoubleLC.MSK)

The layout implementation (Figure 12-85) makes an extensive use of virtual R,L,C elements. This technique is recommended for the tuning of the circuit, but one should remember that the final goal is a complete layout implementation. The simulation performed in figure 12-86 confirms the theoretical assumption: the Fourier transform clearly includes the two main contributions near 1500MHz and 2500MHz, without  $f_{osc}$  in between. Removing the undesired harmonics is quite easy, in order to keep the desired 2500MHz contribution.

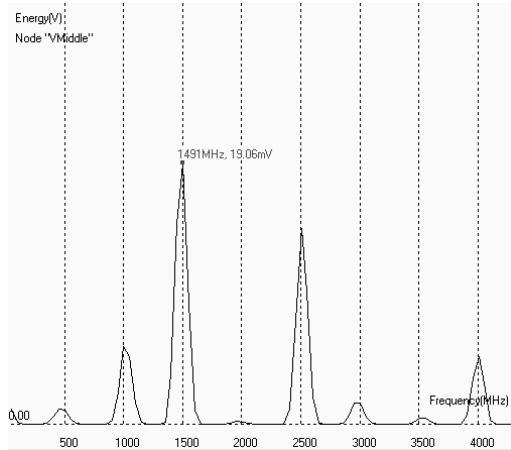


Figure 12-86: Fourier transform of the double-balanced mixer output (MixerDoubleLC.MSK)

### Gilbert Mixer

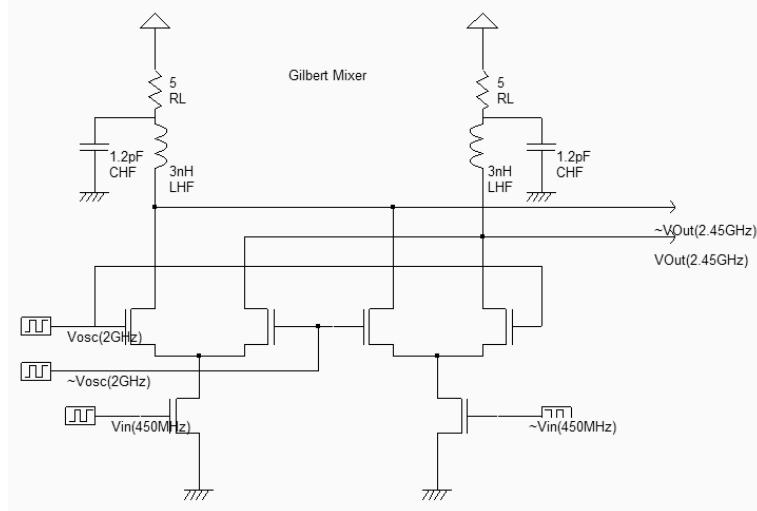


Figure 12-87: The Gilbert mixer (MixerGilbert.SCH)

The double-balanced mixer is not implemented using a resistor based voltage adder, as suggested in the schematic diagram shown previously (Figure 12-84). Most mixers use the Gilbert cell [Gilbert] which consists of only six transistors, and performs a high quality multiplication of the sinusoidal waves [Lee]. The schematic diagram shown in figure 12-87 uses the tuned inductor as loads, so that  $V_{out}$  and  $\sim V_{out}$  oscillate around the supply VDD.

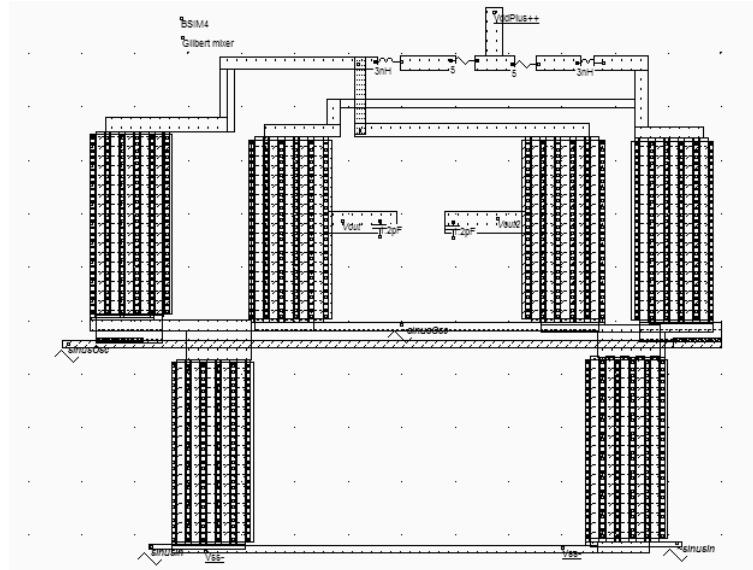


Figure 12-88: The Gilbert mixer implementation with virtual R,L and C (MixerGilbert.MSK)

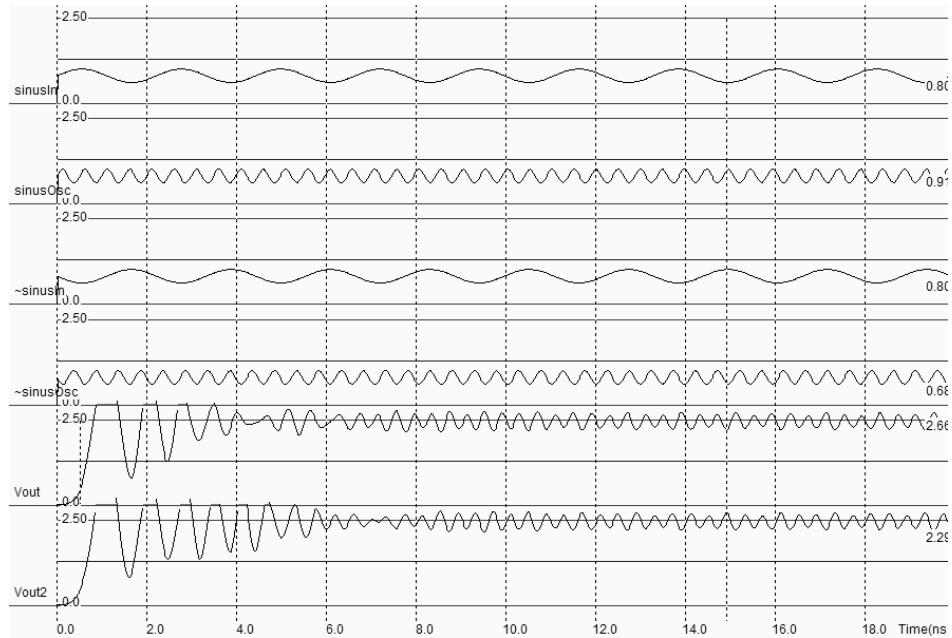


Figure 12-89: Time-domain simulation of the Gilbert mixer (MixerGilbert.MSK)

The implementation shown in figure 12-88 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.

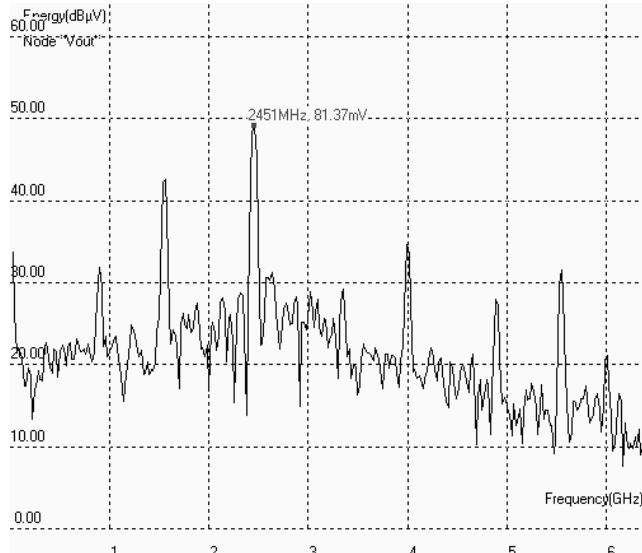


Figure 12-90: Fourier transform of the Gilbert mixer output (MixerGilbert.MSK)

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  $f_{osc}-f_{in}$ .

Notice that the simulation time has an influence on the Fourier Transform result: a short simulation (5ns) would lead to a poor precision in our frequency range of interest, but a high precision on very high frequencies (Above 10GHz). In our case, it is preferable to perform the time domain simulation over a large time (50ns) which will give a high precision at low frequencies (From DC to 5GHz), but to limit the Fourier spectrum to around 10GHz. As the target frequency is around 2.5GHz, a 50ns simulation gives the best results.

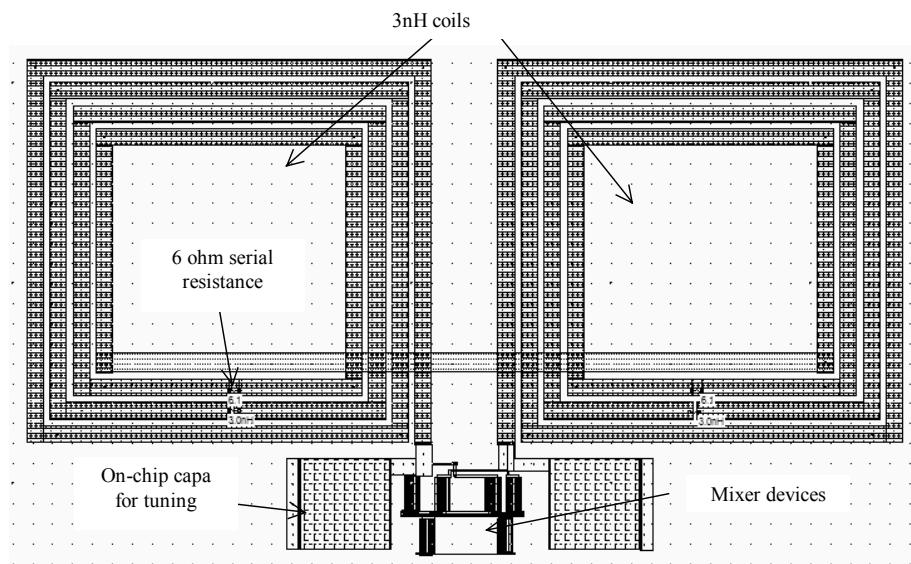


Figure 12-91: The complete implementation of a Gilbert mixer circuit (MixerGilbert2.MSK)

In figure 12-91, a complete implementation of the Gilbert mixer has been realized, so that virtual R,L and C components are replaced by physical elements. The coils have a target 3nH inductance, and their associated parasitic resistance is approaching 6 ohm when the combination of metal6,metal5 and metal4 are used. The tuning capacitor is added to the parasitic coil capacitor to perform the best resonance at the desired 2.5GHz frequency. The design relies on good models for the inductor and capacitor, which is not the case in the Microwind software which uses first order approximations of parasitic resistance, capacitance and coil inductance. In a real case implementation, we may expect significant differences between measurements and simulations. Having accurate predictions of such circuits is quite challenging.

## 7. Sub-sampling Frequency Converter

Let us recall that the frequency down conversion consists in shifting the input signal with a frequency  $f_{in}$  down to a lower frequency  $f_{out}$ , without altering its amplitude or frequency modulation. One interesting solution consists in using a transmission gate with a very accurate tuning of the gate clock.

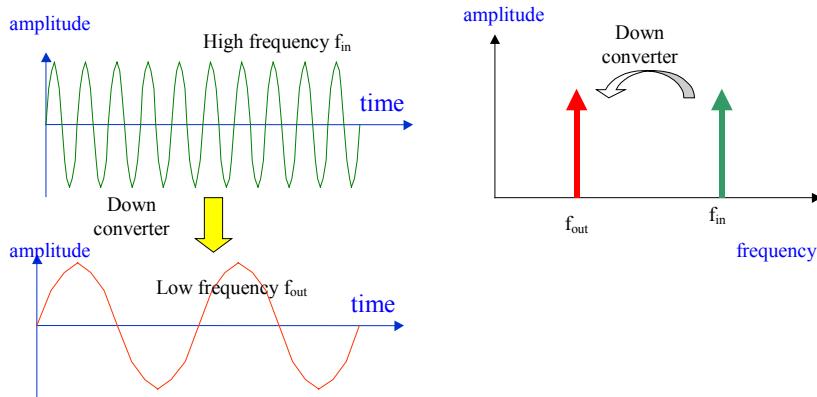


Fig. 12-92: Principles for down conversion

As an illustration, we use a 1.900 GHz sinusoidal wave (*DataIn*), and a 1.818 GHz sampling signal (*Enable*). The expected output frequency is therefore  $1.900 - 1.818 = 0.082$  GHz, that is 82MHz. The layout of the sample circuit is a simple transmission gate with an RC filter (Figure 12-93).

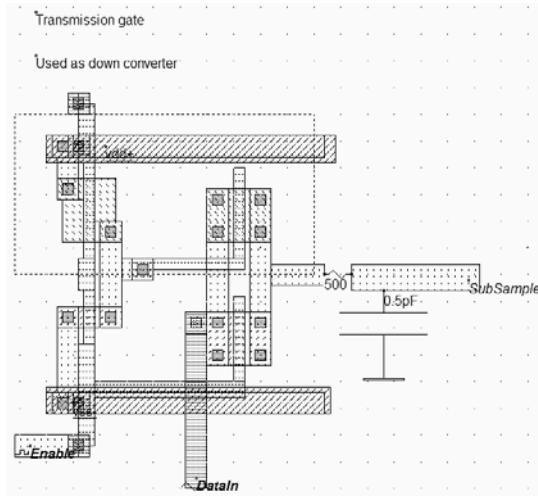


Fig. 12-93. Layout of the transmission gate and RC filter used for down conversion (DownConverter.MSK)

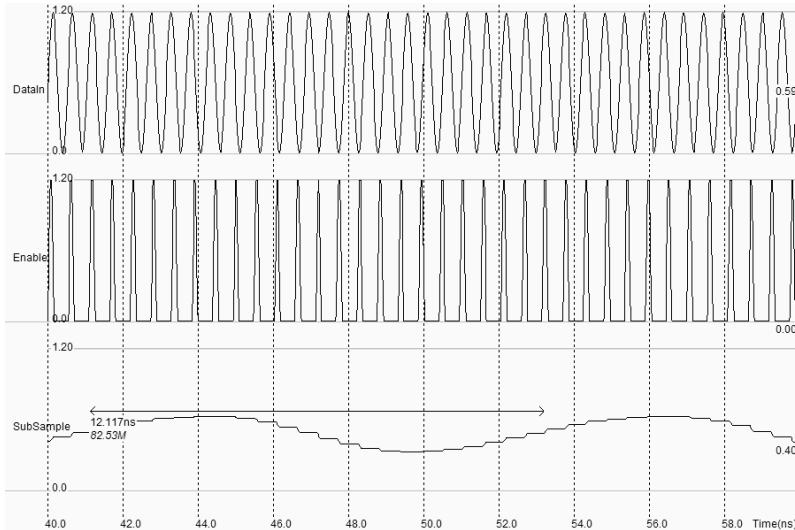


Fig. 12-94. Down conversion of the 1.9GHz input sinusoidal wave to a low frequency near 82MHz

The sampling signal *Enable* operates at a rate slightly slower than the input frequency, which leads to a signal at low frequency at the output of the transmission gate (Figure 12-94). With a simple RC filtering, the output signal becomes a sinusoidal wave with a frequency equal to the difference between  $f_{DataIn}$  and the gate frequency  $f_{Enable}$ . When simulating over a 20ns time interval (Figure 12-94), and asking for the evaluation of the frequency of the output *subSample*, we find 82MHz as expected.

## 8. Conclusion

In this chapter, we have described in the first part the role of on-chip inductor for resonant circuits. Then, we have detailed the power amplifiers and the associated notions of power efficiency and amplifier class. Thirdly, we have presented some circuits used to generate oscillations, based on ring oscillation and passive LC networks. Next, we

have described the main parts of the phase lock loop and illustrated three applications in the GHz range. The frequency conversion was presented through the addition of sinusoidal waveforms in non-linear devices. We also presented the Gilbert mixer and looked at the frequency conversion performances in frequency domain thanks to the Fourier Transform. Finally, the sub-sampling principles were applied to frequency down conversion through a simple example.

## References

- [Macnamara] Macnamara T. "Handbook of Antennas for EMC", Artech House Publishers, ISBN 0-89006-549-7
- [Weste] Neil H. E. Weste, Kamran Eschraghian "Principles of CMOS VLSI Design", Addison-Wesley, 1993, ISBN 0-201-53376-6
- [Niknejad] Niknejad Ali M., Meyer, robert G. "Design, simulation and applications of Inductors and Transfromers for Si RF Ics", Kluwer, 2000, Isbn 0-7923-7986-1
- [Lee] T.H.Lee "The Design of Radio-frequency Integrated Circuits", Cambridge University Press, 1998, ISBN 0-521-63061-4
- [Wheeler] H. A. Wheeler "Simple Inductance Formulas for Radio Coils", Proceedings of the IRE, Oct. 1928, pp 1398-1400
- [Thompson] M. Thompson "Inductance Calculation Techniques – Part II: Approximations and Handbook Methods", Power Control & Intelligent Motion, Dec 1999
- [Hella] Mona M. Hella, Mohammed Ismail "RF CMOS power Amplifiers, theory, Design and Implementation", Kluwer academic publishers, 2002, ISBN 0-7923-7628-5
- [Bendhia] Sonia Bendhia "xxx" Delay cell reference

## Exercises

### Exercise 12-1

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.

### Exercise 12-2

Optimize the power amplifier for a maximum power efficiency delivered to a  $30\Omega$  load, as the radiating resistance is often closer to that value than the standard  $50\Omega$ .

### Exercise 12-3

Design a LC oscillator targeted to 5.4GHz, corresponding to the frequency used in wireless area network protocols such as IEEE 802.11a.

**Exercise 12-4**

Redesign the high-performance VCO to oscillate around 5.4GHz, with a span of 0.5GHz (Corresponds to IEEE 802.11a).

**Exercise 12-5**

The main problem of the XOR phase detector is that the "ideal" position corresponds to a phase difference of  $\pi/2$ . In high performance PLL applications, an other type of phase detector is used, as shown below. Implement the phase detector and extract the effect of the phase difference between  $clkDiv$  and  $clkIn$  to the voltage  $Vc$ .

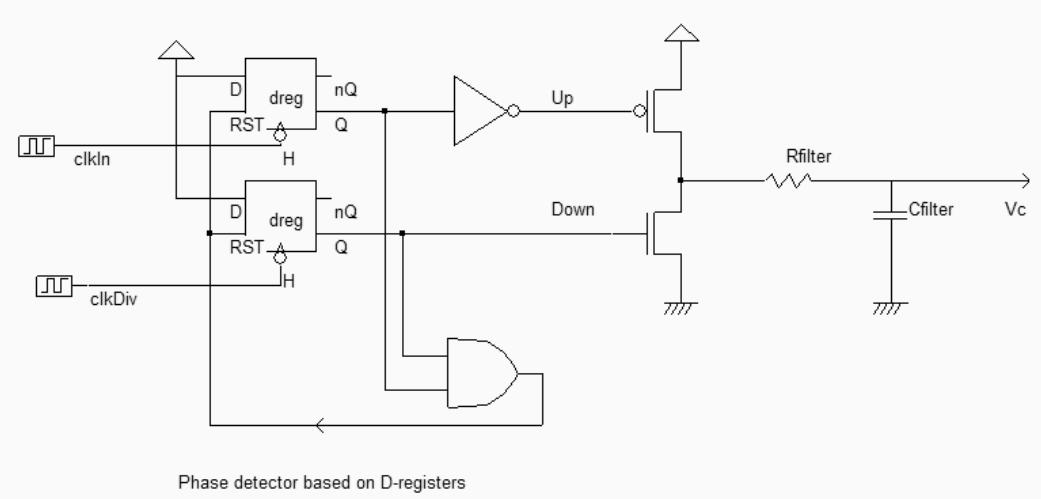


Figure 12-95. The D-Latch phase detector at work (PhaseDetectD.SCH)

*Answer:* See phaseDetectD.MSK

# 13      Converters and Sensors

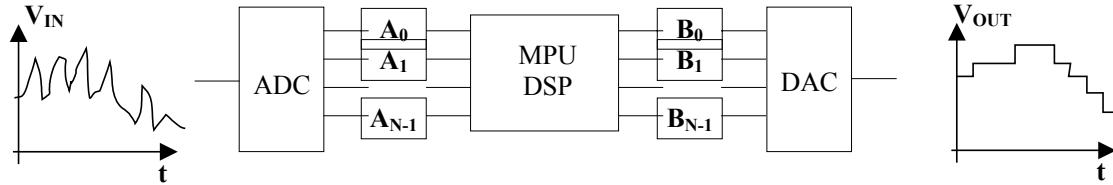
In this chapter, we shall discuss the basic principles of data converters and give an overview of their architecture. The data converter design and implementation is also discussed, with emphasis on basic building blocks, the use of comparators, voltage reference, sampling structures, etc... We also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

## 1. Introduction

Our environment is full of analog signals that we need to monitor, to capture, to treat, to store, to modify and transmit, such as sound, temperature, humidity, light, radio frequency waves, acceleration... A modern way to treat analog signals is to convert them into digital signals. The advantages of using digital techniques called signal processing are the programmability, the stability, the repeatability, the accuracy, the noise immunity, but also the ability to implement special functions such as linear phase filters, error correcting codes ...

The digital signal is a variable whose possible values are 0 or 1, which corresponds to a low or a high voltage. As the opposite of an analog signal is a continuous time signal whose response with respect to time is uninterrupted.

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 13-1. 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  $B$ . In this case example,  $B$  has the same dimension as for the input signal  $A$ . 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 13-1. Basic principle of  $N$  bits analog to digital and digital to analog converters.

A typical function of this signal processing circuit is to filter the high frequency components of the input  $V_{in}$ . Consequently,  $V_{out}$  only contains the slow-varying portion of  $V_{in}$ . The figure below shows some target applications of ADC and DAC converters, with the frequency range in X axis and the converter resolution in Y axis [Gustavsson][Baker]. Low frequency, low resolution data conversion mainly concern low quality voice, as found in phones. Mobile phones typically operate at 8000Hz, 12 bits. Digital audio in CD players works with 20 bit converters at a frequency of 44KHz. Digital video and high speed internet have created specific demand on high-speed data converters. Finally, sampling rates around 1GHz are still a specificity of very high speed instrumentation such as GHz bandwidth oscilloscopes.

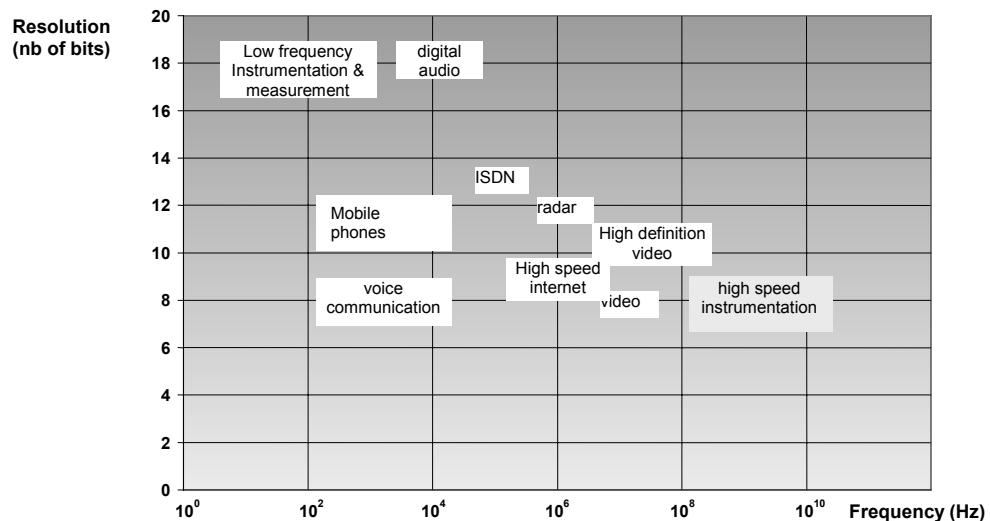


Figure 13-2. Speed and resolution requirements on ADCs

## 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  $V_{OUT}$  (figure 13-3).

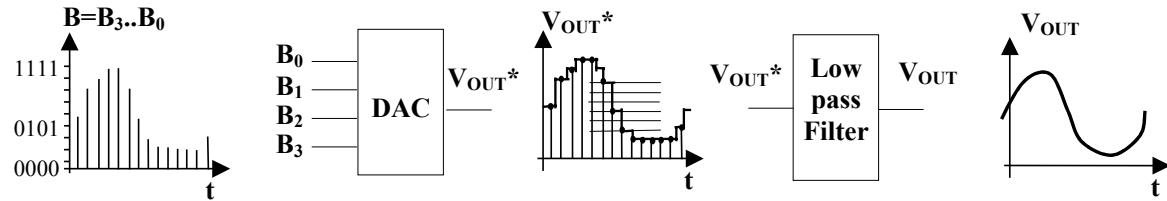


Figure 13-3. A four-bits digital conversion including a filter module.

A wide variety of DAC architectures exists, from very simple to complex ones. Each of them has its own merits and limits. The digital signal can be provided in many different codes, depending on the final application: binary, thermometer, Gray, two's complement , offset binary, and so on. This concepts are presented in the following paragraphs.

### 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 13-4, the resistance ladder includes 8 identical resistors which generate 8 reference voltages equally distributed between 0 and  $V_{dac}$ .

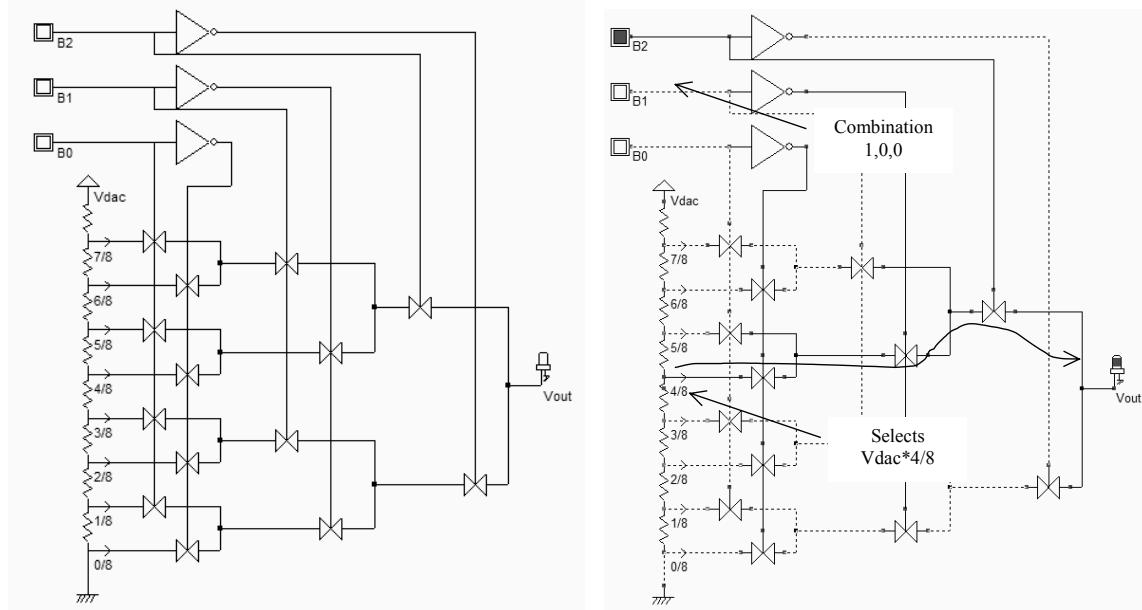
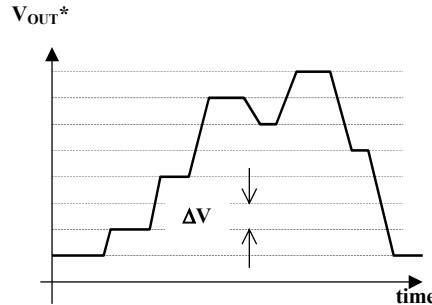


Figure 13-4. Schematic diagram of the digital-analog converter (DAC3bit.SCH)

The digital-analog converter uses the three-bit input ( $B_2, B_1, B_0$ ) to control the transmission gate network which selects one of the voltage references (A portion of  $V_{dac}$ ) which is transferred to the output  $V_{out}$ . Let us consider  $V_{dac}=1.2V$ , which corresponds to the core voltage of the CMOS  $0.12\mu m$  process. The voltage step can be expressed as:

$$\Delta V = \frac{V_{dac}}{2^N} = \frac{1.2}{8} = 0.15V \quad (\text{Eq. 13-1})$$

The correspondence between the input B and the output  $V_{out}^*$  is given in figure 13-5, considering  $V_{dac}=1.2V$ .



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 13-5. The specifications of a 3-bit digital-to-analog converter

### Layout considerations

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. The resistance per square is quite small (Around  $4\Omega$  per square) due to this thin metal coat, as seen in the cross-section of figure 13-6. In order to increase the sheet resistance value, 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  $40\Omega$  per square (figure 13-6 right).

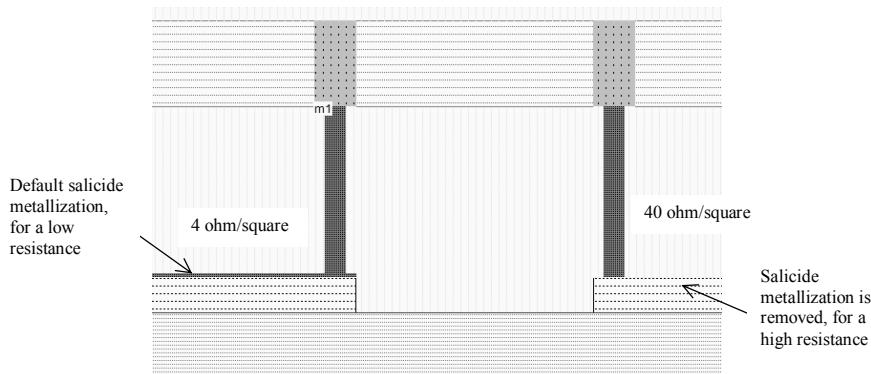


Figure 13-6. Removing the salicidation to increase the sheet resistance

Following a double click in this layer, we activate the property "**Remove salicide to increase resistance**" (Figure 13-7). Consequently, the resistor value is multiplied by 10 and can be used to design an area-efficient resistor network.

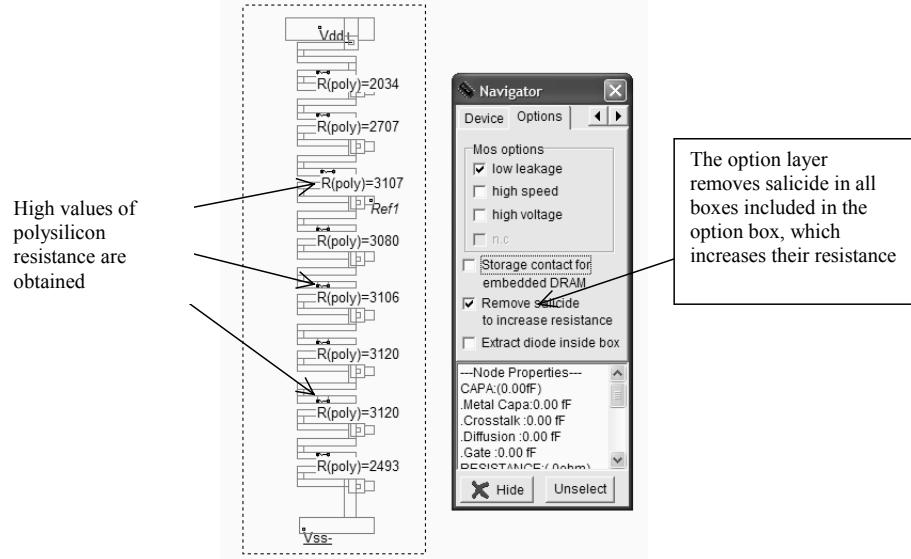


Figure 13-7. The sheet resistance is increased by removing the salicide deposit, thanks to an option layer (ADC.MSK)

The resistor ladder generates intermediate voltage references used by the voltage comparators located in the middle. By default, Microwind does not take into account any serial resistor. This means that the resistor ladder layout on the left of figure 13-8 is considered as a short cut between VDD and VSS.

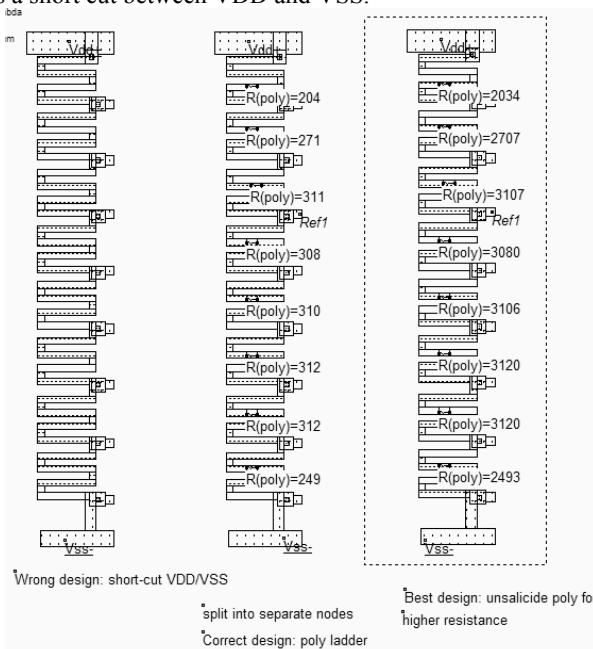


Figure 13-8. Virtual resistor symbols split the polysilicon path into separate electrical regions (ADCRes.MSK)

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 (Figure 13-8 middle and right). The virtual resistor may be found in the upper part of the palette.

Once inserted, the menu of figure 13-9 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.

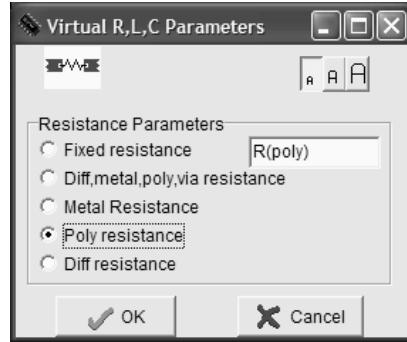


Figure 13-9: Adding a virtual resistor symbol to extract the polysilicon resistance

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 13-10. 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  $50\mu A$  on a 1.2V supply power. Lower DC currents may be obtained by increasing the length of the polysilicon path.

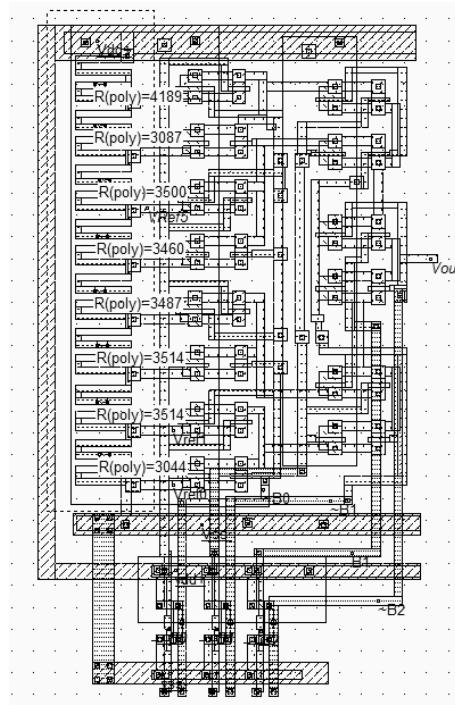


Figure 13-10. Layout of the digital-analog converter (DAC.MSK).

The simulation of the R ladder DAC (Figure 13-11) 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.

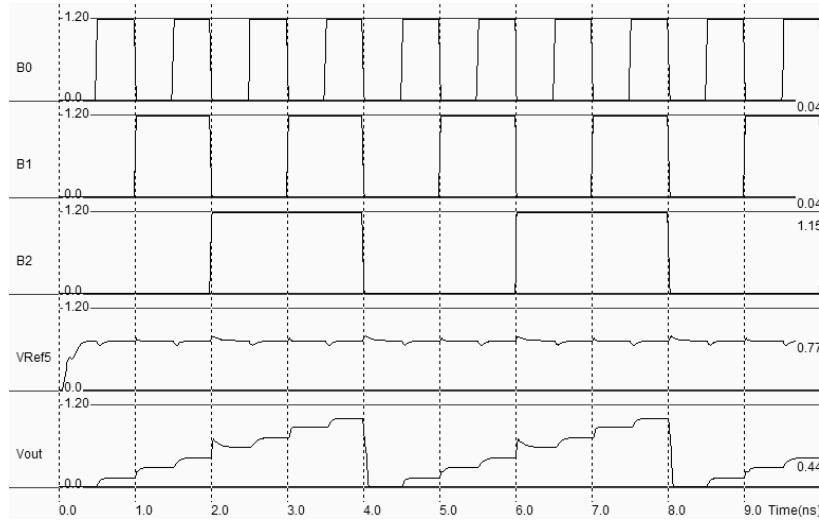


Figure 13-11. Simulation of the digital-analog converter (DAC.MSK).

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  $t=2.0\text{ns}$ , the internal node discharge leads to a voltage overshoot close to one voltage step  $\Delta V$ . Also notice that, according to the schematic diagram of figure 13-4, the output is connected to N switches *On* and N switches *Off*.

### Converter non-linearity

Due to the non-ideal behavior of switches, process fluctuations, leakages and various gradient effects, there is a small difference between the ideal analog output  $V_{out\_ideal}$  and the actual analog output  $V_{out}$ . The deviation of  $V_{out}$  from the ideal value  $V_{out\_ideal}$  is called the integral non-linearity (INL). The normalized integral non-linearity, according to [Gustavsson], can be expressed using equation 13-2. The integral non-linearity is illustrated in figure 13-12: when  $V_{out}$  is exactly equal to the ideal output, the integral non-linearity INL is equal to 0. However, for several values of  $B$ , a small difference is usually observed.

$$INL_i = \frac{V_{out_i} - V_{out_{ideal}}}{\Delta V} \quad (\text{Eq. 13-2})$$

where

$INL_i$ =the integral non-linearity for input i (Relative error between -1 and 1)

$V_{out\_i}$ =the real DAC output for input i (V)

$V_{out\_ideal}$ =the ideal DAC output for input i (V)

$\Delta V$ =ideal voltage step (V)

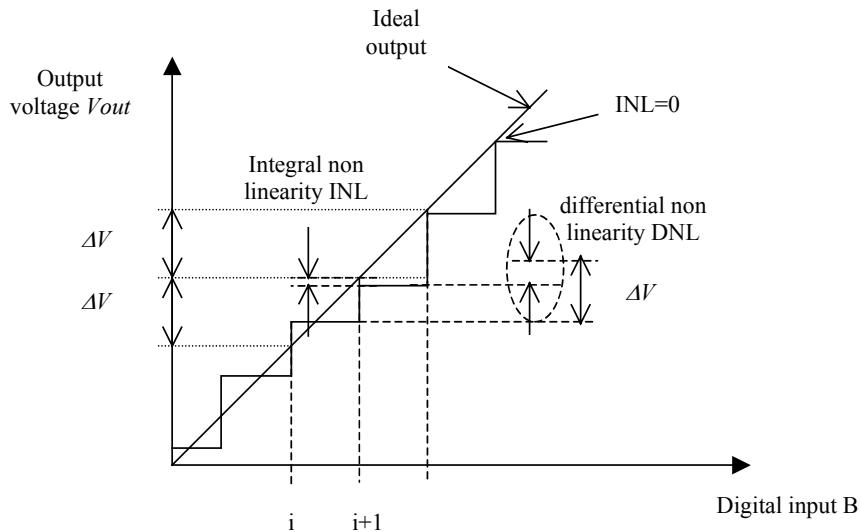


Figure 13-12. The illustration of integrate and differential non-linearity

The difference between two adjacent analog outputs may be significantly different from the theoretical voltage step. This deviation is called the differential non-linearity (DNL). In figure 13-1, the DNL is the vertical mismatch between the ideal voltage step  $\Delta V$  and the measured step between input  $i$  and input  $i+1$ . The normalized differential non-linearity includes the voltage step  $\Delta V$  to get the relative error, and can be described by equation 13-3.

$$DNL_i = \frac{V_{out_{i+1}} - V_{out_i} - \Delta V}{\Delta V} \quad (\text{Eq. 13-3})$$

where

$DNL_i$ =the differential non-linearity for input  $i$  (Relative error, usually between -1 and 1)

$V_{out_{i+1}}$ =the real DAC output for input  $i+1$  (V)

$V_{out_i}$ =the real DAC output for input  $i$  (V)

$\Delta V$ =ideal voltage step (V)

The illustration of integral non-linearity is given in the simulation of the 3-bit DAC. The ideal reference voltages are placed separately in the layout, as shown in figure 13-13. In the simulation mode **Voltage, Current vs. Time**, all voltage values are placed in the same window. The ladder of reference voltages appears, as well as the DAC output  $V_{out}$ . From the simulation shown in figure 13-14, it appears clearly that an integral non-linearity exists which corresponds to the difference between the ideal and actual value of the output, for some values of  $B$ . The origin of this non-linearity is the resistor ladder design which does not create perfectly regular resistance values. Remember that process fluctuation may affect the value of the resistance, which is one other source of non-linearity.

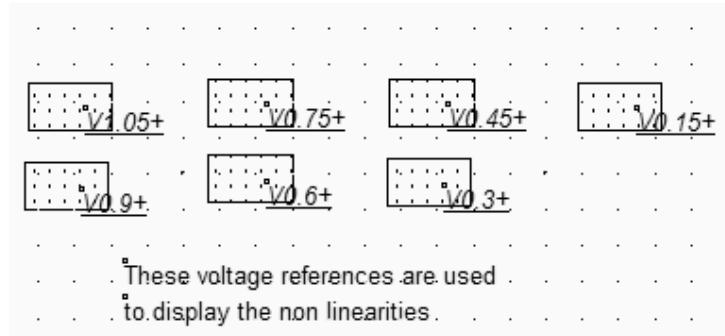


Figure 13-13. The illustration of integral and differential non-linearity (DacNonLinearity.MSK)

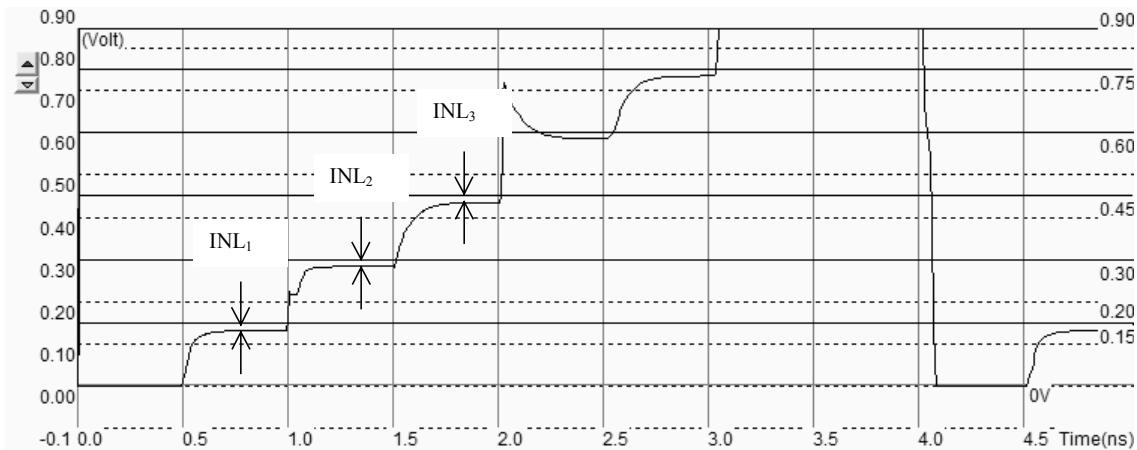


Figure 13-14. A zoom at the analog voltage  $V_{out}$  reveals a non negligible integral non linearity (DacNonLinearity.MSK)

### 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 13-15. At the right end of the network is the output voltage  $V_{out}$ .

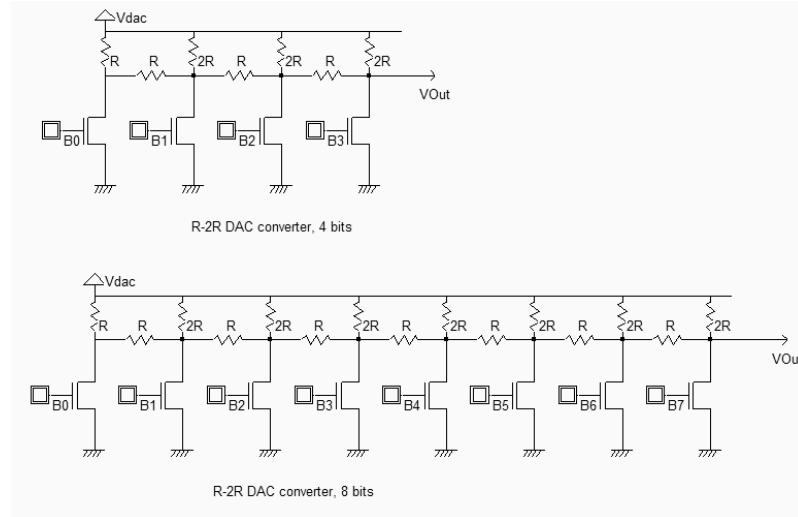


Figure 13-15. 4-bit and 8-bit DAC converter using the R-2R ladder (DACP2R.SCH)

Seven resistors were used for the 4-bit implementation of the R-2R DAC, that is half of the previous R-ladder. The difference is even more significant in the 8-bit circuit, with only 15 resistors, while the simple ladder would require 255 resistors in series.

In the 4-bit implementation of the DAC, the digital inputs ( $B_3, B_2, B_1, B_0$ ) determine whether each cell is switched to ground or tied to  $V_{dac}$ . Each cell's output voltage is a ratio of  $V_{dac}$  because of the ladder network voltage division. The final output voltage  $V_{OUT}$  depends on the value of  $B$  (0 to 15), following the given formula :

$$V_{OUT} = V_{dac} \cdot \frac{(2^N - B)}{2^N} \quad (\text{Eq. 13-4})$$

On this principle, table 13-1 gives the value of  $V_{OUT}$  versus the input code, with  $V_{dac}$  equal to 1.2V.

$B_3$	$B_2$	$B_1$	$B_0$	$V_{OUT}$
0	0	0	0	1.2
0	0	0	1	1.125
0	0	1	0	1.05
0	0	1	1	0.975
0	1	0	0	0.9
0	1	0	1	0.825
0	1	1	0	0.75
0	1	1	1	0.675
1	0	0	0	0.6
1	0	0	1	0.525
1	0	1	0	0.45
1	0	1	1	0.375
1	1	0	0	0.3
1	1	0	1	0.225
1	1	1	0	0.15
1	1	1	1	0.075

Table 13-1. Output voltage produced by the 4-bit R-2R DAC versus input code B

### Layout considerations:

The resolution of the R-2R DAC is linked with the accuracy of the resistors and of the resistance of the switches which must be negligible to avoid a voltage drop and associated non-linearity. It is important to implement a low  $R_{on}$  switch (Large width, minimum length), together with large resistors. In figure 13-16, the design of a four-bit digital-to-analog converter is reported. The elementary resistor pattern has a fixed value of 500 ohm.

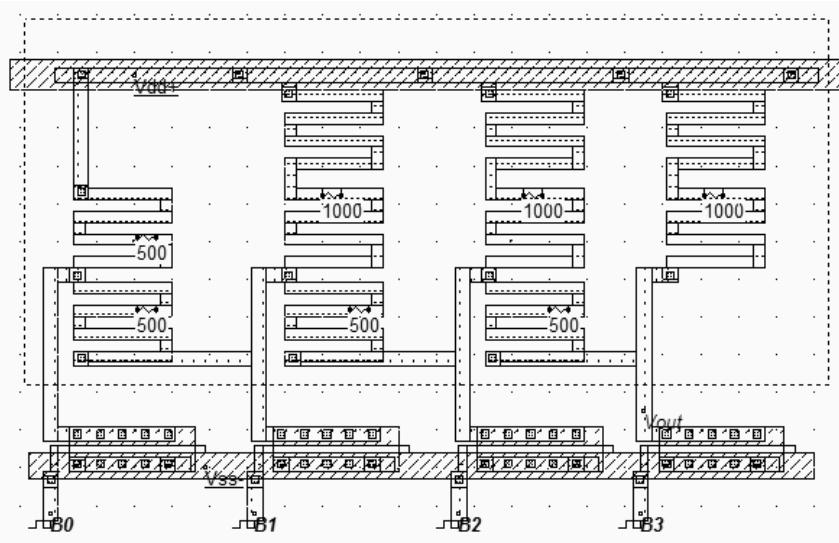


Figure 13-16. A four-bit 2-2R digital to analog converter (DacR2R4Bit.MSK)

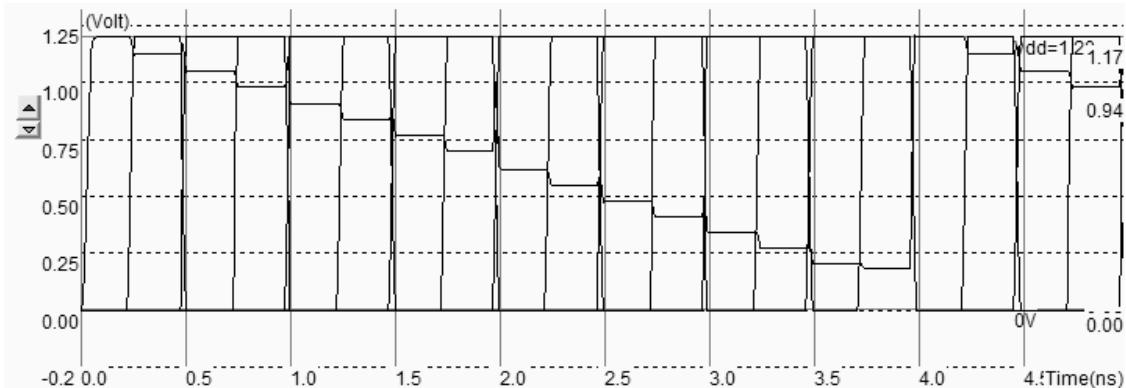


Figure 13-17. Simulation of the 4-bit 2-2R digital to analog converter (DacR2R4Bit.MSK)

The simulation of the four bit 2-2R digital to analog converter (Figure 13-17) shows a regular decrease of the output voltage  $V_{out}$ . As the  $R_{on}$  of the MOS devices is not negligible, the final value  $V_{out}$  ( $B=1111$ ) is higher than the predicted 0.075V (Table 13-1). This non-linearity may be corrected by enlarging the MOS switch and increasing the length of the serpentine resistor. Alternatively, a dummy switch, whose pass resistance is half  $R_{on}$ , may be inserted inside each cell in serial with R.

### Switched capacitors

A very popular DAC architecture used in CMOS technology is based on switched capacitor [Baker]. An array of capacitors is connected to switches, in parallel, as described in figure 13-18. The capacitors are connected in parallel and share one common electrode which is connected to a follower amplifier. Notice that the capacitors are binary weighted, which means that  $C$ ,  $2C$ ,  $4C$  capacitances are implemented. The capacitor array totals  $2^N C$ . Figure 13-18 gives an example of 3-bit ( $B_0, B_1, B_2$ ) charge scaling DAC.

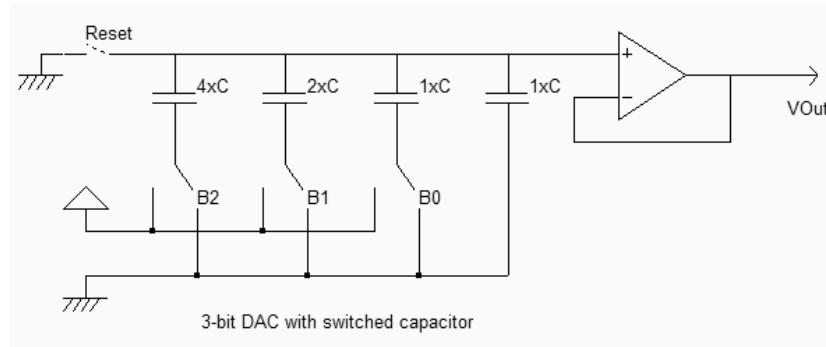


Figure 13-18. The charge-scaling digital to analog converter (DacCapacitor.SCH)

The first step is to discharge all capacitors thanks to the *Reset* switch connected to the ground. Then the switch is disconnected. After the initialization, the digital switches  $B_0, B_1$  and  $B_2$  connect each capacitor to VDD or to VSS, according to the logic value. The output voltage  $V_{out}$  is then a function of the voltage division between capacitors. As an example, if the number to convert  $B=011$ ,  $B_2$  is connected to VSS,  $B_1$  and  $B_0$  are connected to VDD as shown in figure 13-19. The equivalent capacitor divider corresponds to a value of the output equal to  $5/8V_{dd}$ . The conversion table gives the value of  $V_{out}$  versus the input code.

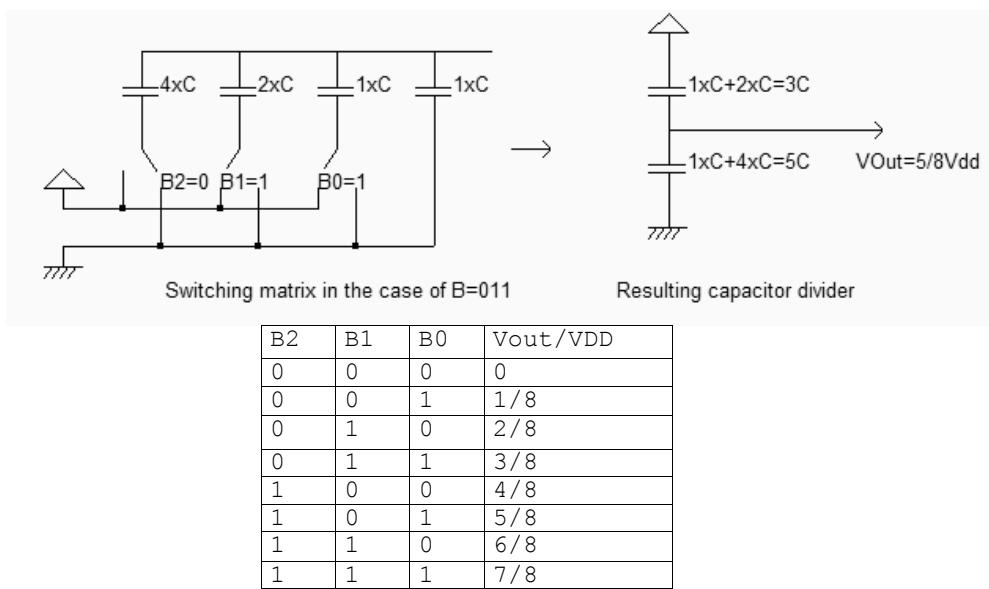


Figure 13-19. The digital to analog converter at work for  $B=011$  (DacCapacitor.SCH)

### Layout considerations

The most important problem is the design of precise capacitors, with values from  $C$  to  $2^N \times C$ . As the number of bits increases, the ratio of MSB to LSB capacitor becomes difficult to control. Moreover, high value capacitors have an important size on the chip. Using metal plates to create them is not realistic as the capacitance value per  $\mu\text{m}^2$  is very low. The solution is to use passive double polysilicon capacitors if available in the CMOS process, as they have good matching accuracy and high capacitance value per  $\mu\text{m}^2$ . In  $0.12\mu\text{m}$  CMOS technology, the capacitance between metals is around  $50 \text{ aF}/\mu\text{m}^2$  and rises to  $2000 \text{ aF}/\mu\text{m}^2$  between polysilicon and poly2.

The implementation of the 3-bit DAC requires 8 sets of capacitor, regrouped in  $4 \times C$ ,  $2 \times C$  and two separate  $C$ . In Microwind, the command **Edit → Generate → Capacitor** gives access to a specific menu for generating capacitor (Figure 13-20). The default capacitor is made with poly/Poly2. The typical capacitance value for  $C$  is around  $1\text{pF}$ . As may be seen in the layout shown in figure 13-21, a  $100\text{fF}$  value for  $C$  already leads to a large layout.

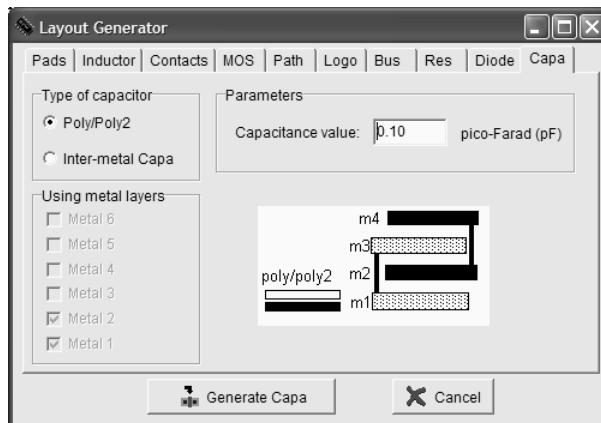


Figure 13-20: The generator menu handles the design of poly/poly2 capacitor and inter-metal capacitors

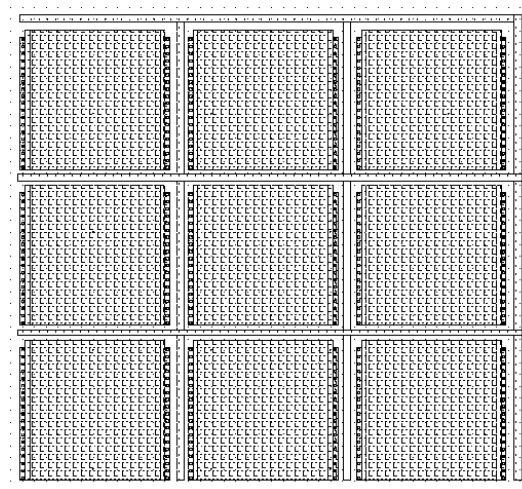


Figure 13-21. Implementation of an array of  $100\text{fF}$  capacitor for the 3-bit DAC (DacCapacitor.MSK)

### 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 13-22).

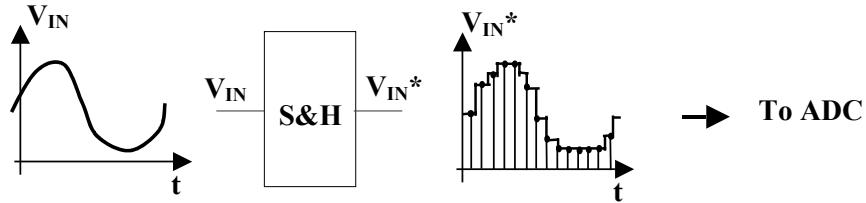


Figure 13-22. The sample-and-hold circuit

The operation is repeated in time with a regular sampling period. We can notice that during the sampling period, the S/H circuit operates alternatively in dynamic mode (sample) and in static mode (hold), as shown figure 13-23. In dynamic mode, the switch lets the input signal flow through the pass transistor and settle  $V_{in}^*$  within the required accuracy. 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.

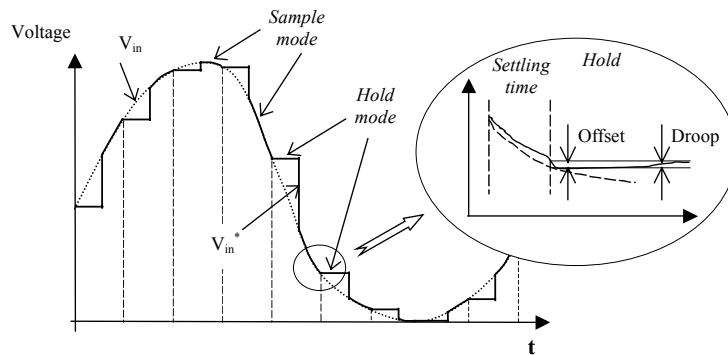


Figure 13-23. Sampling of an analog voltage (sample and hold modes)

The transmission gate can be used as a sample and hold circuit. The schematic diagram of the sample/hold circuit is proposed in figure 13-24. 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.

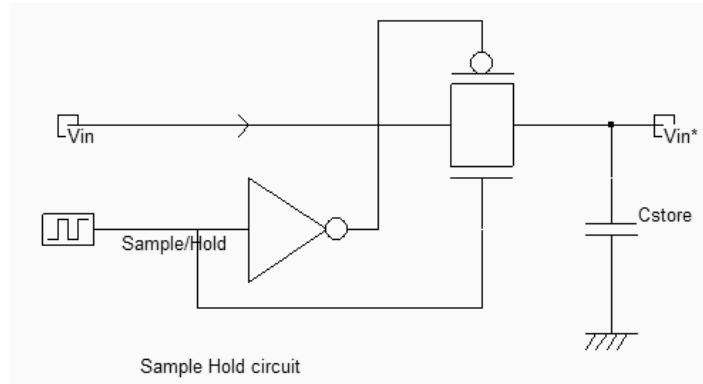


Figure 13-24. Schematic diagram of the Sample-Hold circuit (SampleHold.SCH)

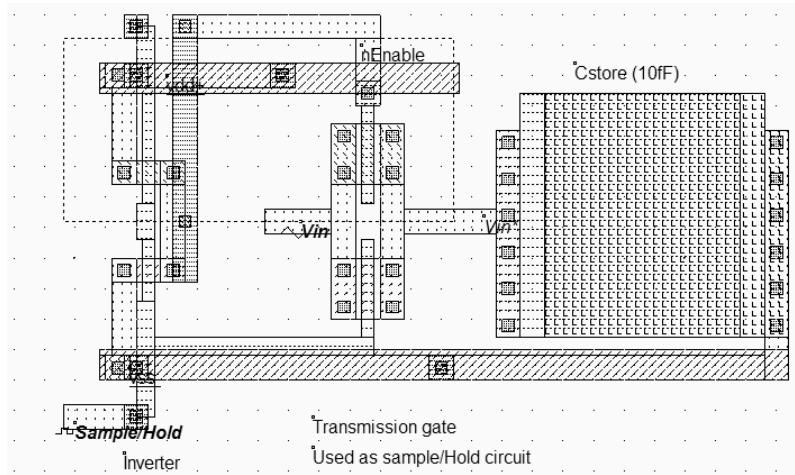


Figure 13-25. The transmission gate used to sample analog signals (SampleHold.MSK)

The layout of the transmission gate is reported figure 13-25. The *sample/hold* command is situated on the left, and controls the transmission gate. The inverter is required for the pMOS device. The  $V_{in^*}$  signal is connected to a 10fF capacitor made of poly/Poly2. The effect of sample and hold is illustrated in figure 13-26. 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  $V_{in^*}$  reaches the value of the sinusoidal wave  $V_{in}$ .

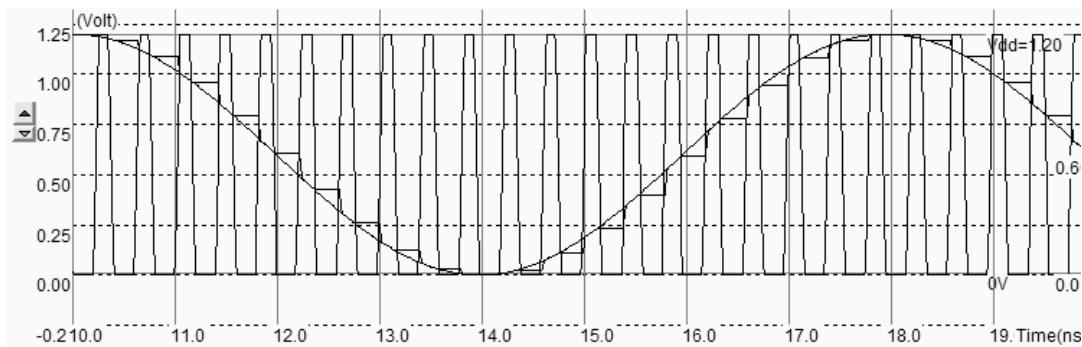


Figure 13-26. Effect of sampling (SampleHold.MSK)

When the gate is off, the value of the sampled data  $V_{in^*}$  remains constant. This is mainly due to the parasitic capacitance of the node which has a value of  $10fF$  as extracted in a CMOS  $0.12\mu m$  process (figure 13-27). This is sufficient to retain the information for several nanoseconds, even in the presence of leakage currents. Higher values of storage capacitor are required if the duration of the analog to digital conversion is of the order of the  $\mu$ -second, with a large voltage precision. In all cases, the sampled voltage must not fluctuate more than by 30% of the least significant bit over the whole sampling cycle.

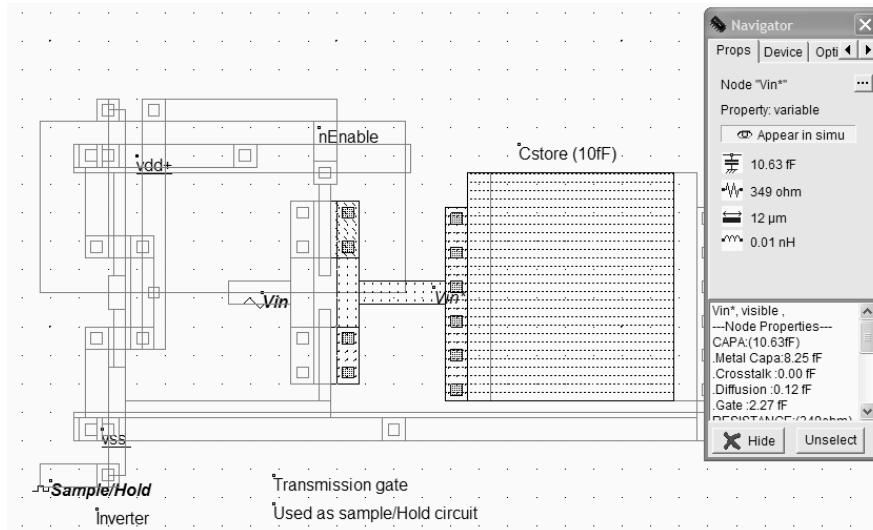


Figure 13-27. The hold effect is related to the parasitic capacitance of the node  $V_{in^*}$

### Layout considerations

If the width and length of the nMOS and pMOS devices are not identical, an error appears between the sampled  $V_{in^*}$  and the original voltage  $V_{in}$ . This voltage difference is a parasitic offset, which depends on the value of  $V_{in}$  in a non-linear way, as shown in the measured transfer characteristics shown in figure 13-28 (Standard sample circuit). The offset is minimized if the nMOS and pMOS sizes are identical, as it is strongly dependent on the parasitic capacitance between the gate and the drain. With identical channel size,  $C_{GD}$  of the pMOS is equal to  $C_{GD}$  of the nMOS. However, as the size of the nMOS is identical to the pMOS, the pMOS switching is slower.

The sampling circuit can be improved by switching the pMOS *before* the nMOS device, in contrast to the circuit proposed in figure 13-27, and by adding so-called "dummy transistors" on the storage node. The curves shown in figure 13-27 are compiled from measured characteristics on a  $0.18\mu m$  test chip [Bendhia], and exhibit a significant reduction of the offset. We often experienced a negative offset (10-20mV) for  $V_{in} < V_{dd}/2$ , and a positive offset for  $V_{in} > V_{dd}/2$ . Unfortunately, Microwind do not modelize these characteristics accurately due to a simplification of coupling capacitance in MOS device models.

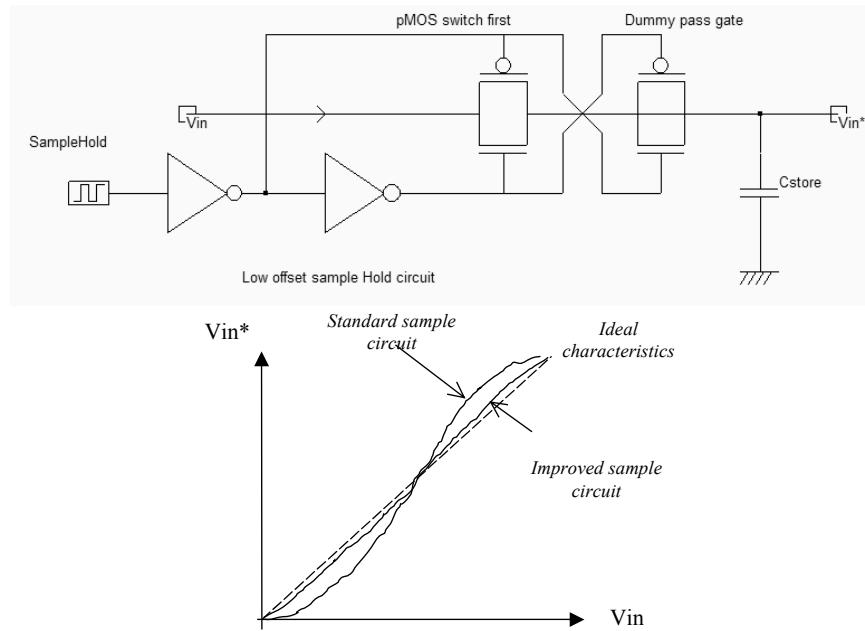


Figure 13-28. Offset reduction techniques (SampleHold.SCH)

### Shannon's Sampling Theorem

The critical element when capturing the analog input voltage accurately is the number of sampled data in the considered time window. We can also talk of the sampling frequency compared to the input voltage frequency. Shannon's sampling theorem gives the minimum frequency required to represent the analog input voltage accurately. The minimum sampling frequency  $f_{sample}$  must be greater than twice the highest frequency component of the original signal  $f_{signal}$ .

$$f_{sample} > 2f_{signal} \quad (\text{Eq. 13-5})$$

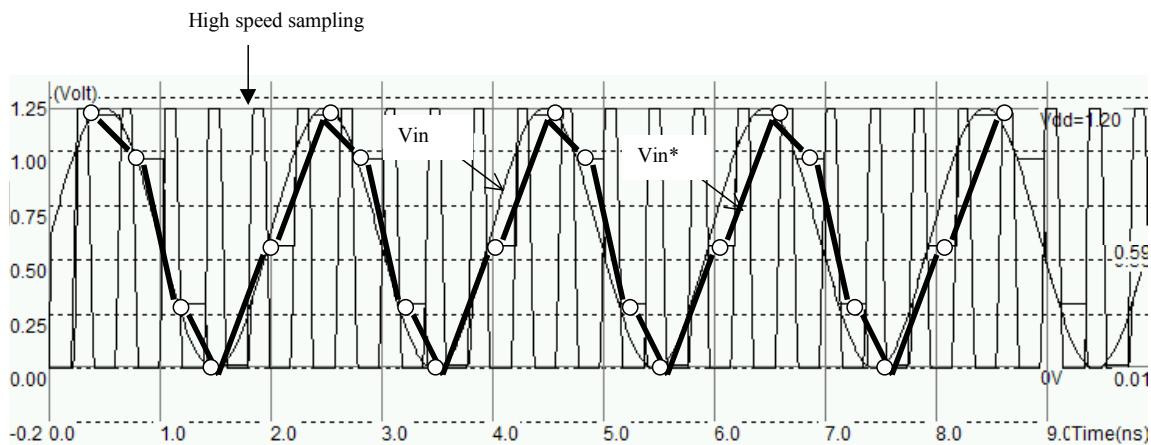


Figure 13-28: The sampling frequency is fast enough to comply with Shannon's theorem (SampleHoldShannon.MSK)

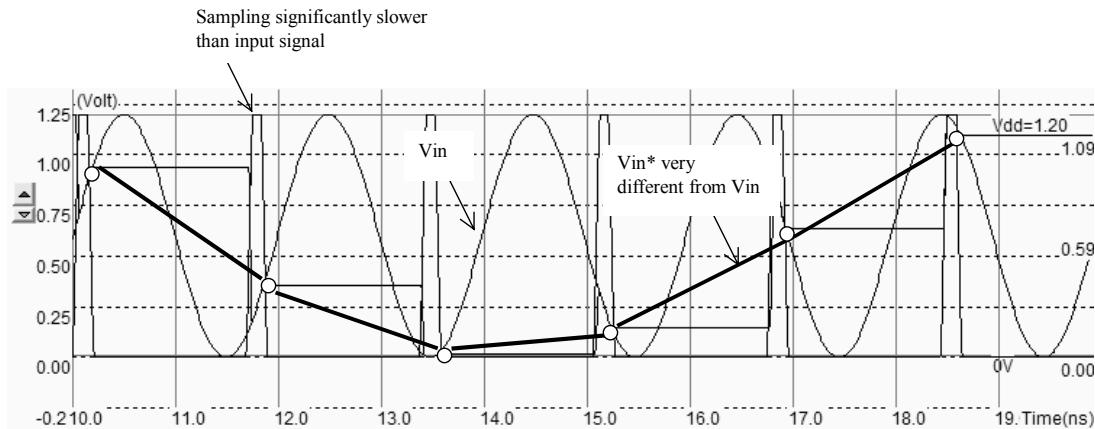


Figure 13-29. The sampling frequency is too slow.  $V_{in}^*$  differs from  $V_{in}$  (SampleHoldShannon.MSK)

Figure 13-28 shows the sampling of a 500MHz sinusoidal input wave ( $f_{signal}$ ) with a sampling frequency  $f_{sample}$  of 2.5GHz which complies largely with the Shannon's Theorem. In figure 13-29 the sampling frequency  $f_{sample}$  is too low (600MHz), consequently the sampled output  $V_{in}^*$  is significantly different from  $V_{in}$ .

## 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. Figure 13-29 shows the complete chain from the analog signal to the digital data using a sampled and hold module and a 4-bit ADC. ADCs can be implemented by employing a variety of architectures. In the following chapters, we describe the Flash converter and successive approach converters.

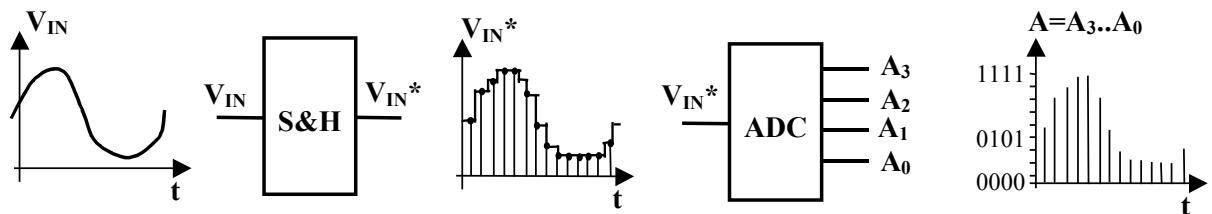


Figure 13-29. A 4 bits digital conversion of a sampled analog voltage

### 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 13-30). The flash converters are widely used for very high sampling rates, at the cost of very important power dissipation.

Analog Input Vin	C2	C1	C0	A1	A0
Vin<Vref0	0	0	0	0	0
Vref0<Vin<Vref1	0	0	1	0	1
Vref1<Vin<Vref2	0	1	1	1	0
Vin>Vref2	1	1	1	1	1

Table 13-3. The specifications for a 2-bit flash ADC converter

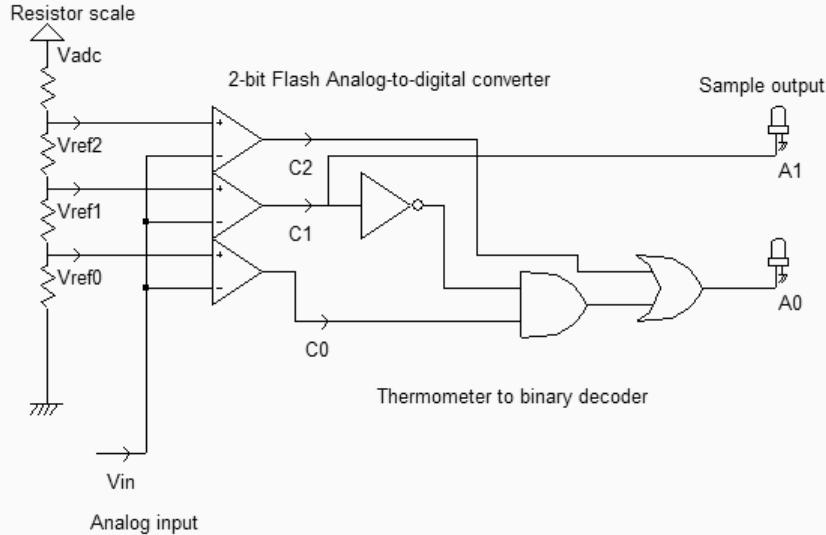


Figure 13-30. The schematic diagram of the 2-bit flash ADC converter (AdcFlash2bits.SCH)

A schematic diagram for the 2-bit flash converter is proposed in figure 13-30. The resistor scale produces 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.

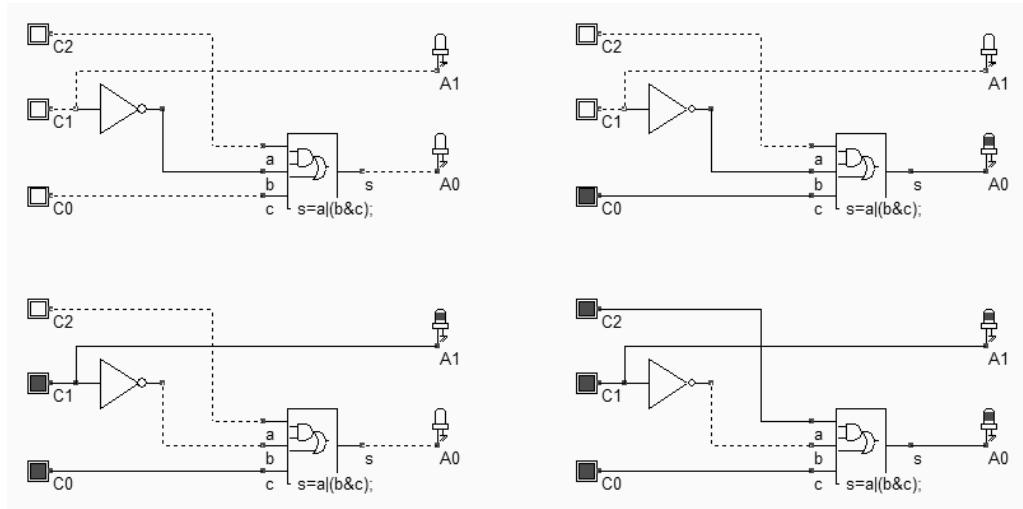


Figure 13-31. The thermometer to binary coder (AdcFlash2bits\_coder.SCH)

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 13-3. 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 13-31), and can be efficiently implemented using one inverter ( $A_1$ ) and a complex gate ( $A_0$ ). For a 3-bit flash converter, the logic circuit starts to rise in complexity. The thermometer code is described in table 13-4.

Analog Input $V_{in}$	$C_6$	$C_5$	$C_4$	$C_3$	$C_2$	$C_1$	$C_0$	$A_2$	$A_1$	$A_0$
$V_{in} < V_{ref0}$	0	0	0	0	0	0	0	0	0	0
$V_{ref0} < V_{in} < V_{ref1}$	0	0	0	0	0	0	1	0	0	1
$V_{ref1} < V_{in} < V_{ref2}$	0	0	0	0	0	1	1	0	1	0
$V_{ref2} < V_{in} < V_{ref3}$	0	0	0	0	1	1	1	0	1	1
$V_{ref3} < V_{in} < V_{ref4}$	0	0	0	1	1	1	1	1	0	0
$V_{ref4} < V_{in} < V_{ref5}$	0	0	1	1	1	1	1	1	0	1
$V_{ref5} < V_{in} < V_{ref6}$	0	1	1	1	1	1	1	1	1	0
$V_{in} > V_{ref6}$	1	1	1	1	1	1	1	1	1	1

Table 13-4. The specifications for a 3-bit flash ADC converter

A 3 bit flash converter requires 7 converters and a complex logic circuit which converts the thermometer code into a binary code, as specified in table 13-4. The 8-bit flash converter would require 255 comparators and a very complex logic decoder. An interesting approach for the encoding consists in using a small memory array, taking into account the specific condition of the thermometer coder. For example, a 0 on  $C_4$  and a 1 on  $C_3$  means that  $V_{ref3} < V_{in} < V_{ref4}$ , which is a sufficient condition to produce  $A=100$  (4) at the output. Using this principle, the implementation of the 3-bit coder is detailed in figure 13-32.

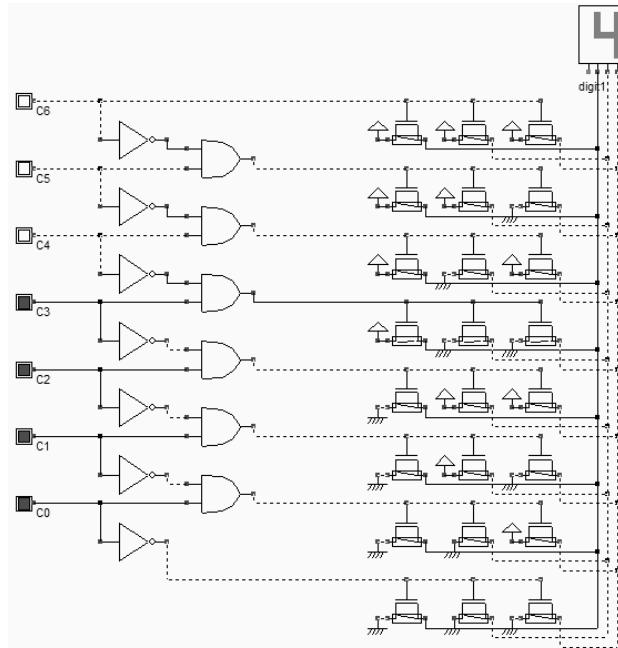


Figure 13-32. The thermometer to 3-bit binary coder using a logic array (AdcFlash3bits\_coder.SCH)

### Flash converter implementation

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.

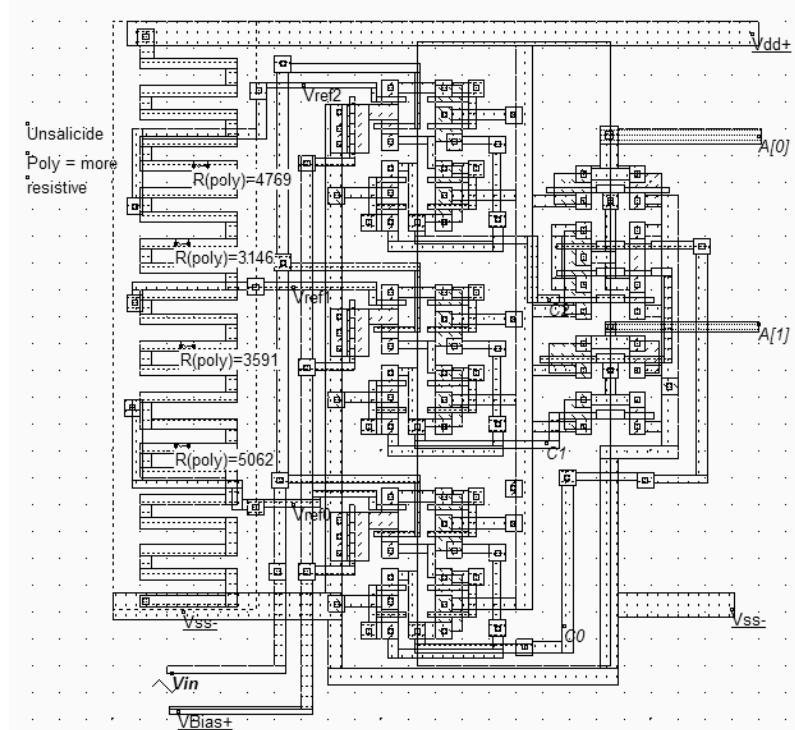


Figure 13-34. Design of the analog-digital converter (ADC.MSK).

In the simulation shown in Figure 13-34, 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 reduced the input voltage range, which means that the resistance scale would be supplied by  $V_{dac-}$  larger than VSS and  $V_{dac+}$  smaller than VDD.

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.

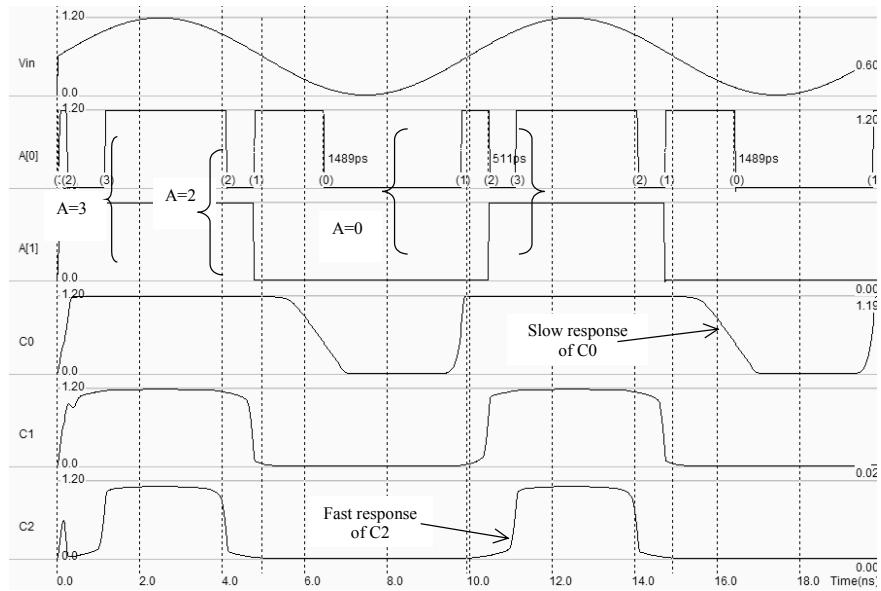


Figure 13-34. Simulation of the analog-digital converter (ADC.MSK).

### Low speed ADC Converters

The most common low speed converter is the iterative converter. As shown in figure 13-35, 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  $V_{in}$ .

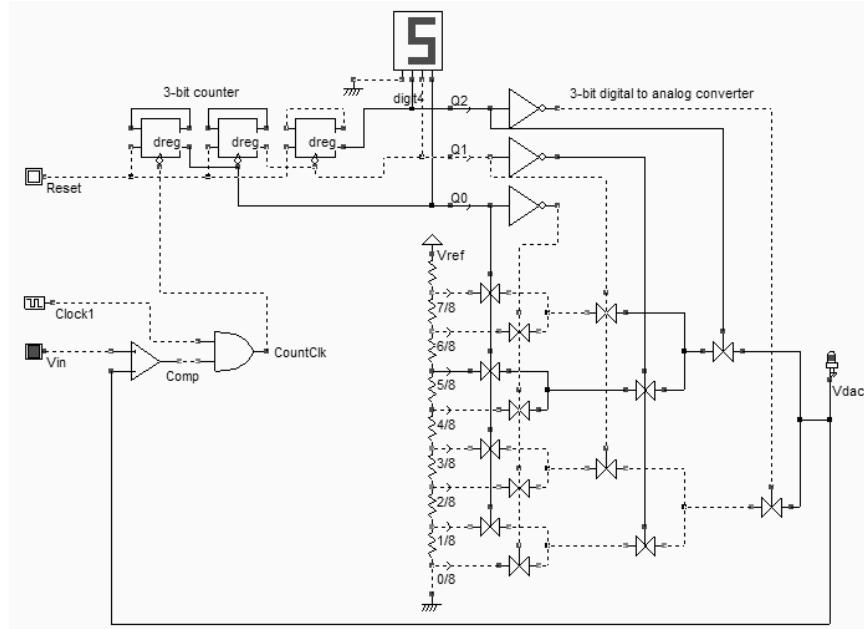


Figure 13-35. Iterative converter using a DAC (ADCIterative.SCH)

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 100MHz clock frequency, the conversion rate is as low as 750 Hz. The implementation of figure 13-35 corresponds to a 3-bit converter. With a high resolution DAC and a high precision amplifier, high resolution ADC converters may be constructed.

A better solution consists in examining the most significant bit  $a_{n-1}$  first and then in determining whether  $V_{in}$  is larger or smaller than  $VDD/2$ . The comparator gives the value of that bit directly. Then, the comparison is performed for the next bit, and so on until all bits are extracted, finishing by the least significant bit  $a_0$ . This type of converter is called successive approximation converter. The complete process is faster than the iterative converter as only N comparisons are necessary. The algorithm is detailed in figure 13-36.

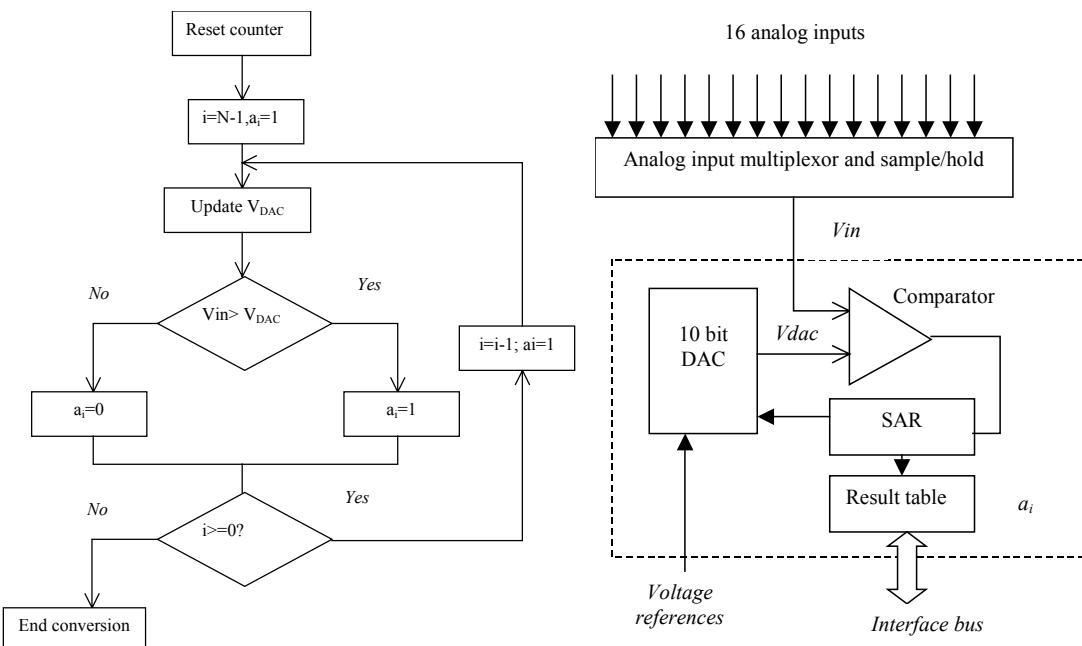


Figure 13-36. Iterative converter algorithm and typical implementation for a 10bit converter (Motorola MPC555)

The schematic diagram of a successive approach converter is given in Figure 13-36. The analog input signal  $V_{in}$  is retained by a sample/hold circuit during the data conversion process. In the first clock cycle, the most significant bit  $a_i$  of the successive approximation register (SAR) is set to 1. The DAC converts the SAR value to an analog voltage  $V_{dac}$  that is compared to  $V_{in}$ . If  $V_{dac} < V_{in}$ , the bit  $a_i$  is validated at 1, and the SAR register is unchanged. Conversely, if  $V_{dac} > V_{in}$ ,  $a_i$  is set to 0. The DAC generation and comparisons processes are repeated for N clock cycles to complete the conversion.

In many industrial 32-bit micro-controller, 8-bit to 16-bit successive approach converters are implemented with 4 to 16 channel multiplexors. In the particular case of a 10-bit converter supplied by a voltage reference  $V_{ref}$ , as much as  $2^{10}$  reference values (1024) are used. An example of conversion is given in figure 13-37. For the first bit,  $V_{dac}$  is set to  $V_{ref}/2$ . As  $V_{dac} > V_{in}$ ,  $a_9$  is set to 0. Then  $V_{dac}$  is set to  $V_{ref}/4$ . As  $V_{dac} < V_{in}$ ,  $a_8$  is set to 1.

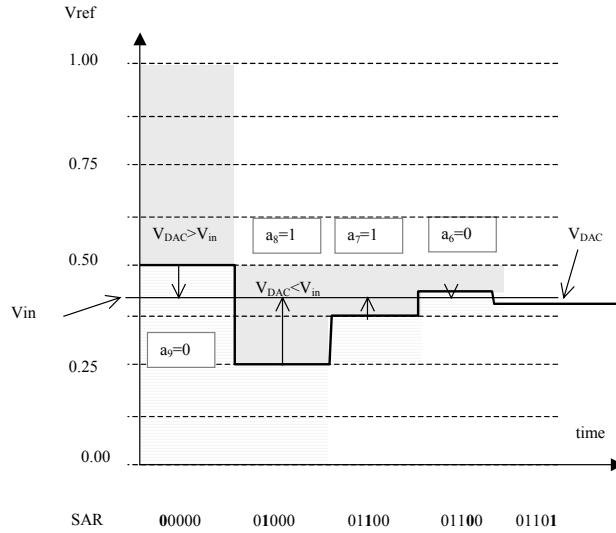


Figure 13-37 The successive approach converter at work

### Pipeline ADC Converters

The pipelined Analog-to-digital converter consists of two or more stages connected in serial, each containing a low resolution ADC and DAC converter. The schematic diagram of figure 13-35 corresponds to a 4-bit ADC, based on a 2-bit flash ADC. The role of the first stage is to generate the most significant bits  $A3$  and  $A2$ . Then the difference between  $V_{dac}$  and  $V_{in}$  is computed. In order to convert the residue, the voltage difference is amplified and sent to a second 2-bit DAC which calculates the least significant bits  $A1$  and  $A0$ . The pipelining approach is very powerful and may be applied for a large number of stages, which permits to convert analog signals with a high resolution without the need of designing high resolutions DAC or ADC blocks. An extensive study of pipeline ADC may be found in [Gustavsson].

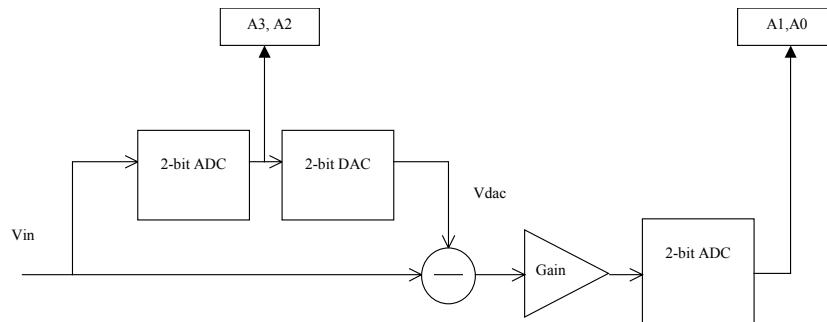


Figure 13-37 Pipeline ADC converter

The successive approach or pipeline converter layout area increase almost linearly with the converter resolution N, as well as with the power consumption, which represents a key advantage on flash converters.

## 5. Temperature Sensor

One of the simplest temperature sensing element is the *pn* diode [Ristic p 289]. The classical model of the diode is given by equation 13-7. The expression of the current includes two strongly temperature-dependent parameters, the exponential term and the reverse saturation current.

$$I_{ak} = I_{sat} S \left( \exp \left[ \frac{q}{kT} V_{ak} \right] - 1 \right) \quad (\text{Eq. 13-7})$$

with

$I_{sat}$ = reverse saturation current per  $\mu\text{m}^2$  ( $\text{A}/\mu\text{m}^2$ )

S=surface of the diode ( $\mu\text{m}^2$ )

Q=electric charge

K=Boltzmann's constant

T=absolute temperature ( $^\circ\text{K}$ )

$V_{ak}$ = diode voltage (V)

For temperature sensing, the *np* diode is forward biased by a small constant current, and the diode voltage  $V_{ak}$  serves as a measure of the temperature. The proposed circuit is given in figure 13-35. The pMOS device serves as a load while the P+/Nwell diode serves as a temperature sensor. The pMOS device itself is sensitive to temperature, but its dependence is negligible as compared to the diode. In this circuit,  $V_{ak}$  is equal to  $V_{ref}$ .

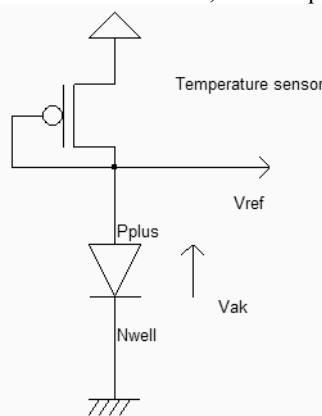


Figure 13-35. Principles for a temperature sensor based on a junction diode (SensorTemperature.SCH)

The implementation of the diode in forward biasing condition cannot be done with a  $N+/Psusbtrate$  diode, as the substrate is connected to ground, which would imply a negative biasing of the  $N+$  diffusion. The only remaining solution consists in using an *nwell* region connected to ground, and a  $P+$  diffusion, which creates a  $P+/Nwell$  diode.

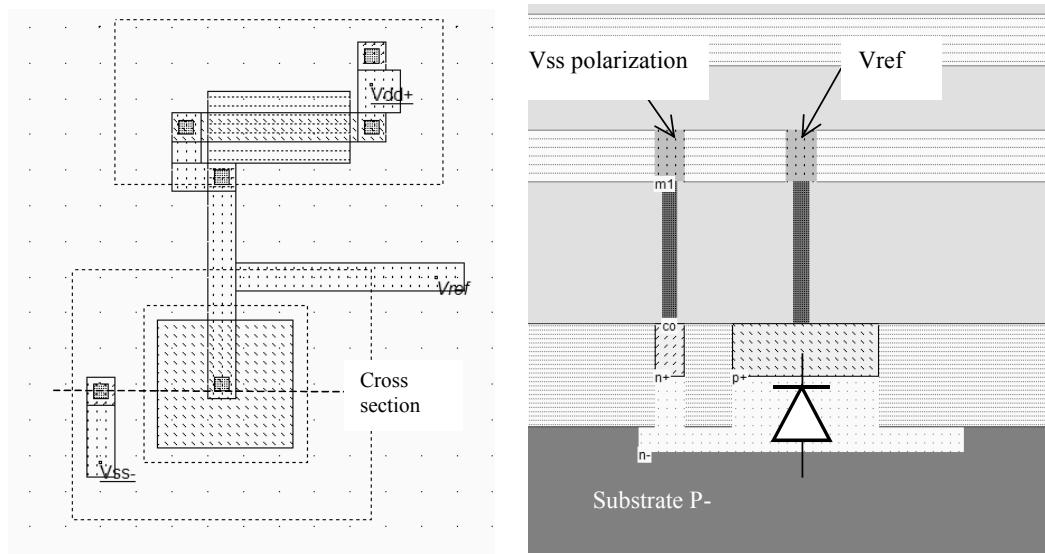


Figure 13-36. Implementing the temperature sensor (SensorTemperature.MSK)

The implementation of the current sensor is reported in figure 13-36. The pMOS channel length is large so as to reduce the DC current and to avoid short channel limiting effects. The simulation of the temperature influence is performed using the parametric analysis, in order to plot the diode voltage  $V_{ref}$  versus the temperature in °C directly. Invoke the command **Analysis→Parametric Analysis**, click inside the layout corresponding to  $V_{ref}$ , and the following screen appears. Select the item "Temp.", and the measurement "Final voltage  $V_{ref}$ ". Click **Start Analysis** to perform the iterative simulation from -40°C to 120°C with a step of 20°C. At the end of each simulation, the final value of  $V_{ref}$  is added to the data array. It can be seen from the result of figure 13-37 that  $V_{ref}$  decreases nearly linearly with temperature, with a slope of around -1.2mV/°C.

Measured results presented in [Ristic] give around -2.4mV, using a stable current reference of  $10\mu A$  instead of the diode-connected on-chip pMOS device. The main problem of this type of sensor is its very strong dependence on process variation which requires a calibration procedure to obtain an exact value for the temperature.

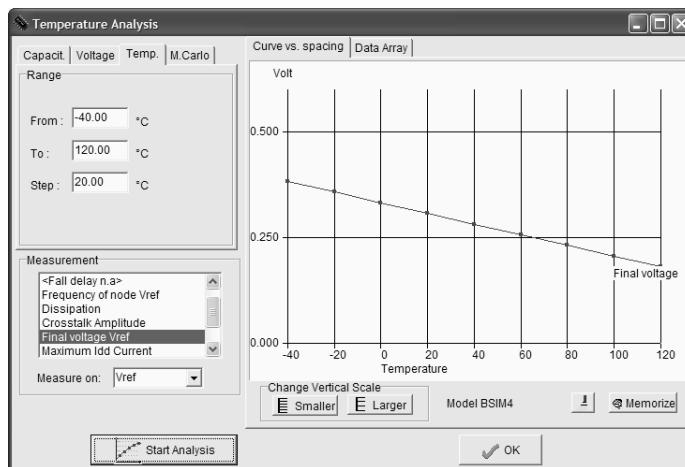


Figure 13-37. Simulation of the voltage dependence with temperature (SensorTemperature.MSK)

## 6. Image Sensors

Recently, new attention has been paid for CMOS image sensors, due to the proliferation of low cost video cameras such as webcams and video on mobile phones. Most high quality cameras use charge-coupled-device <Gloss> image sensors, which feature superior light sensing characteristics over CMOS sensors. However, the use of standard CMOS technology for image sensing offers two key advantages over CCD technology: the analog signal processing can be realized on the same chip, and the price of CMOS imagers is significantly lower than its CCD counterpart, thanks to the availability of deep sub-micron CMOS foundries all over the world.

### The Diode Detector

In CMOS technology, one of the most simple light detection devices is the PN junction. The diode is photo resistive, which means that its characteristics I/V are sensitive to light. The photo diode can be considered as a variable resistor. The most common photo diode consists of an N+ diffusion area in the P-substrate. By default, with no light, the diode is polarized in reverse mode, so that almost no currents flow from the grounded P-substrate to the N+ diffusion region (Figure 13-38). The light photons are converted in the P region into electrons which are attracted by the N+ diffusion and generate a photo current. This current is almost linearly proportional to the incident light. Consequently, the resistance of the diode is linearly decreased with the intensity of the light.

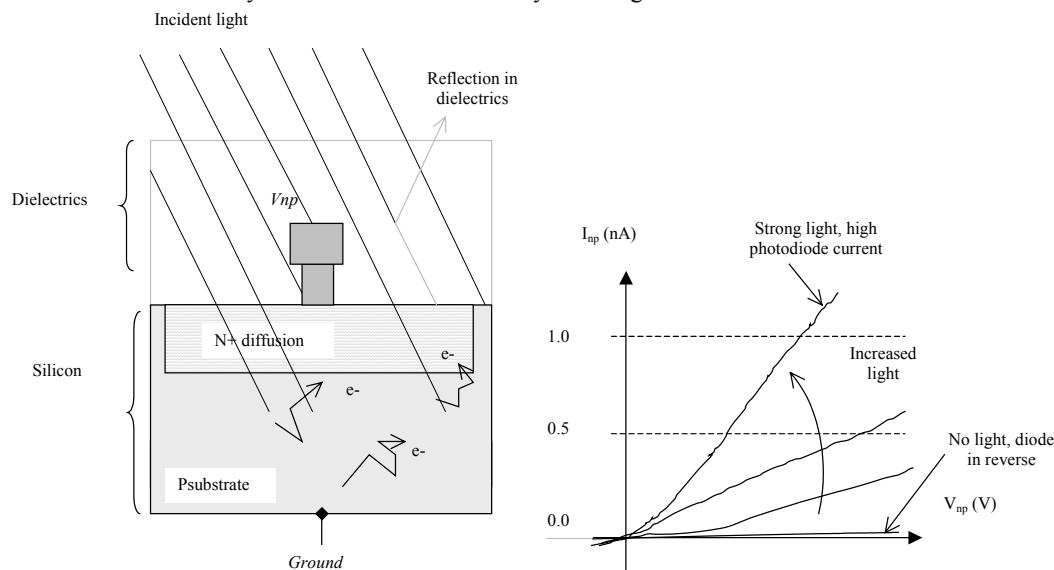


Figure 13-38. The diode as a light detector

Silicon photo resistors are sensitive to the spectral band corresponding to a wavelength from 350 to 1100 nm, which covers the visible spectrum (Figure 13-39). Our eyes are sensitive to a light that corresponds to a wavelength range of 400nm (Blue) to 700nm (Red).

The efficiency in converting photons into electrons is limited by reflection on the dielectric materials which cover the surface of the integrated circuit and by the loss and recombination of electrons during their travel from the substrate to the electrode. The efficiency of the light sensor is maximum near 700nm, with an efficiency around 20%. The current that flows on the photodiode is very small, approximately 1nA for a  $10 \times 10 \mu\text{m}$  diode area.

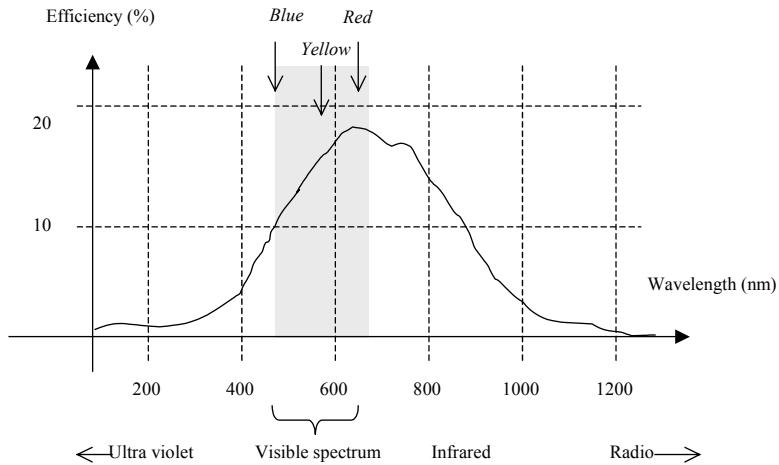


Figure 13-39. Typical performance of a photo diode versus the incident wavelength

Notice that the photoreceptor is subject to an important time delay to reach equilibrium after a change in illumination level. This time constant is of the order of the micro-second.

### Diode Detector Setup

The basic setup for the passive light sensor is shown in figure 13-40. The circuit consists of the diode and a capacitor. First, the capacitor  $C_{store}$  is charged to a high voltage  $Vdd$ , using the precharge switch. Second, the precharge is deactivated and the light sensor starts to discharge the capacitor, thanks to the reverse photo-current  $I_{NP}$ . Without light, the current  $I_{NP}$  is equivalent to the reverse mode current, which is the range of the pico-ampere ( $10^{-12} \text{ A}$ ) for a small PN junction. With a strong light, the current  $I_{NP}$  enters the range of the nano-ampere ( $10^{-9} \text{ A}$ ).

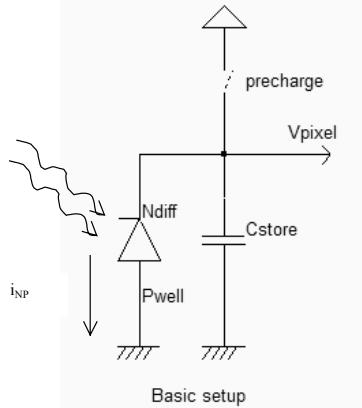


Figure 13-40. Light detector setup including a precharge circuit (ImageSensor.SCH)

The basic diagram for an array of passive light sensors is shown in figure 13-40. Each passive pixel sensor <Gloss> (PPS) converts the photons into an electrical charge which is carried off by the sensor through pass transistors which are laid out as in a memory array. The charges flow through the vertical lines to a voltage amplifier with an important gain. The main problem with the passive sensor is the noise that appears in the resulting image and limits its use to low quality image sensing. Furthermore, a large pixel matrix induces important leakage in the vertical bit lines and limits the use of the passive pixel sensor to very low resolution devices.

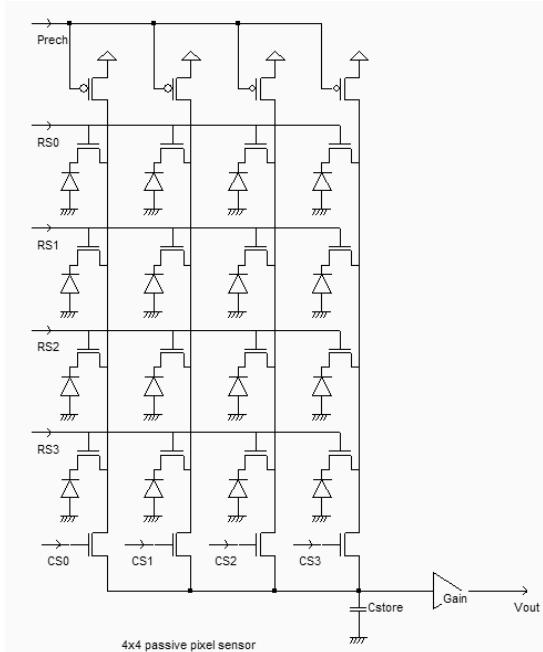


Figure 13-40. A passive pixel array made with the photodiode and pass transistors (ImageSensor.SCH)

The goal of active pixel sensors <Gloss> (APS) is to reduce the noise associated with passive sensors, and to amplify the light-induced charges at each pixel location. The active pixel sensor approach improves the light sensor performances significantly, allowing the design of large pixel arrays, with fast read out access, while keeping power dissipation low. The transistor  $N1$  sets the photodiode voltage  $V_{diode}$  to a high value when  $Set=1$ . Once  $N1$  is cut off, the photodiode discharges  $V_{diode}$ , depending on the light intensity. The source follower  $N2$  buffers the photodiode voltage to the bus, when the row select transistor  $N3$  is active ( $Sel=1$ ).

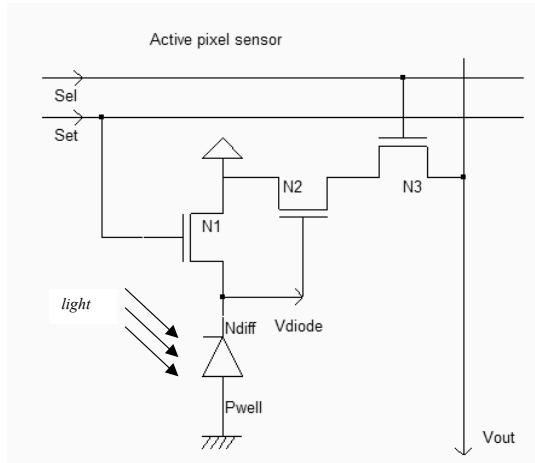


Figure 13-41. The active pixel sensor with a photodiode (ImageSensorActive.SCH)

In Microwind, the photodiode effect is modeled by a virtual resistor added between *Vdiode* and the ground. Values ranging from  $1M\Omega$  to  $9M\Omega$  are added in the  $2 \times 2$  pixel array to account for variable photocurrent. However, the real case photocurrent is very small (Around  $10-100 \text{ pA}/\mu\text{m}^2$  in the best case). When a  $10 \times 10 \mu\text{m}$  pixel is designed, the maximum current is around  $10 \text{ nA}$ , which discharges the pixel capacitance within several microseconds, with an equivalent leakage resistance of hundreds of mega-ohm. Using a mega-ohm resistor speeds up the simulation but accelerates real case chronograms by 3 orders of magnitude.

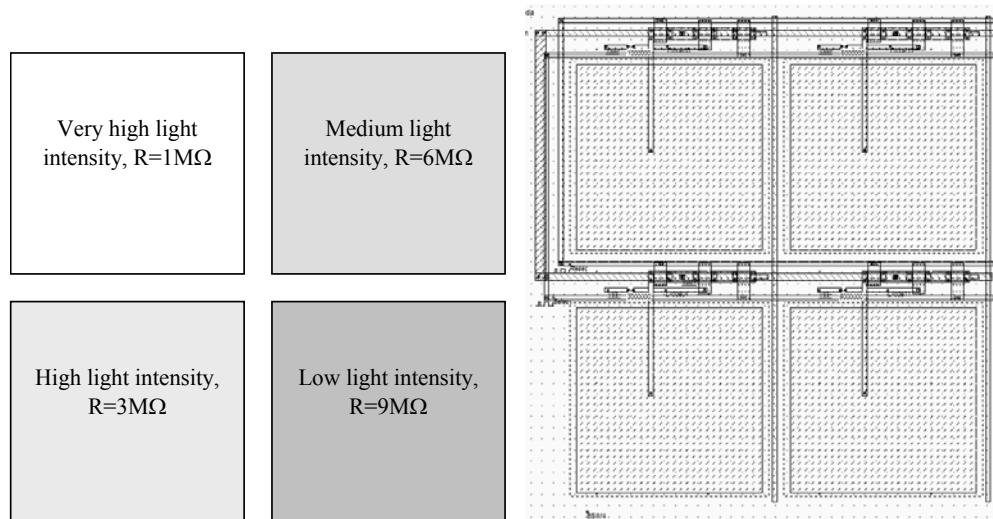


Figure 13-42. A matrix of 4 active pixels (ImageSensor2x2.MSK)

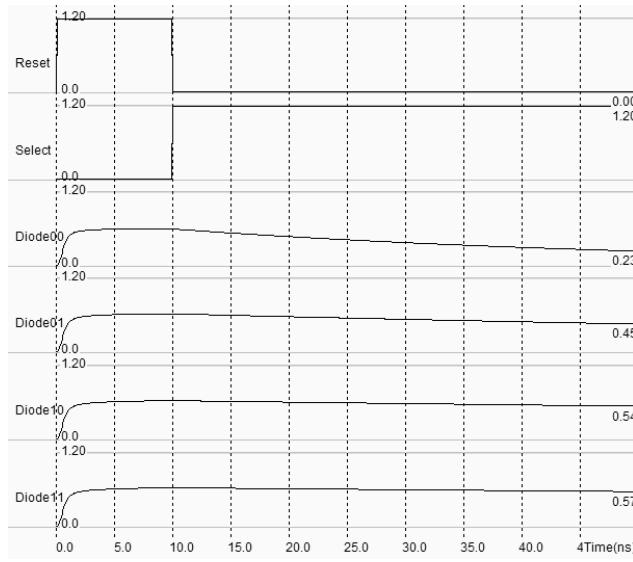


Figure 13-43. Simulation of the active pixels (ImageSensor2x2.MSK)

In the simulation reported in figure 13-43, the role of photocurrents is clearly demonstrated: the diode voltage of each pixel is set to around  $VDD - Vt$ , then each photocurrent discharges the pixel voltage with a slope dependent on the light intensity. The final voltage may be sampled at time 50ns. In real case light sensors, 100 $\mu$ s up to 1ms are required before sampling the pixel voltage and converting it into an image information.

Color filter materials are used to capture selectively the blue, green and red components of the incident image. The transmission of light has a general shape shown in figure 13-40. The blue filter passes electromagnetic waves around a 475nm wavelength, the green filter passes mainly 510nm waves, and the red filter passes 650nm waves. Color filters may be placed mechanically on the top of the pixel array, in order to assign one color to one pixel, according to a regular assignation pattern such as the one shown in figure 13-40.

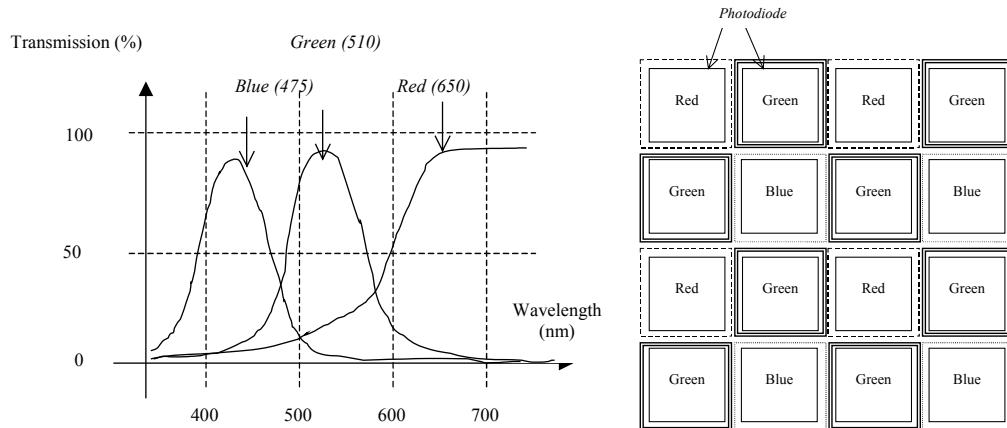


Figure 13-40. Filters used to assign one color to one pixel

## 7. Conclusion

In this chapter we have introduced the basis of digital to analog signal conversion. The implementation of resistor ladders has been detailed, with an illustration of non-linearity effects. The analog-to-digital convert principles have also been described, through the example of the flash converter and successive approach converter. Finally, temperature and light sensors have been briefly introduced.

### Exercises

#### Exercise 13-1

Design of a 3-bit thermometer to binary coder according to the schematic diagram shown in figure 12-xxx

#### Exercise 13-2

Design a 3-bit flash converter using the thermometer coder and a set of 7 comparators.

#### Exercise 13-3

Design a iterative converter using a 4-bit counter and the 2-2R 4-bit DAC.

#### Exercise 13-4

Design a successive approach converter using a specific register and the R-2R 4-bit DAC.

### References

[Backer] R.J Backer, H. W.Li, D. E. Boyce "CMOS design, layout and simulation", IEEE Press, 1998, [www.ieee.org](http://www.ieee.org)

[Razavi] B. Razavi "Design of Analog CMOS integrated circuits", McGraw Hill, ISBN 0-07-238032-2, 2001,  
[www.mhhe.com](http://www.mhhe.com)

[Gustavsson] Mikael Gustavsson, J.Jacob Wikner, Nianxiong Nick Tan "CMOS Data Converters for Communications", Kluwer Academic Publishers, ISBN 0-7923-7780-X, 2000

[Ristic] Ljubisa Ristic "Sensor Technology and Devices", Artech House, ISBN 0-89006-532-2, 1994

[Bendhia] Paper 0.18 $\mu$ m sampling

# 14

## 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. Specific aspects of I/O floorplan, supply clamp and interfacing with packages are also introduced, followed by a short description of IBIS models for describing I/O behavior, and of the signal transport between integrated circuits.

### 1. Power Supply

The power supply of integrated circuits has continuously decreased with the progresses in process integration. Figure 14-1 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. Using a low voltage is attractive for low power applications and to prevent the very thin gate oxide for overstress and possible destruction. Classically, for reliable operations, the maximum voltage handled by the gate oxide is 0.7V/nm.

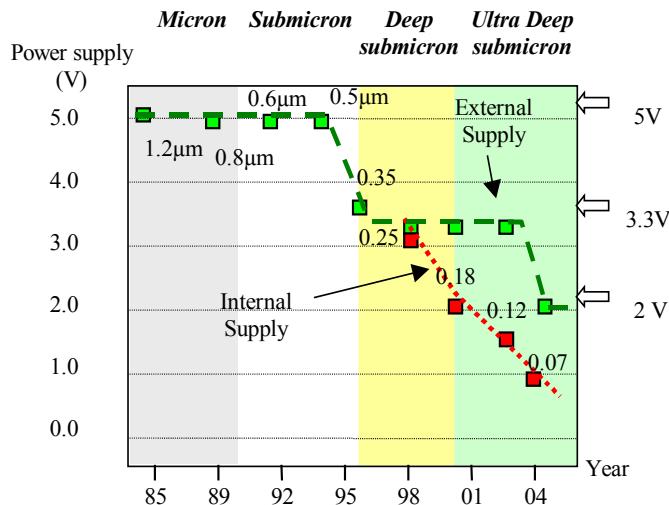


Figure 14-1: Power supply decrease with technology scale down

In  $0.12\mu\text{m}$  CMOS technology, the external supply  $\text{VDDH}$  is  $2.5\text{V}$  and the core supply  $\text{VDD}$  is  $1.2\text{V}$ . 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. Remember that the oxide thickness is  $20\text{ \AA}$  for the core MOS devices in  $0.12\mu\text{m}$ , that the breakdown voltage for high quality  $\text{SiO}_2$  is around  $0.1\text{V}/\text{\AA}$ , which means that the device can handle up to  $2\text{V}$ . Voltage translator ensures the bi-directional conversion between high and low voltage signals (Figure 14-2). In the die, several functions are supplied at high voltage, such as on-chip regulators, phase-lock-loops, ADC and DAC converters, or radio-frequency circuits and power amplifiers.

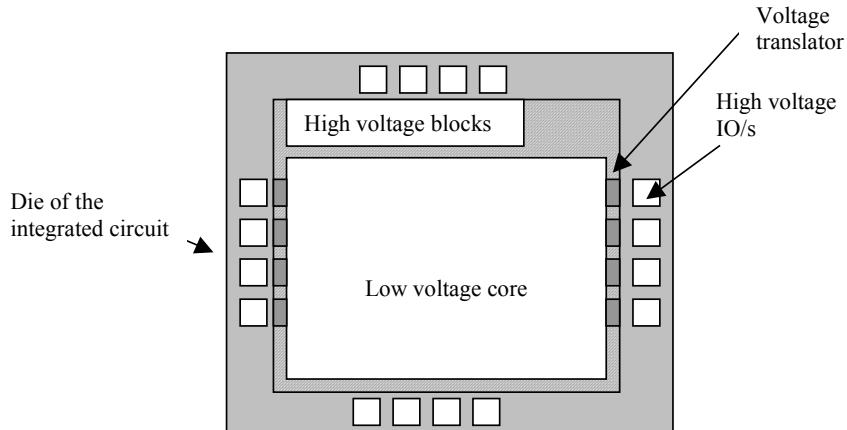


Figure 14-2: Multiple power supply in deep submicron technology

## 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 a connection wire is attached to build the electrical link to the outside word, as shown in figure 14-3.

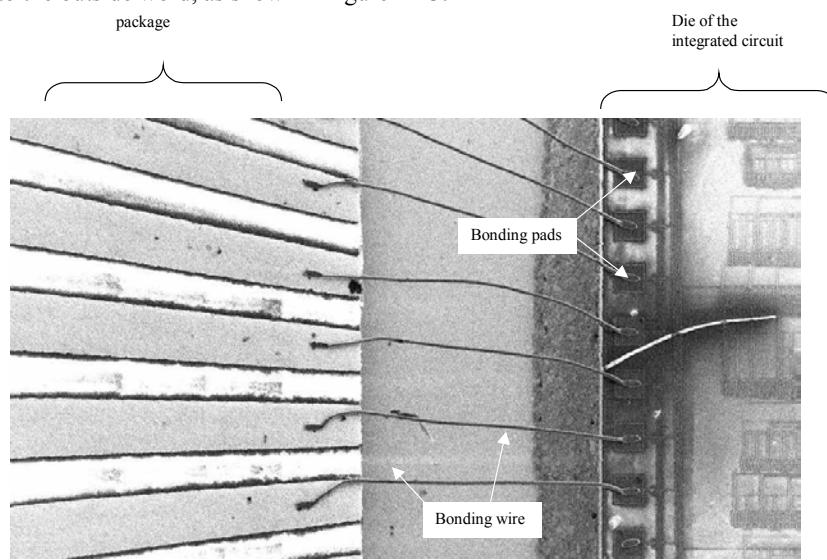


Figure 14-3: The bonding pad is used to connect a wire (called the bonding) which builds the electrical connection between the die and the package

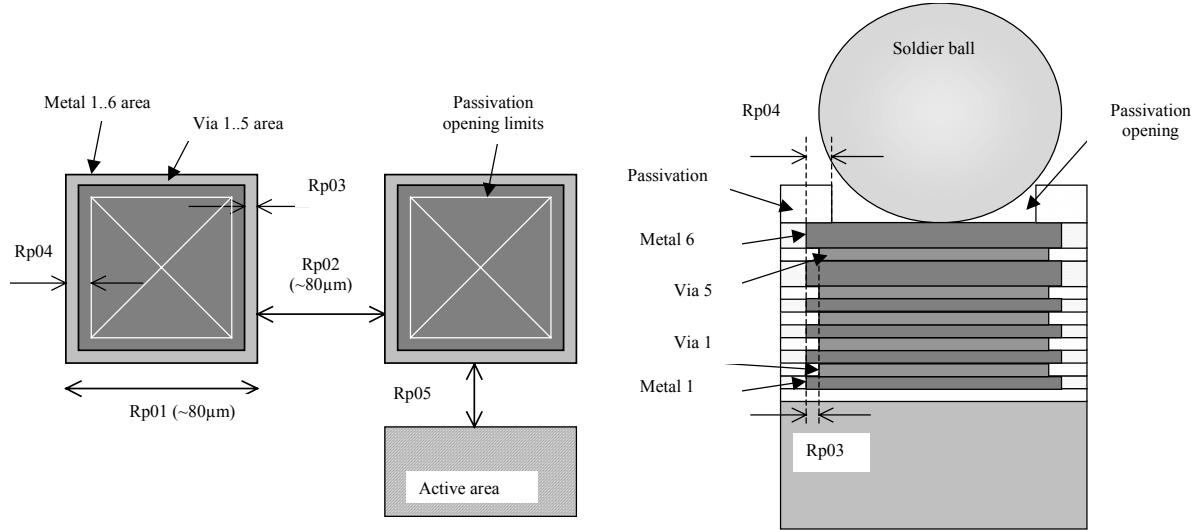


Figure 14-4: The bonding pad design rules

The pad size is approximately  $80\mu\text{m} \times 80\mu\text{m}$ . The basic design rules for the pad are shown in figure 14-4. The pad-to-pad spacing, ( $Rp02$ ), is also around  $80\mu\text{m}$ . New technologies such as 90nm enable the implementation of pad structures with  $50\times 50\mu\text{m}$  openings.

The cross-section shown in figure 14-4 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.

Design rule	Description	Value in $0.12\mu\text{m}$
$rp01$	Pad width:	$80\mu\text{m}$
$rp02$	Between two pads $100\mu\text{m}$	$80\mu\text{m}$
$rp03$	Border of via vs. Metal	$2\mu\text{m}$
$rp04$	Opening in passivation v.s last metal	$5\mu\text{m}$
$rp05$	Between pad and unrelated active area	$20\mu\text{m}$

Table 14-1: The bonding pad design rules

The pad can be generated by Microwind using the command **Edit → Generate → I/O pads**. The menu shown in figure 14-5 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 rind, as detailed later. Select the item **Single pad** and click **Generate**. Then give the location in the layout for the pad. As the pad is giant compared to the usual design scale, click **View All** to see the global pad layout.

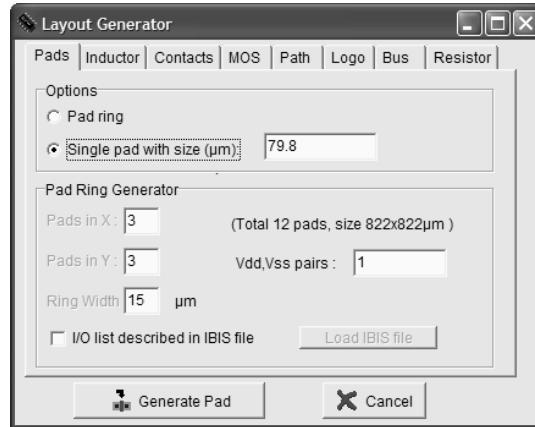


Figure 14-4: The bonding pad menu

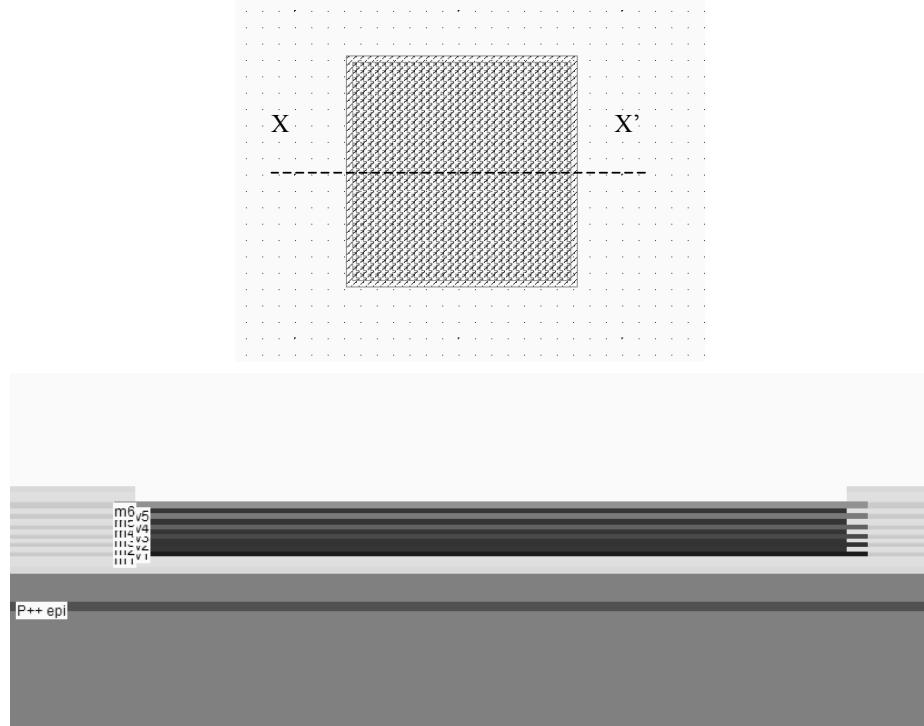


Figure 14-5: The bonding pad generated by Microwind and its cross-section. On top of metal6, the passivation has been removed (IOPad.MSK)

The cross-section of the pad is reported in figure 14-5. We see the stack of metal layers, the vias and the passivation opening for the bonding to the package. In some CMOS technologies, there may exist a constraint on the via size that forces the designer to split the large via surface into an array of elementary via with the 2 lambda size, 4 lambda spacing. This type of design rule has not been implemented in Microwind.

### 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 14-6. The proposed architecture is based on 5 pads on each side, meaning a total of 20 pads.

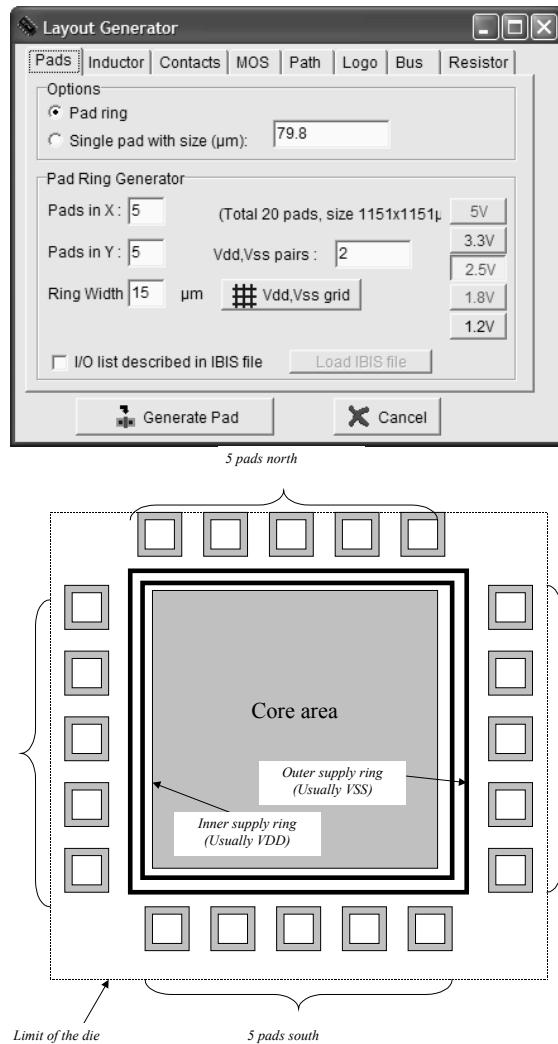


Figure 14-6: The menu for generating the pad ring and the corresponding architecture

The layout of the default pad ring generated in  $0.12\mu\text{m}$  is shown in figure 14-7. 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.

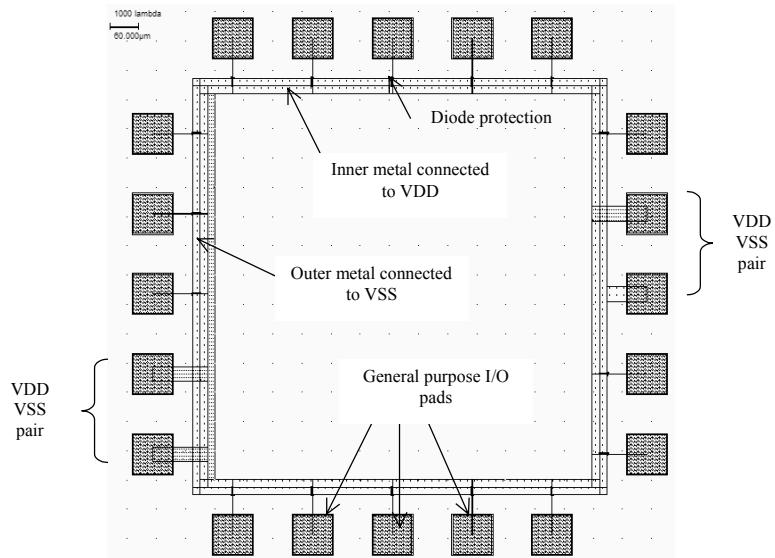


Figure 14-7: The default pad ring generated in  $0.12\mu\text{m}$ , with 20 pads, including two pairs of VDD/VSS supply pins (padRing.MSK)

The way the supply pads are connected to the internal rings is detailed in figure 14-8. All VSS pads are connected to the outer ring, while the VDD pads are connected to the inner ring. The more the circuit needs power, the larger the number of supply pads. In a 800 I/O integrated circuit, nearly 100 pads are dedicated to the voltage supply, split equally between VSS and VDD supply pads.

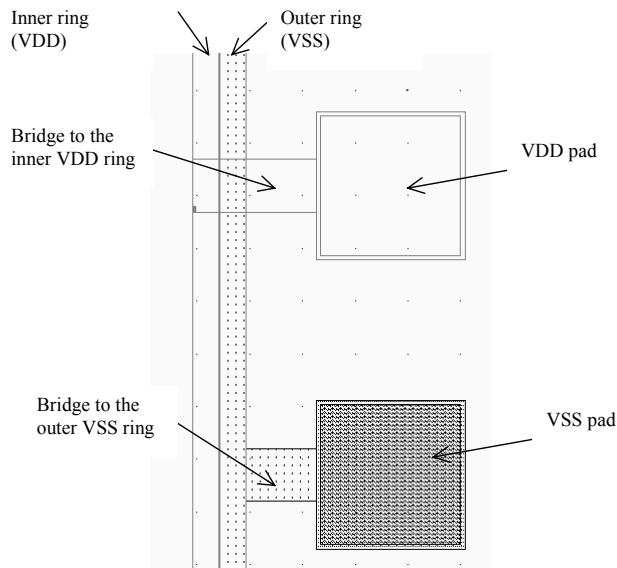


Figure 14-9: Connecting to the VSS and VDD internal ring (PadRing.MSK)

### The supply rails

The supply voltage may be 5V, 3.3V, 2.5V, 1.8V or 1.2V. Most designs in 0.12 $\mu$ m 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). An example of a four ring design is shown in figure 14-10.

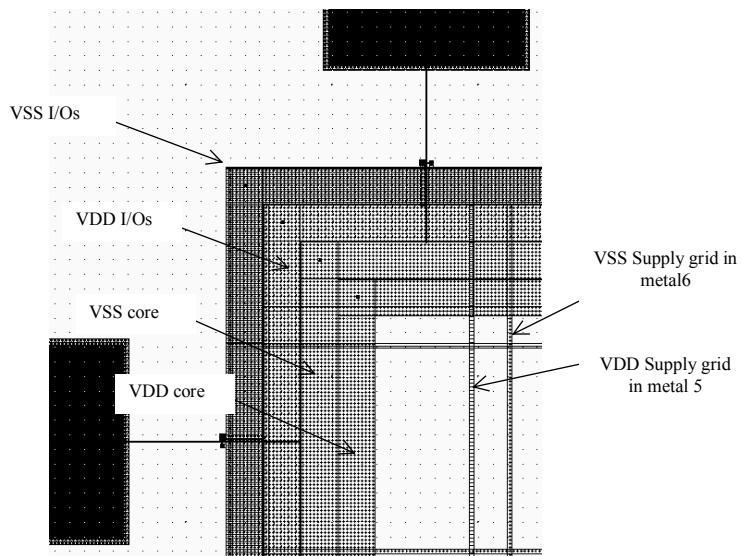
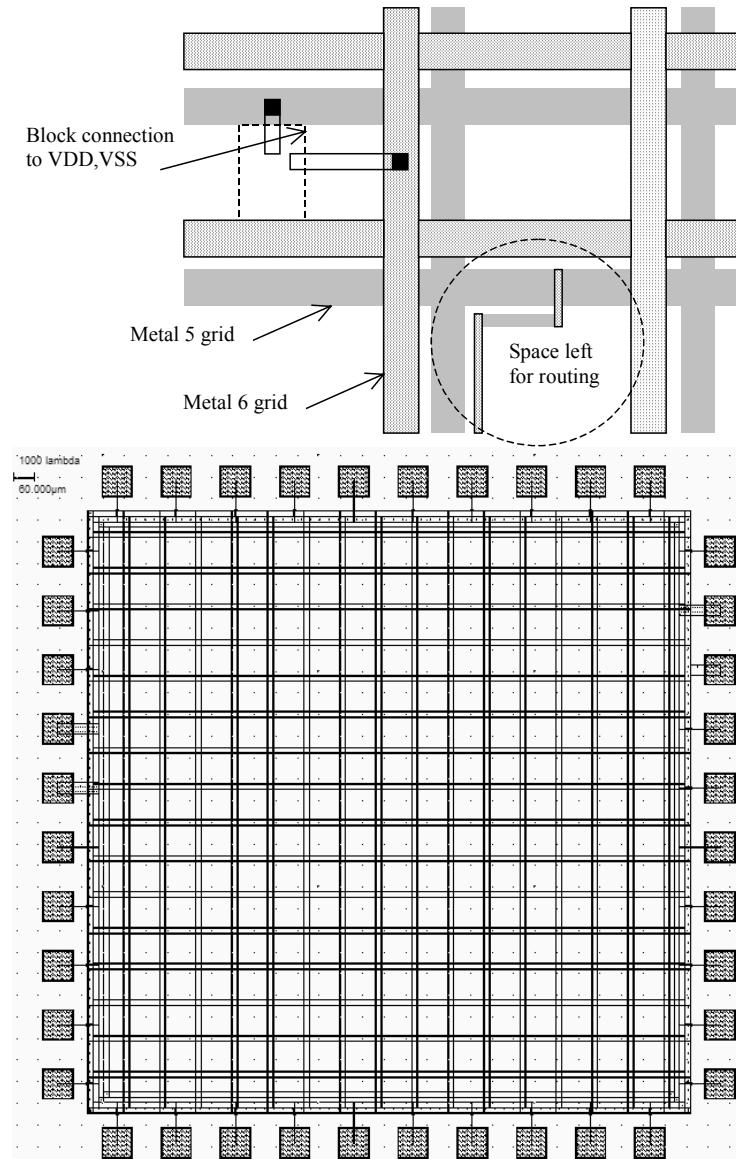


Figure 14-10: Zoom at the supply rings VSS, VDD for I/Os and core (PadRingDoubleGrid)

### Supply rail design

A metal wire cannot drive an unlimited amount of current. When the average current density is higher than  $2.10^9$  A/m<sup>2</sup> [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 2mA/ $\mu$ m width for aluminum supply lines and 5mA/ $\mu$ m for copper, which means that a copper interconnect is superior to aluminum in sustaining large currents.



*Figure 14-11: 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 usual design approach consists in creating a regular grid structure, as illustrated in figure 14-11, 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. These upper layers are thicker than lower metal layers and consequently can drive larger currents. The grid spacing is around  $100\mu\text{m}$ , and each supply conductor width is around  $5\mu\text{m}$ . Enlarging the supply width would reduce the routing area. Reducing the supply width would limit the current drive.

### Metal Slit

Certain precautions should be taken when enlarging metal lines larger than  $20\mu\text{m}$ . The coefficients of thermal expansion of  $\text{SiO}_2$  and metal are significantly different: 0.5 ppm/ $^{\circ}\text{K}$  for  $\text{SiO}_2$ , 2.8 for Silicon and 23 for aluminum. The stress accumulated during important temperature variations may break the large metal traces and crack oxides [Hastings][Shimokura]. Metal tracks less than  $20\mu\text{m}$  width do no suffer such problems. The automatic layout generation tools in Microwind do not take this effect into account. The correct design in the case of large metal tracks is shown in figure 14-12. Holes in the metal are inserted regularly to split the box into parallel conductors to discourage delamination and oxide cracks.

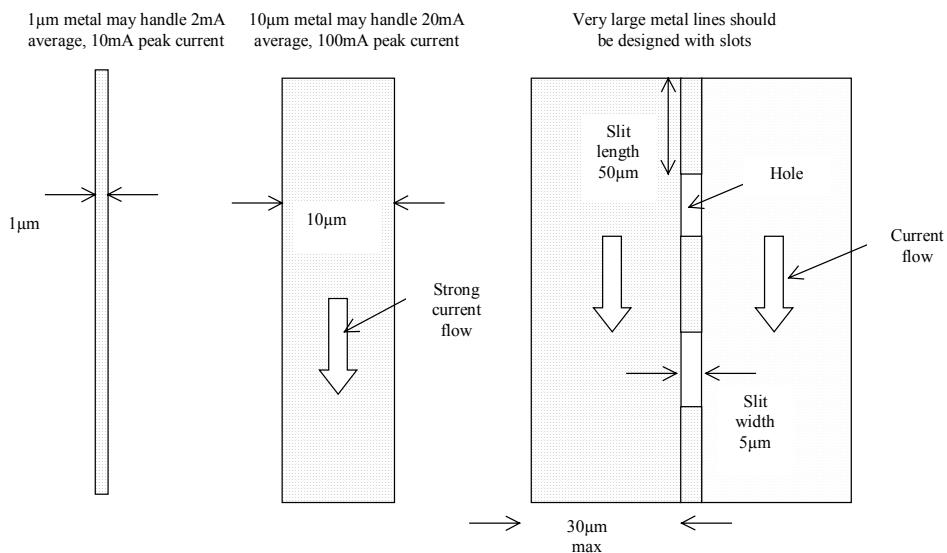


Figure 14-12: Very large metal lines are designed with slots to prevent the damages caused by thermal stress

### Power line connection

The layout design of power line connections must be handled with care because of the strong current flow. An example of poor design is reported in figure 14-13 (a). Metal5 is connected to metal6 using via5. In (a), an unreliable connection is formed between metal5 and metal6 because the number of vias is too small regarding the amount of current. Remember that each via can only drive approximately 5mA average current. A better approach (b) consists in placing contacts on the border of the intersection area. The best approach (c) is to fill the crossing area with contacts. An in-depth study of power line design may be found [Clein].

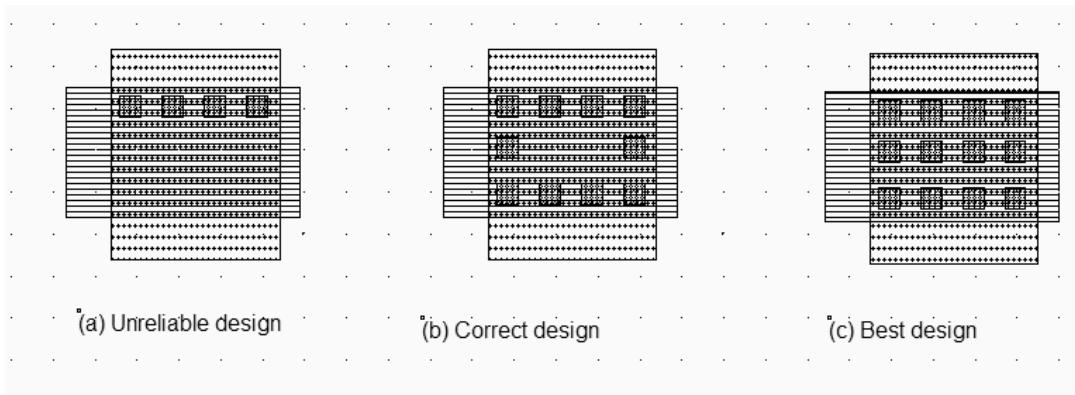


Figure 14-13: The interlayer connection using vias (ConnectLayers.MSK)

#### 4. Input Structures

The input pad includes some over-voltage and under-voltage protections due to external voltage stress, electrostatic discharge (ESD <Glossary>), 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 5000 Volt, with a general shape similar to that of figure 14-14. The design of I/Os which may handle such a high voltage and which may dissipate such a high energy safely requires specific techniques which are introduced in the following sections.

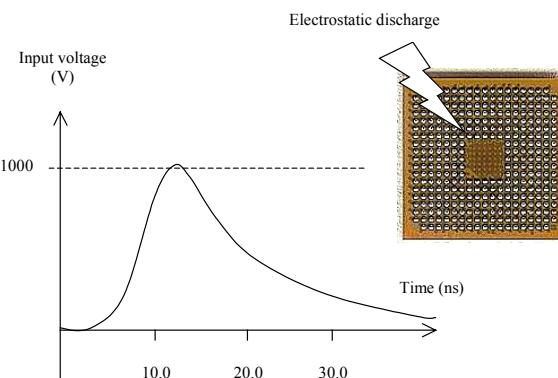


Figure 14-14: Example of electrostatic discharge on the input/output pin of an integrated circuit

One of the most simple ESD protections is made up of one resistance and two diodes (Fig. 14-15). The resistor helps to dissipate the parasitic energy and reduces the amplitude of the voltage overstress <Glossary>. 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.

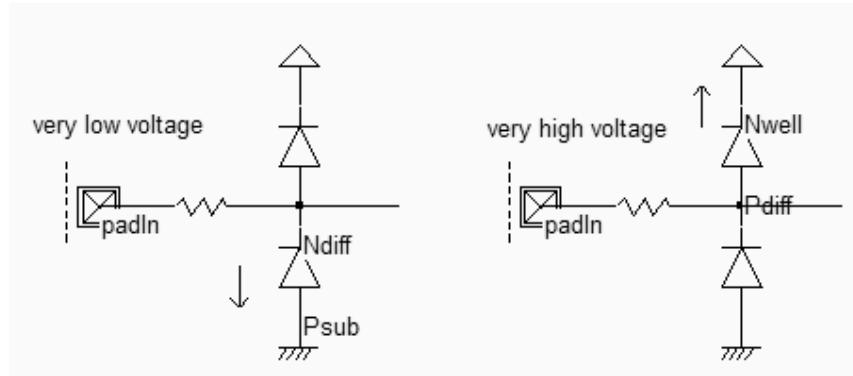


Figure 14-154: Input protection circuit (IOPadIn.SCH)

### Resistor Design

Two types of resistors are available for ESD protection : polysilicon and N-well. The polysilicon resistor is completely embedded in oxide, and may therefore handle very high voltage stress. However the polysilicon is thin and its current capabilities are limited. The default salicidation of the polysilicon layer must be removed to obtain a high resistivity. The design of an ESD resistor is different from usual polysilicon resistors, mainly because of the high current, requiring an enlarged layer width and multiple contacts (Figure 14-16). The usual ESD resistance value ranges between 50 and 2000 ohm. In Microwind, the polysilicon resistor is generated by fixing the target resistance value, and enlarging the width to several  $\mu\text{m}$  to handle strong currents.

The major drawback of polysilicon resistor is the poor heat dissipation. The oxide has a positive role in isolating the polysilicon conductor from the rest of the circuit for over-voltages expressed in Kilo-volts. However, the same oxide plays a negative role by thermally isolating the polysilicon material which would ultimately melt because of overheating.

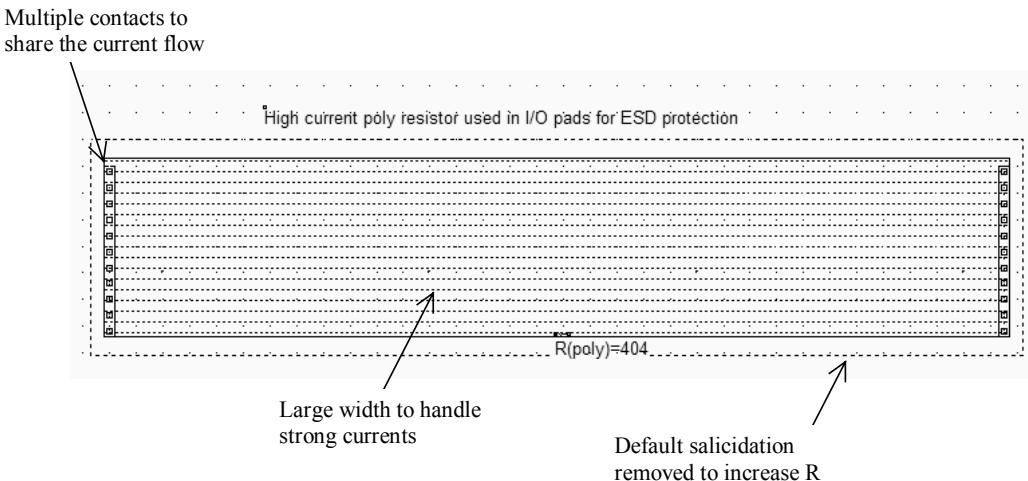


Figure 14-16: Polysilicon resistor used in input protection circuit (ResPoly.MSK)

An alternative solution consists in using the N-well as a resistor. In that case, N+ diffusion contacts are used for the electrical connection between the upper metal and the lower well area. The thermal conduction is excellent but the voltage isolation is limited by the breakdown of the junction between the N-well resistor and the P-substrate.

### Diode Design

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, as shown in figure 14-17.

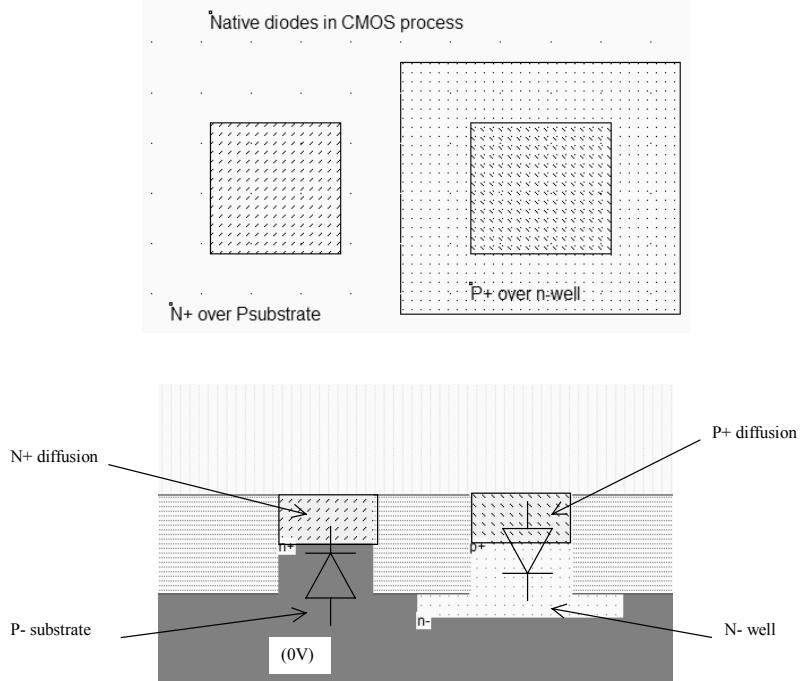


Figure 14-17: The most simple diodes in a CMOS process (Diode.MSK)

As the substrate is at 0V, the P-/N+ diode is turned on only if the N+ voltage is significantly negative (-0.5V). If the N+ voltage is higher than 0V (Which is the usual case), the diode has no effect, and can be considered as a parasitic capacitance. As the n-well is usually connected to a high voltage VDDH, the P+/Nwell diode is turned on only if the P+ voltage is higher than the I/O supply voltage VDDH (VDDH + 0.5V, that is around 3V in 0.12μm).

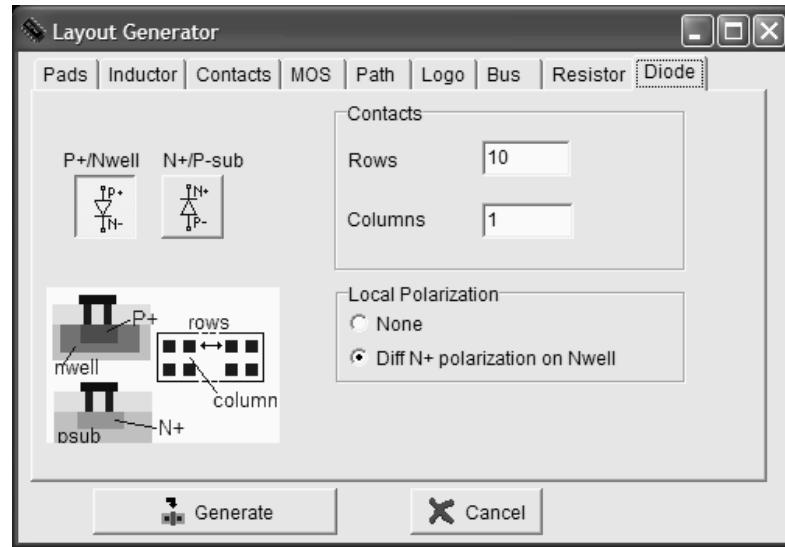


Figure 14-18: The diode generating menu in Microwind (By default a P+/well diode)

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 14-19. The large number of rows ensures a large current capability, which is very important in the case of ESD protection devices.

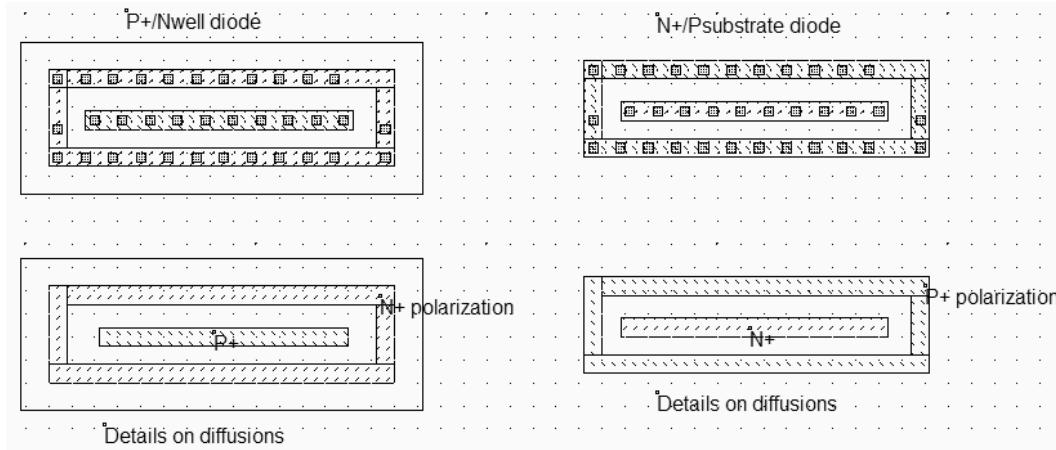


Figure 14-19: Generating a default protection diode (IODiode.MSK)

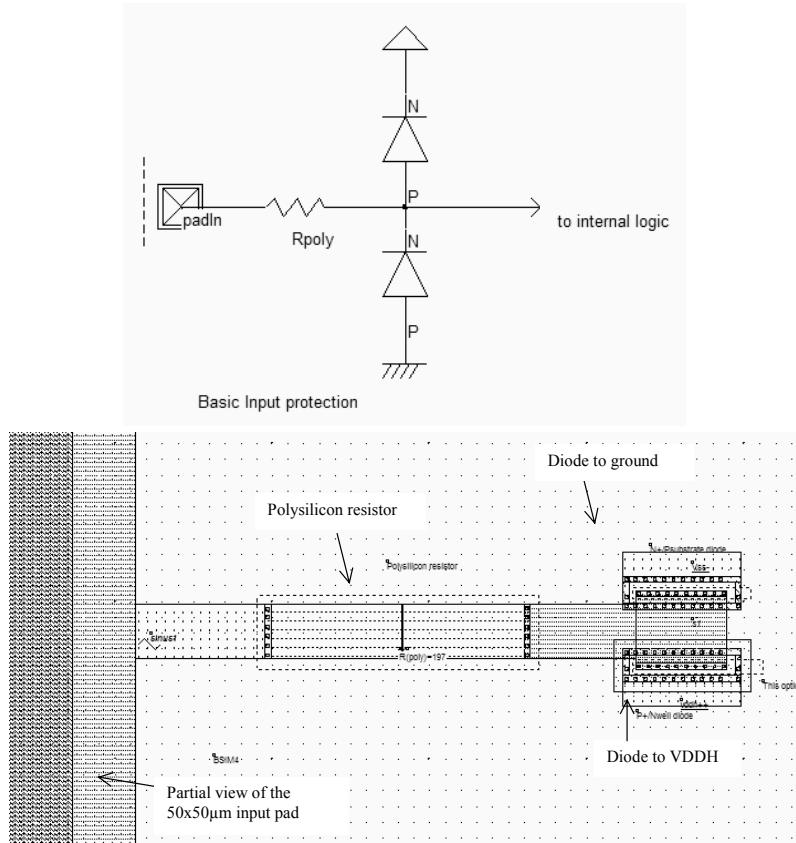


Figure 14-20: A test case to evaluate the role of diodes (IoPadIN.MSK)

A protection circuit example is given in figure 14-20. It consists of a pad  $50 \times 50\mu\text{m}$ , a serial resistor around 200 ohm and two diodes. When a very high sinusoidal waveform ( $\pm 10\text{V}$ ) which corresponds to an electrical overstress 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  $+10\text{V}$  wide. Notice that the current flowing in the diodes is around 1mA (Figure 14-21).

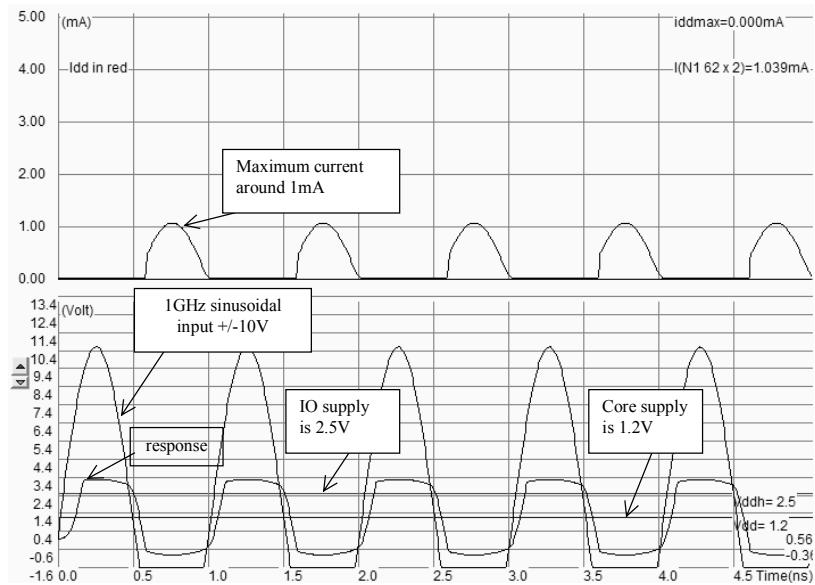


Figure 14-21: The diodes clamp the positive and negative overstress so that the internal voltage keeps close to the voltage range [0..VDDH] (IoPadIN.MSK)

Considering a real case electrostatic discharge, the voltage may rise to 1000V-5000V, which corresponds to a diode current more than 100 times larger, that is 100-500mA. Around 100 contacts would be needed for minimum reliability. In industrial case ESD protections, the diode length is approximately 50 $\mu$ m. Notice that the lateral surface of the diode is more important than its surface, as the current flows mainly horizontally. The design shown in figure 14-22 (b) should be preferred to the one of figure 14-22 (a) because the lateral surface is larger, meaning a better current efficiency.

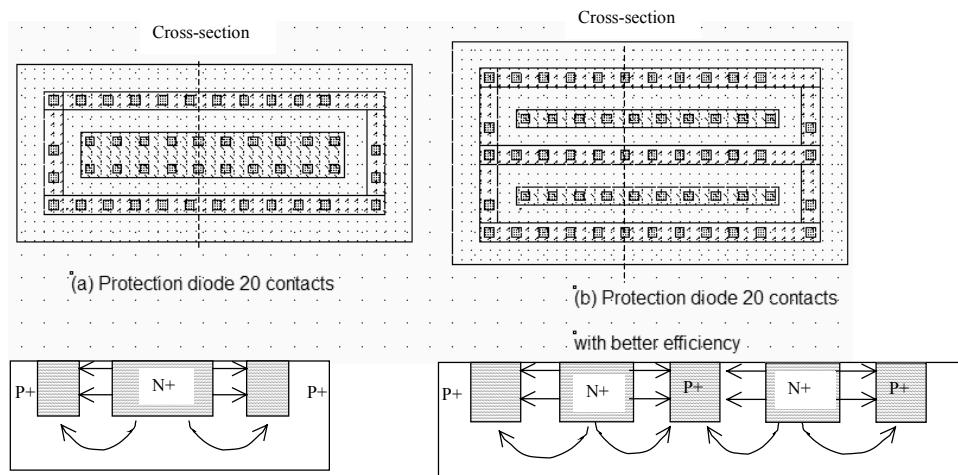


Figure 14-22: The lateral surface of the diode should be maximized for better efficiency (IODiodes2.MSK)

### Clamp MOS Devices

An interesting device for electrostatic discharge prevention is described in [Dabral], called the gate-coupled NMOS. The schematic diagram of the circuit is shown in figure 14-23. It consists of two stages of protection, the first one which handles the majority of the current, and the second one which assists the first stage with relaxed stress constraints. Such protections can handle 5-7KV (Kilo-volt) ESD stress, with several design iterations and experimental testing.

The  $C1-R1$  circuit is a high pass filter. By default the voltage of node  $Ng$  is zero, due to the weak tie to VSS. A very sharp over-voltage such as the one created by an electrostatic discharge induces a positive voltage on node  $Ng$  which turns on the clamp MOS by capacitance coupling. A current path is established between the pad and the ground, until the voltage of  $Ng$  goes below the threshold voltage. Finally, the clamp is turned off and the ESD charges are eliminated. Note that a nominal rise edge from 0 to VDD should not turn on the clamp. Therefore,  $C1$  and  $R1$  must be chosen to eliminate the ESD pulse while keeping quiet in presence of logic edges.

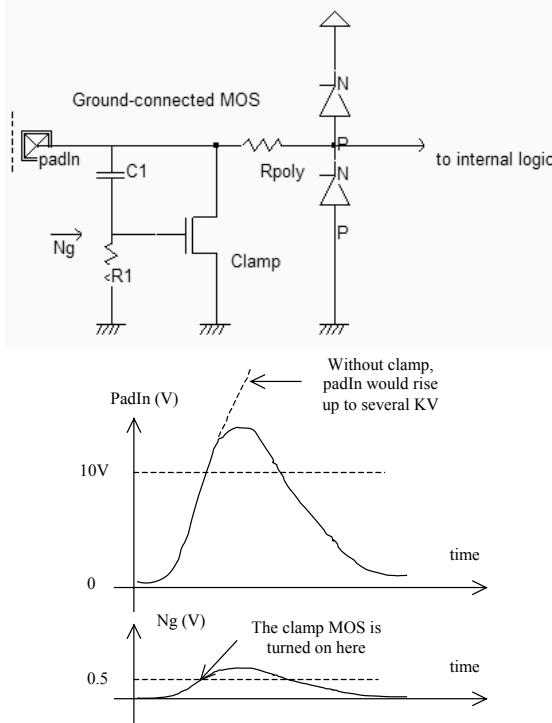


Figure 14-24: The ground-connected MOS turns on and short cuts the over voltage at a very sharp rise edge of the input voltage, such as in an ESD pulse (IOPadIn.SCH)

A lot of subtle layout issues arise with the implementation of high performance electrostatic discharge protections, as described in [Wang]. The clamp MOS is a good example of specific layout techniques for optimized behavior when faced to overstress. The driving idea is to keep the parasitic current flow straight from the input pad to the ground. A double oxide MOS device (See next section) is used to handle strong voltage stress. The diffusion connected to the pad is enlarged to create a small serial resistance which is used as a ballast (Figure 12-25). The salicidation of the drain and source is generally removed to increase this ballasting effect.

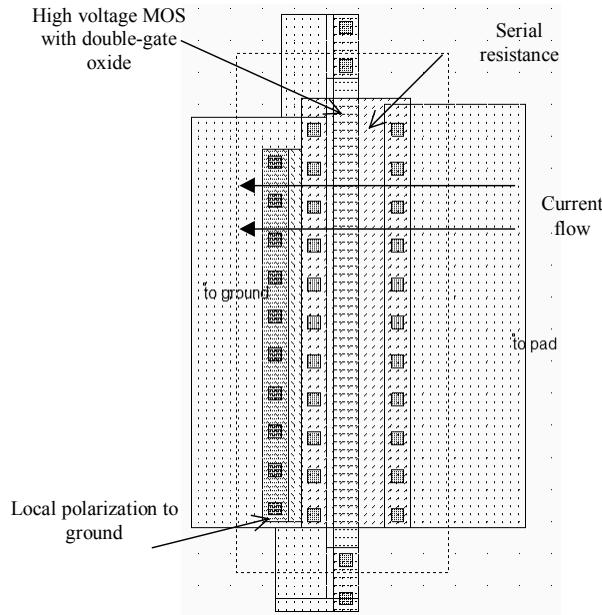


Figure 14-25: Specific layout of the gate-grounded MOS used in advanced input pads (ggMos.MSK)

### Zener Diode

The Zener diode is equivalent to a normal diode, but has a particular behavior in invert mode, as it turns on for a very negative  $V_{PN}$  voltage, that is a very high  $padIn$  voltage. The characteristics of the Zener diode are shown in figure 14-26. For positive  $V_{PN}$ , the diode is in direct mode, for negative  $V_{PN}$ , the diode is off. However, when  $V_{PN}$  is strongly negative (Less than  $V_Z$ ), the diode is turned on again, with a so-called Zener effect. The diode layout is also shown in figure 14-26. An option layer configured to extract the diode surrounds the diode area (Dot rectangle in the layout). The diode model parameters are derived from the BSIM4 model, and are not accessible to the user. In contrast, the surface of the diode has a direct impact on its characteristics.

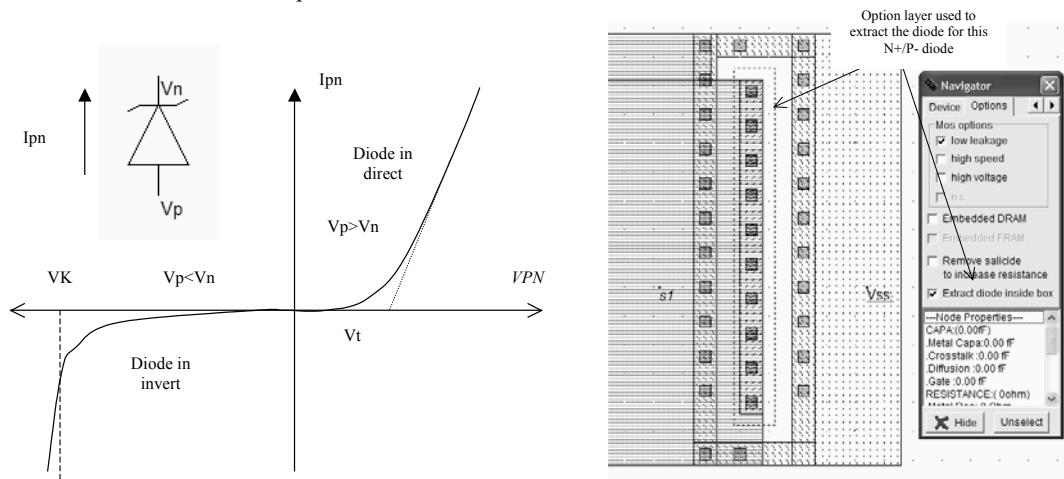


Figure 14-26: The Zener diode (IOPadZener.MSK)

Turning back to the input protection circuit, a significant increase of the pad voltage corresponds to a negative increase of  $V_{pn}$ . When passing the  $V_K$  limit, the Zener diode is turned on and the charges start to flow through the substrate to the ground.

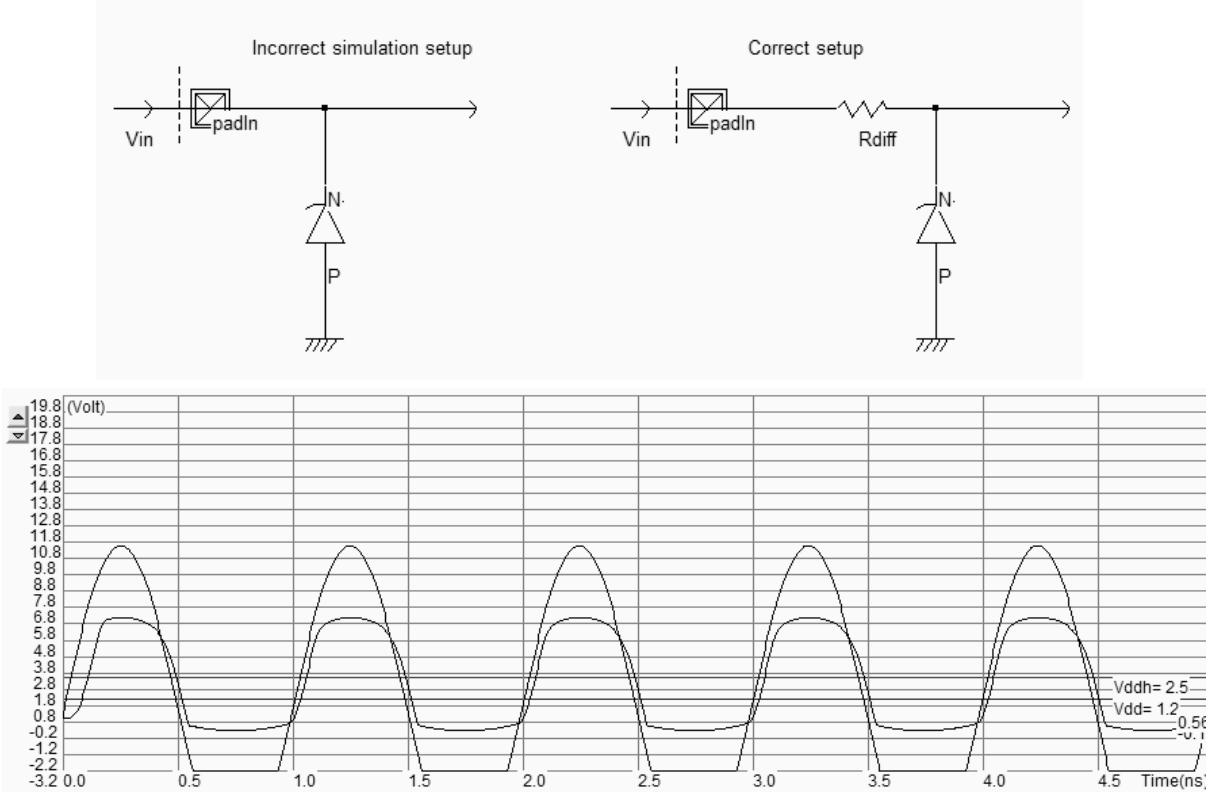


Figure 14-27: The Zener effect can be seen for positive overstress (IOPadZener.MSK)

The simulation of the Zener diode as a protection circuit is proposed with the schematic diagrams of figure 14-27. The simulation setup proposed on the left is incorrect as the direct connection of the voltage source to the Zener diode also forces the output voltage so that no clamp effect can be seen. In contrast, the serial resistor  $R_{diff}$  of the right figure creates the required impedance between the voltage source and the output to enable the observation of the Zener effect. The diode model used in Microwind includes the Zener effect if the simulation is performed in BSIM4 mode. In the simulation of figure 14-27, the large positive voltage provokes the necessary conditions for a negative  $V_{pn}$  and consequently a Zener clamp. The diode in direct mode is observed for negative input values, which corresponds to positive  $V_{pn}$ .

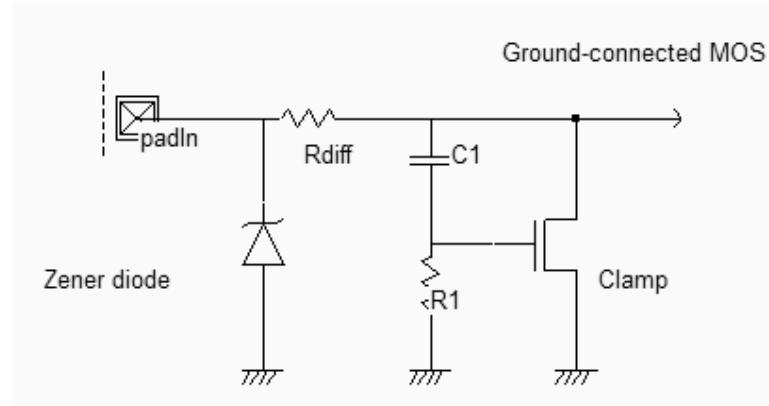


Figure 14-28: An input pad protected with a Zener diode and a diffusion resistor (IOPadIn.SCH)

An input protection circuit which combines the Zener diode as a primary protection circuit , the ground-connected MOS and a secondary protection circuit is proposed in figure 14-28. Such structures may handle severe ESD stress, as well as other parasitic transient pulses found in industrial applications.

### High voltage MOS

The general diagram of an input structure is given in figure 14-29. 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. In DSCH, the high voltage devices are drawn with a double line. The symbol  $Vdd\_HV$  represents the I/O voltage, which is usually 2.5V in CMOS 0.12 $\mu$ m.

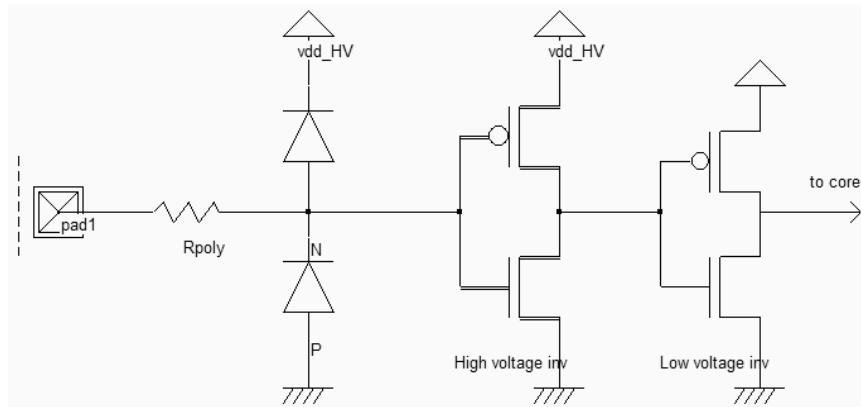


Figure 14-29: The basic principles for an input circuit, including the ESD protection and the voltage translator (IOPadIn.SCH)

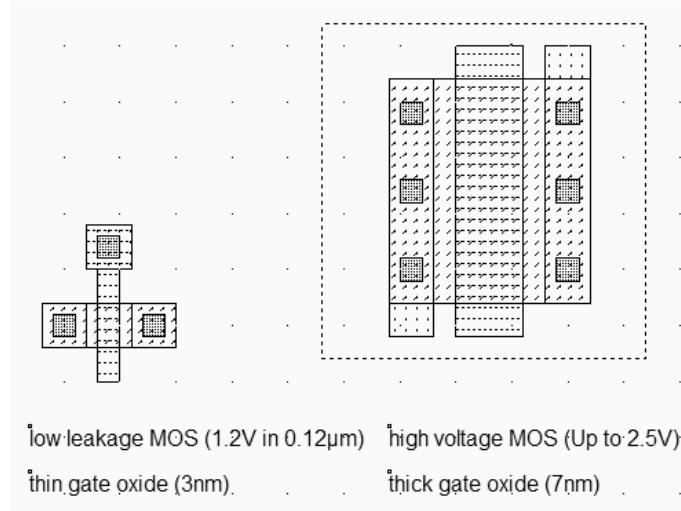


Figure 14-30: High voltage MOS device vs. normal MOS (MOSHighVoltage.MSK)

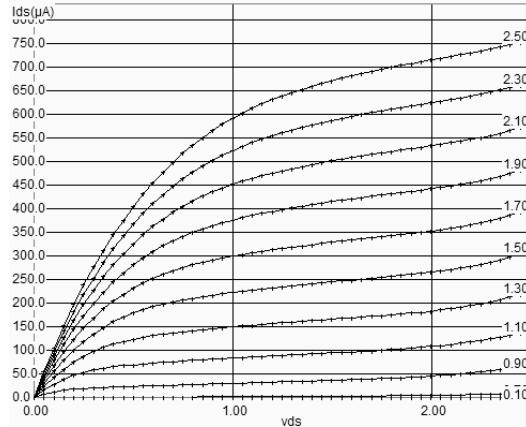


Figure 14-31: Static characteristics of the high voltage MOS (MOSHighVoltage.MSK)

The high voltage MOS layout differs slightly from the normal MOS, as shown in the comparative layout view of figure 14-30. 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. It has been shown that lateral drain diffusion degrades the ESD protection performances [Wang]. One reason is the lower efficiency of LDD devices in enabling strong currents to flow in the channel. Consequently, LDD device are slower to evacuate the parasitic energy. The gate oxide thickness is twice the oxide of the core logic. In 0.12 μm, the gate oxide of the high voltage MOS is around 5 nm, while the core MOS is 2 nm.

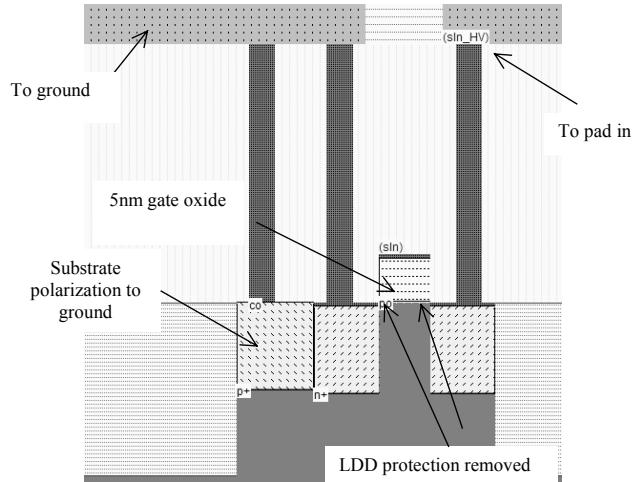


Figure 14-32: The particularities of MOS devices used in input pads: removed LDD and double gate oxide (IOPadMos.MSK)

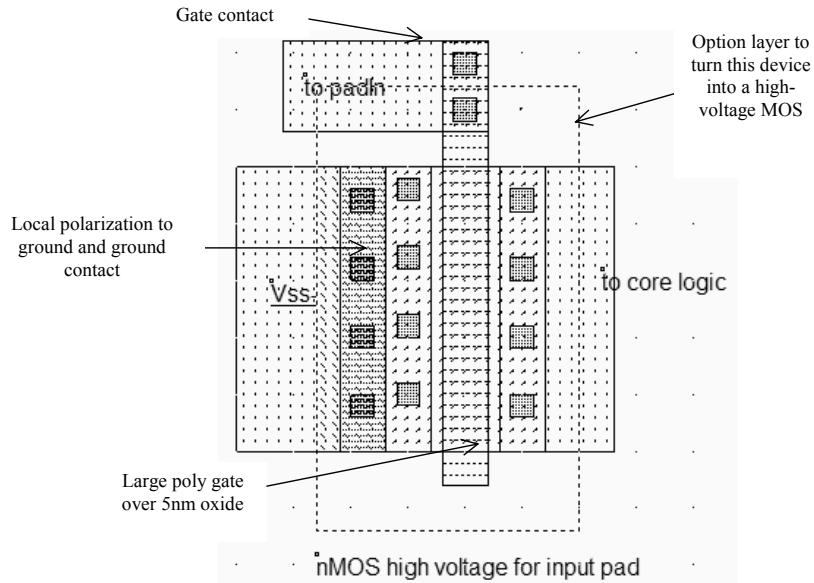


Figure 14-43: Layout of the input MOS device (IOPadMos.MSK)

The bird's view of the layout (Figure 14-43) 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  $0.25\mu\text{m}$  MOS device. To turn a normal MOS into a high voltage MOS, the designer must add an option layer (The dot rectangle in figure 14-43). The tick in front of **High voltage MOS** (Figure 14-44) assigns high voltage properties to the device : double oxide, removed LDD, different rules for minimum length, and different MOS model parameters.

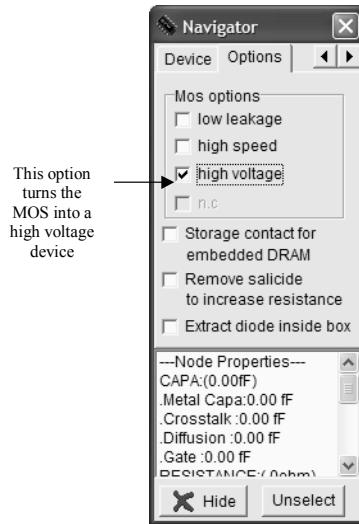


Figure 14-44: Handling the high voltage property through the option layer (IOPadMOS.MSK)

The simulation of the complete input pad is proposed with the schematic diagram of figure 14-45. A slow sinusoidal waveform *DataIn* (10MHz) is generated between 0 and 2.5V, with an additive noise. The noise is a random number mainly concentrated between  $-1$  and  $+1$  V, with a Gaussian distribution. The noise contains virtually all frequencies spread uniformly from very low to very high frequencies. The noise input passes through the serial polysilicon resistor of around  $330\Omega$ , then through the two diodes and serves as the input for the high voltage inverter. The output *SinHV* is connected to the low voltage inverter.

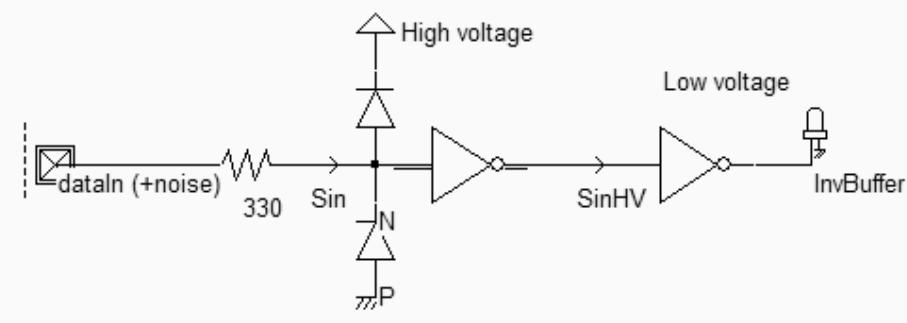


Figure 14-45: Simulation of an input structure with voltage translation from high to low voltage (IOPadIn.SCH)

Click **Add Noise** in the sinusoidal parameter window to activate the noise generation in addition to the desired signal. The Gaussian noise model gives a good approximation of ambient noise (Figure 14-46). The RMS amplitude is derived from an evaluation of the Root-Mean-Square of the noise sampled data *noise[n]*. The RMS voltage is given by equation 14-1. It gives a good indication of the envelope of the noise, as the exact amplitude may not be determined due to the random nature of the signal.

$$V_{rms} = \sqrt{\frac{\sum_{i=0}^n (\text{noise}[i])^2}{n}} \quad (\text{Eq. 14-1})$$

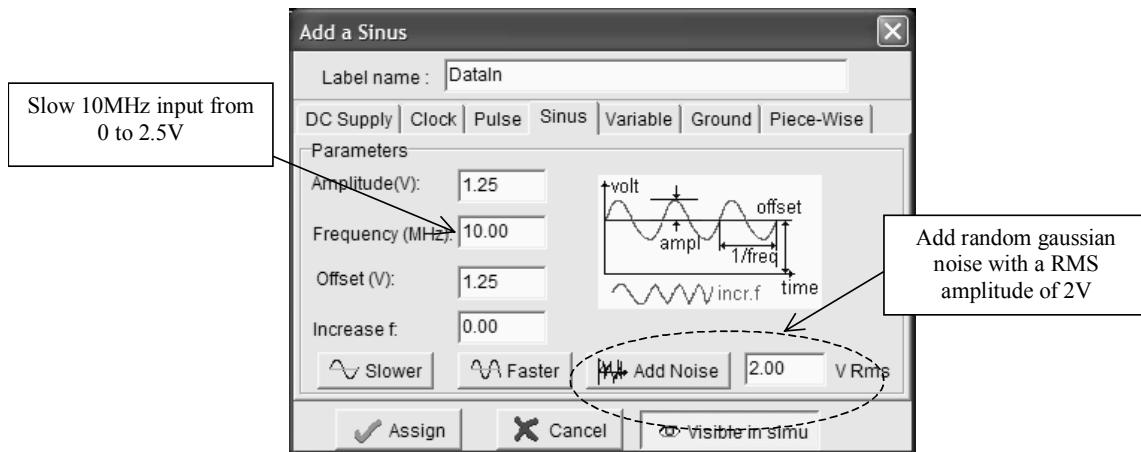


Figure 14-46: Handling the high voltage property through the option layer (IOPadMOS.MSK)

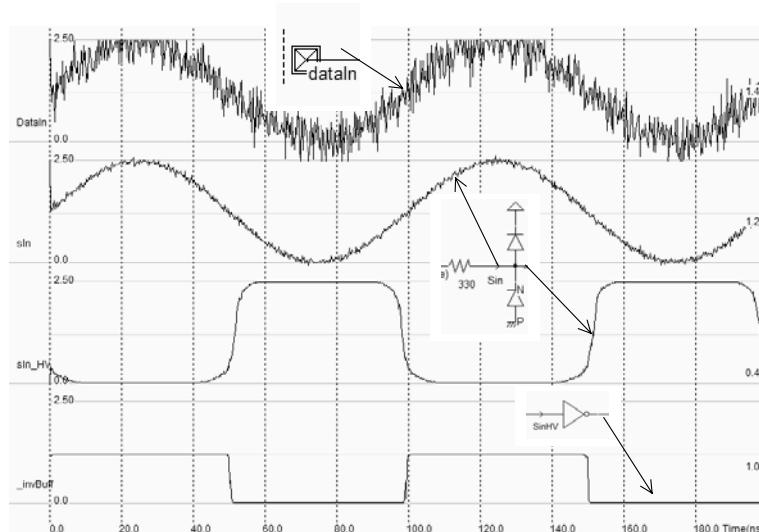


Figure 14-47: Simulation of a noisy input and the response of the input buffer (IOPadInInv.MSK)

The simulation shown in figure 14-47 gives an interesting insight of the signal propagation within the input pad. First, the noisy sinusoidal waveform is significantly filtered by the serial resistance and the parasitic diode capacitance. The noise amplitude of signal *Sin* is greatly reduced. However, due to the slow rise and fall of the input signal, a risk of parasitic glitch may appear. The signal *Sin\_HV* gives a logic translation of the input voltage which is converted to a low swing signal *Inv\_buff* by the low voltage inverter.

### 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 14-48.

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.

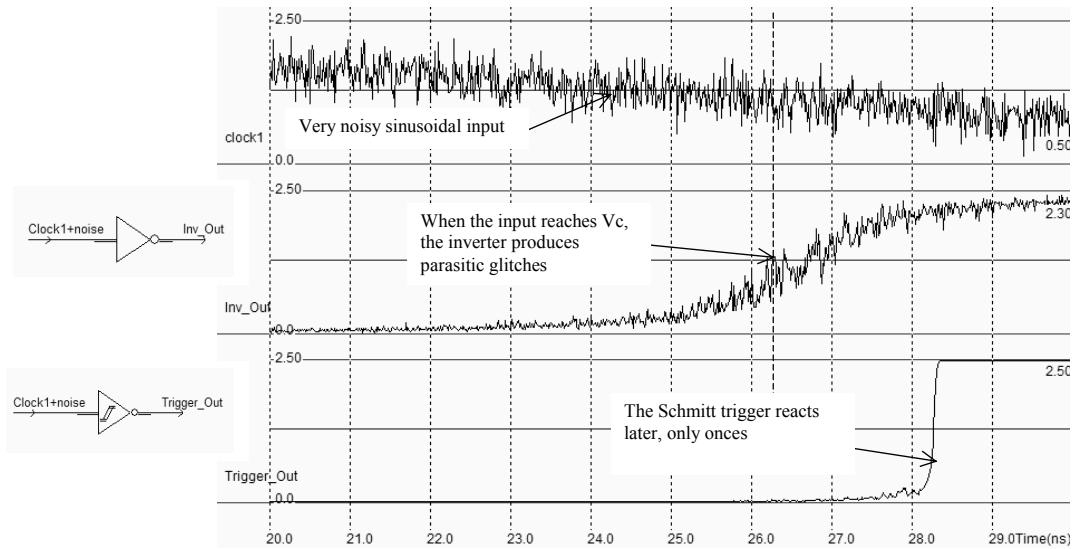
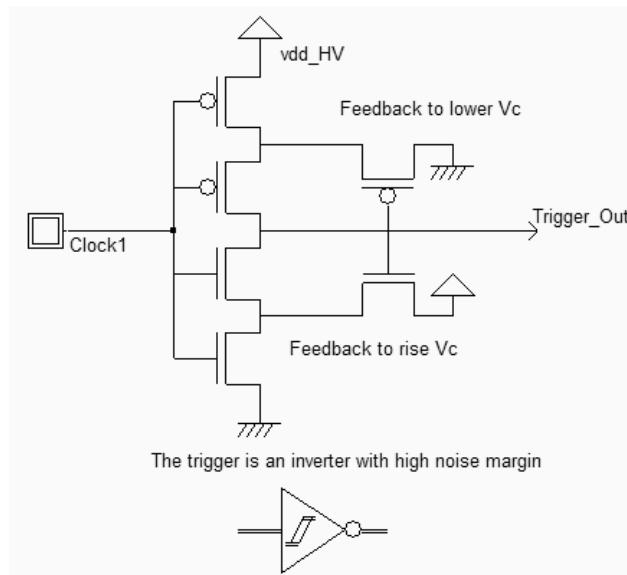


Figure 14-48: The filtering effect of the Schmitt trigger with a noisy input signal (TriggerCompInv.MSK)

The schematic diagram of the trigger is proposed in figure 14-49 [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  $VDD\_HV$  when *Trigger\_Out* is high. Consequently, the threshold voltage has risen to  $V_{C\_high}$ , higher than the commutation point  $V_C$ .



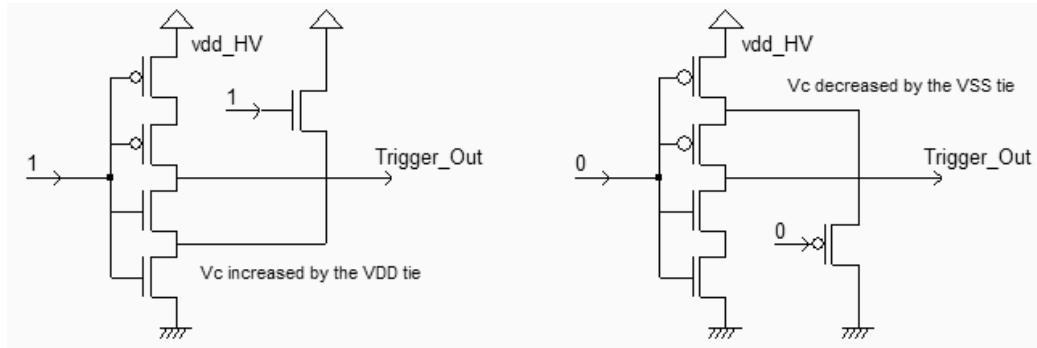


Figure 14-49. Schematic diagram of the trigger (Trigger.SCH)

The layout of the trigger is shown in figure 14-50. 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 14-51).

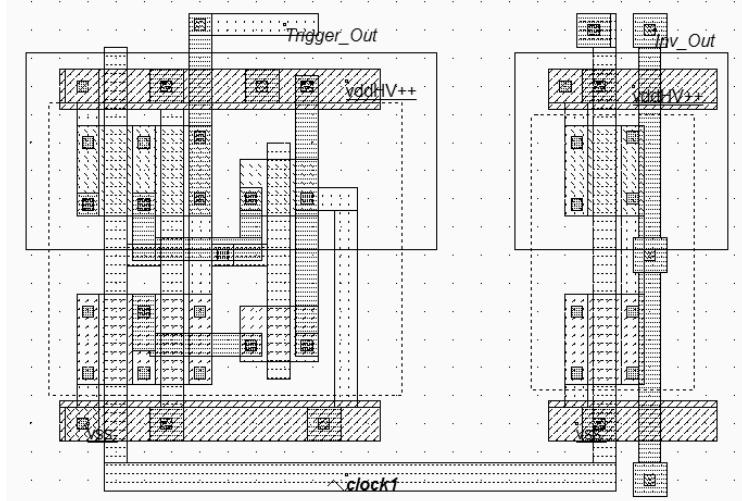


Figure 14-50. Layout of the trigger and the inverter, using high voltage MOS devices (TriggerCompInv.MSK)

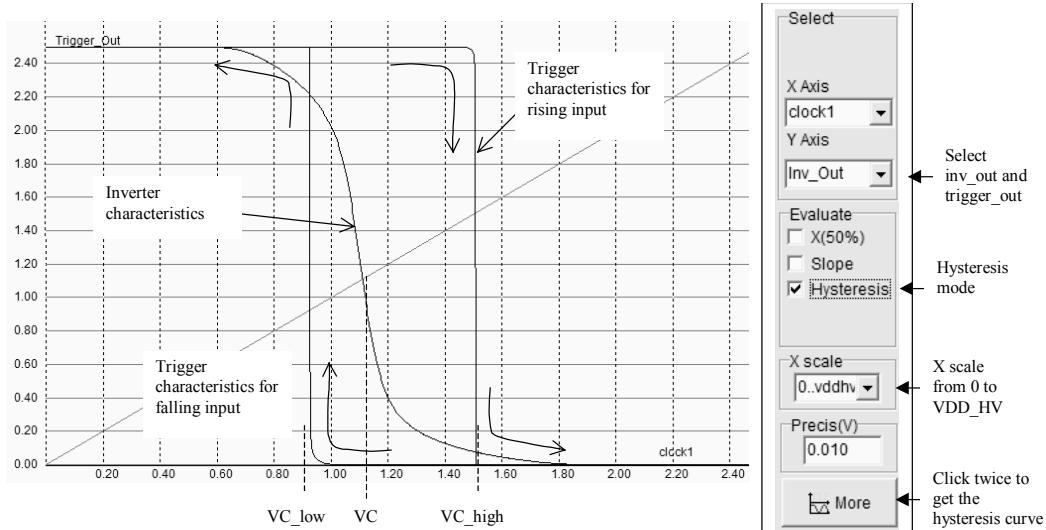


Figure 14-51. Static characteristics of the trigger compared to the inverter (Trigger.MSK)

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.

## 5. Digital Output Structures

The role of the output buffer is to ensure that the signal coming out of the integrated circuit (IC1 in figure 14-52) is propagated safely to the receiver which is usually the input of a second integrated circuit (IC2). The emitter signal comes from a low voltage inverter *Inv\_out1*. In 0.12μm, the voltage range is [0..1,2V]. Most I/O interfaces operate at high voltage (2.5 or 3.3V) for compatibility and speed reasons, as well as robustness to parasitic interference as described in part 4. The signal is transformed into a high voltage signal through the inverter *Inv\_Out2* [0..2,5V] which is directly connected to the pad and to the outside world. The signal goes through the package, the printed circuit board, and the other package which is represented by a transmission line. At the far end of the transmission line, we find the input structure of a receiving integrated circuit IC2, consisting of an inverter *Inv\_In1*, working at high voltage, and of *Inv\_In2*, working at low voltage.

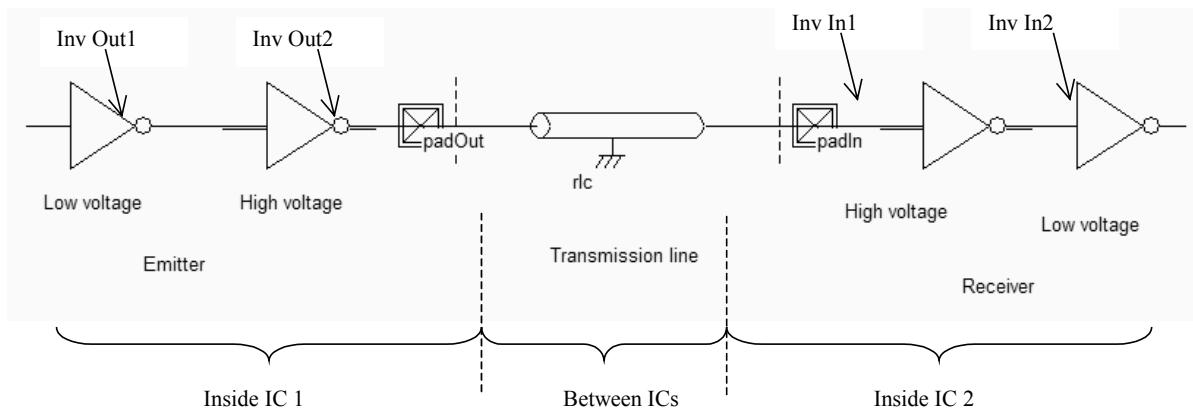


Figure 14-52: The signal propagation between integrated circuits IC1 and IC2

### Output Buffer

The schematic diagram of the basic output buffer is given in figure 14-53. A very simple structure is used to protect the output buffer from electrostatic discharge, and more generally from any over or under voltage. A Zener diode may be used, or a set of two diodes, as for the input pad.

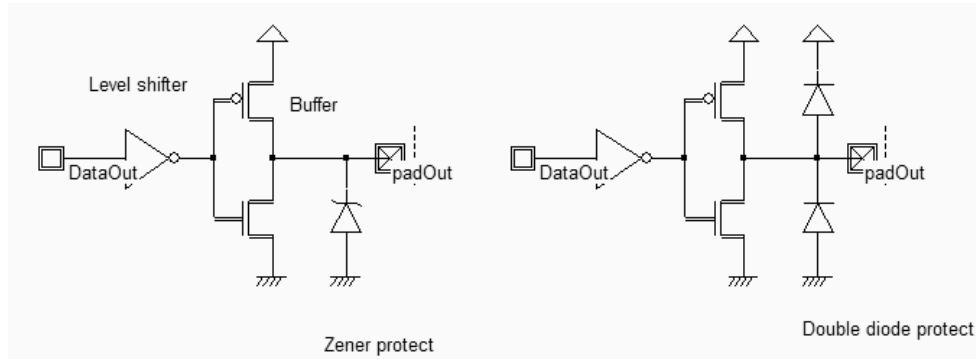


Figure 14-53: The output buffer design including the protection against electrostatic discharge (IOPadOut.SCH)

### 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. An immediate solution would consist in using a high voltage inverter as a level shifter. The signal *Data\_Out* has a 1.2V voltage amplitude, as shown in figure 14-54.

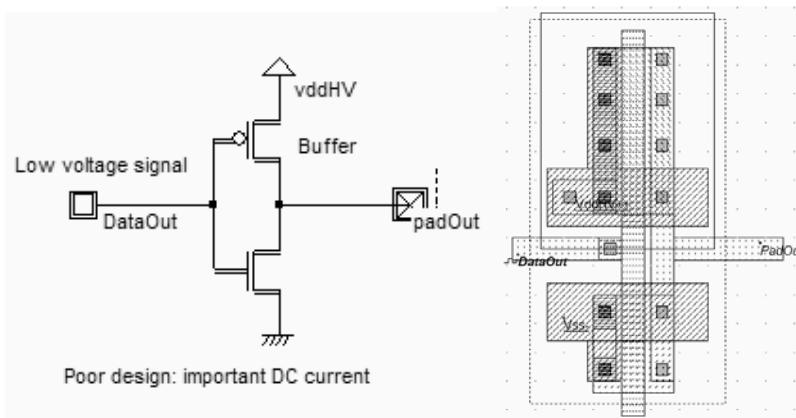


Figure 14-54: The inverter used as a level shifter (LevelShiftBad.MSK)

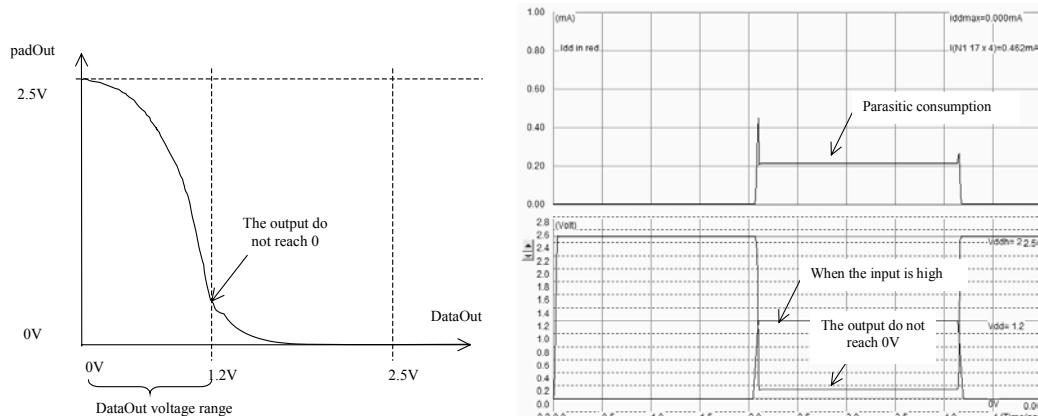


Figure 14-55: Analog simulation of the inverter showing the parasitic DC consumption (LevelShiftBad.MSK)

In the simulation shown in figure 14-55, the output signal *PadOut* is almost correct, except that the low level is not exactly zero. This is due to the input voltage limit to 1.2V. In the inverter characteristics (Figure 14-55 left), the input voltage 1.2V is not sufficient to obtain a "good" zero on the output. The important consequence of this incomplete switching is the large DC dissipation on a high level of *DataOut*, in the range of 200 micro-watts. Since this parasitic consumption appears at a low logic level, the sum of DC currents in the case of a 1000 pin integrated circuit would approach a fraction of ampere, which is not acceptable.

Figure 14-56 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.

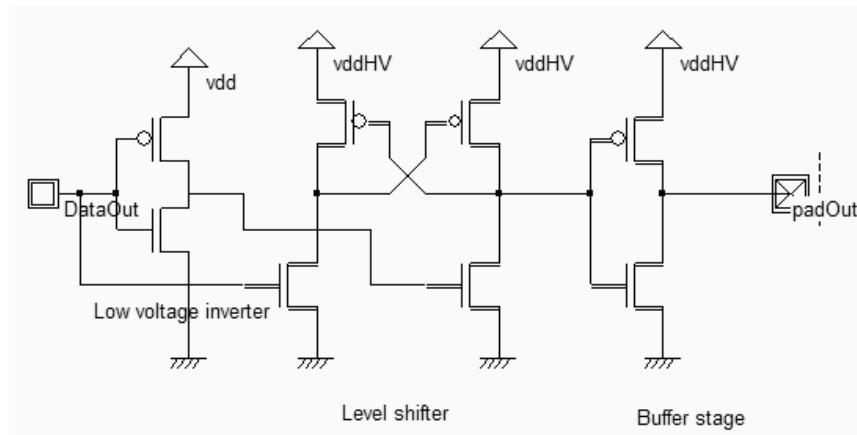
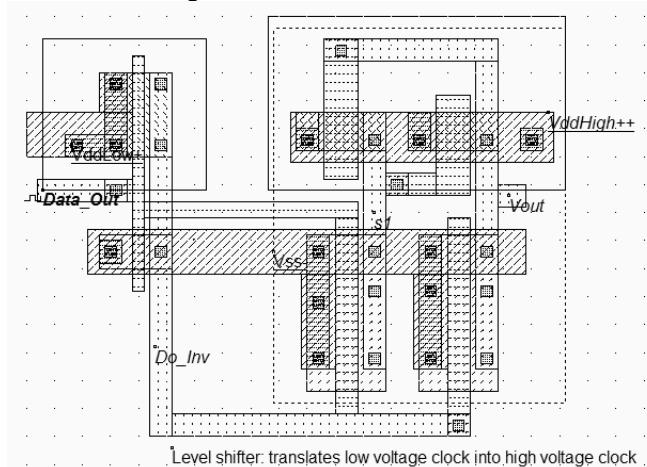


Figure 14-56: Schematic diagram of a level shifter (IOPadOut.SCH)

The layout of the level shifter is shown in figure 14-57. 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 14-57.



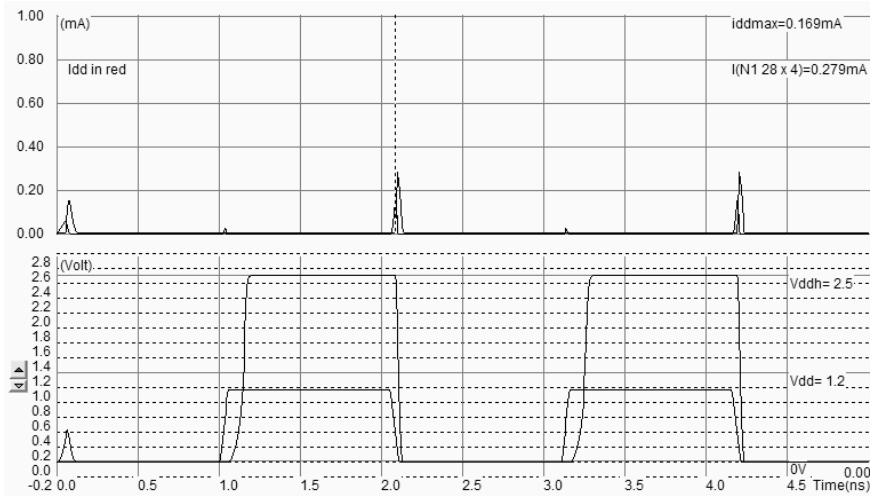


Figure 14-57: Layout and simulation of the level shifter (LevelShift.MSK)

### Output MOS devices

The role of the output buffer is to amplify the logic signal generated by the level shifter in order to switch at the appropriate speed, which depends on the target application. Usually, the buffer stage is built from several MOS devices in parallel, in order to achieve maximum Ion current of 2,4,8 or 16mA.

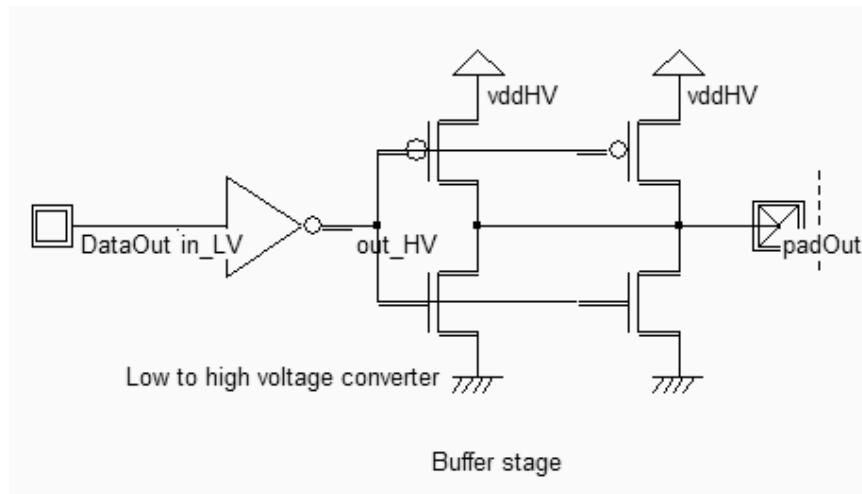


Figure 14-58: The schematic diagram of the buffer stage (IOPadOut.SCH)

The buffer stage is connected to the output of the high voltage inverter, called *Out\_HV* in figure 14-58. Usually, several MOS devices are connected in parallel to achieve a large current flow in order to drive efficiently the large capacitance of the output node. The MOS devices with parallel fingers can be generated by Microwind, using the MOS generator shown in figure 14-59. Assuming that the target current is 4mA, what we need is 2 fingers, the high-voltage MOS option, the minimum length 0.24 $\mu$ m for this type of device, and a width adjusted to 2.5 $\mu$ m. Notice that Microwind also generates a set of polarization contacts, appearing on the left side of the MOS device, for a good connection to ground. The maximum current of typical MOS devices (High voltage option) is listed in table 14-1.

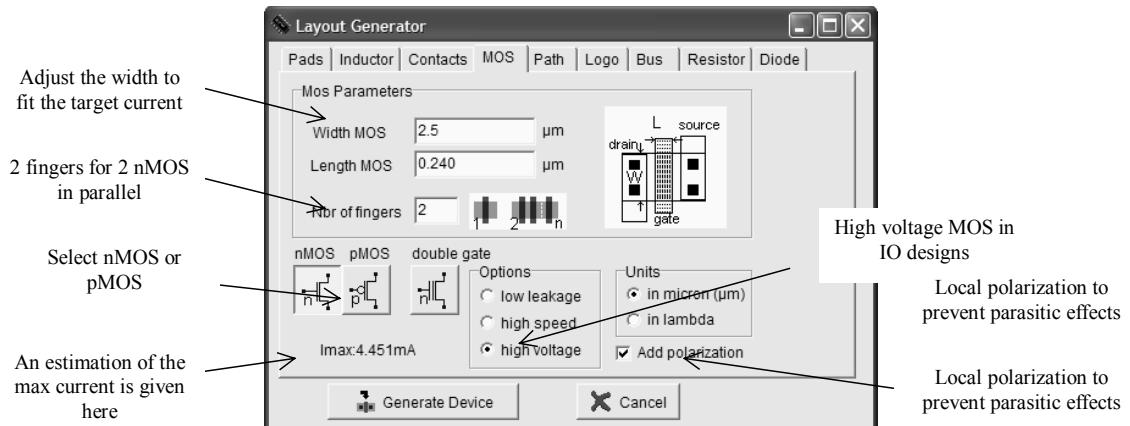


Figure 14-59: The MOS device generated for output pads is high-voltage, multiple fingers, and has a large width to produce the desired maximum current.

MOS width	MOS length	$I_{onN}$ (mA)	$I_{onP}$ (mA)
1.2 μm	0.24 μm	0.85	0.65
2 μm	0.24 μm	1.3	1.0
10 μm	0.24 μm	7.0	5.0
2 μm	0.5 μm	1.1	0.6
10 μm	0.5 μm	5.5	3.0

Table 14-1: Maximum current of basic nMOS and pMOS I/O devices (0.12 μm CMOS)

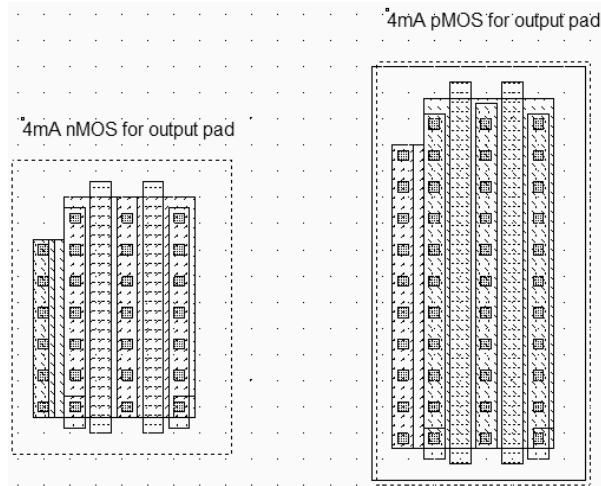


Figure 14-50: Generating multiple finger MOS devices for high current generation (IOPadMos.MSK)

The usual current drive of an output pad is 4mA. The nMOS and pMOS device that can switch 4mA are shown in figure 14-50. A high current drive is mandatory to ensure the rapid charge and discharge of the parasitic output node capacitance.

### Output Buffer Simulation

Let us assume that the output pad structure drives a 5pF load, which is quite low. The *DataOut* signal is shifted again by the level shifter to a 0..2,5V signal, and serves as a command for the MOS devices in parallel, which ensures the charge and discharge of the load (Figure 14-51). The virtual capacitor of 5pF is added to the layout using the capacitor icon in the palette. The simulation reported in figure 14-52 shows a charge and discharge of this capacitor within around 3nS, which is sufficient in the case of low speed applications. In the case of a high speed signal transport, as in memory bus, the current drive must be increased.

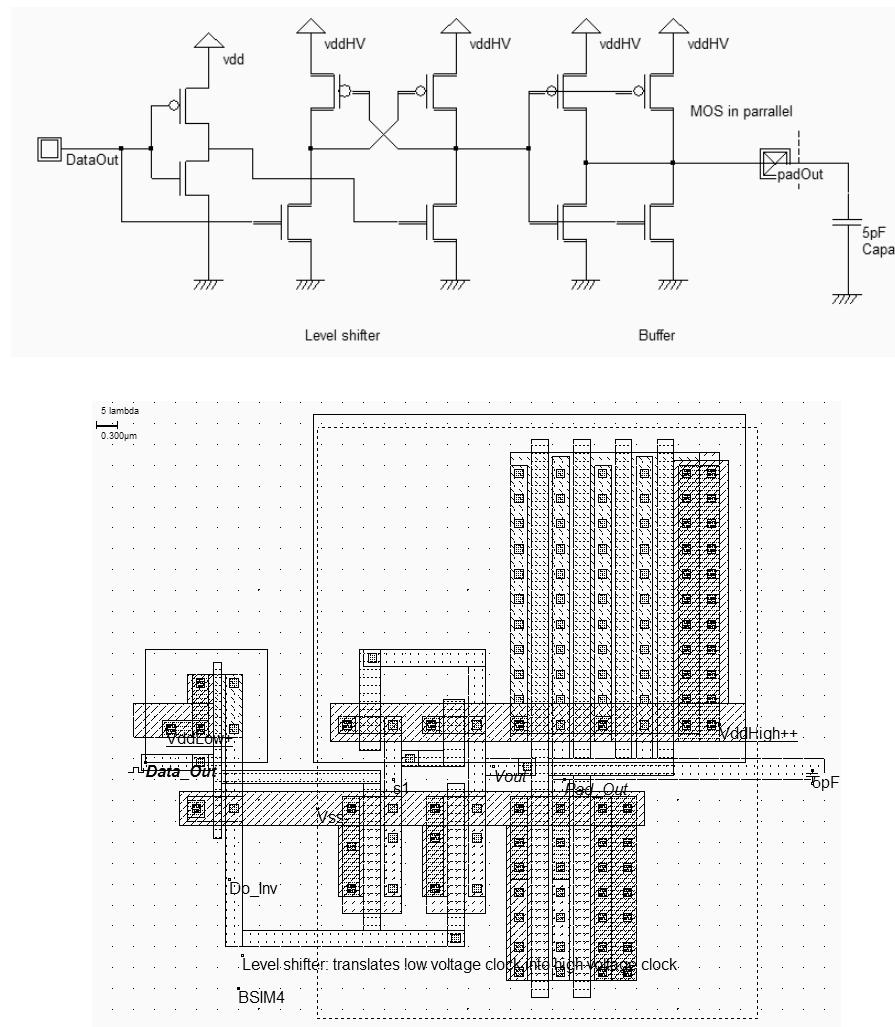


Figure 14-51: The schematic diagram of the output pad with the buffer stage, using MOS devices in parallel (LevelShiftBuff.MSK)

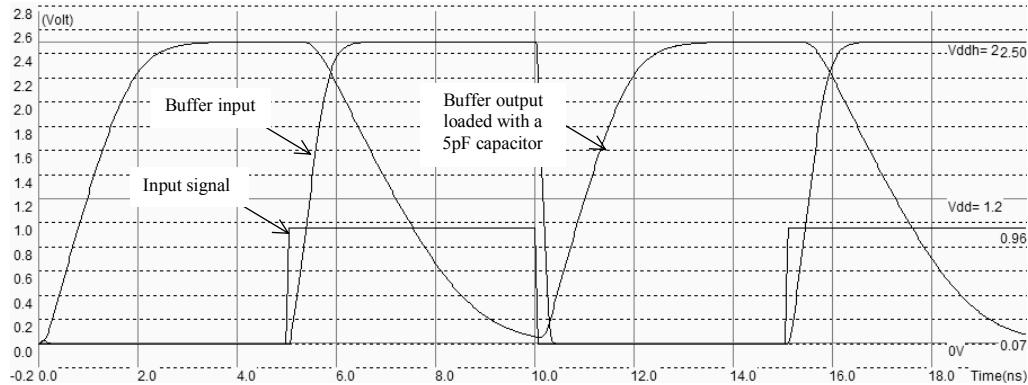


Figure 14-52: The analog simulation of the output pad loaded with 5pF (LevelShiftBuff.MSK)

### Output MOS protection to ESD

To improve the robustness of the output structure to electrostatic discharge and voltage overstress, the MOS layout can be improved. Firstly, the salicidation of the drain should be removed to increase the sheet resistance from the channel to the output pad and to enhance the ballasting resistance effect which has a positive influence on ESD protection performance. The ballasting resistance is mainly used to dissipate over voltage. The salicidation of the gate and source have no impact on ESD protection. The only region that should be resistive is the drain diffusion area connected to the pad. A specific option layer is added in the MOS design of figure 14-53 to increase the serial resistance which aims at protecting the output MOS device from parasitic stress.

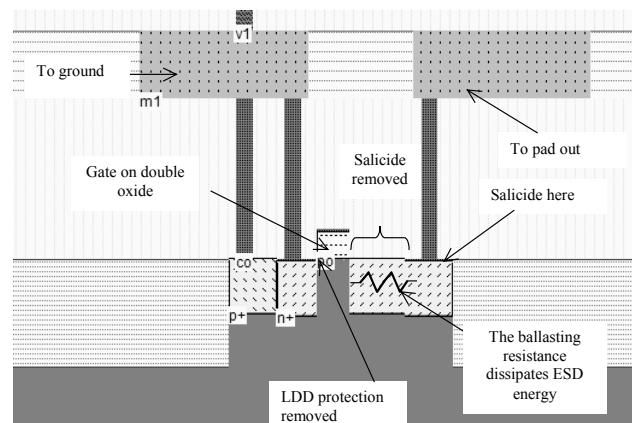


Figure 14-53: Cross-section of the MOS devices used in I/O pads (IOPadMos.MSK)

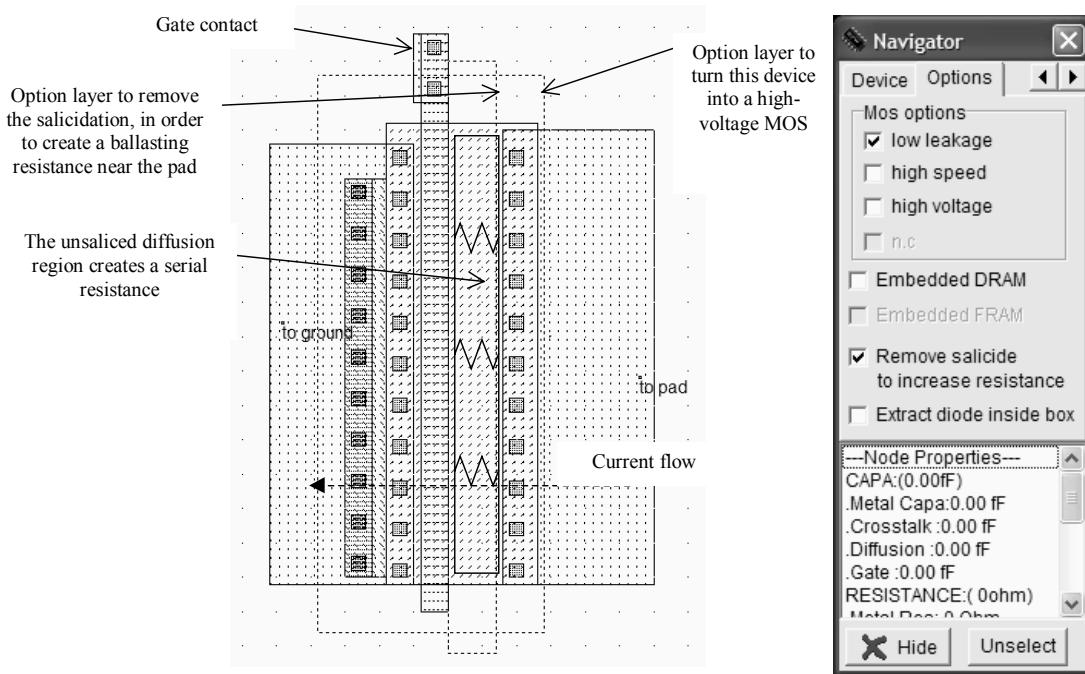


Figure 14-54: The particularities of MOS devices used in I/O pads: removed LDD, double gate oxide, and a ballast region (IOPadMos.MSK). Two option layers serve to declare the high-voltage and salicide removal options

The default salicidation causes a reduction by a factor of 10 of the serial drain resistance from the pad contact to the gate, resulting in significant protection degradation. An option layer is added to the layout covering the ballasting region, with the activation of the property **Remove Salicide** which blocks the titanium salicidation, and keeps the parasitic serial resistance intact.

### 3-state and Programmable Drive Buffer

The programmable drive buffer is used to adapt the current drive to the load, through a programming logic circuit. The circuit shown in figure 14-55 is able to switch the output signal *Pad\_Out* with a 2mA, 4mA or 6mA current.

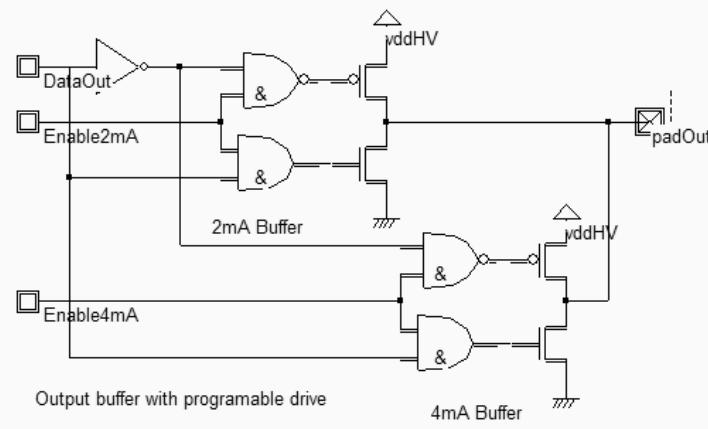


Figure 14-55: The programmable drive buffer (IOOutProgDrive.SCH)

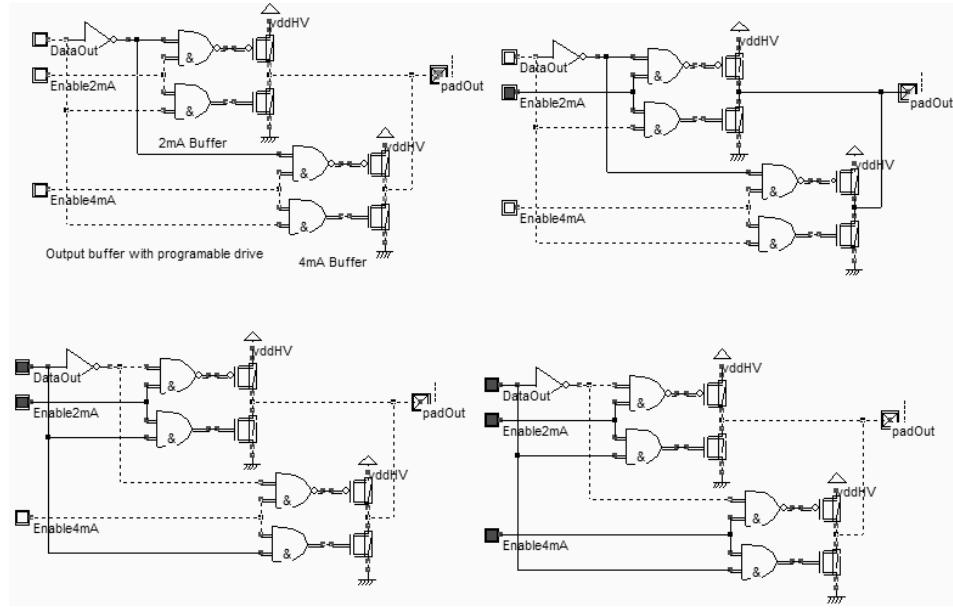


Figure 14-56: The simulation of the programmable drive buffer (IOOutProgDrive.SCH)

The nMOS and pMOS drivers are controlled independently. If all enables are inactive (Figure 14-56 top left) and all switches are off, the pad is in high impedance state. This is convenient to realize a 3-state buffer. If *Enable\_2mA* is asserted, the 2mA buffer is activated (Top right, bottom left). When both *Enable\_2mA* and *Enable\_4mA* are active, both buffers work in parallel, adding their currents to a total of 6mA. These specifications are summarized in table 14-2.

Enable_4mA	Enable_2mA	PadOut current
&0	0	0mA (3-state)
0	1	2mA
1	0	4mA
1	1	6mA

Table 14-2: The output current depends on the enable signals (IOOutProgDrive.SCH)

## 6. Pull-up, pull-down

It might be interesting to add the possibility of a weak tie to a defined voltage, particularly in the case of shared data buses. The role of the pull-up resistor shown in the schematic design of figure 14-57 is to recall the output node to *VDD\_HV* when the connection is floating. Using a MOS device as a  $10\text{K}\Omega$  switched resistance is efficient in terms of silicon area. The only drawback of this pull-up resistance is that the active logic level “0” is a little higher than 0.0V due to the leakage to *VDD\_HV*, which leads to a non negligible DC current consumption. Similarly, a high resistance nMOS device may create a weak tie to ground, which is equivalent to a pull-down device.

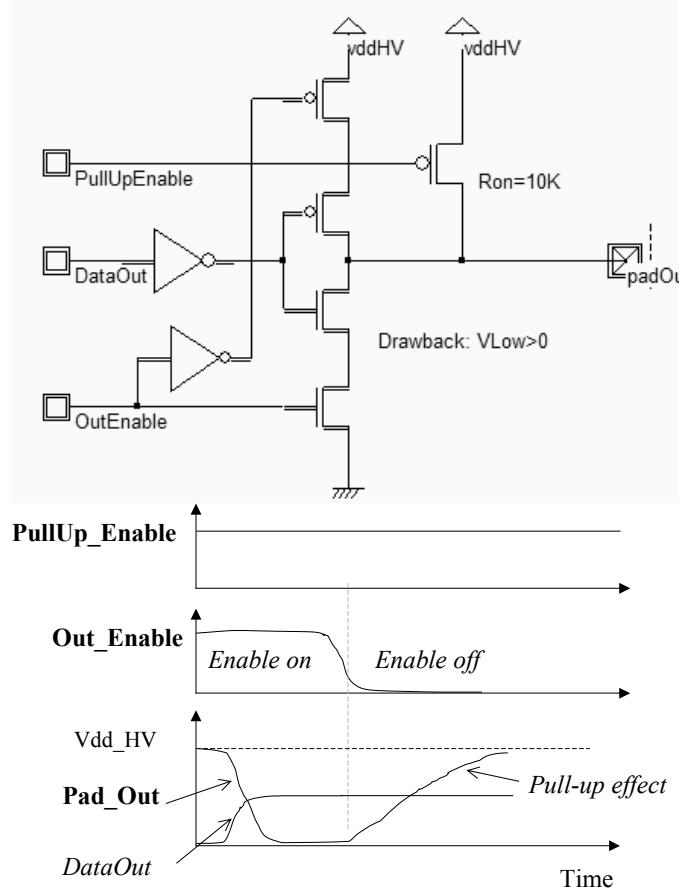


Figure 14-57: The 3-state output pad with a 10Kohm pull-up (IOPadOut.SCH)

### I/O pad

The input-output pad structure is a combination of input and output pad structure. The input-output pad contains one input stage together with one output stage, usually with extensive programmable functionality. In figure 14-58, the output stage can be turned off when the signal *Out\_Enable* is inactive. The pull up and pull down devices can be activated through *PullUp\_Enable* and *PullDown\_Enable* signals. More complex I/O pads may include programmable drives, as described previously.

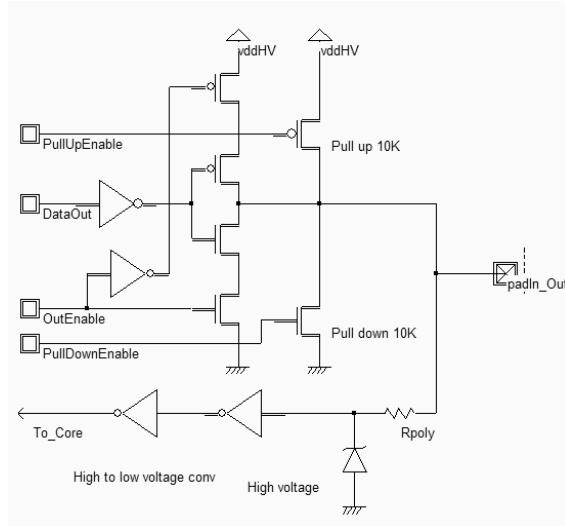


Figure 14-58. Design of an input-output pad (IOPad.MSK)

## 7. Low voltage differential swing

The main speed limitation in signal propagation is the time required to reach the logic state "1" or "0". Working at low supply voltage reduces the voltage swing and thus decreases the time required for the signal to reach the final logic area. Unfortunately, decreasing the logic cell supply also reduces the *Ion* current of the MOS devices which drives the output line. The differential swing logic circuit [Uyemura] uses two information signals *vin* and  $\sim\text{vin}$  instead of one, where  $\sim\text{vin}$  represents the logical complement of *vin*. The receiver works in differential mode, with signals that have a low swing amplitude.

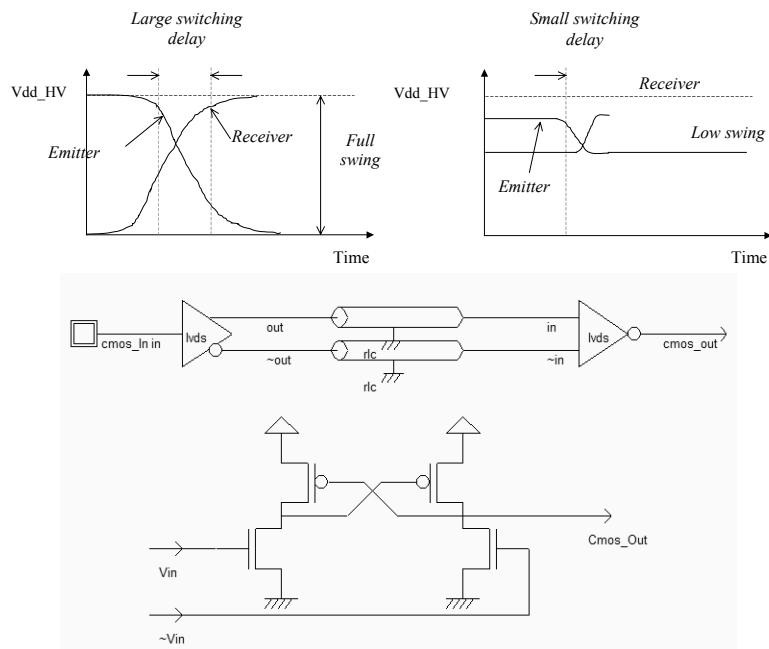


Figure 14-59: Low voltage differential swing logic to improve signal transport

Low voltage differential swing circuits operate at much higher frequencies than conventional CMOS drivers. However, LVDS circuits dissipate a significant amount of DC current, even when there is no switching activity. In the simulation of figure 14-60, a biasing current of  $100\mu\text{A}$  appears in the upper window. Combined with the fact that LVDS circuits require two interconnects instead of one, differential circuits are limited to very high speed functions such as fast data buses.

From a layout design point of view, the sizing of n-MOS and p-MOS devices has a strong influence on the LVDS buffer operation. The specification of the LVDS signal has a direct impact on the width ratio between n-MOS and p-MOS devices. In the case of a voltage swing of 400mV, from 0.4V to 0.8V, the p-MOS should be significantly smaller than the n-MOS to ensure reliable working conditions.

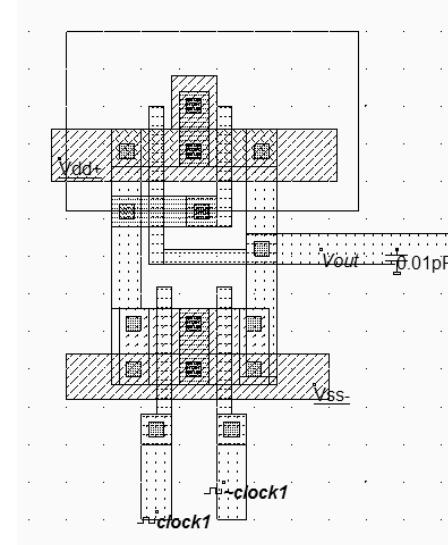


Figure 14-60: Low voltage differential swing layout (Lvds.MSK)

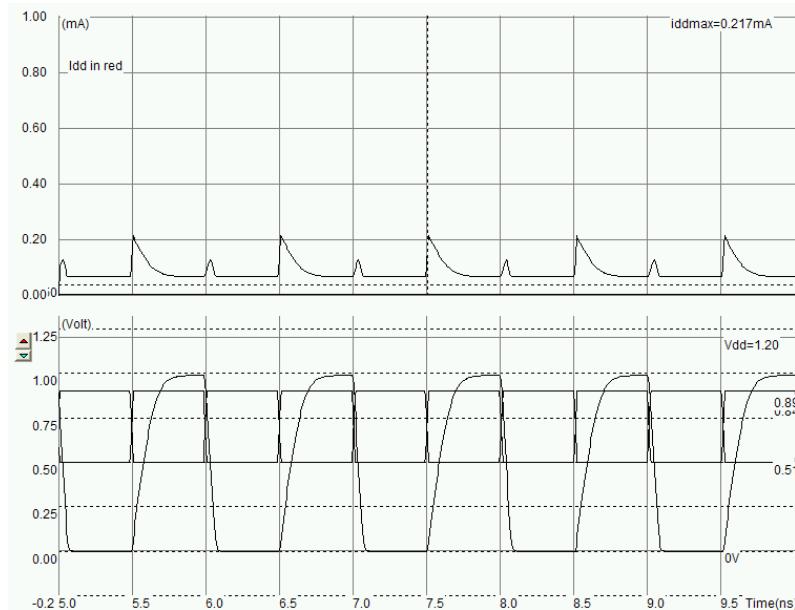


Figure 14-60: Low voltage differential swing simulation (Lvds.MSK)

## 8. Power Clamp

The power clamp is an efficient circuit that protect the logic core from oxide destruction, as a result of an electrostatic discharge appearing on the supply line. A simple circuit for this power clamp is proposed in figure 14-61, which has clear similarities with the ground gated MOS placed in input pads. The nMOS clamp is normally off, as  $R1$  ties the gate to a zero voltage, and  $C1$  is charged.

Assuming that an ESD pulse appears at the VDD supply, the nMOS transistor is turned on at a violent rise of VDD, which induces a positive voltage peak on  $Vg$  by capacitance coupling through  $C1$ . Consequently the MOS clamp is turned on, and the over voltage is limited. The MOS device width and the values for  $R1$  and  $C1$  are optimized to limit the core supply over-voltage below the gate oxide breakdown, without being sensitive to small VDD fluctuations. In  $0.12\mu\text{m}$ , the oxide breakdown of the 2.5V logic is around 6V, but decreases to 3.0V for the  $0.12\mu\text{m}$  devices.

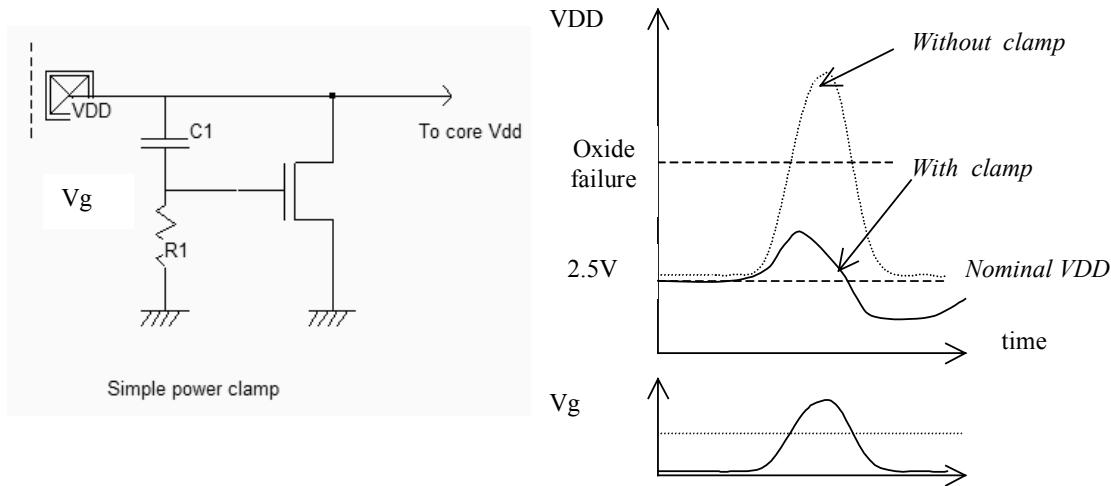


Figure 14-61. Design of a power clamp (PowerClamp.SCH)

## 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 14-62. 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.

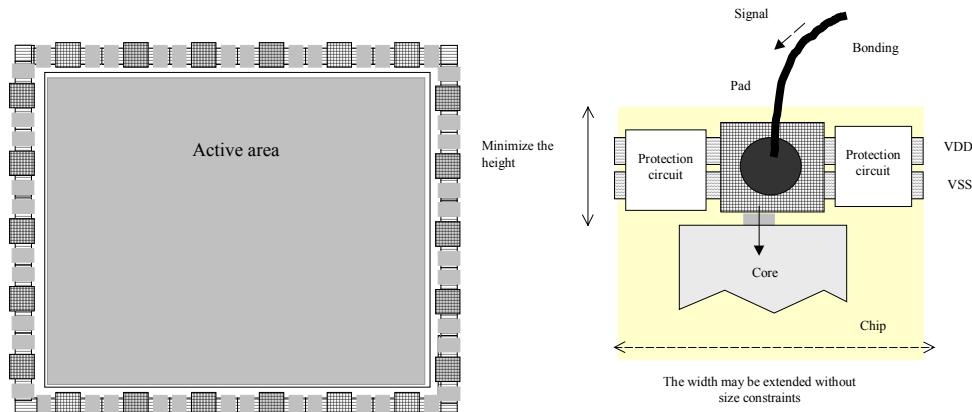


Figure 14-62 : Chip size fixed by the core

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 14-63. 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.

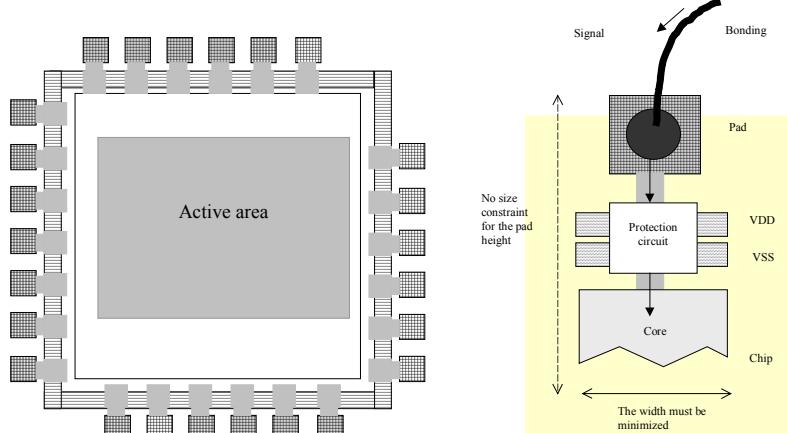


Figure 14-63. Chip size fixed by the number of pads

The spared silicon area may be avoided by using a double pair of I/O pads, as illustrated in figure 14-64. This attractive feature has been made available starting  $0.25\mu\text{m}$  technology. An example of a test-chip using a double pad ring is reported in figure 14-64, which corresponds to a CMOS  $0.18\mu\text{m}$  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.

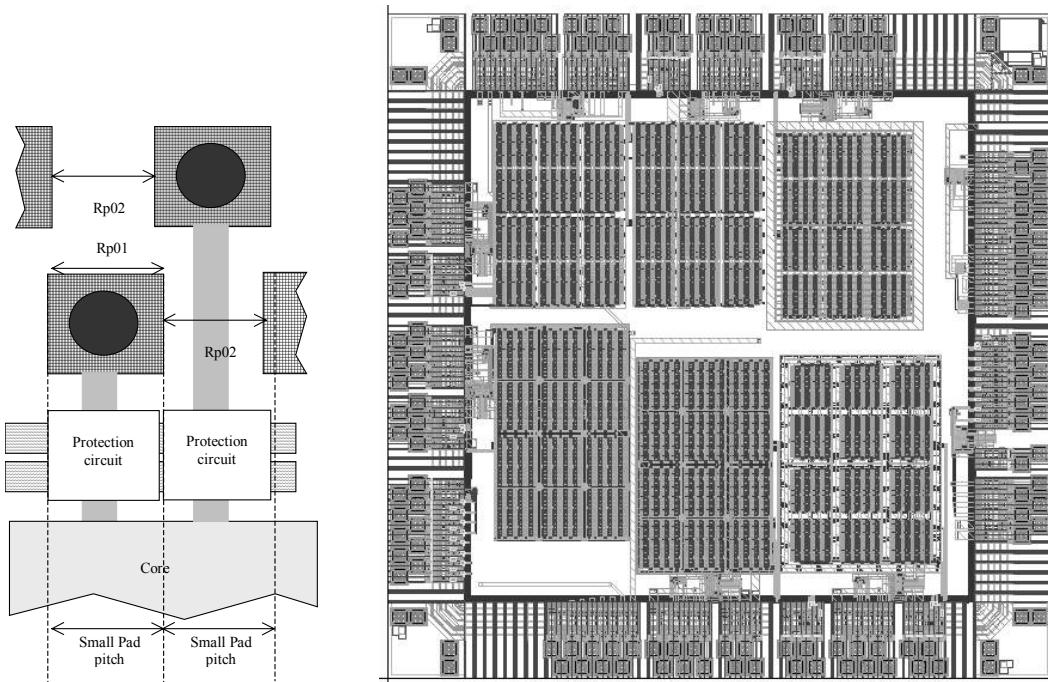


Figure 14-64. An example of a double ring test-chip in  $0.18\mu\text{m}$  technology (Courtesy of ST-Microelectronics)

There exists possibilities to place three rows of bonding pads in some circuits, such as for some state-of-the art processors and micro-controllers. Future trends may include the use of matrix of bonding balls all over the surface of the chip. This technique, called chip-scale packing, is already in use for some low complexity integrated circuits.

## 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 IBIS Modeling Data electronically 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.

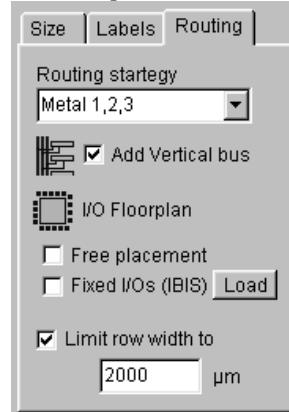


Figure 14-66. Controlling the I/O pin assignment by an IBIS description file

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**. The screen shown in figure 14-67 appears.

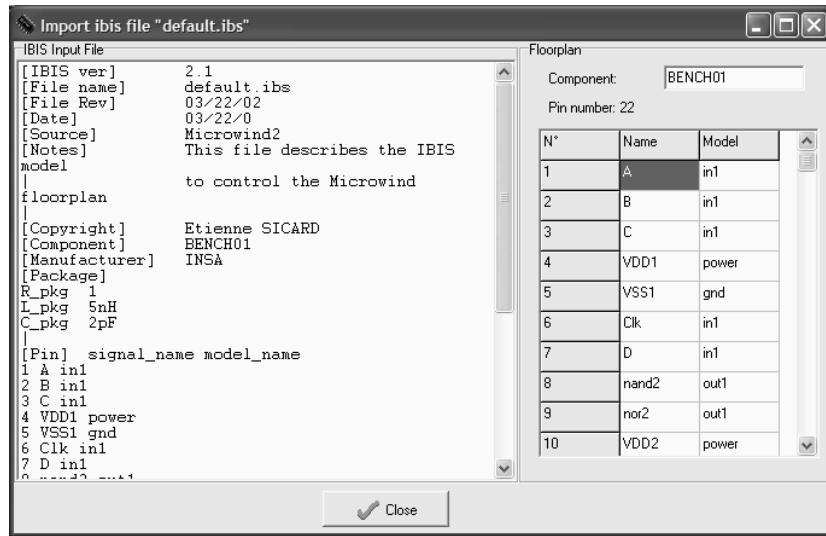


Figure 14-67. The IBIS description file loaded for controlling the pin assignment

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 14-3.

[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 14-3. The IBIS keywords understood by Microwind

At a click on **Generate Pad**, the layout of figure 14-68 is created, which corresponds to the list of pins declared in the IBIS file, as appearing in figure 14-67.

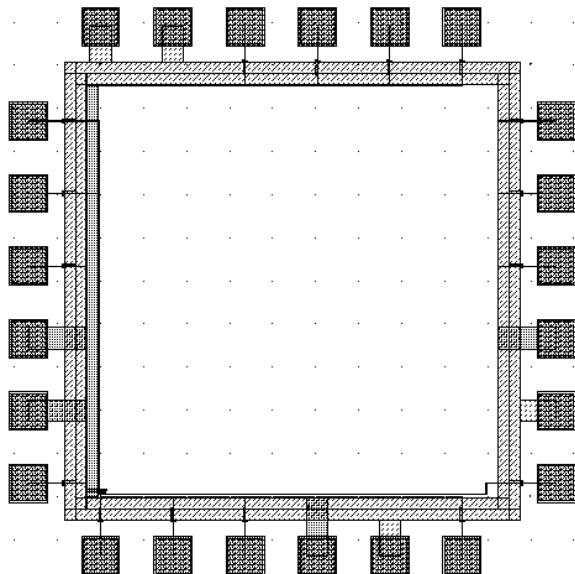


Figure 14-67. The I/O pad generation constructed using the IBIS file default.IBS

## 11. Connecting to the package

The integrated circuit is usually connected to the package by bonding wires or solder balls. In the first case, the bonding wires are made of gold, with a usual diameter of  $25\mu\text{m}$ . The wires build the link between the pads and the package leads. An example of package connection using bonding wires is shown in figure 14-68.

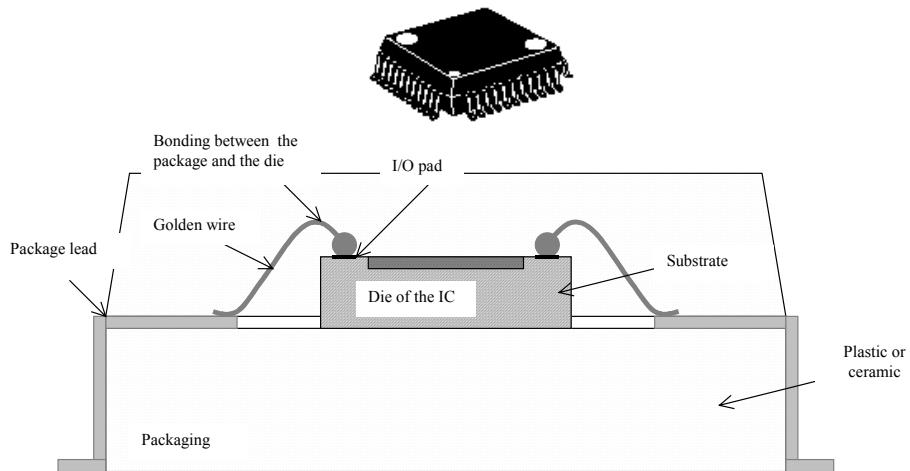


Figure 14-68: The structure of a Quad flat pack (QFP)

As the complexity of integrated circuits has constantly increased, a new type of link has been invented which creates in one single step all the connections between the die and the package. This technology, called ball grid array, was introduced some years ago and is now commonly used for integrated circuits with more than 200 pins. The cross-section of a ball-grid array and one integrated circuit example are proposed in figure 14-69. The die of the integrated circuit is flipped and connected using small solder balls to a specific package.

The package serves as a routing matrix from the IC pads (Pitch close to 100 $\mu\text{m}$ ) to the ball gate array (Pitch between 500 $\mu\text{m}$  and 2mm). The package is a complex network of very thin copper conductors embedded in an insulator. The BGA substrate may include from 2 to 6 metal layers to achieve the routing of general purpose signals and the distribution of power supply.

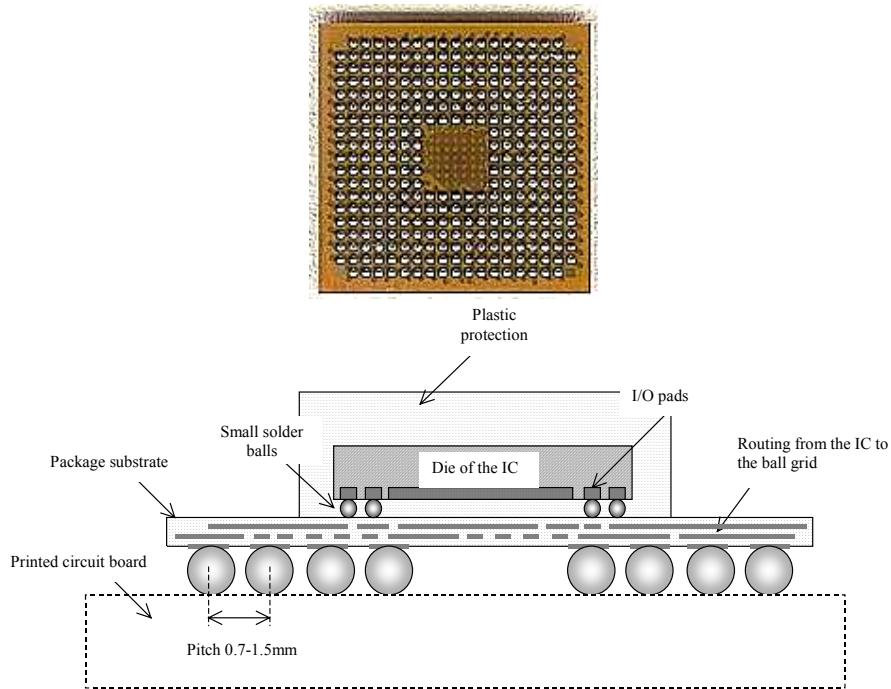


Figure 14-69: The assembly of the die to the package using micro balls

### Stacked Integrated Circuits

To minimize the surface of the electronic systems, the trend is to stack integrated circuits within a single package, also called System in Package <Glossary>. The benefit of this technique is mainly a much more compact system, at the price of a much more complex assembly, reliability and thermal dissipation issues. One example of stacked integrated circuits is shown in figure 14-70-a. Stacked integrated circuits are particularly attractive when mixing processors, memories, power management, actuators, sensors and radio-frequency elements. Due to cost and reliability issues, the stacking of heterogeneous integrated circuits may be preferred to a single all-integrated die solution.

The chip scale packaging (CSP <Glossary>), shown in 14-70-b, consists in connecting directly the chip to the printed circuit board without any intermediate package substrate. The die is flipped and electrically connected to the board via solder balls. The routing constraints in the printed circuit board are very severe as the ball pitch may be as low as 200 $\mu\text{m}$ .

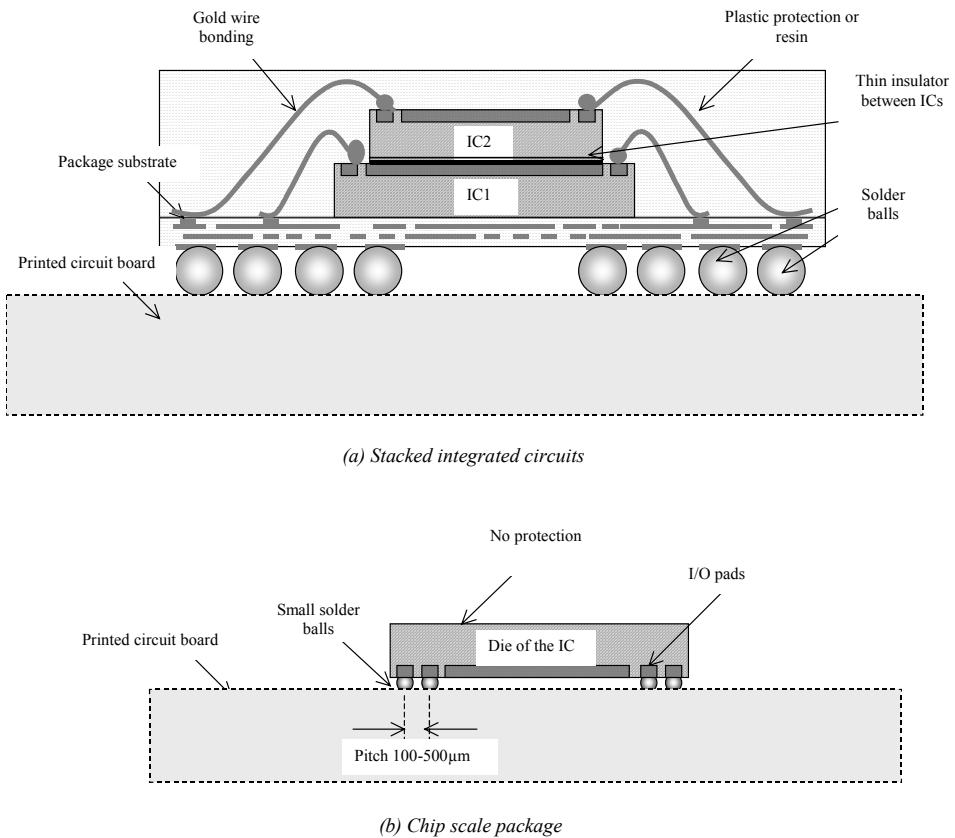


Figure 14-70: The stacking of two different dies in the same package (a) and the chip scale package (b)

## 12. Signal propagation between integrated circuits

The communication between two integrated circuits raises a set of issues that are briefly introduced in this paragraph. The emitter signal comes from a logic circuit *IC1* (figure 14-71). To be exported outside the integrated circuit, the signal is buffered by an inverter. The path between two integrated circuits is a conductor that can be a few millimeters to several meters long. Typically, the distance between two ICs in standard printed circuit boards is of the order of some centimeters. Finally, the signal enters a buffer of the receiver situated in *IC2*.

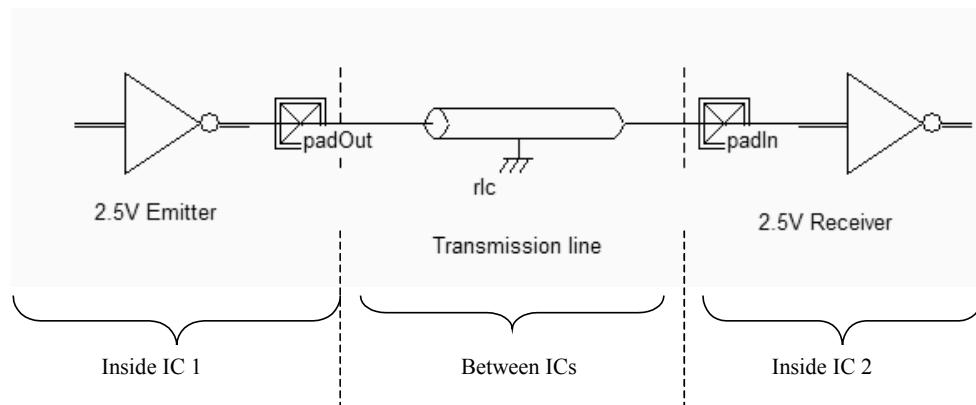


Figure 14-71: The signal propagation between integrated circuits

The transmission line effect can be seen in the simulation of figure 14-72. The original signal is quite clean and looks like a standard square waveform. However, the transport of the signal outside the integrated circuit, through the package, all the way along the interconnect and then inside the other integrated circuit has a significant impact on the shape of the resulting signal: propagation delay, overshoot and ringing. These effects are illustrated in figure 14-72

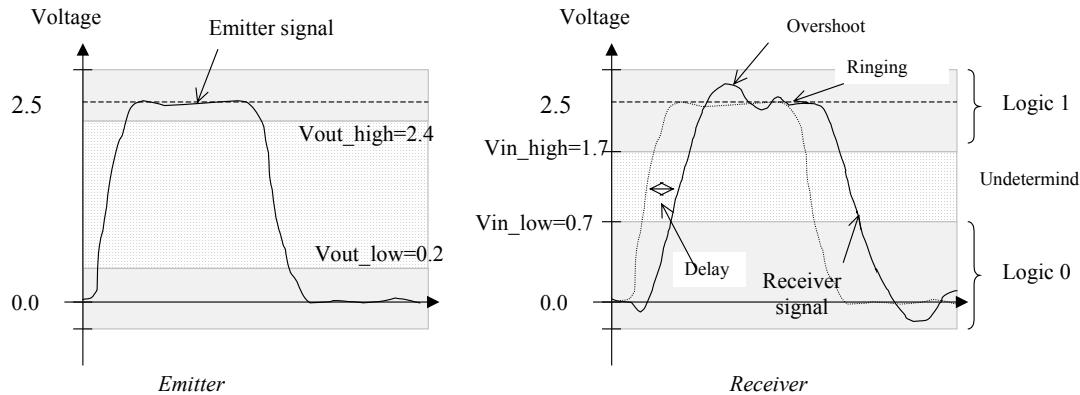


Figure 14-72: The signal is modified by the propagation within the package and interconnects.

Some limits are defined in figure 14-72 for low and high logic levels. These levels differ for the emitter and the receiver. The delay finds its origin in the flight time, linked to the light speed. In a normal epoxy printed circuit board (Also called FR4, with a permittivity of 4.2), the signal propagates according to equation 14-1 :

$$v = \frac{c}{\sqrt{\epsilon_r}} \quad (\text{Equation 14-1})$$

The resulting propagation time is :

$$t_{prop} = \frac{\sqrt{\epsilon_r}}{c} = \frac{2.1}{300,000 \text{ km/s}} \cong 140 \text{ mm/ns}$$

The overshoot and ringing are due to the inductive and capacitive behavior of the interconnection situated between the emitter and the receiver. The combination of C and L provokes resonance effects. As each portion of conductor has its own inductance and capacitance, several resonance effects may be observed at different frequencies.

There are various standards for input/output supply voltages, as shown in the table 14-3. The TTL standard and the low voltage TTL standard (LV-TTL) work with non-symmetrical low and high levels. All CMOS standards are almost symmetrical.

Standard	VSS (V)	VDD (V)	Vin low	Vin high	Vout low	Vout high
TTL	0.0	5.0	0.8	2.0	0.4	2.4
LV-TTL	0.0	3.3	0.8	2.0	0.4	2.4
LVC-MOS2V5	0.0	2.5	0.7	1.7	0.2	2.1
LVC-MOS1V8	0.0	1.8	0.63	1.17	0.45	1.35
LVC-MOS1V2	0.0	1.2	0.43	0.78	0.30	0.9
LVC-MOS1V0	0.0	1.0	0.35	0.65	0.25	0.75

Table 14-3: The format of some basic I/O standards in TTL and CMOS integrated circuits

The main limitation of a conventional I/O is the full swing of the voltage output, at the cost of a significant delay in the signal switching at the far end of the receiver. A very interesting idea consists in limiting the full voltage swing of the signal (Some volts) to only hundreds of mV. The flight time linked to the light speed remains unchanged but the charge and discharge time of the complete interconnect is significantly reduced.

Standard	VDD (V)	Vref (V)	Vin low	Vin high	Vout low	Vin high
SSTL3	3.3	1.65	Vref-0.2	Vref+0.2	0.90	2.10
SSTL2	2.5	1.25	Vref-0.15	Vref+0.15	0.65	1.85
SSTL18	1.8	0.9	Vref-0.125	Vref+0.125	0.40	1.30

Table 14-4: The format of high speed differential I/O standards used in recent CMOS integrated circuits

Details of small swing voltage standards (SSTL) are given in table 14-4. The SSTL circuit at work is illustrated for a 2.5V voltage supply (SSTL2), and a 1.25V voltage reference  $V_{ref}$ . A high speed bus using SSTL drivers is shown in 14-73. Notice the four SSTL drivers and receivers, plus the voltage reference  $V_{ref}$ . The input data must be higher than  $V_{out\_high}$  or lower than  $V_{out\_low}$ , on the near end, close to the emitter.

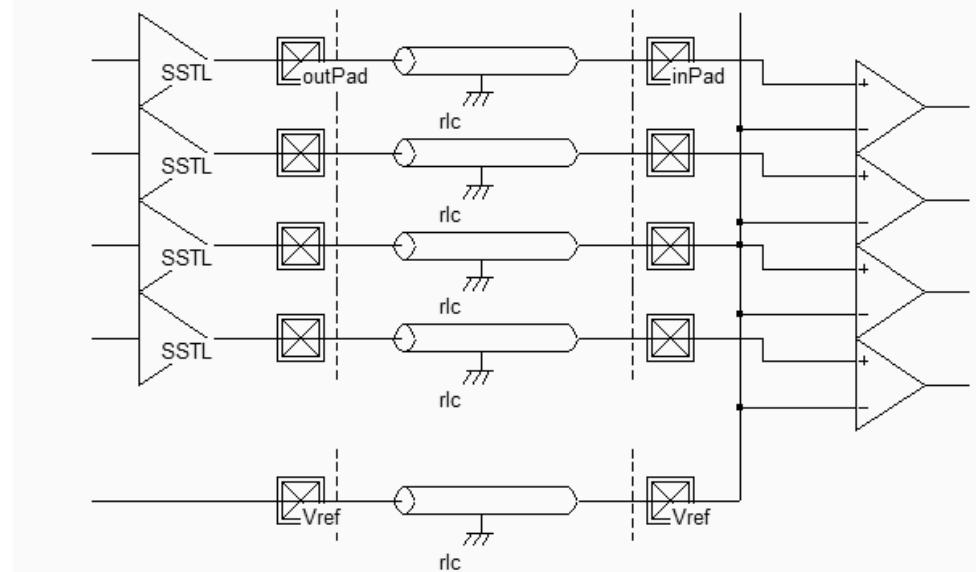


Figure 14-73: The SSTL bus used for double data rate RAM interfaces with high speed micro-processors (Iosstl.SCH)

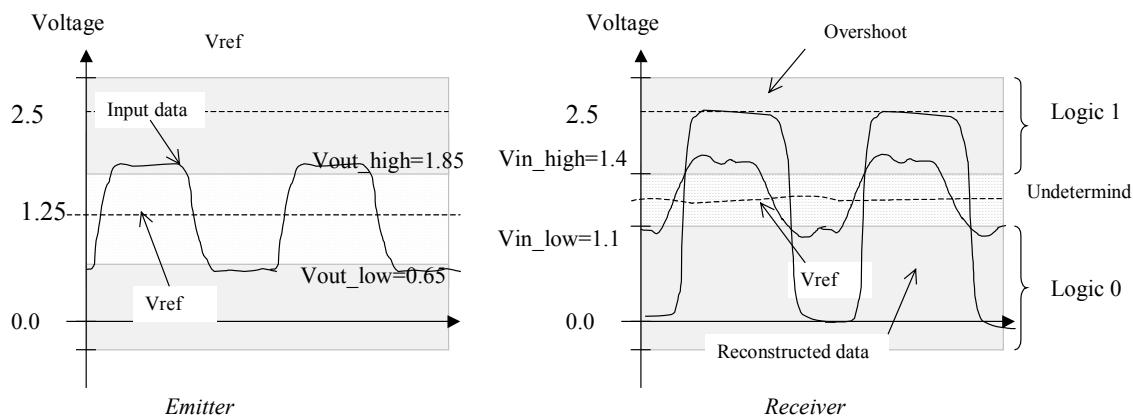


Figure 14-74: Typical waveforms for the SSTL2 emitter and receiver structure

The received signal is considered as a 1 if higher than  $V_{in\_high}$  (Only 150mV higher than  $V_{ref}$ ), and considered as a 0 if lower than  $V_{in\_low}$  (at least 150mV lower than  $V_{ref}$ ). These margins appear on the right of figure 14-74. The small voltage swing enables faster data rates: the SSTL2 input/outputs are used for double data rate memories with up to 800Mbit data rate.

## 13. Conclusion

This chapter has described the input/output interfacing of the integrated circuit. The power supply network has been described, with emphasis on metal grid strategy. Then, several aspects of the electrostatic discharge prevention have been addressed, and the basic elements for the input protection circuit have been detailed. Concerning the output structures, the buffer architecture, the 3-state option and programmable drive design principles have been presented. A brief presentation of IBIS has also been provided, followed by some insights in the connection between the integrated circuit and the external world. Quad flat pack, ball grid array, chip scale and stacked packages have also been described. Finally, the main standards for low voltage and small swing input/output signals have been listed.

## Exercises

### Exercise 14-1

Design a clamp circuit sensitive to a 100V pulse. The capacitor should be a coupling capacitor, and the simulation should be performed with the option "With crosstalk". Compare the performances of the clamp circuit and the Zener diode protection circuit proposed in this chapter.

### Exercise 14-2

Design a programmable I/O pad, according to the schematic diagram of figure 14-57, with 2,4 or 6mA drive capabilities.

### Exercise 14-3

Build a differential emitter/receiver circuit with a 2Gb/s bandwidth. What is the critical distance to perform the detection correctly?

### Exercise 14-4

Evaluate the I/O density per mm<sup>2</sup> for QFP, BGA, µBGA and CSP.

### Exercise 14-5

Build a SSTL bus transfer system for long interconnects on-chip. The interconnect may be routed in metal 6, and correspond to 10mm. Use the command **Edit → Generate → Metal Bus** to generate long bus lines automatically.

## References

[Dabral] Sanjay Dabral, Timothy J. Maloney, « Basic ESD and I/O Design », 1998, John Wiley and sons, ISBN 0-471-25359-6

[Hastings] Alan Hastings "The Art of Analog Layout", Prentice Hall, 2001, ISBN 0-13-087061-7

[Bellaouar] Abdellatif Bellaouar, Mohamed I. Elmasry "Low-power Digital VLSI Design", Kluwer Academic Publishers, 1996, ISBN 0-7923-9587-5

[Wang] Albert Z.H. Wang "On-Chip ESD protection for Integrated Circuits", An IC Design Perspective, Kluwer Academic Publishers, 2002, ISBN 0-7923-7647-1

[Ibis] [www.eia.org/ibis](http://www.eia.org/ibis)

# 15

## Silicon On Insulator

### 1. Introduction

The use of Silicon-On-Insulator (SOI) technology is bringing interesting new possibilities compared to conventional bulk technology. This chapter highlights the extra performance and advantages offered by SOI, as well as the limiting parasitic effects. Performance improvements concern the power consumption and the commutation speed. In the best case, the SOI technology may cut the power consumption nearly by half, with speed improvements close to 30%. The speed improvement itself is equivalent to about two years of progress in bulk CMOS technology. The insulator material used in SOI is a buried SiO<sub>2</sub> layer, illustrated in figure 15-1.

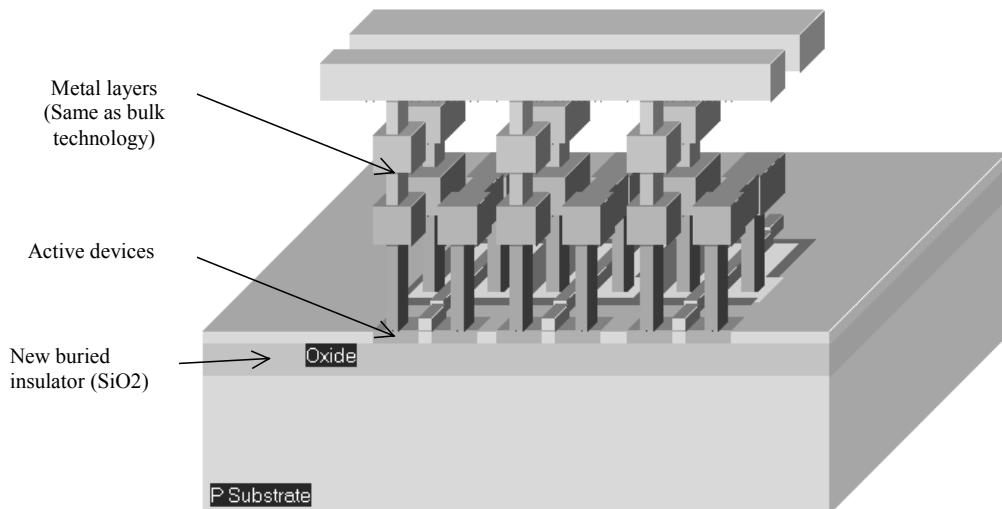


Figure 15-1 : 3D View of SOI ring inverter showing the SiO<sub>2</sub> buried layer(inv3Soi.MSK)

In fact, the SOI technology has been available for more than 20 years, but its applications were mainly restricted to space and army due to very low sensitivity to radiation. The road to a commercial use of SOI still faces several issues: one is the cost of the substrate, which is 5 to 10 times the cost of a bulk wafer, an other is the need to train designers to specific design techniques and rules, as the behavior of a SOI MOS device differs slightly from the bulk MOS device. Although the MOS device fabrication is slightly modified, the fabrication of the metal interconnects is identical to that of the bulk CMOS process.

### The SOI Substrate

SOI refers to placing a thin layer of silicon on top of a silicon oxide insulator, as illustrated in figure 15-2. The transistors are built on top of this thin layer of SOI. A  $0.12\mu\text{m}$  SOI technology, namely **soi012.RUL** is available in Microwind. It enables the comparative simulation with the corresponding bulk technology (**cmos012.rul**, the standard CMOS  $0.12\mu\text{m}$  technology).

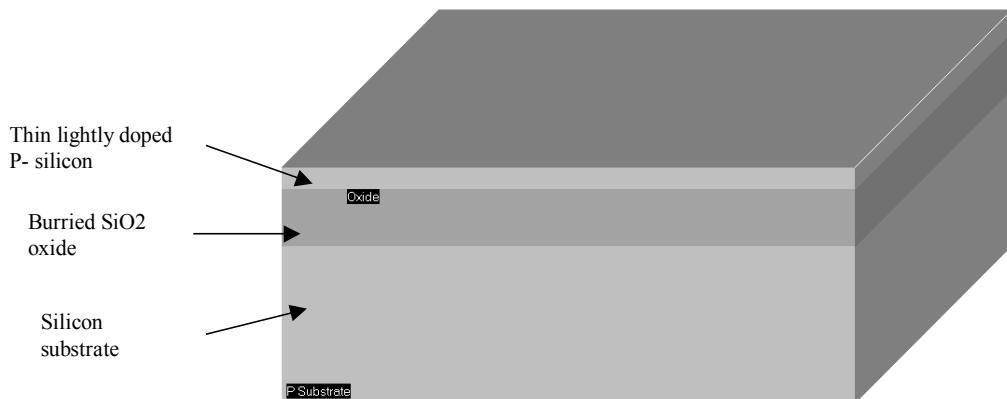


Figure 15-2 : 3D View of SOI ring inverter showing the SiO<sub>2</sub> buried layer(inv3Soi.MSK)

The basic idea is that the SOI layer will reduce the parasitic junction capacitance of the switch, so it will operate faster. Every time the transistor is turned on, it must first charge all its internal capacitance before it can begin to switch. Among these parasitic capacitances are the junction capacitance  $C_{sb}$  and  $C_{db}$ , which are strongly reduced by the silicon dioxide, as described in the two-dimensional cross-section of figure 15-3. The thicker the SiO<sub>2</sub> oxide, the smaller the parasitic capacitance. The typical insulator thickness is between 200 and 500nm. In the SOI CMOS  $0.12\mu\text{m}$  technology provided with Microwind, the dielectric thickness is 300nm, and the silicon thickness on the top of the insulator is 150nm. As illustrated in the figure, the junction capacitance  $C_{sb}$  and  $C_{db}$  are significantly reduced, which is meant to speed up the switching of devices.

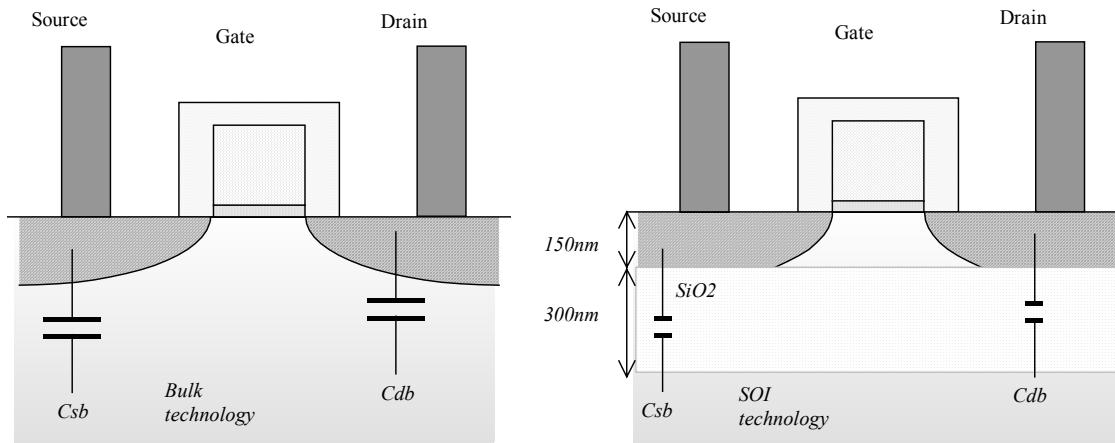


Figure 15-3: The junction capacitance between the source and bulk is almost eliminated in the case of SOI.

### Low voltage Operation

An important feature of SOI devices is the steeper sub threshold slope due to a reduction of the substrate body effect. Typical sub threshold slope factors (NFACT in the BSIM4 menu) are close to 1.0 for SOI devices, as compared to 1.5 for bulk devices. For a given  $I_{off}$  current, the SOI circuit may have a much smaller threshold voltage, which means that the circuit can operate at lower supply.

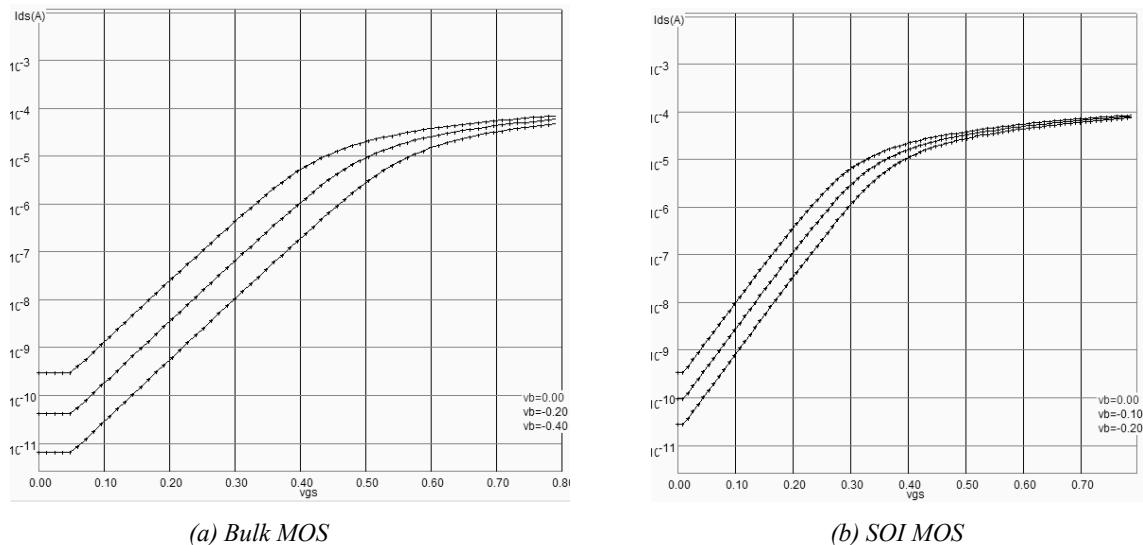


Figure 15-4: The steeper sub-threshold slope enables low voltage and low power operations.

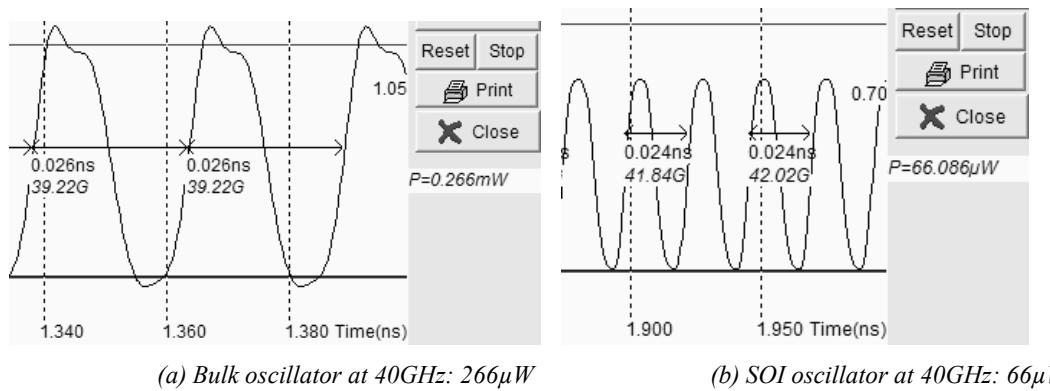


Figure 15-5: The lower  $I_{off}$  current and steeper sub-threshold slope enables low voltage and low power operations.

Recall that the power is proportional to the total circuit capacitor and the square of the supply voltage. This means that SOI circuits are very good candidates for low power operations as the parasitic capacitance is reduced and the supply voltage can be lowered. Considering the ring oscillator with three inverters, we obtain a 42GHz oscillation at a supply voltage of 0.7V in SOI technology, rather than 1.2V in bulk technology (Figure 15-5). The power gain is approaching a factor of 4.

Furthermore, the lower sub threshold combined with a stepper slope is of key interest for analog circuits, which can provide the same functionality and approximately the same bandwidth performances, with a lower power consumption.

### Increased density

One important feature of the SOI technology concerns the CMOS cell density increase thanks to relaxed design rule constraints between N+ and P+ diffusions. In CMOS bulk technology, the n-channel device is separated from the p-channel device by at least 12 lambda. In SOI technology, the design rule drops to only 2 lambda, as shown in figure 15-6.

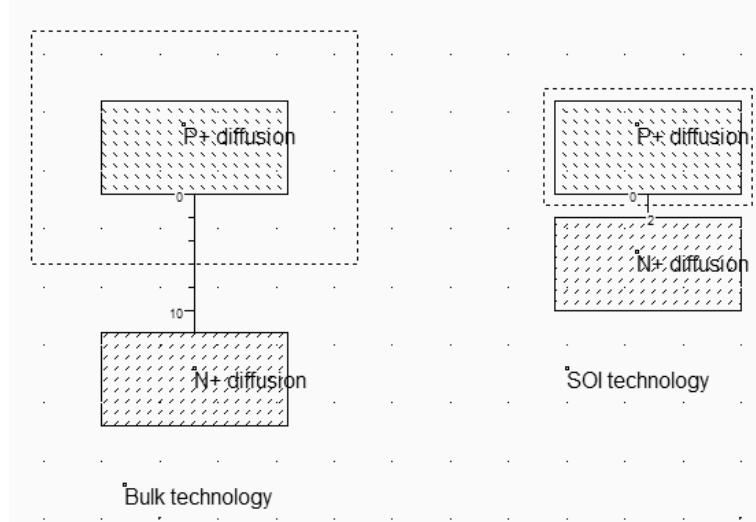


Figure 15-6: The increased density due to relaxed design rules between N+ and P+ diffusions (SOIDiffusion.MSK)

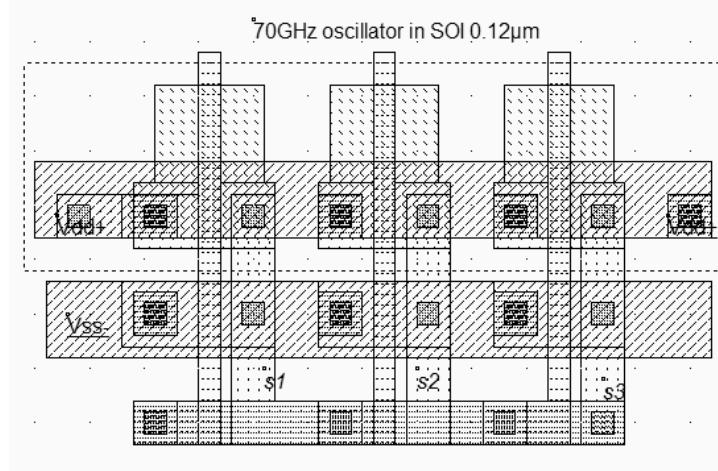


Figure 15-7: The ring oscillator in SOI technology (Inv3SOI.MSK)

Consequently, the layout implementation of a CMOS cell is more compact as the nMOS and pMOS devices almost touch each other. As an example, the 3-inverter ring oscillator in SOI technology is 20% more compact than the bulk version (Figure 15-7), for an identical sizing of nMOS and pMOS devices.

### Increased Operating Frequency

The comparison between the SOI ring inverter and the bulk ring inverter is given in figure 15-8. We observe a very significant gain in terms of speed, nearly 80% in this case. In bulk technology, the 3-inverter oscillator (Inv3.MSK) is operating near 19GHz, when using the model BSIM4. In SOI technology, the same inverter oscillates (Inv3Soi.MSK) around 35GHz.

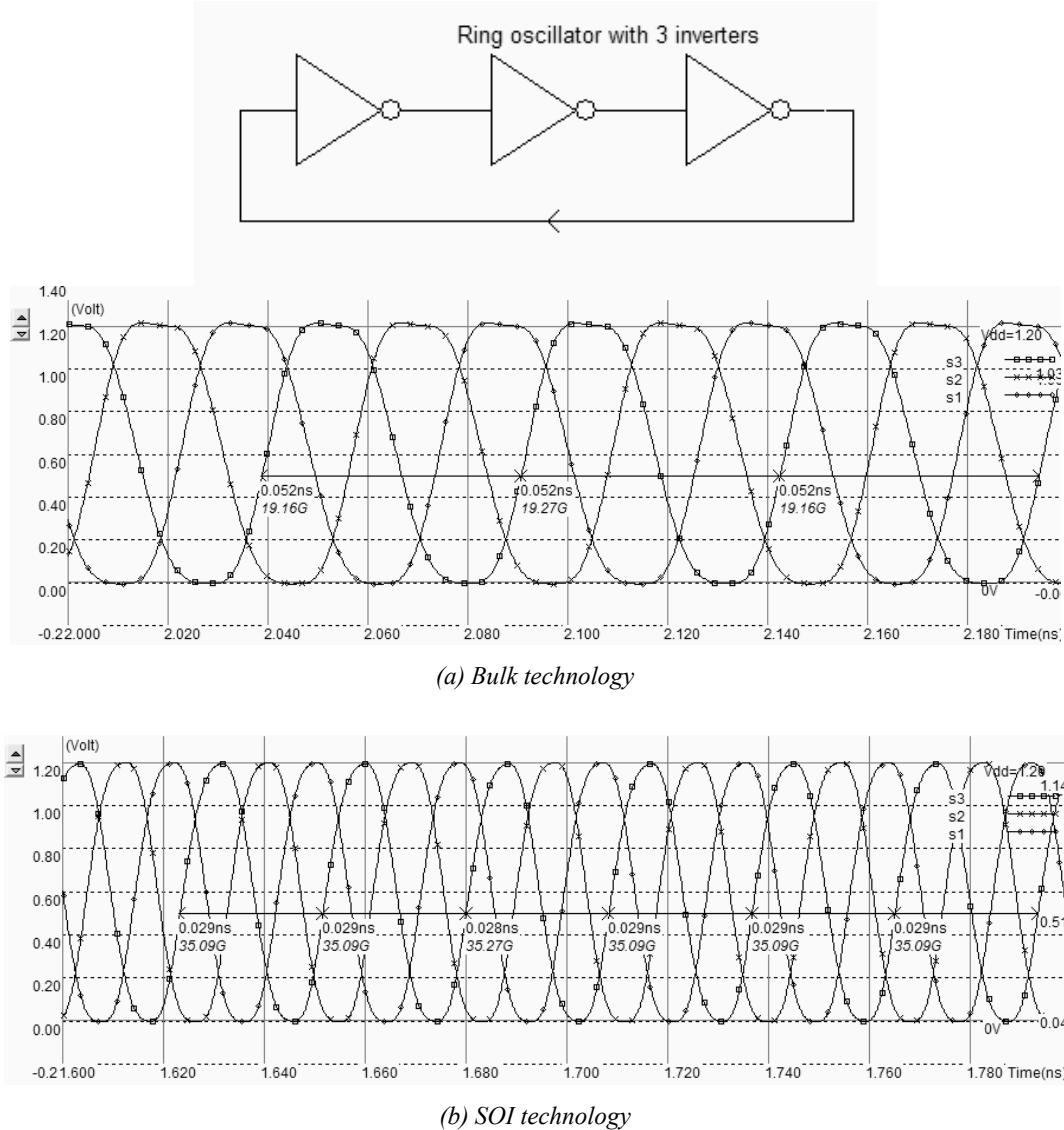


Figure 15-8: The simulation of the ring oscillator in bulk and SOI technologies (Inv3.MSK, Inv3SOI.MSK)

The very important frequency increase observed in figure 15-8 mainly finds its origin in the decreased parasitic capacitance of the drain junctions of the MOS devices. As no long interconnect was needed in this design, the reduction of capacitance has a very clear impact on the final frequency.

Furthermore, the maximum current available with the SOI MOS is 20% higher than for the bulk version, due to a particular undesired effect (The kink effect) described later in this chapter.

### Decreased couplings

The oxide isolation has a positive impact on the noise immunity between blocks. One of the main contributors to noise is the substrate in bulk technologies. A high power, high frequency circuits such as a power amplifier may inject a fraction of their switched energy to the substrate, which may parasitically affect sensitive parts such as amplifier inputs or analog to digital converters. The insulator provided in the SOI technology has very efficient decoupling capabilities which facilitate the embedding of incompatible functionalities within the same silicon substrate (Figure 15-9).

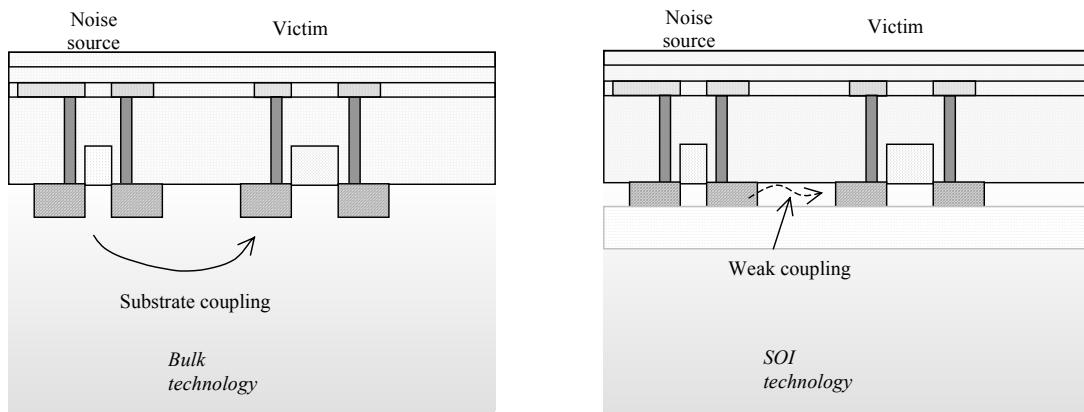
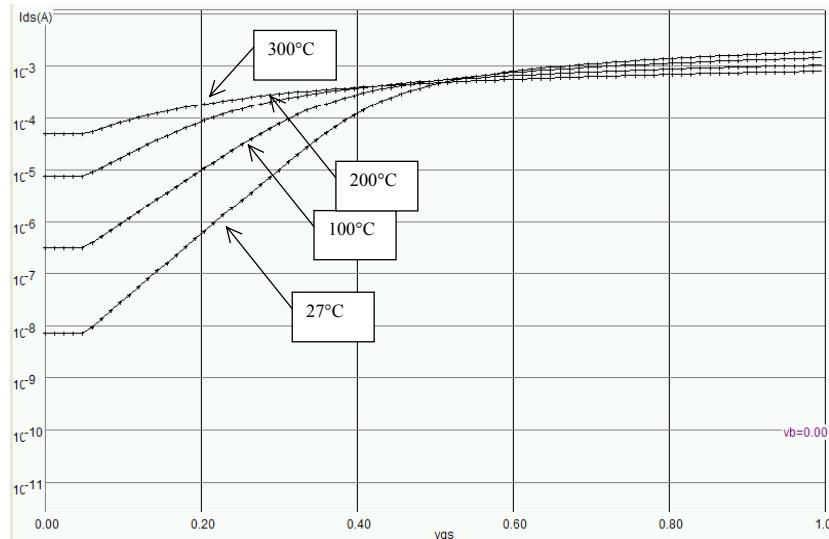


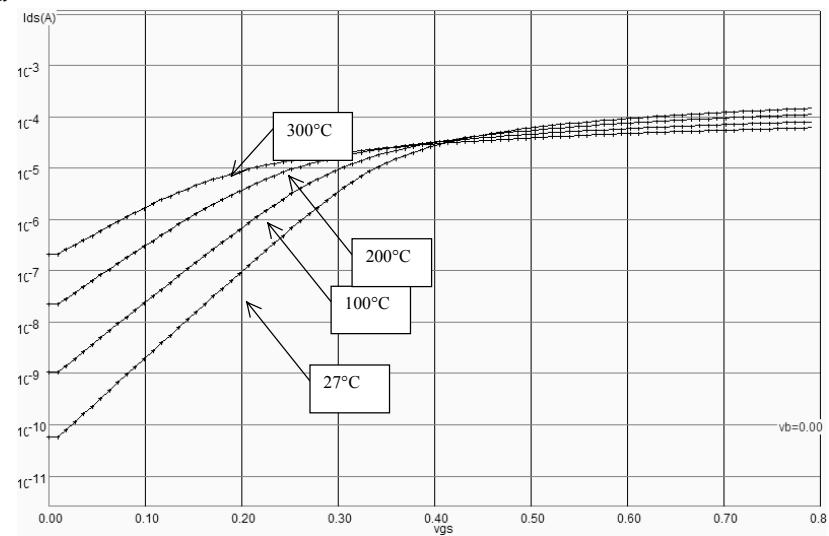
Figure 15-9: Increased decoupling between noisy and sensitive circuits thanks to the insulator

### High Temperature Leakage

The  $I_{off}$  current, corresponding to a zero gate voltage, determines the parasitic leakage current of the MOS device. A low leakage is important for low power operation. The behavior of SOI devices is better than the bulk device in terms of  $I_{off}$  current at high temperature [Kuo]. In the comparative simulations shown in figure 15-10, the sub-threshold slope is steeper for SOI at nominal temperature, as presented earlier. When the temperature is increased up to 200°C, the leakage current in the bulk device is rapidly increased up to 10 $\mu$ A, while in SOI technology, the leakage is kept below 0.1 $\mu$ A. Consequently, at high temperature, the SOI device has a standby current nearly 100 times lower than for bulk technology.



(a) Bulk technology



(a) SOI technology

Figure 15-10: Temperature dependence for bulk and SOI MOS devices (Low leakage  $W=10\mu m$ ,  $L=0.12\mu m$ )

## 2. SOI technology issues

### Kink effect

In SOI technology, when an n-channel MOS transistor passes strong current between the drain and the source, a parasitic phenomenon called Kink effect appears [Kuo]. The current  $I_{ds}$  suddenly rises and provokes a conductance discontinuity, usually between 0.5V and 1V in  $0.12\mu m$  CMOS process. The origin of this parasitic effect is the impact ionization of high energy electrons entering the drain region, which creates supplementary positive and negative charges below the gate.

While electrons participate to the  $Ids$  current, the underlying insulator prevents the positive charge from being evacuated to the substrate, as it would happen in bulk technology thanks to the natural ground connection of the substrate. The positive charges accumulate below the gate (Figure 15-11). The body of the SOI MOS device may rise significantly, without any direct control. The rise of the local voltage below the gate has an instant impact on the threshold voltage which is lowered.

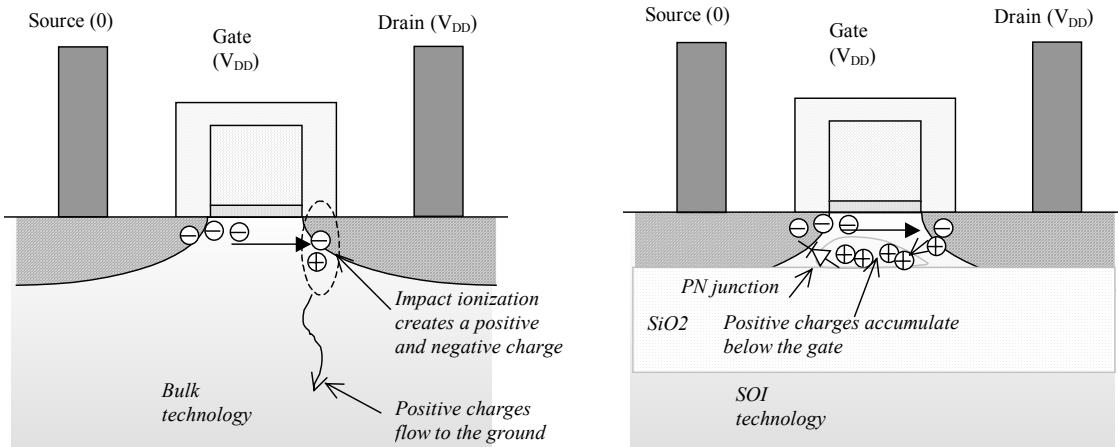


Figure 15-11: The impact ionization creates an accumulation of positive charges below the gate in the case of SOI

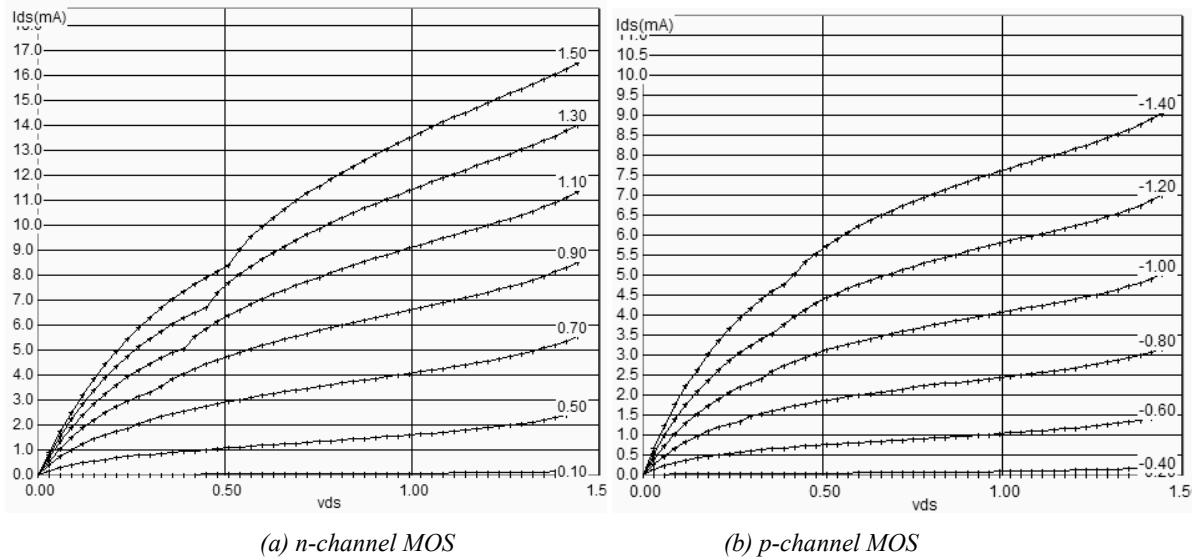


Figure 15-12: The drain current characteristics of the n-channel and p-channel SOI devices show a kink effect near saturation

At a certain point, the bias of the PN junction between the P-doped bulk and the N<sup>+</sup> source diffusion is high enough to turn on the junction, which leads to a sudden channel current increase, as seen in the  $Id/Vd$  characteristics (Figure 15-12). This effect is also called floating body effect (FBE). As the impact ionization is more severe for n-channel MOS devices than for p-channel MOS devices, the kink effect is more pronounced for the n-channel than for the p-channel.

### Fully Depleted MOS

A possibility for reducing the floating bulk effect is to use a very thin diffusion for the channel, so that there is no more room for accumulation of positive charges, and consequently almost no kink effect. The source and drain diffusions are usually manufactured with an increased thickness on top of the SiO<sub>2</sub> insulator.

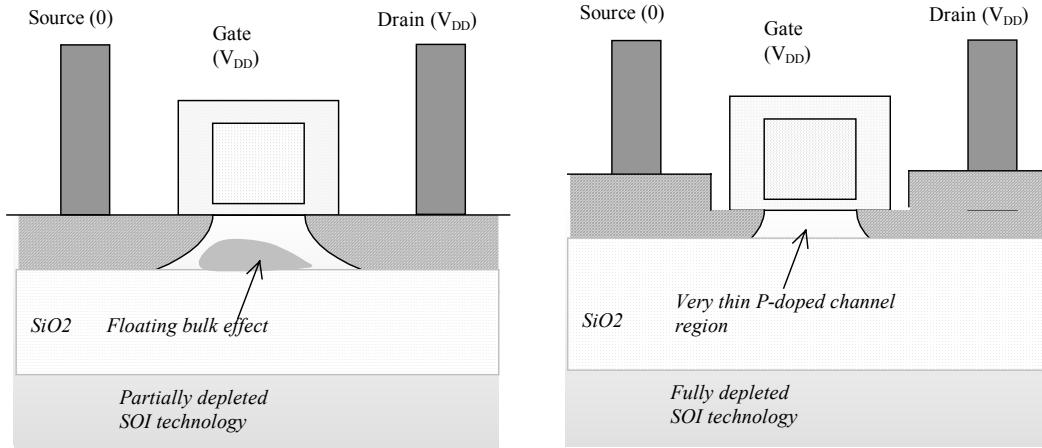


Figure 15-13: The fully depleted MOS device has no more Kink effect, but several manufacturing and design drawbacks

The fully depleted MOS devices are much harder to manufacture and control. The process-controlled threshold adjustment required for low  $V_t$ , high speed and ultra-high speed MOS devices is very complex due to the very thin diffusion area below the gate. These drawbacks have made the fully depleted MOS less attractive than partially depleted MOS. The SOI process parameters provided in Microwind correspond to a partially depleted MOS technology.

## 3. SOI Device Model

Bulk silicon models such as LEVEL 3 or BSIM4 typically do not include source/bulk diode currents because the junctions are usually reverse-biased, and can be considered as junction capacitors. This is not the case for SOI devices where the source/bulk junctions can be significantly forward-biased due to the impact ionization which provokes the accumulation of positive charges below the gate.

### Fully depleted MOS model

The kink effect is very weak in fully depleted SOI MOS devices. Consequently, the BSIM4 model may be applied with reasonable accuracy as the underlying physics and working principles are similar.

## Partially depleted MOS model

In Microwind, the kink effect is modelized in the case of partially depleted SOI devices, thanks to a new parameter *ASOI*. Details on the SOI model in SPICE are provided in [Kuo], who considers the lateral bipolar device made of the source, the channel and the drain regions. The SOI MOS model includes a complete NPN device model in the case of an N-channel MOS, and a PNP device model in the case of a P-channel MOS. A more simple implementation proposed in Microwind consists in modifying the saturation current model directly, where the Kink effect is the most important.

The new parameter is introduced, called *ASOI*. The kink effect occurs when  $Vds$  is higher than the saturation voltage  $Vdsat$ . The parameter *Asoi* determines the amplitude of the kink. A new term is introduced, as shown in equation 15-1. This approach is a simplified version of the model used in BSIM3 SOI device model [Berkeley].

$$I_{ds} = I_{ds\_bsim4} \left( 1 + \frac{\text{ASOI}}{L_{eff}.Vt.\sqrt{V_{DS} - V_{dsat}}} \right) \quad (\text{Equ. 15-1})$$

L = device channel length (m).

$V_{ds}$ = voltage difference between drain and source (V)

$V_{dsat}$ = saturation voltage as defined in chapter 3 (V)

$V_t$  = threshold voltage of the MOS device (V)

ASOI = technological parameter for handling the kink effect (default 2e6 V/cm)

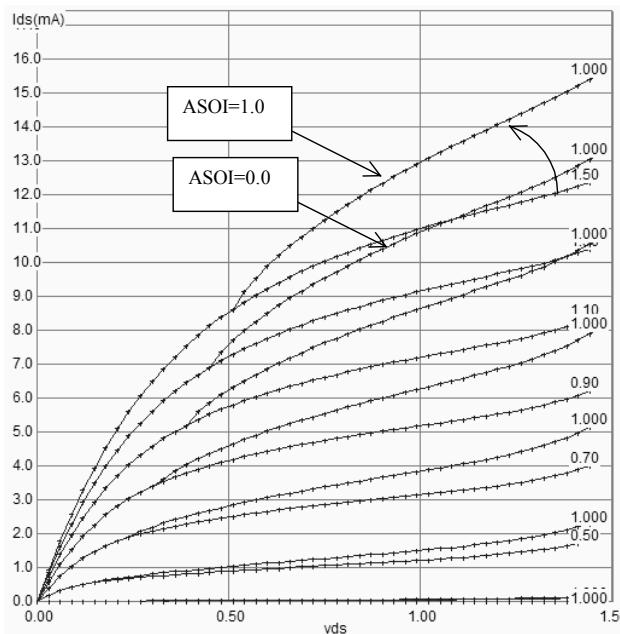


Figure 15-14: The effect of the ASOI parameter on the  $Id/Vd$  characteristics (Using soi012.RUL)

As the oxide thickness scales down to 2nm and below, the quantum mechanism of direct tunneling through the gate oxide rises exponentially. The gate current becomes large enough to compete with the channel current and consequently to affect the body potential. Much more complex models such as BSIMPD [BsimPd] have been developed for an accurate simulation of these nano-scale MOS devices.

## 4. SOI Design

Assuming a partially depleted SOI technology, the Kink effect may be reduced by adding a polarization contact to ground which helps evacuating the accumulated charges outside the channel. The T-shape and H-shape MOS with body tie to ground are shown in figure 15. The MOS device on the left has no body contact, and may suffer from Kink effect as soon as the VDS voltage is higher than 0.5V. The T-shape MOS device includes a supplementary P+ diffusion which is connected to the P- channel region on one side and the VSS ground contact on the other side.

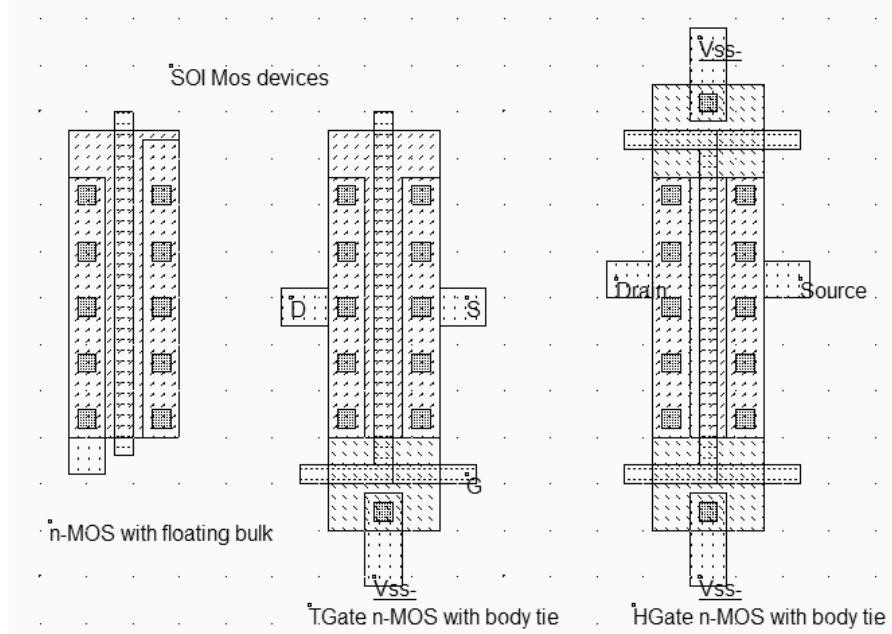


Figure 15-15: Adding a contact in partially depleted MOS to avoid the kink effect (mosSoi.MSK)

The body tie is very efficient at the bottom of the T-shape MOS but cannot evacuate charges accumulated on the upper part of the channel rapidly. An improved design (H-shape) consists in placing two body ties, one at the bottom and one on the top, which almost eliminates the Kink effect. The main disadvantage of the body tie is the important device surface increase and the needs for VSS connections at each MOS device. Important benefits of the SOI technology in terms of compact layout are lost as the body ties takes up valuable silicon space.

### The Memory effect

Accounting for the floating body effect (FBE<Glossary>) requires specific models which handle the "memory effect" of accumulated charges below the channel. Without body tie, the time constant for eliminating these charges is of the order of the millisecond, far larger than the switching delay within the logic gates. However, only a very small percentage of the transistors in a typical logic circuit are unable to work properly with a floating body and require a body tie to ground.

In figure 15-16 a functional error example linked to floating body effect is described, adapted from [Kuo].

<Etienne: add schema>

<Etienne: add layout>

*Figure 15-16: Logic circuit with a functional error due to floating body effect (soiProblem.SCH)*

<Etienne: add simu>

*Figure 15-17: Logic circuit with a functional error due to floating body effect (soiProblem.MSK)*

<Etienne: add comments on simu>

## 5. The Tera-Hertz MOS device

The Tera-Hertz ( $10^{12}$  Hertz) transistor is the key device for the development of 10-to-20-gigahertz processors. A MOS device with Tera-Hertz transit frequency is expected to be fabricated in phase with the 65nm CMOS process. The Tera-Hertz MOS device combines a SOI substrate, narrow gate length and a high K dielectric insulator for the gate. Technological issues to be solved concern the gate and transistor leakage currents and the reliability of the high K dielectric.

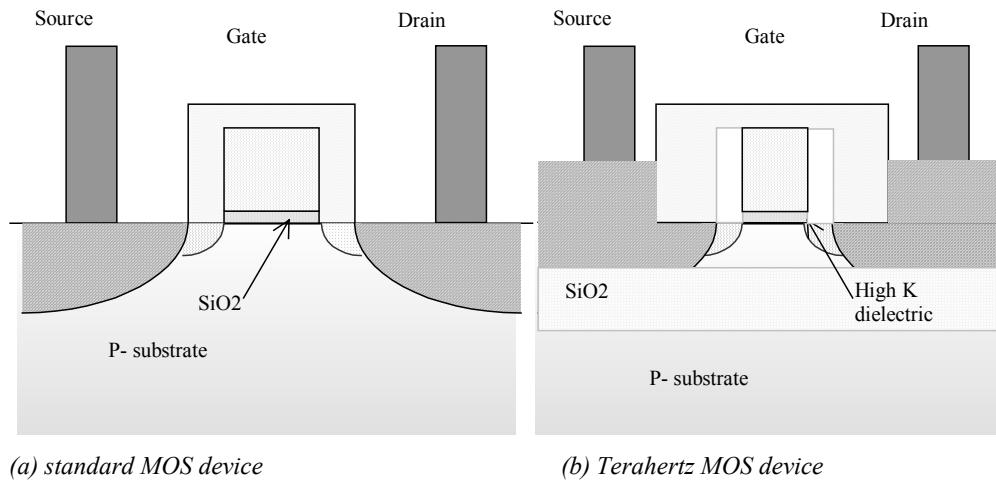


Figure 15-18: The Tera-hertz transistor

## 6. Conclusion

This chapter has described briefly the SOI technology, its main advantages and drawbacks. Using Microwind, some comparison may be performed between bulk and SOI technology in terms of MOS characteristics, switching speed and power consumption. The Kink effect has been described and a simplified model has been proposed, together with body tie technique to limit its consequences. The adoption of SOI as a mainstream technology is not yet a reality, maybe because of the steady progress in bulk CMOS technology and of the drawbacks linked to Kink effect at device level. However, fully functional microprocessors utilizing SOI have recently been introduced, with impressive gains in terms of speed and power consumption, which have convinced many skeptics that SOI is a serious candidate for the fabrication of the next generation of chips.

### Exercises

15-1. Design a NAND gate in SOI technology. What is the switching speed and standby current improvement as compared to bulk technology?

15-2 <Tbd>

15-3 <Tbd>

### References

- [Kuo] James B. Kuo, Shih Chia Lin "Low Voltage SOI CMOS VLSI Devices and Circuits", Wiley Intersciences, ISBN 0-471-41777-7, 2001
- [Berkeley] BSIM3v3 Manual, University of California at Berkeley, USA, <http://www-device.eecs.berkeley.edu> 1998

[BSIMPD] BSIMPD version 2.0 MOSFET model user's manual <http://www-device.eecs.berkeley.edu>

[CompactModel] Cmpact Model Concil Homepage <http://eigroup.org/cmc>

# 16

## Future and Conclusion

### 1. Predicting the unpredictable

Will the semiconductor industry run out of process technology soon? The international technology roadmap for semiconductors sets industry technology targets and milestones for the next 15 years [Itrs]. This prospective is probably one of the most referenced document in micro-electronics. The roadmap predicts transistor-gate lengths in microprocessors shrinking to 25 nanometers (0.025 micron) by 2008, five years earlier than earlier editions

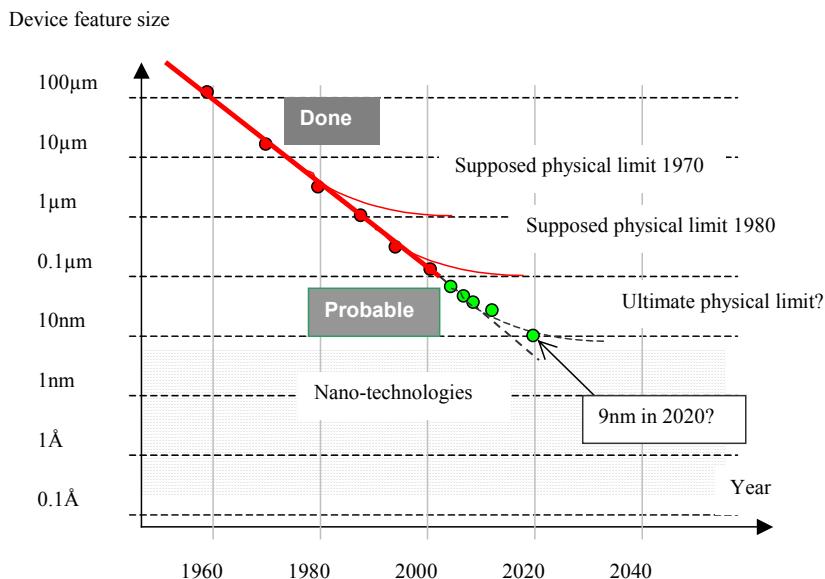


Figure 16-1. The art of predicting the future of micro-electronics

#### The technology scale down beyond 90nm

The 90nm CMOS technology was introduced in 2004 for system-on-chip fabrication. The next technological steps are presented in figure 16-2: 65nm, 35nm, 25nm, 15nm and 9nm. By 2007, processors with 1 billion transistors are expected to be developed, running at 10GHz.

The power consumptions of these processors could rise to around 100 Watts per square centimeters on the die, which could represent the second limiting factor in CMOS design, after the cost of the foundry and the masks.

By 2016, the ITRS roadmap projects the minimum physical gate length of transistors to be close to 9 nm (0.009 micron), which is considered by most researchers to be the physical limit of silicon. Will the transistor-gate length scaling continue below 9nm? This prospect is now causing the industry to consider post-CMOS technologies.

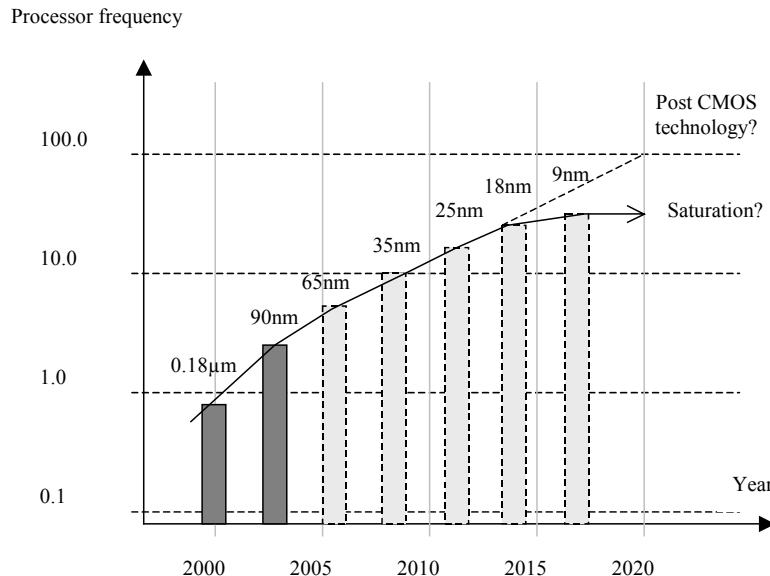


Figure 16-2. CMOS technologies forecast until 2020

## 2. As a conclusion

This book has described several aspects of the CMOS circuit design, using the Microwind tool as an illustration. A very important gap exists between the educational PC-based tool and the professional tools used in industry for real-case designs. However we hope that the readers have caught the essential parts of MOS devices, logic circuits, memories, analog cells and interfacing, through the illustrations and numerous examples.

No book, no teaching can replace the practical experience. Although the simulations should never be trusted, the access to microelectronic technology tends to be more and more costly, which justifies the relevance of such simple tools.

The authors have dedicated around two years to build the technical contents of this book, and tried their best to improve the Microwind and Dsch tools, trying to make attractive and simple something which tends to be more and more complicated. Still, some bugs needs to be corrected, the user's interface should be improved, and important new features should be included. As the tools are in constant evolution thanks to user's feedback and comments, we encourage the reader to download the updated version of Microwind and Dsch from their web page, as detailed in the companion CD-ROM introduction. The tools have benefit from the real-case experiments conducted in 0.35, 0.25 and 0.18 μm CMOS technologies in partnership with ST-Microelectronics and Motorola.

We hope that the reader will find the contents of this book and the companion tools useful. It is our hope that the reader will design logic and analog circuits by himself, understand by a practical approach the principles of deep submicron VLSI design, and later contribute to innovative designs to support the electronic systems of the future.

## References

[Irts] Tech. Roadmap <web>

# A      Design Rules

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.

## 1. 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

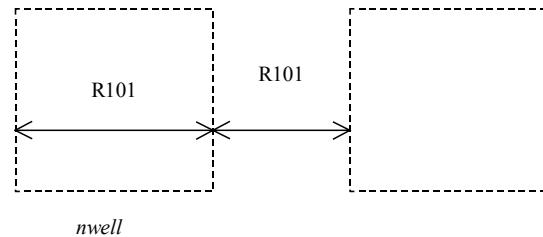
*Table 1-xxx: correspondence between technology and the value of lambda in µm*

## 2. Layout Design Rules

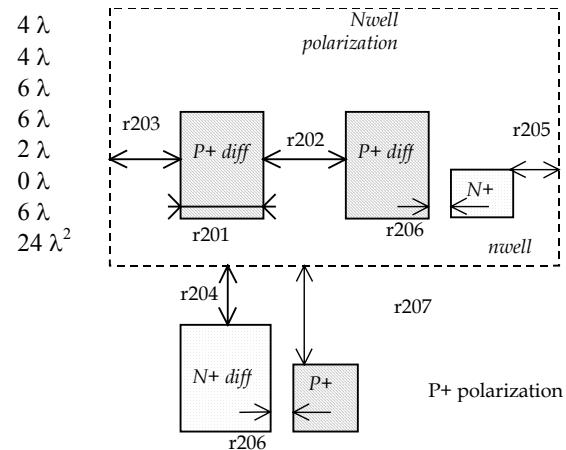
The software can handle various technologies. The process parameters are stored in files with the appendix '.RUL'. The default technology corresponds to a generic 6-metal 0.12µm CMOS process. The default file is CMOS012.RUL. To select a new foundry, click on **File -> Select Foundry** and choose the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

**N-Well**

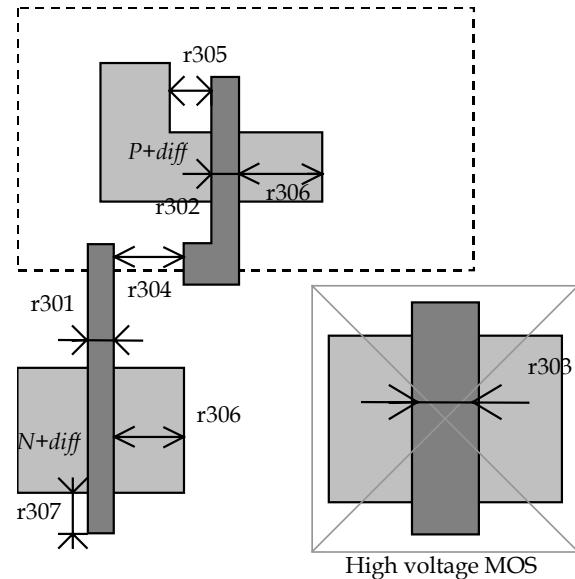
r101	Minimum well size	$12 \lambda$
r102	Between wells	$12 \lambda$
r110	Minimum well area	$144 \lambda^2$

**Diffusion**

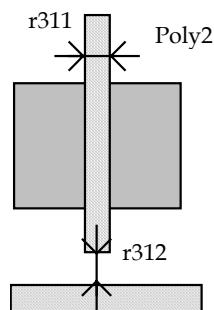
r201	Minimum N+ and P+ diffusion width	$4 \lambda$
r202	Between two P+ and N+ diffusions	$4 \lambda$
r203	Extra nwell after P+ diffusion :	$6 \lambda$
r204:	Between N+ diffusion and nwell	$6 \lambda$
r205	Border of well after N+ polarization	$2 \lambda$
r206	Between N+ and P+ polarization	$0 \lambda$
r207	Border of Nwell for P+ polarization	$6 \lambda$
r210	Minimum diffusion area	$24 \lambda^2$

**Polysilicon**

r301	Polysilicon width	$2\lambda$
R302	Polysilicon gate on diffusion	$2 \lambda$
R303	Polysilicon gate on diffusion for high voltage MOS	$4 \lambda$
R304	Between two polysilicon boxes	$3 \lambda$
R305	Polysilicon vs. other diffusion	$2 \lambda$
R306	Diffusion after polysilicon	$4 \lambda$
R307	Extra gate after polysilicon	$3 \lambda$
r310	Minimum surface	$8 \lambda^2$

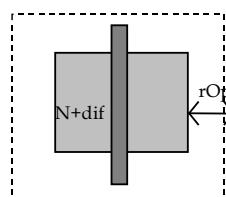
**2<sup>nd</sup> Polysilicon Design Rules**

r311	Polysilicon2 width	$2\lambda$
r312	Polysilicon2 gate on diffusion	$2\lambda$
r320	Polysilicon2 minimum surface	$8\lambda^2$

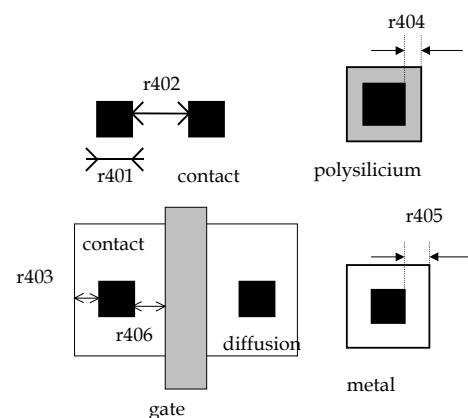
**MOS option**

rOpt Border of “option” layer over diff N+ and diff P+

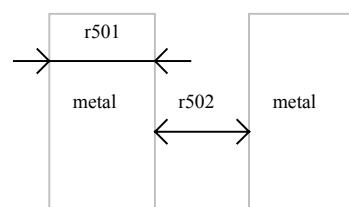
7λ

**Contact**

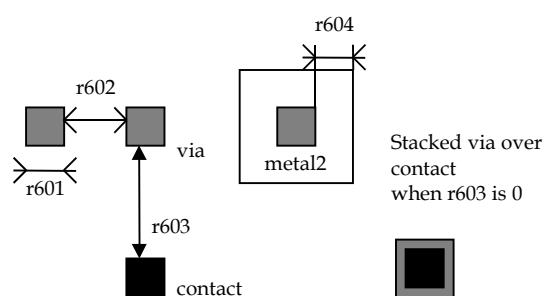
r401	Contact width	$2\lambda$
r402	Between two contacts	$5\lambda$
r403	Extra diffusion over contact	$2\lambda$
r404	Extra poly over contact	$2\lambda$
r405	Extra metal over contact	$2\lambda$
r406	Distance between contact and poly gate	$3\lambda$
r407	Extra poly2 over contact	$2\lambda$

**Metal 1**

r501	Metal width	$4\lambda$
r502	Between two metals	$4\lambda$
r510	Minimum surface	$16\lambda^2$

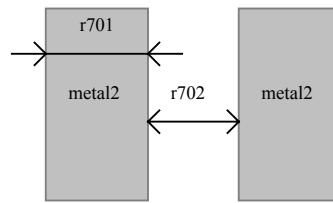
**Via**

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

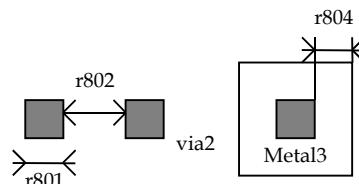


**Metal 2**

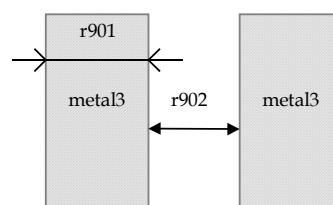
r701	Metal width::	$4 \lambda$
r702	Between two metal2	$4 \lambda$
r710	Minimum surface	$16 \lambda^2$

**Via 2**

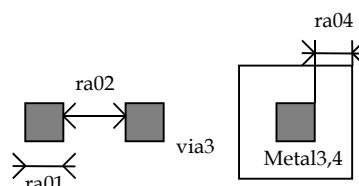
r801	Via2 width : $2 \lambda$
r802	Between two Via2: $5 \lambda$
r804	Extra metal2 over via2: $2 \lambda$
r805	Extra metal3 over via2: $2 \lambda$

**Metal 3**

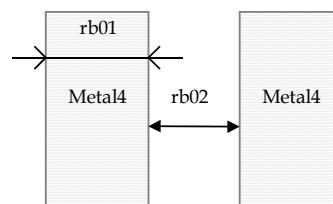
r901	Metal3 width: $4 \lambda$
r902	Between two metal3 : $4 \lambda$
r910	Minimum surface : $32 \lambda^2$

**Via 3**

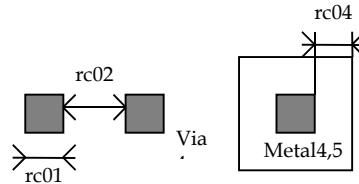
ra01	Via3 width : $2 \lambda$
ra02	Between two Via3: $5 \lambda$
ra04	Extra metal3 over via3: $2 \lambda$
ra05	Extra metal4 over via3: $2 \lambda$

**Metal 4**

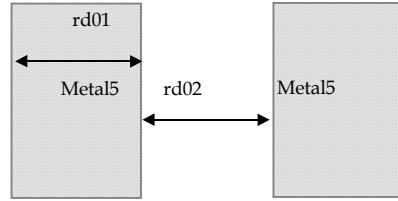
rb01	Metal4 width: $4 \lambda$
rb02	Between two metal4 : $4 \lambda$
rb10	Minimum surface : $32 \lambda^2$

**Via 4**

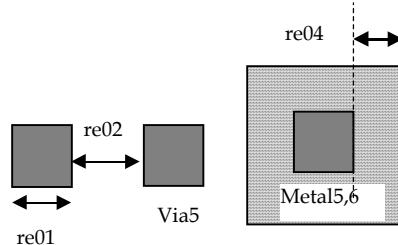
rc01	Via4 width : $2 \lambda$
rc02	Between two Via4: $5 \lambda$
rc04	Extra metal4 over via2: $3 \lambda$
rc05	Extra metal5 over via2: $3 \lambda$

**Metal 5**

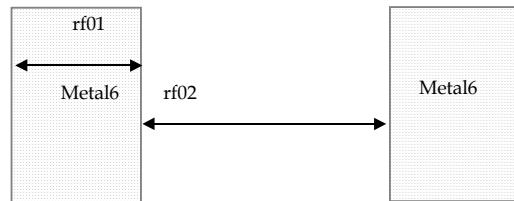
rd01	Metal5 width: $8 \lambda$
rd02	Between two metal5 : $8 \lambda$
rd10	Minimum surface : $100 \lambda^2$

**Via 5**

re01	Via5 width : $4 \lambda$
re02	Between two Via5: $6 \lambda$
re04	Extra metal5 over via5: $3 \lambda$
re05	Extra metal6 over via5: $3 \lambda$

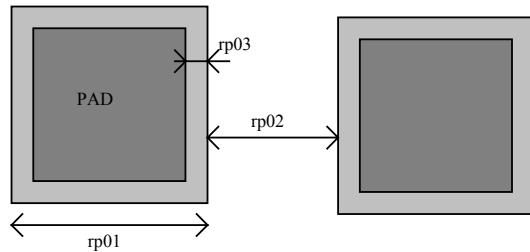
**Metal 6**

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

**3. Pads**

The rules are presented below in  $\mu\text{m}$ . In .RUL files, the rules are given in lambda. As the pad size has an almost constant value in  $\mu\text{m}$ , each technology gives its own value in  $\lambda$ .

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 $\mu\text{m}$

**4. 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 figure A-xxx.

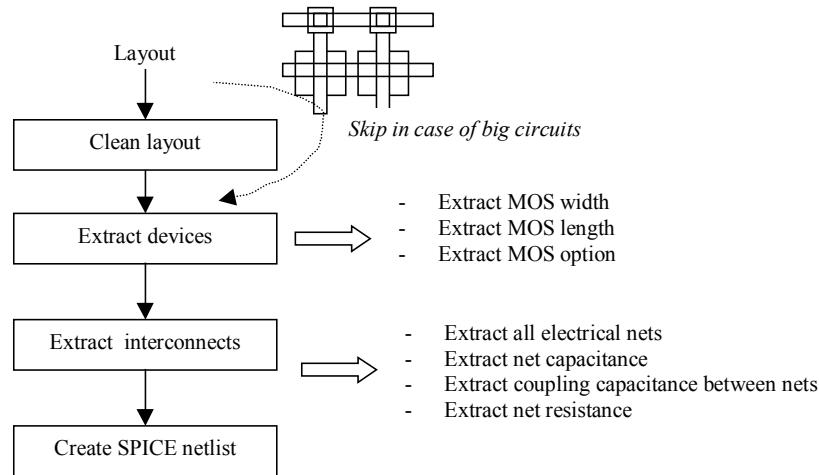


Figure A-xxx: Extraction of the electrical circuit from layout

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.

## 5. Node Capacitance extraction

Each deposited layer is separated from the substrate by a SiO<sub>2</sub> oxide and generated by a parasitic capacitor. The unit is the aF/ $\mu\text{m}^2$  (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 0.12 $\mu\text{m}$ ). Surface capacitance refers to the body. Vertical crosstalk capacitance refer to inter-layer coupling capacitance, while lateral crosstalk capacitance refer to adjacent interconnects.

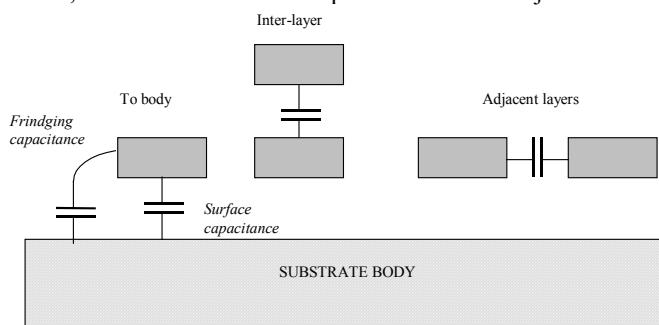


Figure A-1: Capacitances

### SURFACE CAPACITANCE

NAME	DESCRIPTION	LINEIC (aF/ $\mu\text{m}$ )	SURFACE (aF/ $\mu\text{m}^2$ )
CpoOxyde	Polysilicon/Thin oxide capacitance	n.c	4600

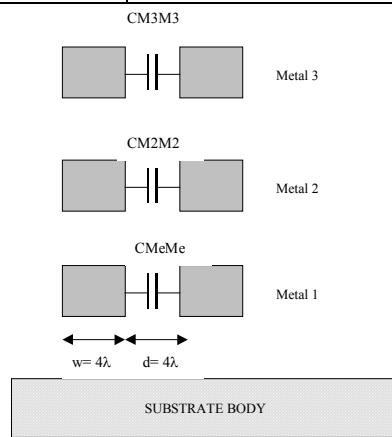
CpoBody	Polysilicon to substrate capacitance	n.c	80
CMEBody	Metal on thick oxide to substrate	42	28
CM2Body	Metal2 on body	36	13
CM3Body	Metal3 on body	33	10
CM4Body	Metal4 on body	30	6
CM5Body	Metal5 on body	30	5
CM6Body	Metal6 on body	30	4

**INTER-LAYER CROSSTALK CAPACITANCE**

NAME	DESCRIPTION	VALUE (aF/ $\mu\text{m}^2$ )
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

**LATERAL CROSSTALK CAPACITANCE**

NAME	DESCRIPTION	VALUE (aF/ $\mu\text{m}$ )
CMeMe	Metal to metal (at $4\lambda$ distance, $4\lambda$ 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



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.

**Figure A-2: Crosstalk capacitance**

**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 A-3 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.

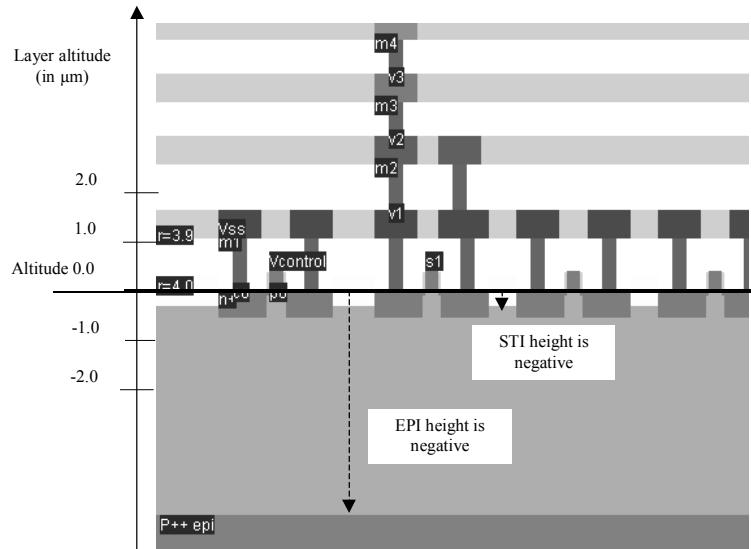


Figure A-2: Description of the 2D aspect of the CMOS technology

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 SiO <sub>2</sub> 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 Si <sub>3</sub> N <sub>4</sub> .	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

## 6. Resistance Extraction

NAME	DESCRIPTION	VALUE ( $\Omega$ )
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

### 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 (SiO <sub>2</sub> )	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)

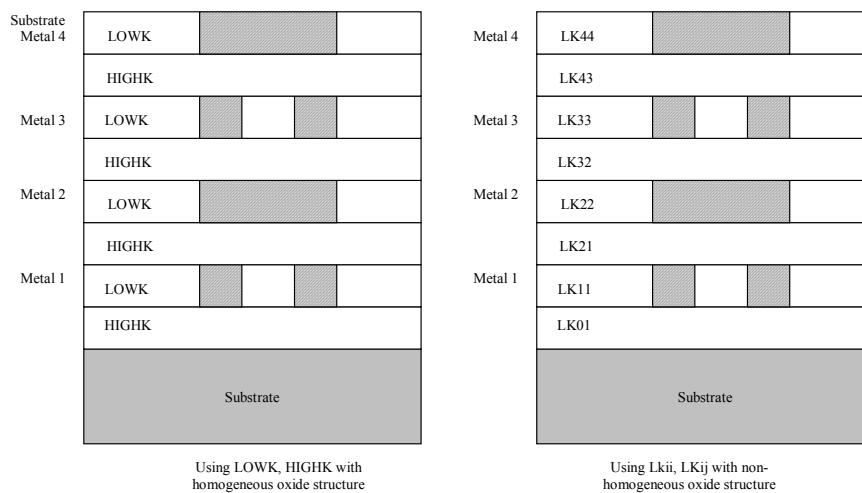


Fig. A-xxx: Illustration of the use of LOWK, HIGHK dielectric constants (left figure) or detailed permittivity for each layer (right figure)

## 7. 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
DELTAT	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

**Models Level1 and Level3 for analog simulation**

Four types of MOS devices may be described as detailed in figure 12-4 (Data from SIA, 0.12 $\mu\text{m}$  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 $\mu\text{m}$  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 12-5: Description of MOS options in 0.12 $\mu\text{m}$  technology (cmos012.RUL)

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

PARAMETER	KEYWORD	DEFINITION	TYPICAL VALUE 0.25 $\mu\text{m}$	
			NMOS	pMOS
VTO	l3vto	Threshold voltage	0.4V	-0.4V
U0	l3u0	Low field mobility	0.06 m <sup>2</sup> /V.s	0.025 m <sup>2</sup> /V.s
PHI	l3phi	Surface potential at strong inversion	0.3V	0.3V
LD	l3ld	Lateral diffusion into channel	0.01 $\mu\text{m}$	0.01 $\mu\text{m}$
GAMMA	l3gamma	Bulk threshold parameter	0.4 V <sup>0.5</sup>	0.4 V <sup>0.5</sup>
KAPPA	l3kappa	Saturation field factor	0.01 V <sup>-1</sup>	0.01 V <sup>-1</sup>
VMAX	l3vmax	Maximum drift velocity	150Km/s	100Km/s
THETA	l3theta	Mobility degradation factor	0.3 V <sup>-1</sup>	0.3 V <sup>-1</sup>
NSS	l3nss	Sub-threshold factor	0.07 V <sup>-1</sup>	0.07 V <sup>-1</sup>
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 0.25μm	
			NMOS	pMOS
VTO Mos2	13v2to	Threshold voltage for MOS2	0.5V	-0.5V
VTO Mos3	13v3to	Threshold voltage for MOS3	0.7V	-0.7V
U0 Mos2	13u2	Mobility for MOS2	0.06	0.025
U0 Mos3	13u3	Mobility for MOS3	0.06	0.025
TOX Mos 2	13t2ox	Thin oxide thickness for MOS2	3nm	3nm
TOX Mos 3	13t3ox	Thin oxide thickness for MOS3	7nm	7nm

#### BSIM4 Model for analog simulation

The list of parameters for BSIM4 is given below:

Parameter	Keyword	Description	NMOS value in 0.12μm	PMOS value in 0.12μm
VTHO	b4vtho	Long channel threshold voltage at Vbs = 0V	0.3V	0.3V
VFB	b4vfb	Flat-band voltage	-0.9	-0.9
K1	b4k1	First-order body bias coefficient	0.45 V1/2	0.45 V1/2
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 Vth	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 Vgs.	1	1
U0	b4u0	Low-field mobility	0.060 m2/Vs	0.025 m2/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°-6μm	0.01°-6μm
LINT	b4lint	Channel-length offset parameter	0.01°-6μm	0.01°-6μm
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 reduction	4.24e8 V/m	4.24e8 V/m
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	b4nd ep		0.54	0.54
XJ	b4xj	Junction depth	1.5e-7	1.5e-7

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

## 8. 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"

```

## 9. Design Rule File

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

```

MICROWIND 2.0
*
* Rule File for CMOS 0.18µm
* 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 0.18µm - 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 1=enable, 0= disable)
tox = 0.004 (fast MOS oxide in µm 0.0=disable)
hvt ox= 0.007 (high voltage MOS oxide)
salicide = 0 (Enable salicide 1=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>

```

## 10. Simulation parameters for DSCH2

The following list of parameters is used in Dsch2 to configure the simulation.

CODE	DESCRIPTION	TYPICAL VALUE
VDD	Supply voltage of the chip	2.0 V

<add default.rul>

# B      Microwind2 Program Operation & Commands

## 1. Getting Started

To get your MICROWIND2 program started, use the following procedure:

- ① Insert the companion CD-Rom into drive
- ② Under Windows 98/NT/XP, double-click index.html
- ③ Click "Download Microwind2". Unzip the mw2.zip file to the desired directory.
- ④ Double click on the Microwind2 icon to start the software

The software runs on Windows 95, 98, NT and XP operating systems.

### **Command line parameters**

The command line may include two parameters:

The First parameter is the default mask file loaded at initialization

The Second parameter is the design rule file loaded at initialization

For example, the command « microwind2 test.MSK cmos018.rul » executes MICROWIND2 with a default mask file « test.MSK » and the rule file « cmos018.RUL ».

### **Configure the Microwind2 Icon**

You may program the Microwind2 icon by a click with the right button, then “properties”. The default target do not contain any parameter. Just add the default layout filename and the default design rule file. In the example below, Microwind uses the file "TEST.MSK" and the design rule file "CMOS018.RUL" as initial parameters.

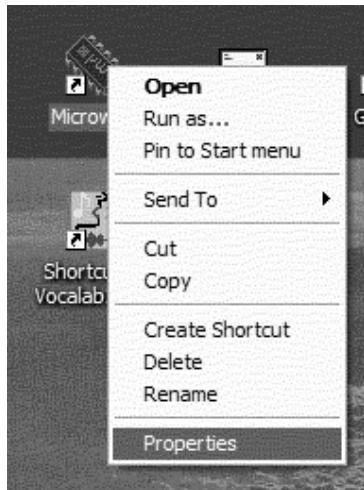


Figure B-xxx: access to Microwind2 icon properties

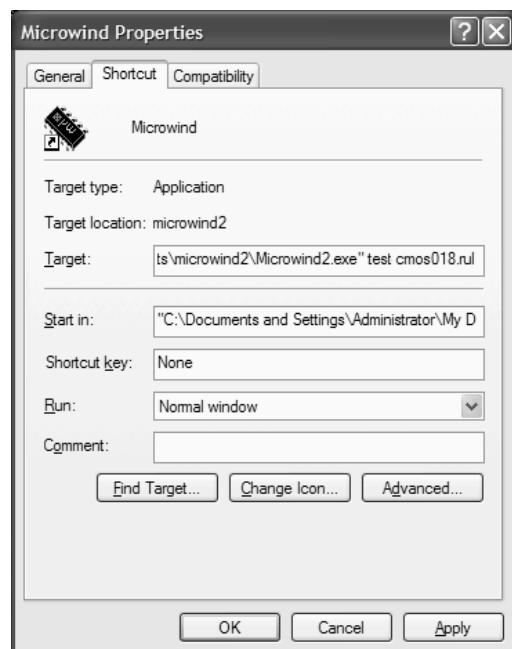


Figure B-xxx: Configuring Microwind with a default filename "test.MSK" and default technology "cmos018.RUL"

## 2. List of Commands in Microwind2

### About Microwind2

Information about the software release and contact for support.

#### Add Text to Layout



Use this icon to fix a text to one box or location in the design. That text illustrates the layout and should be used as much as possible for each significant node such as inputs and outputs. To add some text to a particular place, proceed as follows:

- ① Click on the icon
- ② Set the text location with the mouse. A dialog box appears
- ③ Enter the text in front of “Label name:” and press “Assign”. The text is set in the drawing

A text can be modified as follows: click on the icon, click inside the existing text. The old text appears. Modify it and click on “Assign”. You may add a clock, a pulse, a VDD or VSS voltage source to the text.

#### Convert into

MICROWIND2 converts the MSK layout into CIF using a specific interface, invoked by “File -> Make CIF file”. The CIF file can be exported to VLSI CAD software. The right table of the screen (Figure B-xxx) gives the correspondence between MICROWIND2 layers and CIF layers, the number of boxes in the layout and the corresponding over-etch.

The over etch is used to modify the final size of the CIF boxes in order to fit the exact design rules.

Click on « Convert to CIF » to start conversion. Some parts of the result appear in the left window

The main unit is 1nm. You may change it to fit the requirements of the target CAD tool.

For CMOS 0.25µm rule file (cmos025.RUL), notice the over etch applied to contact and via. This over etch is mandatory to obey the final design rules, while keeping the user-friendly and portable lambda-based design.

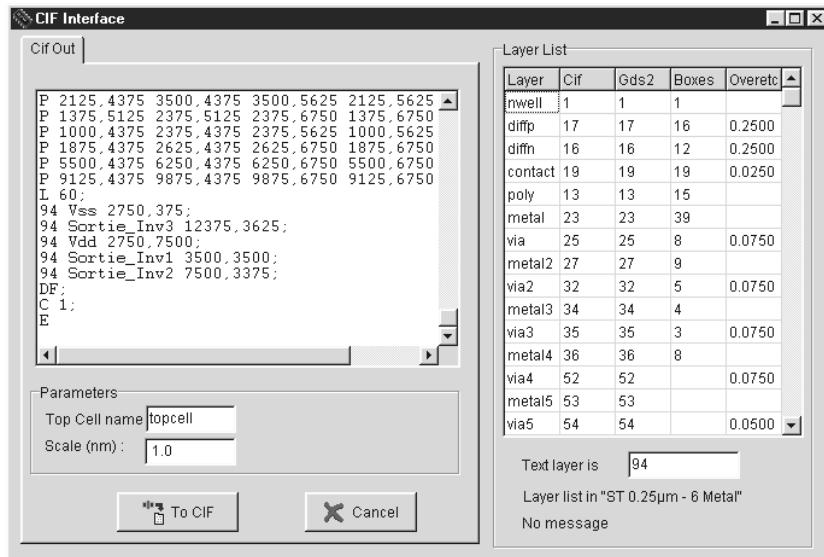


Figure B-xxx: the CIF generation screen

Concerning diffusions, notice that the CIF generator produces active areas and implants. Microwind2 uses simple n+diffusion and p+diffusion while most industrial layout design tools use the concept of active area surrounded by implants, either n+ or p+, as illustrated below.

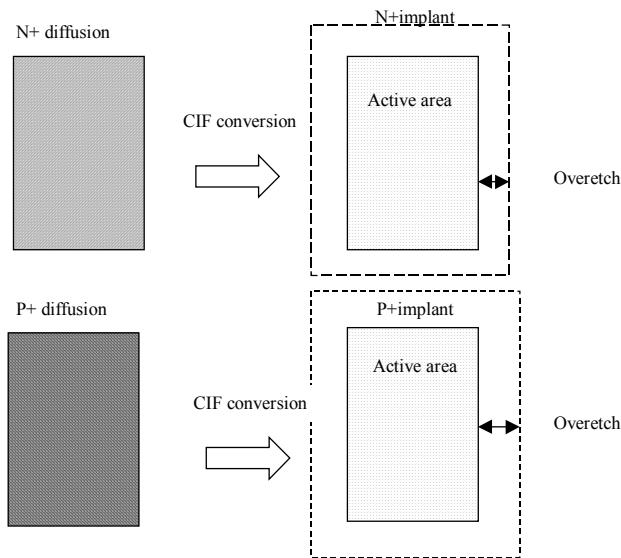


Figure B-xxx: The CIF conversion produces active areas and implants, to be compatible with industrial process

This means that each n+ diffusion box drawn in Microwind is converted into two boxes, one linked with "Active Area", with a code name declared in the design rule file, a second box linked to "diffn", with a given overetch. Each p+ diffusion box is converted into two boxes too. One is linked with the same "Active Area", a second box linked to "diffp", with a given overetch.

## Colors

- Switch to monochrome: the layout is drawn in black and white. This type of drawing is convenient to build monochrome documentation. Press “Alt”+”Print Screen” to copy the screen to the clipboard. Then, open “Word”, click “Edit-> Paste”. The screen is inserted into the document.
- White background. The layers appear with a palette of colors on a white background.

**Copy**

Click on the Copy icon. Move the cursor to the design window, and delimit the active area with the mouse. Consequently, all the graphics included in this area are copied. The external shape of the copied elements appears. Fix those copied elements at the desired location by a click on the mouse. Click on Undo to cancel the copy command.

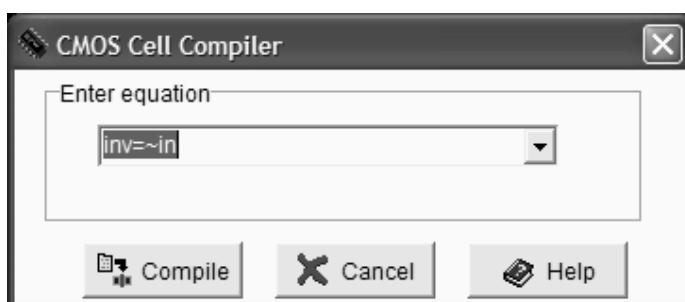
**Cut**

Click on the Cut icon. Move the cursor to the design window, and delimit the active area with the mouse. Consequently, all the graphics included in this area are erased. Click on Undo to fix those elements back into the design.

- A layer is protected from erasing if you remove the tick in the palette twice. In the palette, an empty square to the right of the layer indicates a protected layer.
- A layer is unprotected from erasing if you select it again in the palette. A tick in the square to the right of the layer indicates an unprotected layer.
- One box only can be erased by a click inside that box when the cut command is active. The box is then erased.

**Compile one Line**

The cell compiler is a specific tool designed for the automatic creation of CMOS cells from logic description. Click on Compile -> Compile One Line. The menu below appears. The default equation corresponds to a 3 input NOR gate. If needed, one can use the keyboard in order to modify the equation and then click on Compile. The gate is compiled and its corresponding layout is generated.



*Fig. B-xxx: The cell compiler window*

- The first item of the one-line syntax corresponds to the output name.
- The latter is followed by the sign « = », by the list of input names separated by operators AND '&', OR '|', XOR '^', NOT '¬', XNOR '¬^'. If need be, parenthesis can be added.
- The input and output names are 8 character strings maximum.

Cell	Formula
Inverter	out=/in
NAND gate	n=/ (a.b)
3 Input OR	s=a+b+c
3 Input NAND	out=/ (a.b.c)
AND-OR Gate	cgate=a. (b+c)
CARRY Cell	cout=(a.b)+(cin.(a+b))

*Table B-xxx: Examples of logic cell descriptions*

The p-channel transistors are located on the top of the n-channel transistor net. If some layout already exists near those icons, the cell origin is moved to the right until enough free space is found. If the NOT operator (Symbol '¬') has not been specified after the '=' sign, an inverter is added at the right hand side of the compiled cell. That is why an AND gate is compiled as a NAND gate followed by an inverter.

### Compile VERILOG file

The cell compiler can handle layout generation from a primitive-based VERILOG description text into a layout form automatically. Click on Compile -> Compile Verilog File. Select a VERILOG text file and click on “Generate”. For instance, the microwind2 directory contains the « FADD.TXT » file which corresponds to the description of a full-adder.

```
// DSCH 2.5d
// 21/04/02 14:00:50
// C:\Dscht2\Book on CMOS\fadd.sch

module fadd( C,B,A,Sum,Carry);
    input C,B,A;
    output Sum,Carry;
    xor #(12) xor2(w4,A,B);
    nand #(10) nand2(w5,A,C);
    nand #(10) nand2(w6,B,C);
    nand #(10) nand2(w7,B,A);
    xor #(12) xor2(Sum,w4,C);
    nand #(10) nand3(Carry,w7,w6,w5);
endmodule

// Simulation parameters
// C CLK 10 10
// B CLK 20 20
// A CLK 30 30
```

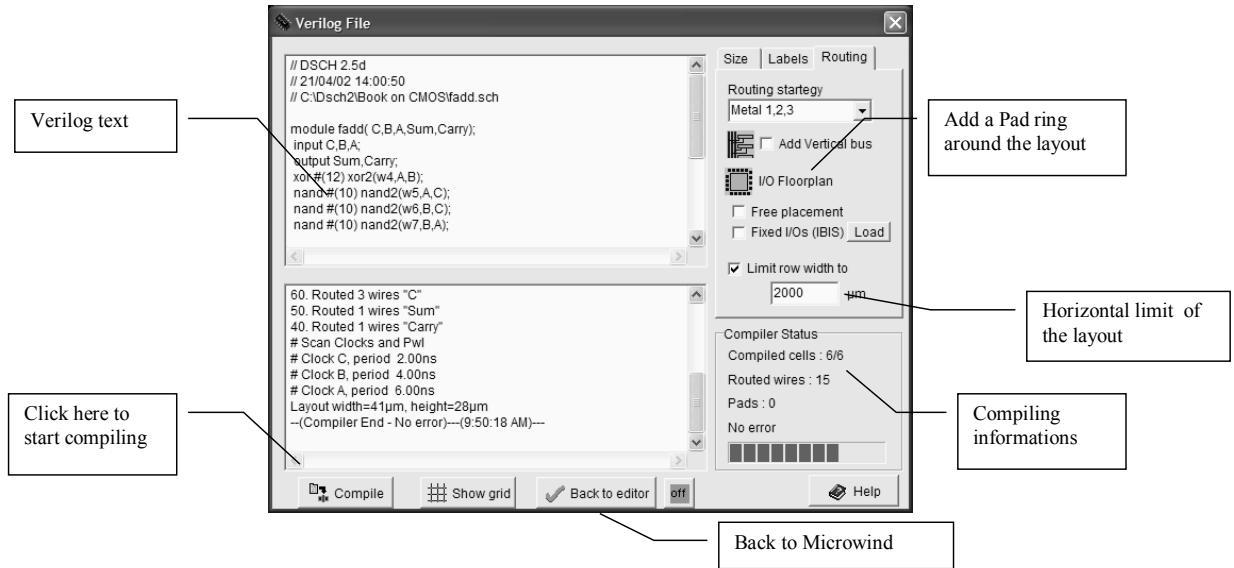


Fig. B-xxx: The VERILOG compiler window

PRIMITIVE	NODES	EXAMPLE
dreg	Inputs : Data, RESET, CLOCK Outputs: Q, nQ	dreg reg1(d,rst,h,q,nq);
Inv, not	Inputs : IN Outputs: OUT	inv inv1(s,e); // both 'inv' and 'not' not inv1(s,e); // can be used
and	Inputs : 2 to 4 Outputs: S	and and1(s,a,b,c,d); // limit inputs to 4
nand	Inputs : 2 to 4 Outputs: S	nand nand1(s,a,b,c,d);
or	Inputs : 2 to 4 Outputs: S	or or3(s,a,b,c);
nor	Inputs : 2 to 4 Outputs: S	nor my_nor4(s,a,b,c,d);
xor	Inputs : a,b Outputs: S	xor xor_gate(xor_out,d0,d1);
Nmos	Inputs: gate, source Outputs: drain	nmos nmos1(d,s,g);

Fig. B-xxx: The VERILOG primitives supported by the CMOS compiler

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. Click **Compile** → **Show grid** to superpose the routing grid on the layout. In figure xxx, the routing of input/output between basic cells is presented. Notice that the routing is performed either on the top or on the bottom of the active parts.

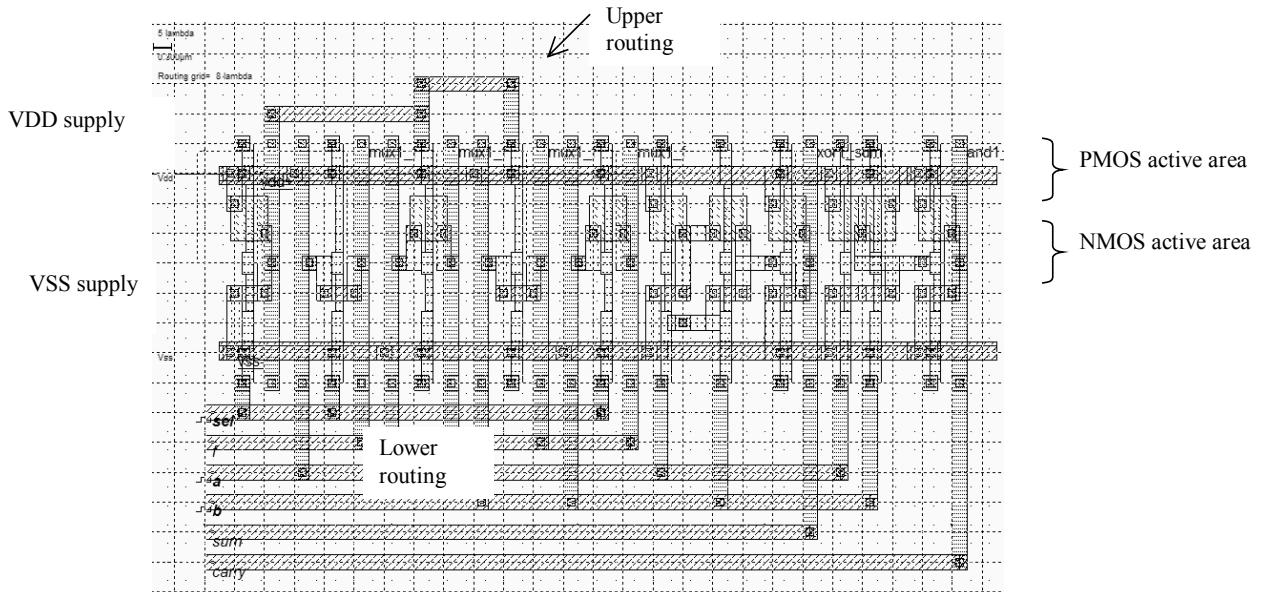


Fig. B-xxx : The compiler grid

**Design Rule Checker**

The design rule checker (DRC) scans all the design and verifies that all the minimum design rules are respected. Click on the icon above or on Analysis ->Design Rule Checker to run the DRC. The errors are highlighted in the display window, with an appropriate message giving the nature of the error. Details about the position and type of the errors appear on the screen.

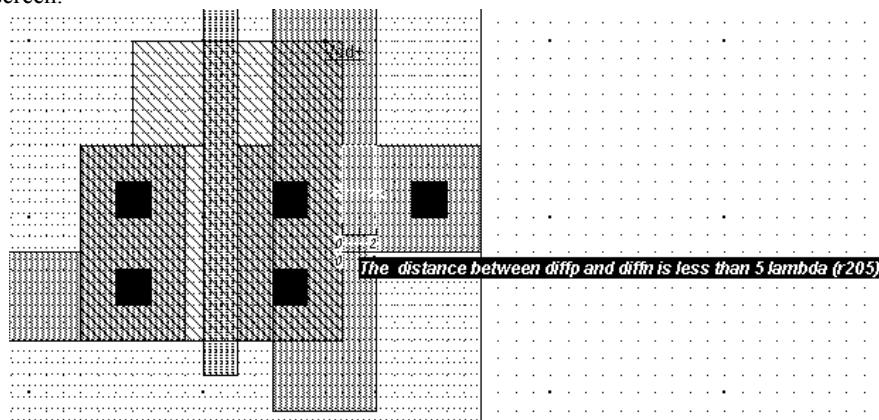


Fig. B-xxx: Example of design rule error

**Draw a Box**

The « Draw Box » icon is the default icon. It creates a box in the selected layer. The default layer is polysilicon. If the « Draw Box » icon is not selected, click on it. Then, move the cursor to the display window and fix the first corner of the box with a press of the mouse. Keep pressed and drag the mouse to the opposite corner of the box. Release the mouse and see how the box is created.

- The active layer is selected in the palette.
- The red color indicates the active layer.
- The gray key on the right of the layer button specifies that all boxes using the layer can be erased, stretched or copied.
- A click on the gray key turns the key to a red color. A red key protects the layer.

### Duplicate XY

The command « Duplicate XY » is very useful to generate an array of identical cells such as RAM cells for example. Click on Edit -> Duplicate XY, include the elements to duplicate in an area defined by the mouse, and the screen shown in figure B-xxx appears. In both X and Y, the default multiplication factor is x 2. You may adjust the space between cells. By default, the cells touch each other. Selected boxes appear in the right window, and also in yellow on the main layout window.

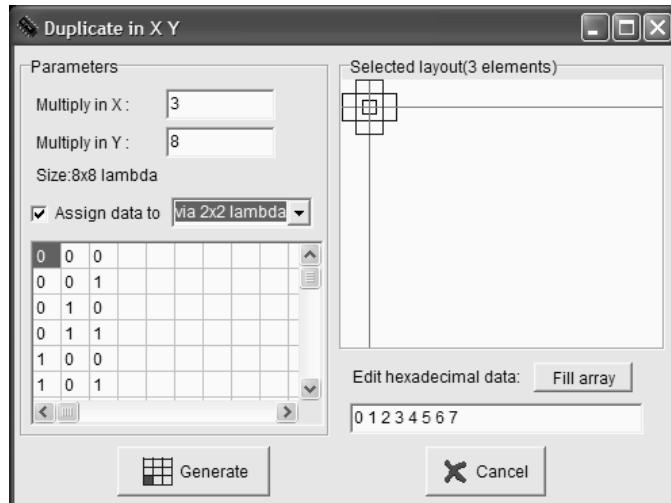
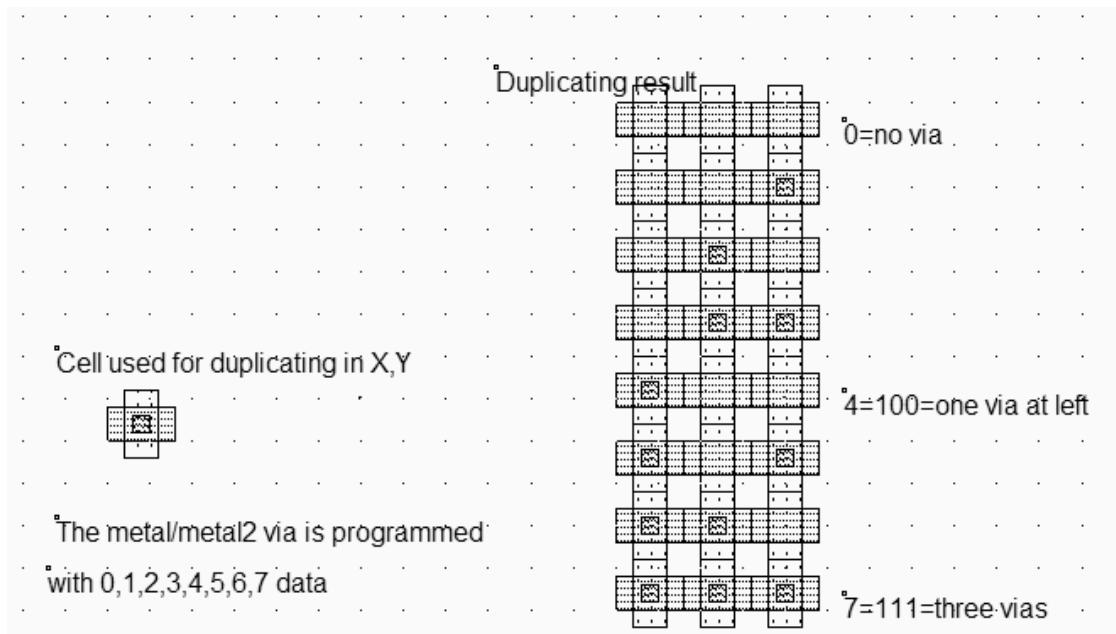


Fig. B-xxx: The Duplicate X,Y menu, used for a metal1/Metal2 crosspoint

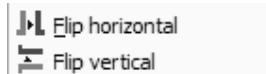
The data option (Assign data, Edit Hexadecimal data, Fill Array) is very useful to generate ROM masks or decoder arrays. An example of decoder array is given in figure B-xxx. The programming affects the via between metal1 and metal2, according to the list of hexadecimal values given in the editing list.

- Click the desired data on the "Edit hexadecimal data" editing area. The values must be separated by a space.
- Click "Fill array" to convert these data into Boolean values, according to the X,Y array size.
- Select the appropriate box for programming. In this case, the 0/1 create of not a via at each X,Y location.
- Click "Generate". The following result appears.



*Fig. B-xxx: Example of duplicating a pattern with programmed via (DuplicateXYExample.MSK)*

## Flip



To apply a rotation or a flip to one part of the design, click on Edit -> Flip and Rotate, and choose the appropriate Flip command (Horizontal or vertical). Delimit the active area of the boxes in the layout which will be modified.

## Help

Provides an on-line help for using Microwind2. Includes a summary of commands, and some details about the design rules, as shown in figure B-xxx.

The screenshot shows the 'Design Rules' dialog box with a title bar 'Design Rules'. Below it is a section titled 'Design rules and electrical parameters'. A table lists various layers (nitride, passiv, metal6, via5, metal5, via4, metal4, via3, metal3, via2, metal2, via, metal, poly) along with their respective width, spacing, surface, and various capacitive and resistive values. At the bottom right of the dialog box, it says 'Techno: CMOS 0.12µm - 6 Metal loaded from file "default.rul"'.

Layer	Width	Spacing	Surface		Surf capa	Lin capa	Ctk capa	Res	Thickn	Height	Permitt
lambda	lambda	lambda	lambda2		aF/µm <sup>2</sup>	aF/µm	aF/µm	ohm	µm	µm	
nitride	0	0	0								
passiv	1330	1330	0								
metal6	8	8	144		100.00		50.00	0.05/sq	0.70	6.60	3.10
via5	5	5	0					1.00/via	0.50	6.10	4.00
metal5	8	8	100		120.00		50.00	0.05/sq	0.70	5.40	3.10
via4	2	4	0					1.00/via	0.50	4.70	4.00
metal4	3	4	16		140.00		50.00	0.06/sq	0.50	4.20	3.10
via3	2	4	0					2.00/via	0.50	3.70	4.00
metal3	3	4	16		160.00		50.00	0.06/sq	0.50	3.20	3.10
via2	2	4	0					2.00/via	0.50	2.70	4.00
metal2	3	4	16		180.00		50.00	0.06/sq	0.50	2.20	3.10
via	2	4	0					2.00/via	0.50	1.70	4.00
metal	3	4	16		200.00		30.00	0.06/sq	0.50	1.20	3.10
poly	2	3	16		400.00			4.00/sq	0.20	0.01	4.00

Figure B-xxx: design rules and electrical rules proposed in the help menu

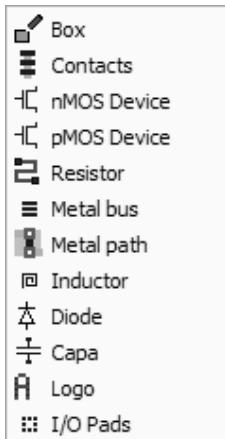
## Insert Layout

The command "File -> Insert Layout" is used to add a MSK file to the existing files. The inserted layout is fixed at the right lower side of the existing layout. The current file name remains unchanged.

## Leave Microwind2

Click on "File -> Leave Microwind2" (Or CTRL+Q) in the main menu. If you have made a design or if you have modified some data, you will be asked to save it. After confirmation, you can return to Windows.

## Generate



The layout generator includes a set of predefined layout macros such as box, contacts, n-MOS and p-MOS devices, resistor, metal bus, metal path, inductor, diode, capacitor, logo and I/O pads. Those cells are built according to design rules, and user-size parameters.

### Generate Box



Same as the « Draw Box » command described in page xxx.

### Generate Contacts



This macro generates contacts such as polysilicon/metal, n-diffusion/metal, p-diffusion/metal and metal/metal2 metal2/Metal3,(etc..), or stacked contacts can be obtained here. You may also click the above icon in the palette. Multiple contacts can be generated when entering number of contacts in X and Y greater than one.

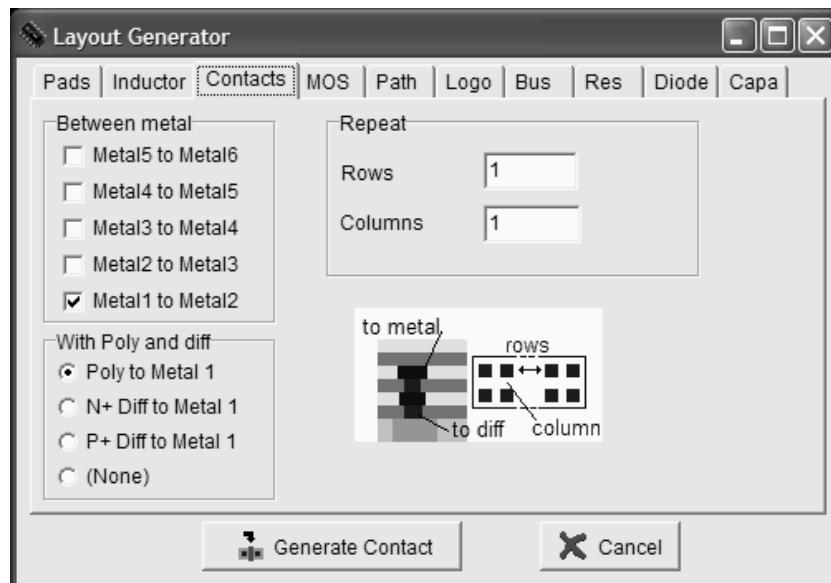


Figure B-xxx: The contact menu

**Generate nMOS, pMOS devices**

This macro generates either a n-channel or a p-channel transistor. The double gate MOS is also available in some CMOS technologies, for building the EEPROM memory. The parameters of the cell are the channel length (default value is given by the design rules), its width, and the number of gates. Once those parameters are defined, the device outline appears. Click on the mouse to place it in the appropriate place.

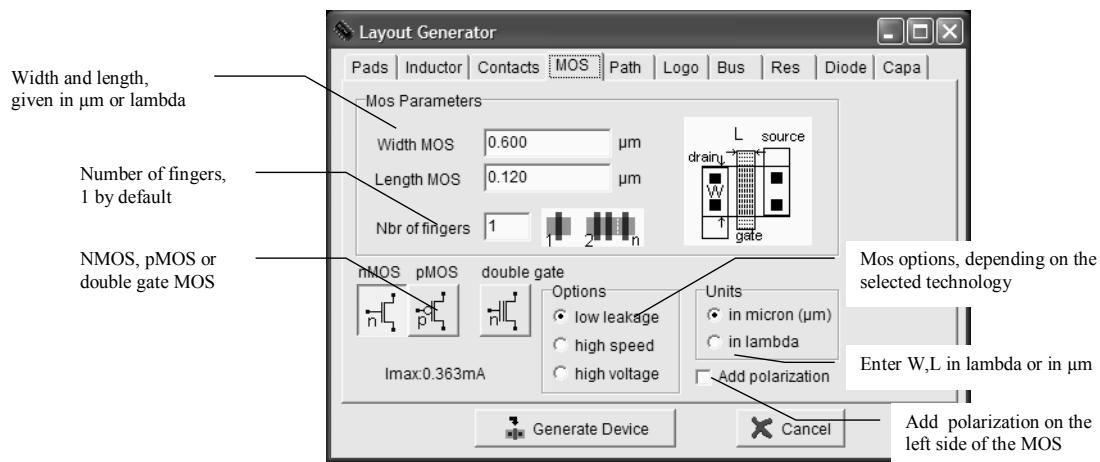


Figure B-xxx: The MOS generator menu

**Generate Resistor**

This command generates a resistor in n-well, polysilicon or poly2, N+ or P+ diffusion. The default aspect of the resistor is a Z with three bars (Parameter n in the menu). A virtual resistor symbol may be inserted in the resistor layout, to ensure the handling of the resistance effect during simulation. By default, an option layer configured to remove salicidation is added to the resistance layout. By default, all polysilicon and diffusions have a salicide surface metallization to decrease by a factor around 10 the sheet resistance. The unsalicide option is recommended for a high resistance value in a small area. Finally, contacts are added by default to the near and far ends of the resistance to facilitate further interconnection.

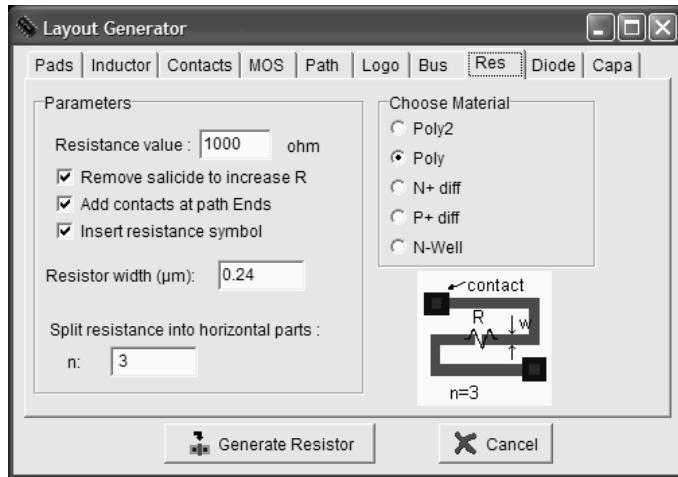


Figure B-xxx: The resistor generator menu

### Generate Inductor

This command generates a coil made from user-defined metal layers. This item is used for very-high frequency oscillators. This inductor is viewed as an inductance thanks to a virtual inductor symbol inserted in the layout. An evaluation of the inductance is proposed at a click of the button **Update L,Q**. The inductor may be a stack of several layers, thanks to the layer menu.

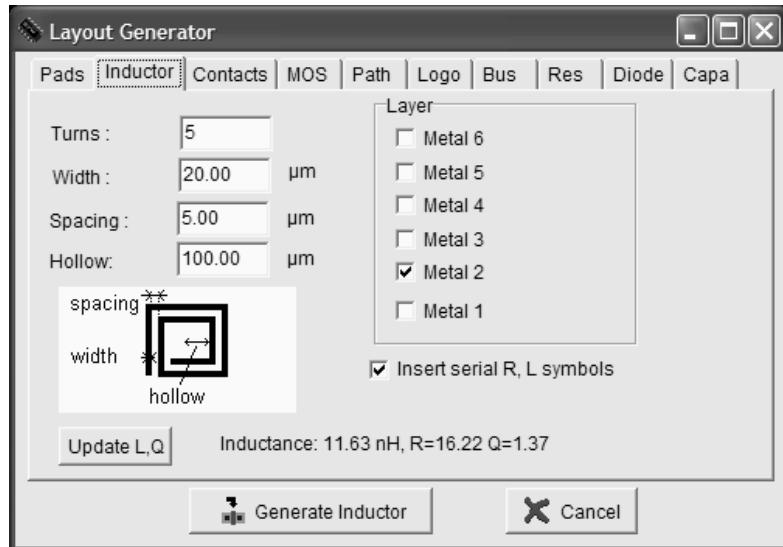


Figure B-xxx: The inductor generator menu

### Generate Bus

This command generates a set of parallel lines with user-defined layer, width and spacing. This command is useful to build coupled interconnects, or bus path used in the final routing of a chip.

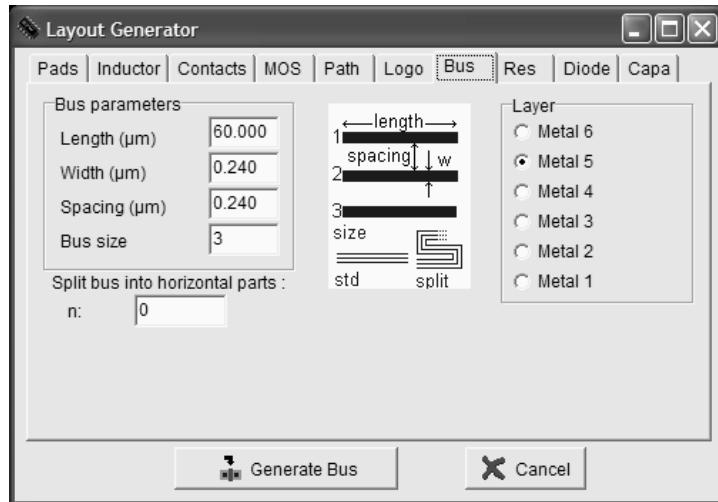


Figure B-xxx: The bus generator menu

### Generate Path

This command generates a path of interconnects using one single layer. The path width can be changed, as well as the alignment to the routing grid. A set of contacts can also be placed at both ends of the path. This command is very useful for VDD and VSS supply drawing and single layer interconnects.

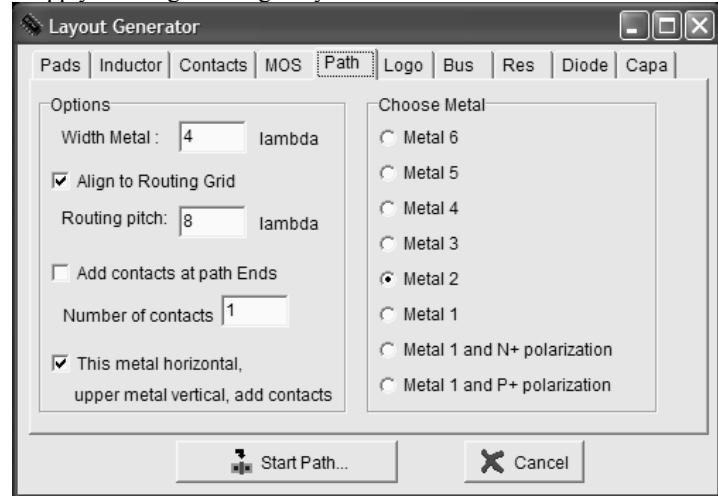


Figure B-xxx: The path configuration menu

### Generate I/O pads

It is possible to add various items such as a single pad, (usually 80x80 μm), or even a set of pads all around and to the layout using the VDD and VSS power rings. In the last case (adding more than one pad), give the number of pads each side of the chip and if need be modify the width of the VDD and VSS tracks, as well as the number of VDD/VSS pad pairs.

### Generate Diode

You may also generate a polarization seal around the contact to create a pad diode protection for example.

**Make Spice File**

Click on File -> Make Spice File to translate your design into a SPICE compatible description. The circuit extractor included in the software generates the equivalent circuit diagram of the layout and a spice compatible netlist ready to be simulated. You may select the model you will be using for simulation. The choice lies between model 1, model 3 and model 9.

- ◆ The SPICE description includes the list of n-channel and p-channel transistors and their associated width & length extracted from the layout.
- ◆ The text file also details the node names, parasitic capacitances, and device models.
- ◆ The SPICE filename corresponds to the current filename with the appendix .CIR

**Measure distance**



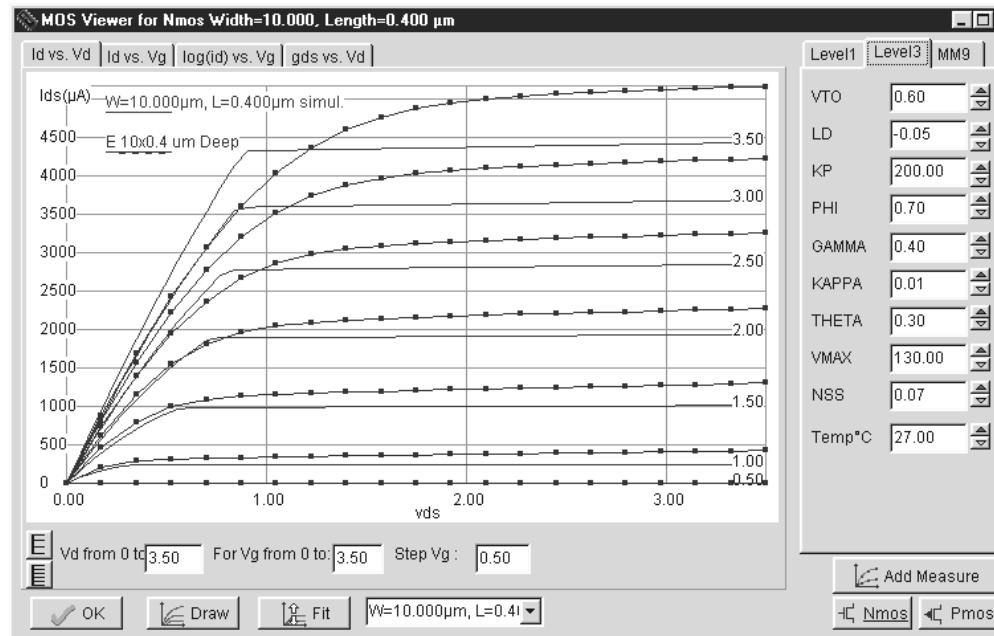
The ruler gives the horizontal and vertical measurements (dx and dy) between two points, directly on the screen in lambda and in micron. The algebraic distance (d) is also given in  $\mu\text{m}$ . The ruler is simply erased by the command View -> Refresh the screen or by a press of <ESC>.

**MOS Characteristi**



Click on the icon. The Id/Vd curve of the default MOS ( $W=20\mu\text{m}$ ,  $L=L_{\text{minimum}}$ ) appears.

- ◆ The effects of the changing of the model parameters can be seen directly on the screen by a click on the little arrows (Up/down), which change the parameter values.
- ◆ Click “Id vs. Vg” to highlight the threshold voltage
- ◆ Click “Id(log) vs. Vg” to see the sub-threshold behavior.
- ◆ Add measurements by selecting a « .MES » file
- ◆ Skip from NMOS to PMOS device by a click on the corresponding button
- ◆ Select the size of the device in the lower list menu.



Three models can be used :

- MOS Model 1 (Berkeley Spice level 1) for long channel devices. This model is obsolete but remains interesting for comparison with advanced models.
- MOS Model 3 (Simplified version of Berkeley MOS level 3). Still in use for first order estimation of the circuit performances.
- BSIM4 (Simplified version of Berkeley MOS BSIM4). The state-of the art model for deep sub-micron device modeling.

#### MOS List

Click “Edit -> MOS List” to get the list of n-channel and p-channel MOS devices currently edited in the layout. The MOS list is displayed in the navigator window. Click the desired MOS in the list to zoom at the corresponding location in the layout.

Move,  
Stretch a  
Box

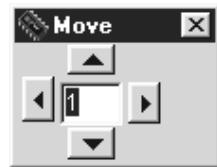


To move one box, click on the above icon. Using the mouse, create an area that includes the box. Then, drag the mouse to the new location and release the mouse. As a result, the box has been moved the new place. Repeat the same in order to move a set of boxes.

- ◆ To protect a layer from moving, click in the rectangle the palette that is situated on the right side of the layer. This will remove the tick.
- ◆ To stretch a box, click on one side of the box that you want to stretch. The box outline appears. Drag the mouse to the new location and release the button. The box is stretched.

TIP: To catch the desired border of the box, draw a line perpendicular to the border, entering the box.

- Move Step by Step**
- To move one box lambda per lambda, click “Edit > Move Step by Step”. Using the mouse, create an area that includes the boxes. The selection appears in yellow. Then, click the arrow until the selection has been moved the new place. The step value (in lambda) is fixed in the edit line.



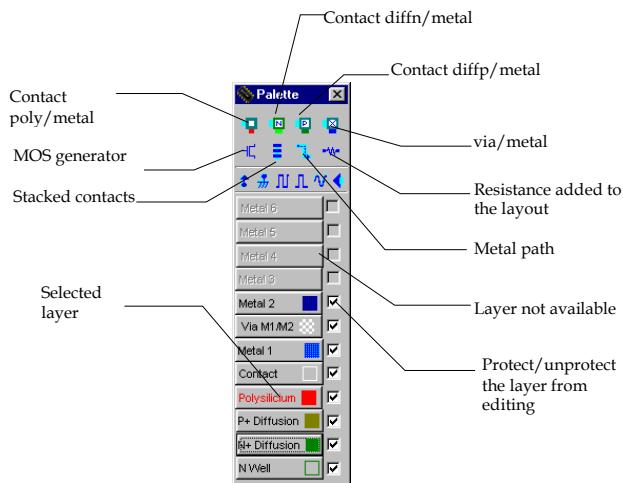
- New**
- Click on File-> New in order to restart the software with an empty screen. The current design should be saved before asserting this command, as all the graphic information will be physically removed from the computer memory. No Undo is available to disable the New command.



Click on the above icon. In the list, double-click on the file to load. « .MSK » is the default extension that corresponds to the layout files. The CIF files « .CIF » can also be loaded. The appropriate conversion program transforms the input CIF into MSK format.



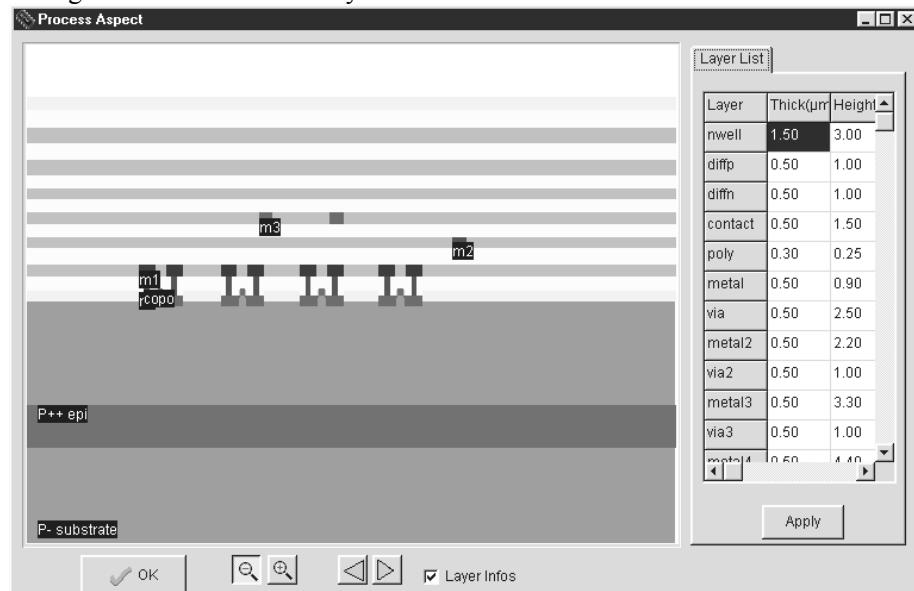
The palette is located on the right side of the screen. A little tick indicates the current layer. The selected layer by default is a polysilicon (PO). The list of layers is given below.



- ◆ If you remove the tick on the right side of the layer, the layer is switched to protected mode. The Cut, Stretch and Copy commands no longer affect that layer.
- ◆ Use “View->Protect all” to protect all layers. The ticks are erased.
- ◆ Use “View->Unprotect all” to remove the protection. All layers can be edited.

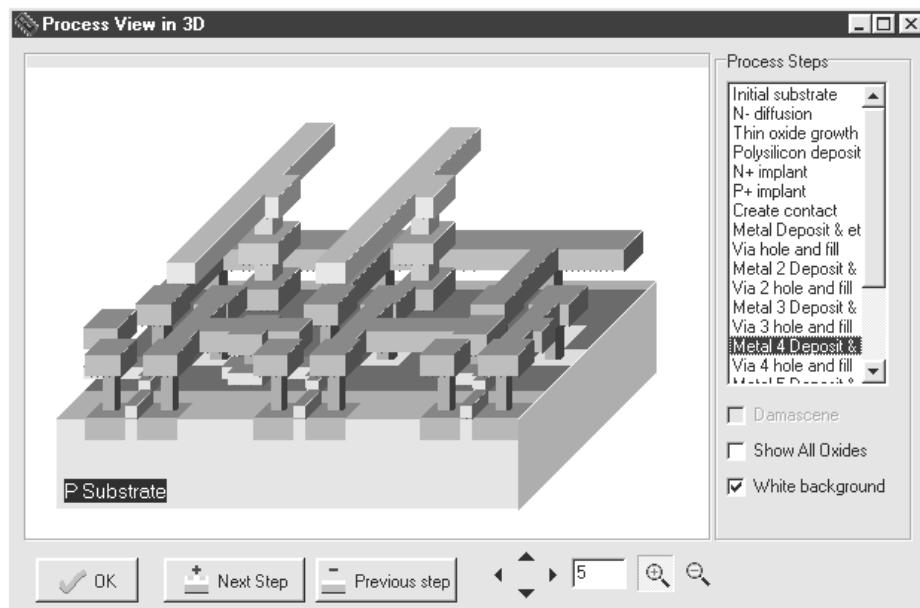
### Process Section in 2D

Click on the above icon to access process simulation. A mouse-operated line is given and embodies the cross section. The screen below appears. The arrows can be used to move the cross-section to the right or to the left in the X axis, and forward and backward in the Y axis. Zooms in and out are available. Remove the layer names by removing the tick in front of “Layer infos”.



### Process Section in 3D

Click “Simulation -> Process Steps in 3D”. Click “Next step” to watch how the layout currently edited on the screen will be fabricated using the selected technology. Use the arrow to shift the displayed portion. Zooms in and out are available.



### Protect All

Click on “View ->Protect All” to protect all layers for editing purpose. All ticks in the palette are removed.

### Print Layout

Click on File ->Print Layout to transfer the graphical contents of the screen to the printer. Alternatively, you can make a copy of the window into the clipboard in order to import the screen into your favorite text editor by pressing **<Alt>+<Print Screen>**. In the text editor or in the graphic editor, simply click on « Edit ->Paste » We recommend that you switch to monochrome mode first by invoking the function File -> Colors -> Switch to Monochrom. In that case the layout will be drawn in a white background color using gray levels and patterns.

### Rotate

To apply a rotation to one part of the design, click on Edit -> Rotate . Delimit the active area of the boxes in the layout so that it can be modified using the mouse.

### Save



Click on File -> Save to save the layout with its current name. The default name is « EXAMPLE.MSK »

### Save As

A new window appears, into which you are to enter the design name. Use the keyboard and type the desired file name. Press « Save ». Your design is now registered within the .MSK appendix.

### Search Text

The most convenient way to find a text in the layout is to invoke Edit -> Search text.

The list of text labels appears in the navigator menu. If you click on the desired text, the screen is redrawn so that the text label is at the center of the window, with two lines drawing a cross at the text location. Its properties appear in the navigator menu.

- ◆ Click on Hide to close the navigator window.
- ◆ Click on Extract to add the electrical properties of the selected text if the layout has not been previously extracted.
- ◆ In the case of a very long text list, select the first letter of the text at hand, press that letter on the keyboard. This will automatically effect an alphabetic research and the selector will move to the first label starting with the selected letter.

### Select Foundry

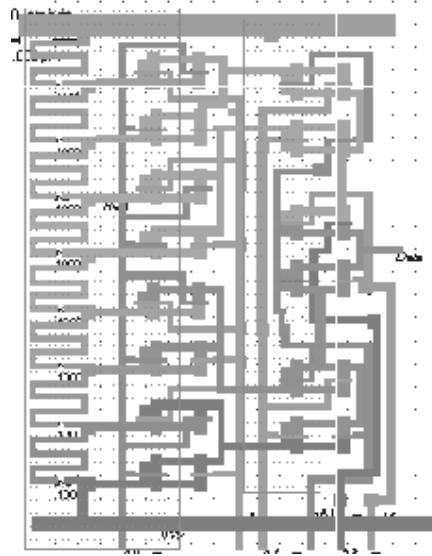
Click on File -> Select Foundry. The list of available processes appears. The default design rule file is written in bold characters. Various technologies are available from 1.2 down to 0.12  $\mu\text{m}$ . Click on the rule file name and the software reconfigures itself in order to adapt to the new process

### Simulation Parameters

- ◆ The default extraction includes the removal of redundant boxes (Purge) and the removal of overlaps (Merge). The fast extraction does not handle Purge nor Merge operations.
- ◆ The MOS level can be chosen between level 1,3 and 9. See chapter 2 for more details about those models.
- ◆ Other options concern the computation of lateral capacitance and vertical crosstalk capacitance

### Simulation on layout

The simulation is performed directly on the layout with a palette of colors. The most interesting layout files to be simulated in this mode are analog blocks such as the DAC.



During switching, the MOS devices of the inverter behave alternately as closed and opened switches, as illustrated earlier. In a first order approximation, the equivalent model of the switch is a resistance. In the illustration of figure 4-51, we plot the operating point of the n-channel MOS and p-channel MOS during switching. What we see is a complex trajectory of the operating point in the  $Id/Vd$  characteristics, both for the nMOS and the pMOS devices. Therefore, the resistance is not a simple function, but rather a varying resistance between several values.

The figure 4-xxx has been obtained using a specific simulation mode called **Simulation on Layout** in the simulation menu. The I/V characteristics of the selected devices are updated during simulation to track the functional point.

During the analog simulation, the node voltage is superimposed to the layout appears with a palette of colors:

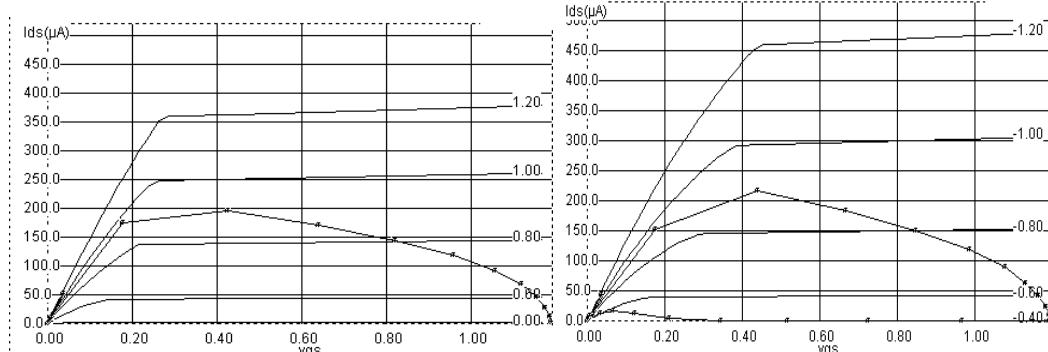


Fig. 4-51: The nMOS and pMOS devices during switching (Inv.MSK)

### Start Simulation



The above icon or the command Simulate -> Start Simulation both give access to the automatic extraction and analog simulation of the layout.

- Click on Voltage vs Time to obtain the transient analysis of all visible signals. The

delay between the selected start node and selected stop node is computed at VDD/2.

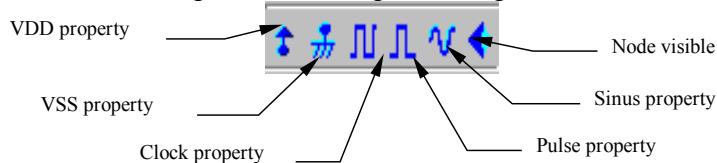
You can change the selected start node in the node list, in the right upper menu of the window. You can do the same for the selected stop node.

- Click on Voltage and Currents so as to make all voltage curves appear in the lower window, and the VDD, the VSS and the desired MOS currents appear in the upper window. In that mode, the dissipated power within the simulation is also displayed.
- Click on Voltage vs. Voltage to obtain transfer characteristics between the X-axis selected node and the Y-axis selected node. Initially the start node is the first clock or pulse of the node list, and the stop node is the first varying node. This mode is useful for the computing of the Inverter characteristics (commutation point), the DC response of the operational amplifier, or for the Schmitt trigger to see the hysteresis phenomenon. The first simulation computes the value of the stop node for start node varying from 0 to VDD. The second click on « Simulate » computes the same for start node varying from VDD to 0. This feature is interesting for circuit with memory effects (Schmidt trigger). Note that the curves may not be exactly the same. You may increase the precision by reducing the computational step "Precision", accessible in the menu, expressed in mV.

NOTE : You can modify the minimum simulation step  $\Delta t$ , but it may be dangerous. If you increase  $\Delta t$  the simulation speed improves but the numerical error may lead to unstable simulations. If you decrease  $\Delta t$ , the simulation speed is decreased too but the numerical precision is improved. The risk of computing divergence is reduced.

## Simulation Icons

The simulation icons add properties to the nodes. Properties are applied to the electric nodes of the circuit in order to serve as simulation guides. You must specify which node is assigned to which voltage before starting the analog simulation.



VDD & VSS. The node is pushed to the power supply voltage with icon Vdd, and pulled to the ground 0V with icon Vss.

CLOCK. When a node becomes a clock, the parameters of the latter are divided as follows : rise time, level one, fall time, and level zero. All values are expressed in nanosecond (ns). If you ask for a second clock, the period will be multiplied by two.

- ◆ You may alter level 0 and level 1 by entering a new value with the keyboard.
- ◆ To generate a clock starting from VDD instead of VSS, click on Invert L/H.
- ◆ Use Period \* 2 to multiply the clock period by two.

- ◆ Use Period /2 to divide the clock period by two.

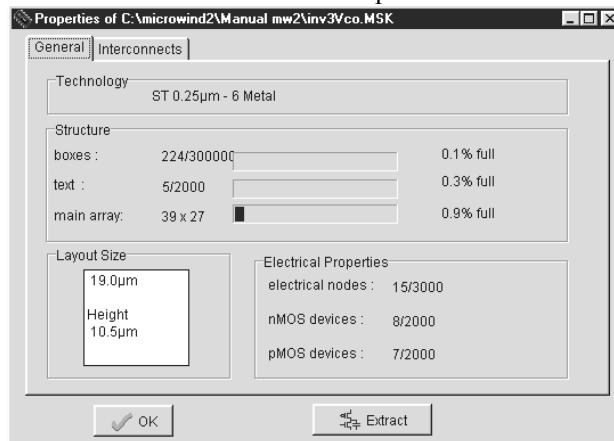
PULSE. The pulse switches from “Level 0” (0 by default) to “Level 1” (VDD by default” depending on the user-defined time table.

- ◆ Enter the string “0101100” and press “Insert”. The time-table is updated
- ◆ Click “Erase”: all lines situated after the selected element of the time-table are erased.

SINUS. The sinusoidal waveform parameters are the amplitude, the offset, frequency and phase.

VISIBLE NODE. Click on the “eye” and click on the existing text in the layout to make the chronograms of the node appear. Initially, all nodes are invisible, but the clocks and impulse nodes are subsequently made visible.

**Statistics** The command File -> Statistics provides some information about the current technology, the percentage of memory used by the layout and the size of the layout plus its detailed contents. If the layout has previously been extracted or if you click « extract now », the number of devices and nodes will be updated.



**Undo** The Undo command (Edit -> Undo) is useful to not take into account the last editing command. It is possible to undo the commands Cut, Paste, Copy, Move, Stretch, Edit and Compile.

**Unprotect All** Click on “View ->Unprotect All” to select all layers for editing purpose. All ticks in the palette are asserted.

**Unselect All** Click on View -> Unselect All (or <ESC>) to unselect the layout. This command is useful to draw the layout back into its default colors after commands such as View

Interconnect or View Node which highlight one single node.

View All

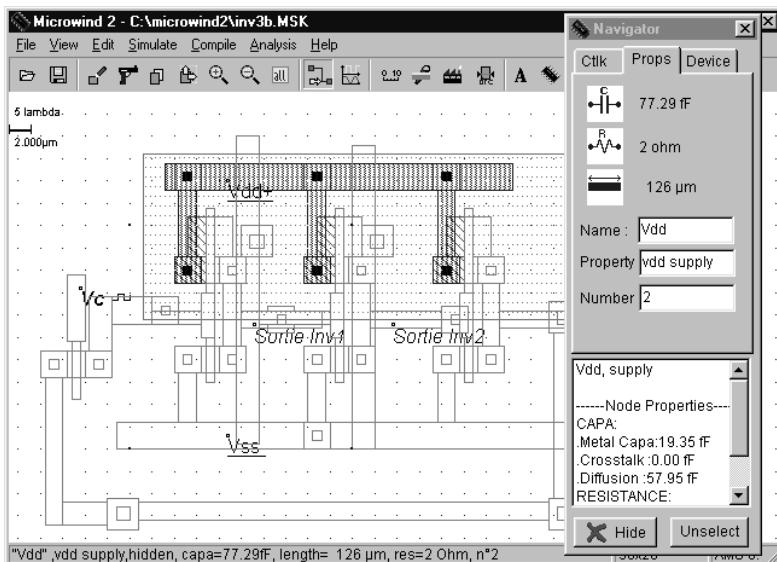


Click View -> View All to fit the screen with all the graphical elements currently on display.

View  
Node



Click on the icon above or on View ->View Node. Then, click in the desired box in the layout. After an extraction procedure has been carried out, you will see that all the boxes connected to that node. In the case of a large layout, the command may take time. The associated parasitic capacitance, the list of text labels added to the selected boxes, and the node properties are also displayed in a separate navigator window. Click “Unselect”, “Hide”, <Escape> or View -> Unselect All to unselect the layout.

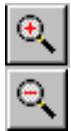


View  
Interconn-  
ect

The command View -> View Interconnect performs an electrical extraction of the metal and polysilicon boxes connected to the desired point. Compared to View Node, this command works faster but does not consider diffused layers that can extend the node interconnect network. The command gives the list of connected text labels.  
Click on <Escape> or on View -> Unselect All to unselect the layout.

Zoom In  
& Out

The above icons perform Zoom In and Zoom Out. When zooming in, the area determined by the mouse will be enlarged to fit the display window. When zooming



out, the area determined by the mouse will contain the display window.

- ◆ If you click once, a zoom is performed at the desired location.
- ◆ Press Ctrl+A for « View All », and Ctrl+o for zoom out.

# C Dsch Logic Editor Operation and Commands

## 1. Getting Started

To get your DSCH2 program started, use the following procedure:

- ① Insert the companion CD-Rom into drive
- ② Under Windows 98/NT/XP, click the file "Index.html"
- ③ Click "Install DSCH2". The Dsch2.ZIP file is copied to the hard disk at the desired location.
- ④ Unzip the Dsch2.ZIP file.
- ⑤ Double click the DSCH2.EXE icon

The software runs on Windows 95, 98, NT and XP operating systems.

## 2. Commands

### About Dsch2

Information about the software release and support.

### Connect



Use the "Connect" icon to create the electrical contact between crossing interconnects.

### Cut



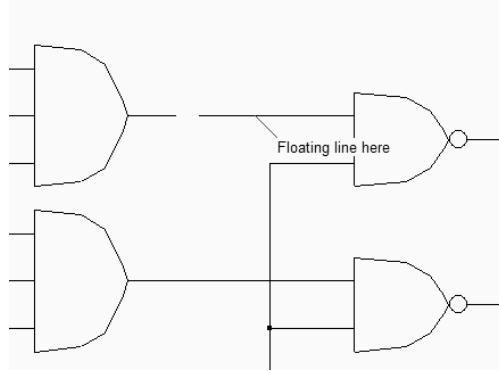
(CTRL+X)

Click on the Cut icon. Move the cursor to the design window, and delimit the active area with the mouse.

Consequently, all the graphics included in this area are erased. Click on Undo to fix those elements back into the design. One symbol only can be erased by a click inside its shape when the cut command is active. The symbol is then erased. One single interconnect can be erased by a click on its wire when the cut command is active.

### Check Floating Lines

The command "**Simulate → Check Floating Lines**" may be found in the Simulation menu. The schematic diagram is scanned in order to detect interconnects with a wrong connection to the symbol or other interconnects, as in the example of figure C-1.



*Fig. C-1: Example of floating line*

### Copy



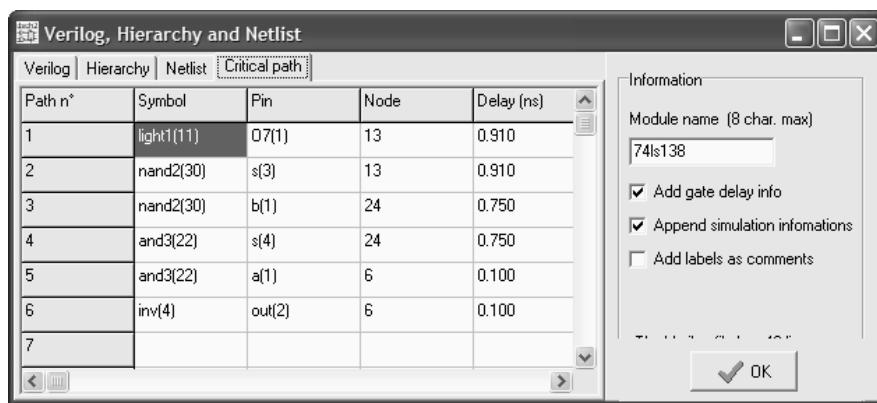
(CTRL+C)

Click on the Copy icon. Move the cursor to the design window, and delimit the active area with the mouse. Consequently, all the graphics included in this area are copied. The external shape of the copied elements appears. Fix those copied elements at the desired location by a click on the mouse.

Click on Undo to cancel the copy command.

### Critical Path Details

The critical path is the series of logic gates between the output and input with the longest propagation delay. The command **View → Critical Path Details** gives the list of symbols and cumulated delays which build the critical path. The graph of the critical path may be highlight using the command **Simulate → Find Critical Path**.



*Fig. C-2: Critical path details*

## Design Hierarchy

The design hierarchy command gives an interesting insight in the hierarchical structure of symbols, together with the list of input and output symbols.

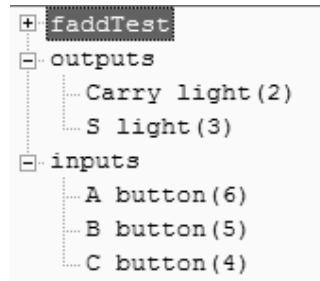


Figure C-3: An example of design hierarchy

## Electrical Net



Click on the icon above or on View → Electrical Net. Then, click in the desired interconnect or pin in the schematic diagram. After an extraction procedure has been carried out, you will see that all the wires connected to that node. Click <Escape> or View → Unselect All to unselect the diagram.

## Find Critical Path

The critical path is the series of logic gates between the output and input with the longest propagation delay. The command **Simulate → Find Critical Path** shows the graph of the critical path. Invoke the command **View → Critical Path Details** to extract the list of symbols and cumulated delays which build the critical path.

## Flip Vertical/Horizontal

To apply an horizontal or vertical flip to one part of the design, click on **Edit → Flip**. Then, delimit the area inside which the elements will be changed.

## Help

Provides an on-line help for using Dsch2. Includes a summary of commands, some details about the design rules, and some precision about the current version of the software.

## Insert another Schema

The command "**File → Insert an other Schema**" is used to add a SCH file to the existing files. Its contents is fixed at the right lower side of the existing schematic diagram. The current file name remains unchanged.

### **Insert User Symbol**

The command **Insert → User Symbol** is used to add a user defined symbol to the existing schematic diagram. The user symbol is created using the command **File → Schema To new Symbol**. The inserted symbol can be fixed at the desired location.

### **Leave Dsch2**

Click on "File → Leave Dsch2" (Or CTRL+Q) in the main menu. If you have made a design or if you have modified some data, you will be asked to save it. After confirmation, you can return to Windows.

#### **Line**



(Or right click with the mouse)

The " Line " icon is the default icon. It creates an interconnection between two points in the schematic diagram. If the " Line " icon is not selected, click on it. Then, move the cursor to the display window and fix the start point of the interconnect with a press of the mouse. Keep pressed and drag the mouse to the interconnect end. Release the mouse and see how the line is created.

### **List of Symbols**

The command gives the complete netlist corresponding to the schematic diagram. The internal structure of hierarchical symbols also appears. The symbol name, list of pins, related node numbers and modelm number are listed.

### **Make Verilog File**

DSCH2 converts the schematic diagram into VERILOG using a specific interface, invoked by **File → Make Verilog file**. The Verilog text can be exported to VLSI CAD software. The right table of the screen (Figure C-4) gives the list of options: module name, gate delay information, list of labels, and general information about the size of the design. The conversion of the schematic diagram into a VERILOG description is useful for compiling the schematic diagram into layout using Microwind. The Verilog description is a text with a predefined syntax. Basically, the text includes a description of the module (name, input, output), the internal wires, and the list of primitives. An example of Verilog file generated by DSCH is given below.

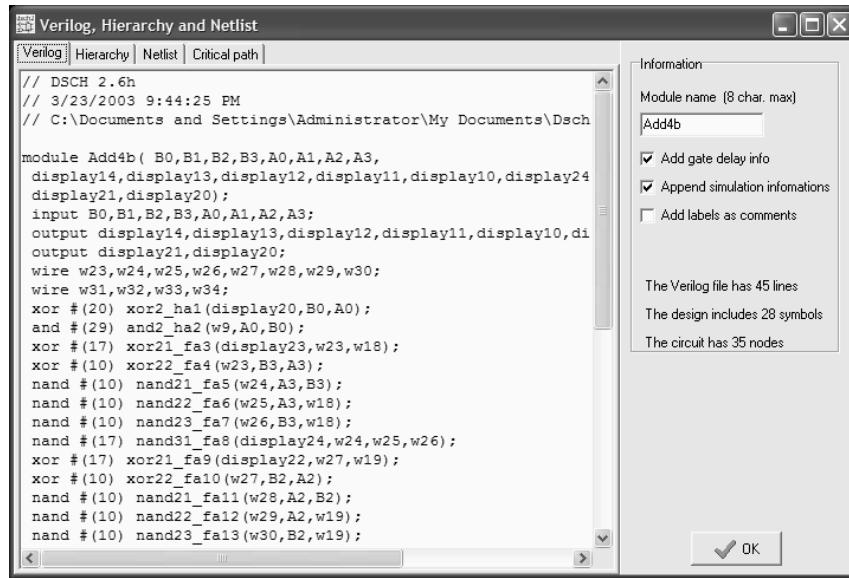


Figure C-4: Conversion into Verilog

### Monochrom/Color (File Menu, F5)

Switch to monochrome mode: the layout is drawn in black and white. This type of drawing is convenient to build monochrome documentation. Press "Alt"+"Print Screen" to copy the active window to the clipboard. Then, open "Word", click **Edit→Paste**. The screen is inserted into the document.

### Move

To move one graphical element, click on the "Move" icon or "**Edit → Move**". Then using the mouse, draw an area that includes the elements. Then, drag the mouse to the new location and release the mouse. As a result, the elements are moved the new place. One single line can be moved or stretched (depending where you click) by a direct click on the line. One single text can be moved by a direct click on the text location.

### New

Click on **File→ New** in order to restart the software with an empty screen. The current design should be saved before asserting this command, as all the graphic information will be physically removed from the computer memory. No Undo is available to disable the New command.

### Open



Click on the above icon. In the list, double-click on the file to load. ".SCH" is the default extension that corresponds to the schematic diagrams.

### Paste

Invoke the *Paste* command **Edit→ Paste**. All previously copied elements are pasted at the desired location. Deleted elements can be replaced that way. Click on **Undo** to cancel the *paste* command.

### Print

Click on **File → Print Layout** to transfer the graphical contents of the screen to the printer. Alternatively, you can make a copy of the window into the clipboard in order to import the screen into your favorite text editor by pressing **<Alt>+<Print Screen>**. In the text editor or in the graphic editor, simply click on « **Edit → Paste** » We recommend that you switch to monochrome mode first by invoking the function **File → Monochrome/color**. In that case the layout will be drawn in a white background color using gray levels and patterns.

### Properties

The command **File → Properties** provides some information about the current technology, the percentage of memory used by the schematic diagram and its detailed contents. In the **Technology** part, details on the time unit, voltage supply, typical delay and typical wire delay are provided, which configure the delay estimation and current estimation during logic simulation.

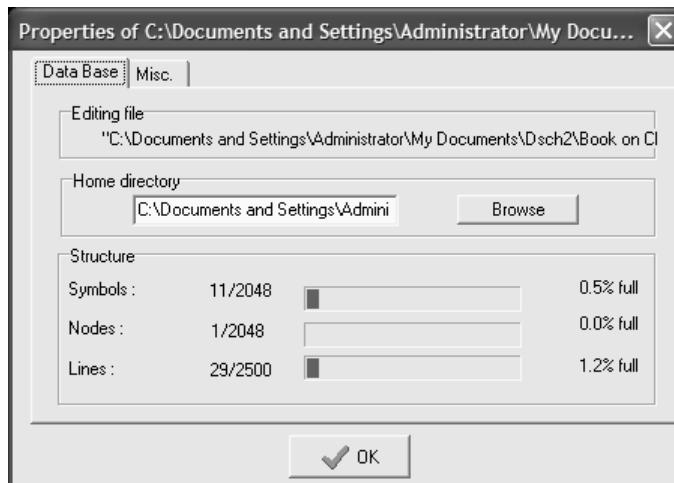


Fig. C-5: File properties, including statistics about the number of symbols, nodes and lines

### Rotate

To apply a rotation to one part of the design, click on **Edit → Rotate**. Select one of the proposed action:

- Rotate right or 90°
- Rotate left or -90°

Then, delimit the area inside which the elements will be rotated.

### Save, Save As

Click on **File → Save** to save the schematic diagram with its current name. The default name is " EXAMPLE.SCH ". In the case of "Save As...", a new window appears, into which you are to enter the design name. Use the keyboard and type the desired file name. Press " Save ". Your design is now registered within the **.SCH** appendix.

### Select Foundry

Click on **File → Select Foundry**. The list of available processes appears. The initial design rule file is "default.tec". Various technologies are available from 1.2 $\mu$ m down to 90nm. Click on the rule file name and the software reconfigures itself in order to adapt to the new process.

### Simulation Options

The simulation parameters are: the simulation step (10ps by default), the gate delay, wire delay, supply voltage, and elementary gate current. These parameters are loaded from .TEC files at initialization or with the command **File → Select Foundry**.

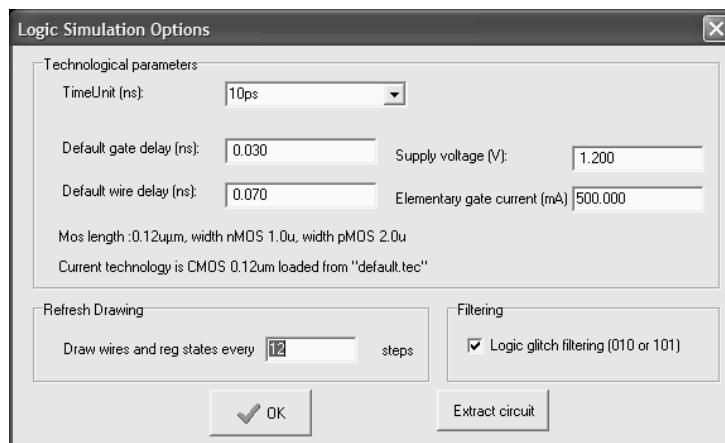


Fig. C-6: Logic simulation parameters

### Start Simulation

The command "**Start Simulation**" launches the electrical net extraction and the logic simulation. The simulation speed may be controlled by the cursor "Fast-Slow". The simulation may be paused, run step by step and stopped. By default, the logic state of all interconnects is made visible. You may also see each pin state by a tic in front of "Show pin state".



Fig. C-7: The logic simulation control window

### Schema to New Symbol

This command is very important to create user-defined symbols in order to build hierarchical designs. As an example, the full adder diagram based on primitives can be translated into a single symbol which includes the structure, input and outputs, as shown below.

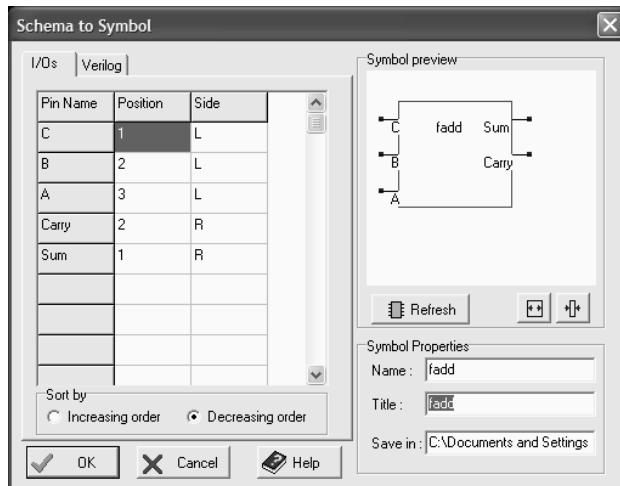


Fig. C-8: Logic simulation parameters

I/Os. The list of I/Os is based on active symbols (buttons, clocks, leds, keyboards, etc...). The position and side in the symbol may be changed in the table.

VERILOG. The structural description based on primitives is described in verilog format and included to the symbol description.

REFRESH. Update the layout of the user symbol.

SIZING. Act on the icons to change the shape of the user symbol.

SYMBOL PROPERTIES. This properties may be changed by the user.

### Symbol Library

The symbol library contains basic logic and electrical symbols, sources, displays and switches. The aspect of the logic library is reported in figure C-9. Most standard logic symbols (Inverter, Buffer, NAND, AND, NOR, OR, XOR) and D-latches are part of the "Basic" symbol menu. The analog components such as resistor, inductor, capacitor,

operational amplifiers are reported in the "Advanced" menu. Notice several input/output symbols, as well as a variety of switches for programmable arrays. Some more symbols may be found in the IEEE directory, accessible through the command **Insert → User Symbol**.

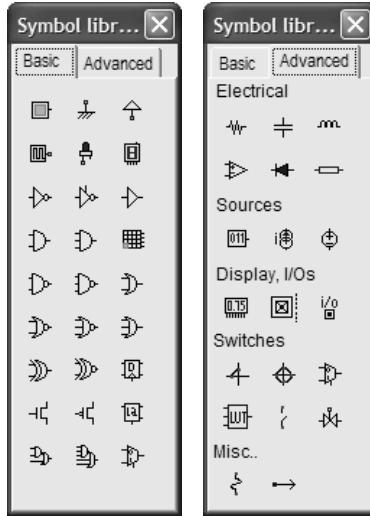


Fig. C-9: The symbol library

### Text



Use this icon to fix a text to one box or location in the design. That text illustrates the layout and should be used as much as possible for each significant node such as inputs and outputs. To add some text to a particular place, proceed as follows:

- ① Click on the icon
- ② Set the text location with the mouse. A dialog box appears
- ③ Enter the text in front of "Text:" and press "Ok". The text is set in the drawing

A text can be modified as follows: click on the icon, click inside the existing text. The old text appears. Modify it and click on "Ok". Text is added for information only. It has no impact on simulation.

### Timing Diagram

The timing diagram gives the time-domain aspect of all input and output nodes. An example of timing diagram is reported in figure C-10. You may zoom on a specific time window, add the evaluation of the consumed current, and get the exact value of each input/output at a desired location.

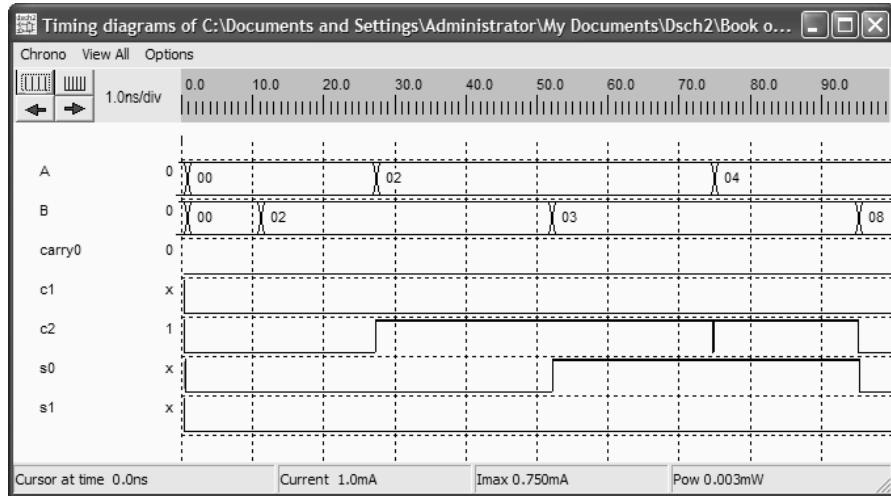


Fig. C-10: The timing diagrams of a logic simulation

**Undo**

The Undo command (**Edit → Undo**) is useful to not take into account the last editing command. It is possible to undo the commands Cut, Paste, Copy, Move, Stretch, & Edit.

**Unselect All**

(Escape Key)

Use the command (**View → Unselect All**) to cancel undesired commands, or to redraw the complete schematic diagram.

**View All**

Click **View → View All** to fit the screen with all the graphical elements currently on display.

**View Same**

Draw again the schematic diagram without changing the scale. Used to refresh the screen.

**Zoom In & Out**

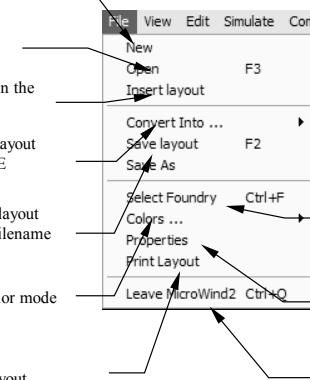
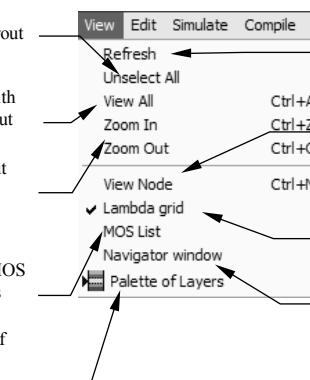
The above icons perform Zoom In and Zoom Out. When zooming in, the area determined by the mouse will be enlarged to fit the display window. When zooming out, the area determined by the mouse will contain the display window.

- If you click once, a zoom is performed at the desired location.
- Press Ctrl+A for « View All », and Ctrl+o for zoom out.

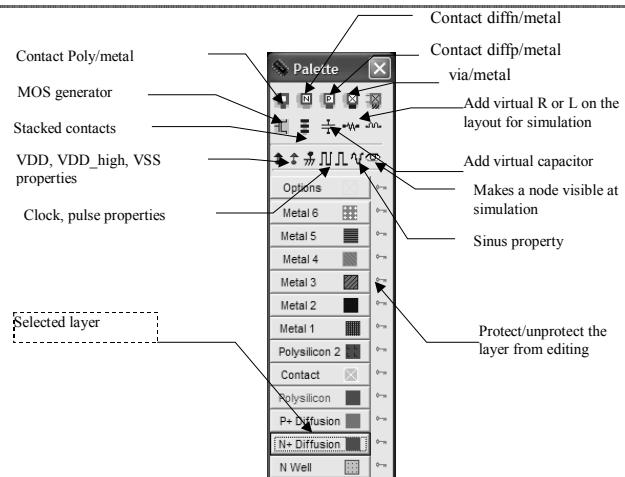
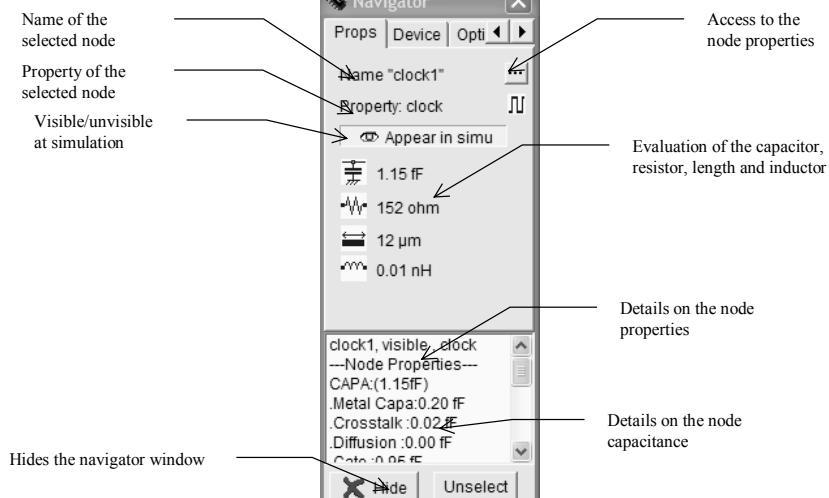
# D

## Quick Reference Sheet

### 1. Microwind2 Menus

<b>FILE MENU</b>	 <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>Select Foundry</p> <p>Colors ...</p> <p>Properties</p> <p>Print Layout</p> <p>Leave Microwind2</p> <p>Ctrl+F</p> <p>F2</p> <p>F3</p> <p>Ctrl+Q</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>
<b>VIEW MENU</b>	 <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>Refresh</p> <p>Unselect All</p> <p>View All</p> <p>Zoom In</p> <p>Zoom Out</p> <p>View Node</p> <p>Lambda grid</p> <p>MOS List</p> <p>Navigator window</p> <p>Palette of Layers</p> <p>Ctrl+A</p> <p>Ctrl+Z</p> <p>Ctrl+O</p> <p>Ctrl+N</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>

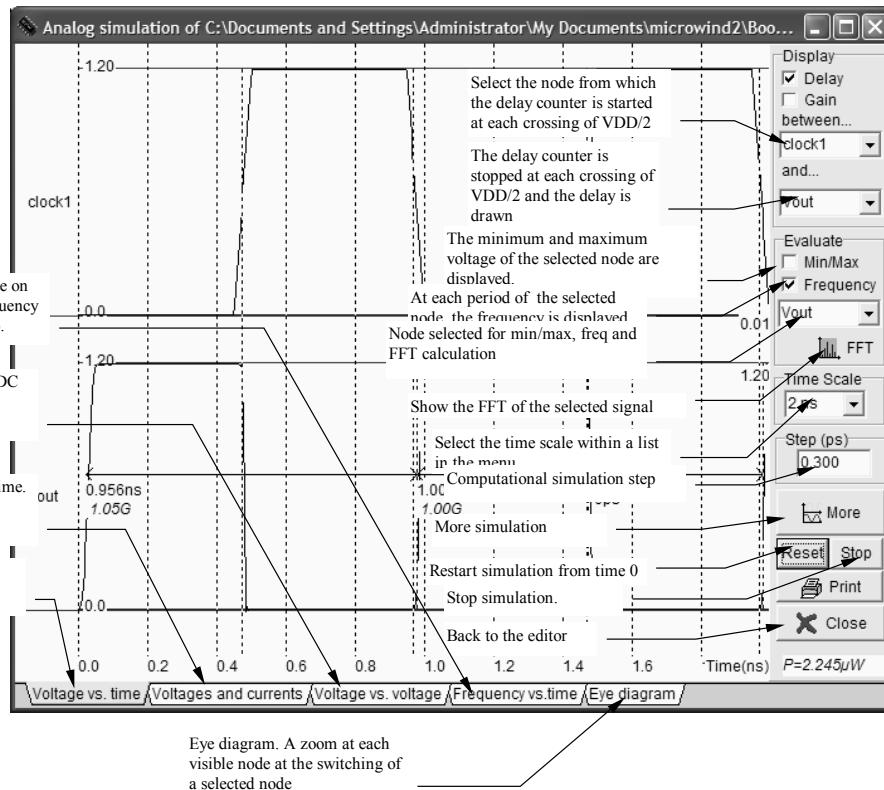
<b>EDIT MENU</b>	<p>Cancel last editing command</p> <p>Cut elements included in an area</p> <p>Duplicate elements included in an area</p> <p>Flip or rotate elements included in an area</p> <p>Generate MOS, contacts, pads, diodes, resistors, capacitors, etc...</p> <p>Connect layers at a desired location</p>
<b>SIMULATE MENU</b>	<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>
<b>COMPILE MENU</b>	<p>Compile one single line (on-line)</p>
<b>ANALYSIS MENU</b>	<p>Verifies the layout and highlight the design rule violations</p> <p>Measure the distance in the layout window, in <math>\mu\text{m}</math> and lambda</p>

**PALETTE****NAVIGATOR  
WINDOW**

**LIST OF ICONS**

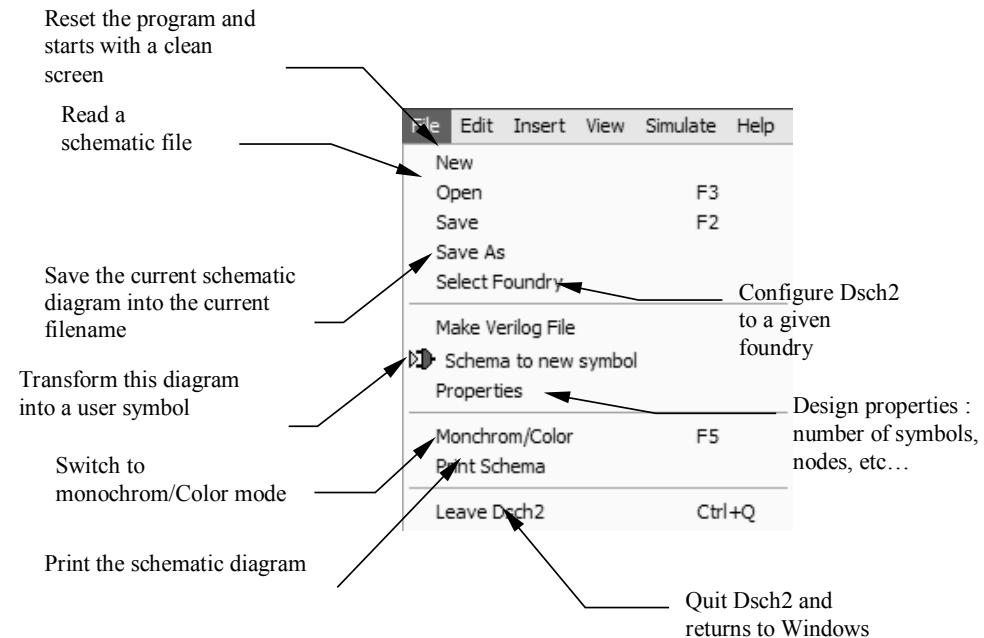
	Open a layout file (MSK format)		Extract and simulate the circuit
	Save the layout file in MSK format		Measure the distance in lambda and micron between two points
	Draw a box using the selected layer of the palette		2D vertical aspect of the device
	Delete boxes or text.		Step by step fabrication of 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

## 2. Microwind2 Simulation menu

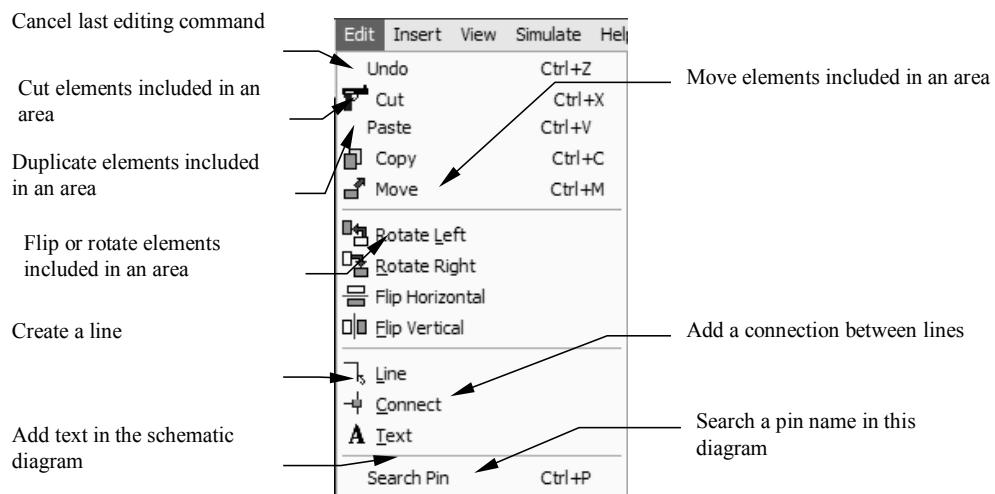


### 3. Dsch2 Menus

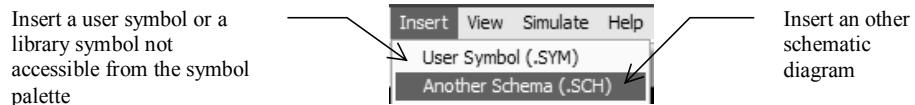
#### File Menu



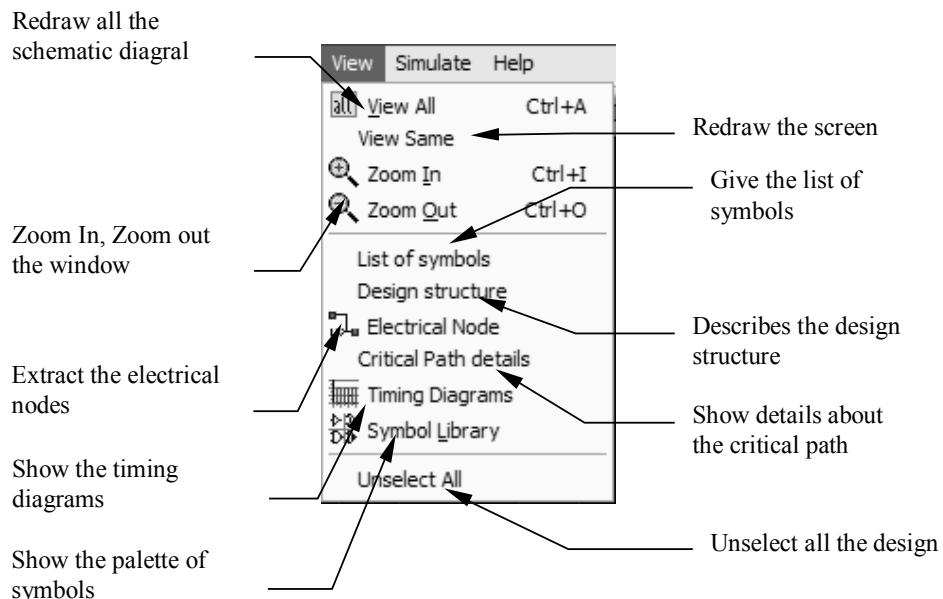
#### Edit Menu



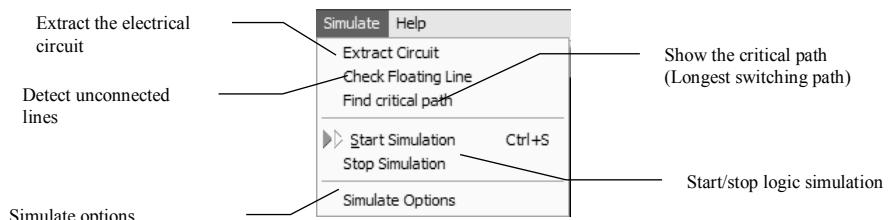
## Insert Menu



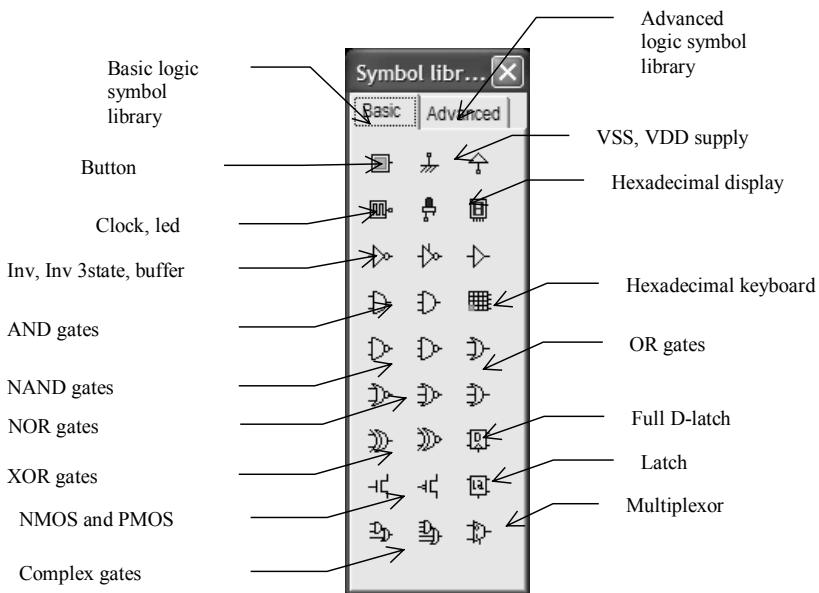
## View Menu



## Simulate Menu



## Symbol Palette



## 4. 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

## 5. 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 $\mu$ m ST)

Chip « b » (0.25 $\mu$ m ST)

Chip « c » (0.18 $\mu$ m ST)

Chip "d" (0.8 $\mu$ m ATMEL-ES2)

Measurement files	Description
0.35 $\mu$ 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 $\mu$ m, L=0.4 $\mu$ m (0.35 effective) Nmos W=10 $\mu$ m, L= 10 $\mu$ m Nmos W=10 $\mu$ m, L= 2 $\mu$ m Nmos W=1 $\mu$ m, L= 0.4 $\mu$ m Nmos W=80 $\mu$ m, L= 0.4 $\mu$ m Pmos W=10 $\mu$ m, L= 0.4 $\mu$ m Pmos W=10 $\mu$ m, L= 10 $\mu$ m Pmos W=10 $\mu$ m, L= 2 $\mu$ m Pmos W=1 $\mu$ m, L= 0.4 $\mu$ m Pmos W=80 $\mu$ m, L= 0.4 $\mu$ m
0.25 $\mu$ m CMOS Nb10x0,25.mes Nb10x10.mes	Nmos W=10 $\mu$ m, L=0.25 $\mu$ m Nmos W=10 $\mu$ m, L=10 $\mu$ m
0.18 $\mu$ m CMOS Nc10x10.mes NcHS4x0.2.mes NcHV4x0.2.mes Nc4x0.2.mes	Nmos W=10 $\mu$ m, L=10 $\mu$ m Nmos W=4 $\mu$ m, L=0.2 $\mu$ m high speed option Nmos W=4 $\mu$ m, L=0.2 $\mu$ m high voltage option Nmos W=4 $\mu$ m, L=0.2 $\mu$ m normal (low leakage)
0.8 $\mu$ m CMOS Nd20x20.mes Nd20x0,8.mes	Nmos W=20 $\mu$ m, L=20 $\mu$ m Nmos W=20 $\mu$ m, L=0.8 $\mu$ m

### 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
```

# E Solutions for exercises

## 1. Chapter 1

### Exercise 1.1

<tbd>

### Exercise 1.2

<tbd>

### Exercise 1.3

<tbd>

### Exercise 1.4

<tbd>

## 2. Chapter 2

### Exercise 3.1

<tbd>

### Exercise 3.2

<tbd>

### Exercise 3.3

<tbd>

### Exercise 3.4

<tbd>

### 3. Chapter 3

#### **Exercise 3.1**

<tbd>

#### **Exercise 2.2**

<tbd>

#### **Exercise 2.3**

<tbd>

#### **Exercise 2.4**

<tbd>

## 4. Chapter 4

### Exercise 4.1

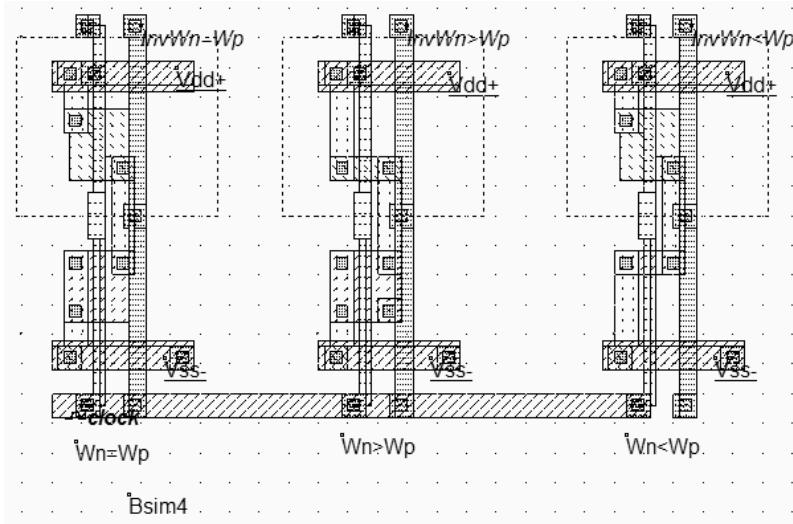


Figure F-xxx: the three inverters and their associated characteristics (Ch41.MSK)

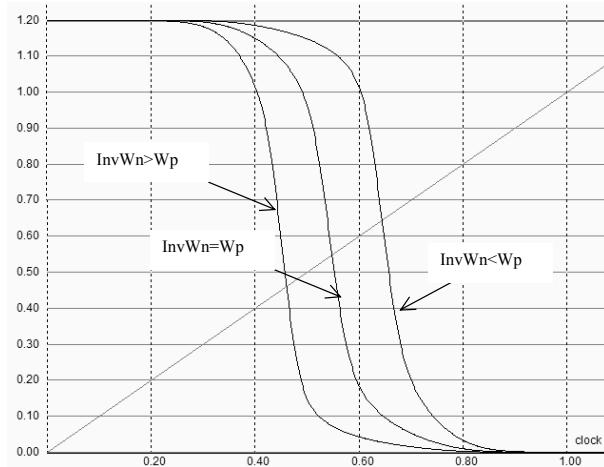


Figure F-xxx: Static characteristics of the three inverters (Ch41.MSK)

### Exercise 4.2

$$t_{PLH2} = \frac{2}{3} * t_{PLHI}$$

### Exercise 4.3

<Sonia>

**Exercise 4.4**

&lt; Sonia &gt;

**Exercise 4.5**

&lt; Sonia &gt;

**Exercise 4.6**

To understand the difference click on Simulate/Simulate on Layout and select the PMOS. <Sonia>

**Exercise 4.7**

The evolution of the circuit *inv5.MSK* with the process parameter fluctuations is obtained using the Monte Carlo simulation accessible in the **Analysis → Parametric Analysis** menu. Click for example the output node n5, then clock **M.carlo** index (Figure F-xxx), and **Start Analysis**.

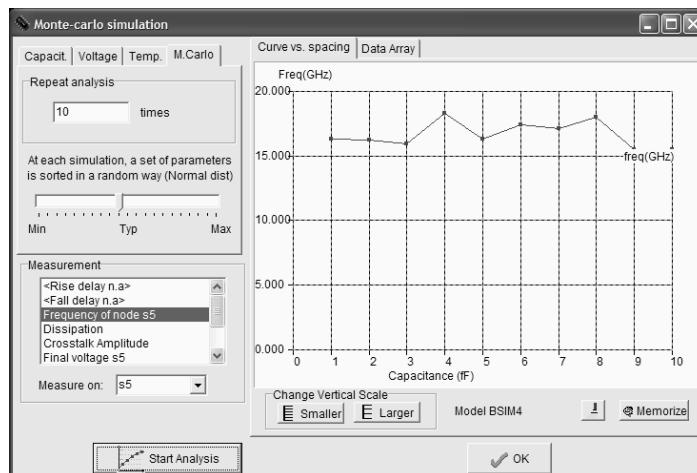


Figure F-xxx: Monte-Carlo analysis of the oscillating frequency vs. the technological parameter fluctuations (Inv5.MSK)

## 5. Chapter 5

### Exercise 5.1

Using the command **Analysis → interconnect analysis with FEM** in Microwind, we extract the total serial resistance  $R$ , ground capacitance  $C_{sub}$  and crosstalk capacitance  $C_{12}$  of the 5mm interconnect in  $0.12\mu\text{m}$ . For lowK=2, we obtain the values reported in table F-xxx. For signal integrity, the best technological option is <Sonia>

Case	$R$	$C_{12}$	$C_{sub}$
N°1: Al, SiO <sub>2</sub>			
N°2: Al, Low K			
N°3: Cu, SiO <sub>2</sub>			
N°4: Cu, Low K			

Table F-xxx:  $R, C$  and crosstalk parameters

### Exercise 5.2

<tbd>

### Exercise 5.3

<tbd>

### Exercise 5.4

<tbd>

## 6. Chapter 6

### Exercise 6.1

<tbd>

### Exercise 6.2

<tbd>

### Exercise 6.3

<tbd>

### Exercise 6.4

<tbd>

## 7. Chapter 7

### **Exercise 7.1**

<tbd>

### **Exercise 7.2**

<tbd>

### **Exercise 7.3**

<tbd>

### **Exercise 7.4**

<tbd>

## 8. Chapter 8

### Exercise 8.1

<tbd>

### Exercise 8.2

<tbd>

### Exercise 8.3

<tbd>

### Exercise 8.4

<tbd>

## 9. Chapter 9

### Exercise 9.1

<tbd>

### Exercise 9.2

<tbd>

### Exercise 9.3

<tbd>

### Exercise 9.4

<tbd>

## 10. Chapter 10

### Exercise 10.1

<tbd>

### Exercise 10.2

<tbd>

### Exercise 10.3

<tbd>

### Exercise 10.4

<tbd>

## 11. Chapter 11

### Exercise 11.1

<tbd>

### Exercise 11.2

<tbd>

### Exercise 11.3

<tbd>

### Exercise 11.4

<tbd>

## 12. Chapter 12

### Exercise 12.1

<tbd>

### Exercise 12.2

<tbd>

### Exercise 12.3

<tbd>

### Exercise 12.4

<tbd>

## 13. Chapter 13

### Exercise 13.1

<tbd>

### Exercise 13.2

<tbd>

### Exercise 13.3

<tbd>

### Exercise 13.4

<tbd>

## 14. Chapter 14

### Exercise 14.1

<tbd>

### Exercise 14.2

<tbd>

### Exercise 14.3

<tbd>

### Exercise 14.4

<tbd>

## 15. Chapter 15

### Exercise 15.1

<tbd>

### Exercise 15.2

<tbd>

### Exercise 15.3

<tbd>

### Exercise 15.4

<tbd>

# F

## 8051 and 16f84 macro- models

### 1. 8051

#### Introduction

The 8051 micro controller symbol is given in figure F-1. The core uses an 8-bit instruction set, and interfaces with three general purpose 8-bit ports P1, P2 and P3. The model in DSCH is only controlled by one clock pin and one reset pin. The Reset pin is active low.

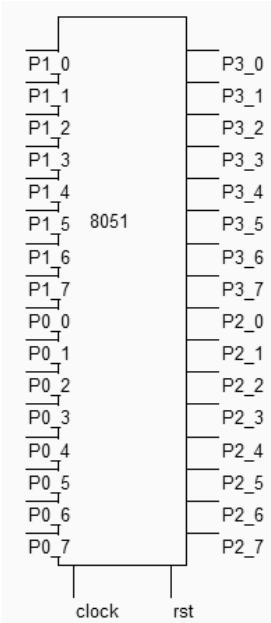


Figure F-1: The 8051 symbol (8051.SYM)

#### Supported Mnemonics

Code	Description	Comments
R	Internal register	R0..R7
A	Accumulator	
SP	Stack pointer	
P	I/O port	P0..P3
#	Immediate	Mov A, #0
X	Hexadecimal data	x0c (hexa) = 12 (dec)
*	Comment	

### Instruction Set

Mnemonic	Operand	Description
ACALL	Address	Absolute call
ADD	A, R	Add
AJMP	Address	Absolute jump
ANL	A, R	And
CJNE		Compare and jump if not equal
CLR	A, R	Clear
CPB	A, P	Complement bit
CPL		
DA		
DB		
DW		
DEC	A, R	Decrement
DIV		
INC	A, R	Increment
JB		
JBC		
JMP		Jump
JNB		
JNZ		
MOV		
NOP		
ORL		
POP		
PUSH		
RET		
RL		
RR		
SETB	A, P	Set one bit
SUBB		
XLR		

### Operating the 8051

Dsch2 includes the model of the 8051 micro-controller [8051]. An example file can be found in **8051.SCH**. The minimum requirements are described in figure F-2:

- One button (Or a VDD symbol) on RESET, which is active low. The button must be pressed to inhibit the RESET function
- One clock on the input *Clock*.

Most standard instructions are supported in the model. However, the ports are assigned predetermined I/O programming conditions: P1 is used for input, P2 and P3 can only be used for output. No interrupt is supported, but subroutine calls are handled in the model.

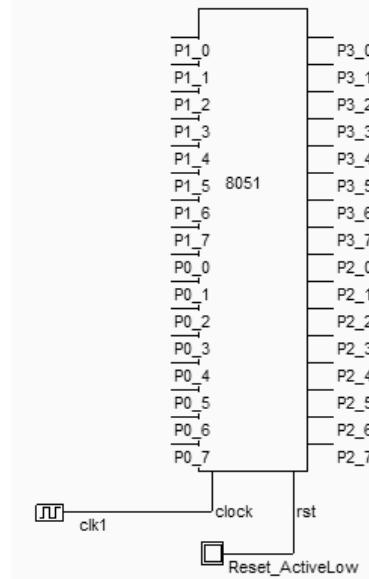


Figure F-2: The 8051 symbol (8051.SCH)

Double click the 8051 symbol, and click **Assembly** to convert the text lines into binary executable code. The window shown in figure F-3 appears. A click on "Assembly" converts the code into binary executable program. The address is situated on the left column. The code is added when possible, and the initial text is also displayed.

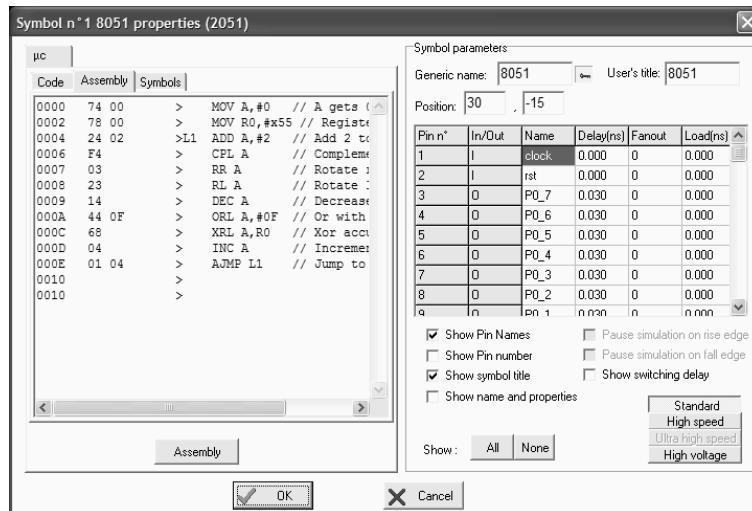


Figure F-3: Editing the assembly lines and converting the text into binary code (8051.SCH)

Then click **OK**, run the simulation. Click the *Reset* button to activate the processor. The default code realizes the execution of several basic instructions, listed in figure F-3 with an infinite loop.

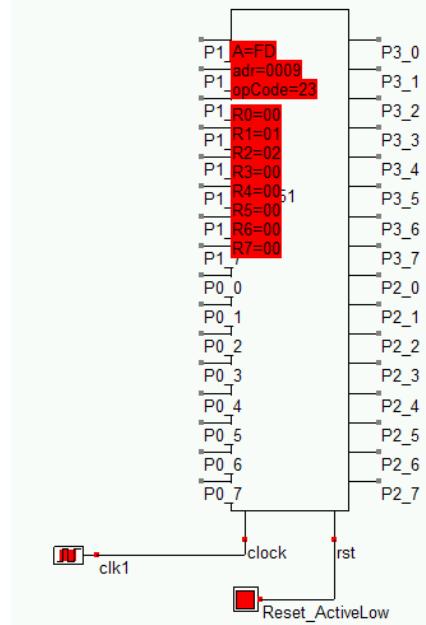


Figure F-4: Executing a simple program on the 8051 (8051.SCH)

## 2. 16f84

### Introduction

The PIC16F84 is part of a family of low-cost, high-performance, CMOS, fully-static, 8-bit micro controllers. All PICmicro™ micro controllers employ an advanced RISC architecture. PIC16F8X devices have enhanced core features, eight-level deep stack, and multiple internal and external interrupt sources. The separate instruction and data buses of the Harvard architecture allow a 14-bit wide instruction word with a separate 8-bit wide data bus. The two stage instruction pipeline allows all instructions to execute in a single cycle, except for program branches (which require two cycles). A total of 35 instructions (reduced instruction set) are available. Additionally, a large register set is used to achieve a very high performance level. The PIC16F84 includes 1Kbyte of Flash memory for the program, 68 bytes of RAM, 64 bytes of EEPROM. The typical operating frequency is 10MHz.

The PIC16F84 is used in applications ranging from high speed automotive and appliance motor control to low-power remote sensors, electronic locks, security devices and smart cards. The Flash/EEPROM technology makes customization of application programs (transmitter codes, motor speeds, receiver frequencies, security codes, etc.) extremely fast and convenient [Pic].

### Supported Mnemonics

Code	Description	
b	Bit address	within an 8-bit file register
d	Destination select;	d = 0 Store result in W d = 1 Store result in file register f.
f	Register file address	(0x00 to 0xFF)
k	Literal field,	constant data or label
W	Working register	accumulator

## Instruction Set

Mnemonic	Operand	Description
addwf	f, d	Add W and f
addlw	k	Add literal and W
andwf	f, d	AND W with f
andlw	k	AND literal with W
bcf	f, d	Bit clear f
bsf	f, d	Bit set f
btfsc	f, d	Bit test f, skip if clear
btfss	f, d	Bit test f, skip if set
call	k	Call subroutine
clrf	f	Clear f
clrW	-	Clear W
clrwdt	-	Clear watchdog timer
comf	f, d	Complement f
decf	f, d	Decrement f
decfsz	f, d	Decrement f, skip if 0
goto	K	Go to address
incf	f, d	Increment f
incfsz	f, d	Increment f, skip if 0
iorwf	f, d	Inclusive OR with f
iorlw	k	Inclusive OR literal with W
movf	f, d	Move f
movwf	f, d	Move W to f
movlw	k	Move literal to W
nop	-	No operation
retfie	-	Return from interrupt
retlw	k	Return with literal in W
return	-	Return from subroutine
rlf	f, d	Rotate left f through carry
rrf	f, d	Rotate right f through carry
subwf	f, d	Subtract W from f
sublw	k	Subtract literal from f
swapf	f, d	Swap nibbles in f
sleep	-	Go into standby mode
xorwf	f, d	Exclusive OR W with f
xorlw	k	Exclusive OR literal with f

## Operating the 16f84

Dsch2 includes the model of the PIC16f84 micro-controller [Pic]. An example file can be found in **16f84adder.SCH**.

The minimum requirements are described in figure F-5:

- One VDD symbol on pin VDD
- One VSS symbol on pin VSS
- One button (Or a VDD symbol) on RESET, which is active low. The button must be pressed to inhibit the RESET function
- One clock on the input *ClkIn*. In the implementation, the pin *ClkOut* is ignored.

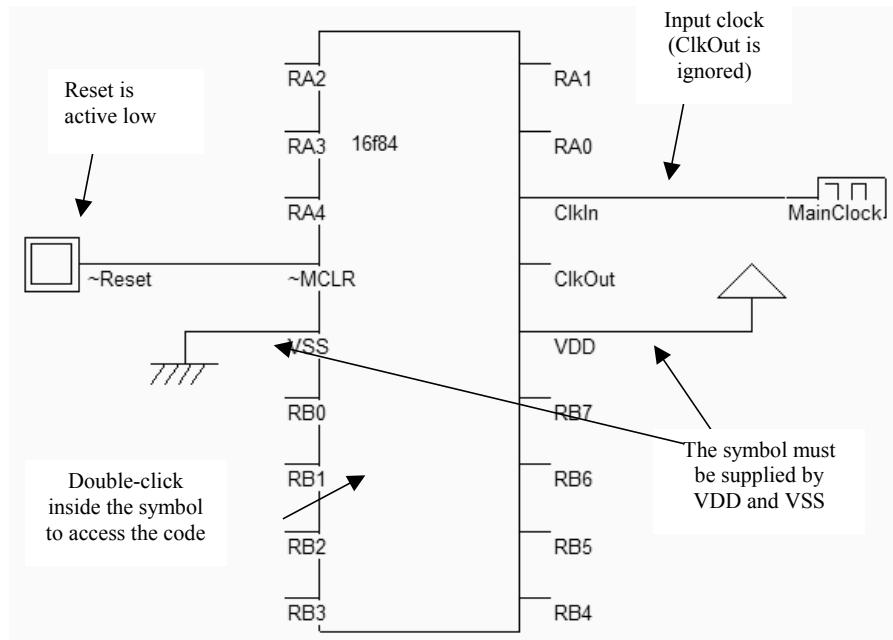


Figure F-5: The 16f84 symbol (16f84adder.SCH)

Double click the 16f84 symbol, and click **Assembly** to convert the text lines into binary executable code. The window shown in figure F-5 appears. The address is situated on the left column. The code is added when possible, and the initial text is also displayed.

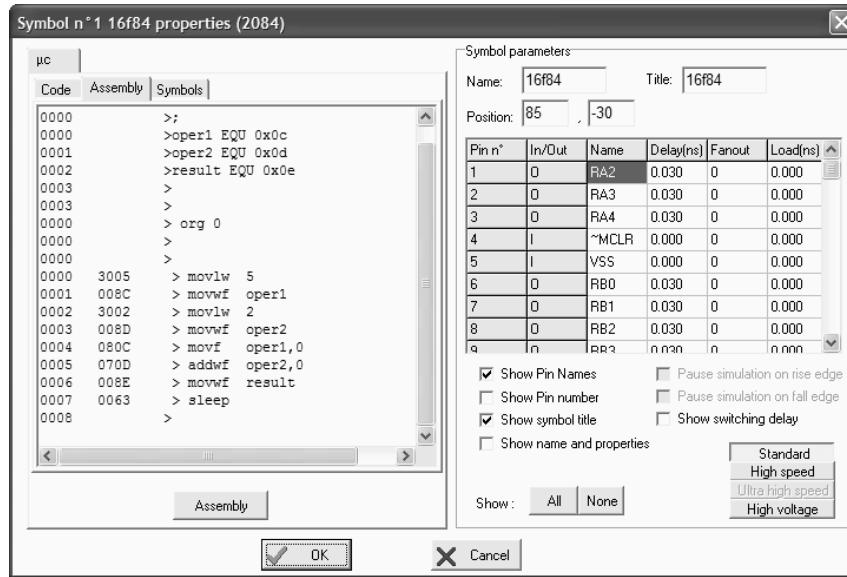


Figure F-6: Editing the assembly lines and converting the text into binary code (16f84adder.SCH)

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.

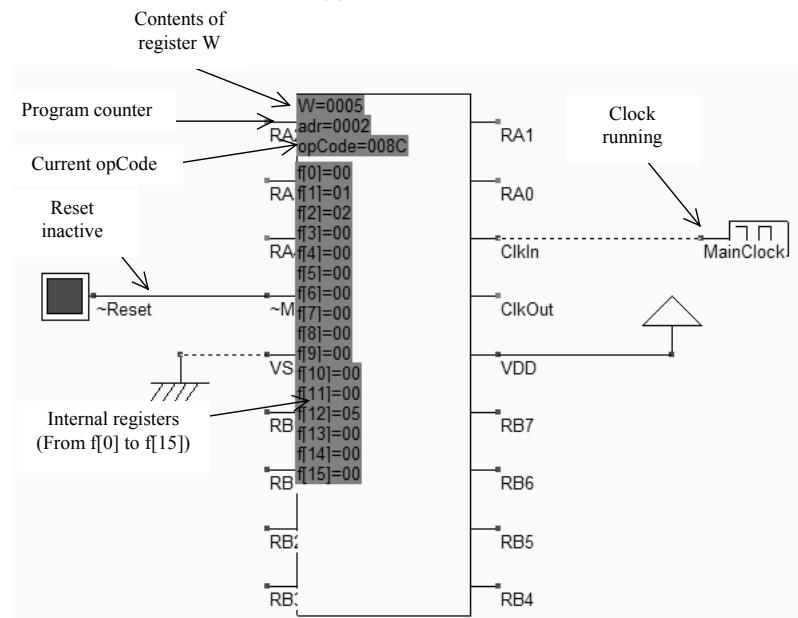


Figure F-7: The program executed by the 16f84 (16f84adder.SCH)

### Remarks on the Assembly Editor

- The comments start with ';' as the first character of the line
- Symbols may be assigned a value using the keyword "EQU"
- The hexadecimal form of a data is 0xxyy
- Any name starting at the first character of the line is considered as a symbol
- The origin of the code is fixed by the Opcode "ORG". Be careful to not write the opcode "ORG" starting at the first character of the line. Leave one character.

### Input/Output with 16f84

The file 16f84.SCH contains the following program. It can be seen that *PortB* is programmed by default as an output port, which is not the case in reality. *PortA* is programmed by default as an input port.

```

; PIC16f84 by Etienne Sicard for Dsch
; Simple program to put 10101010 on port B
;                                01010101 on port B
;
PortB equ 0x06 ; declares the address of output port B

        org    0
loop   movlw 0x55 ; load W with a pattern (hexa format)
        movwf PortB ; Moves the pattern to port B
        movlw 0xaa ; load W with an other pattern
        movwf PortB ; Moves the pattern to port B
        goto  loop ; and again

```

The input *PortA* is updated  
in f[5]

The output *PortB* is updated  
in f[6]

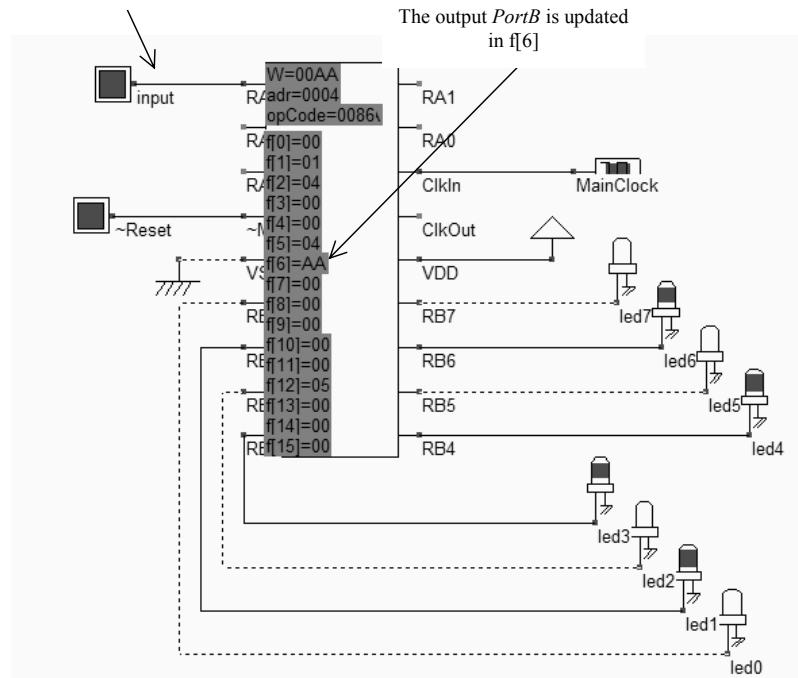


Figure F-8: Using input/output ports in the 16f84 controller (16f84.SCH)

### Features not supported in the model

- The timer is inactive
- The interruptions are inactive
- PortA can only be an input port
- PortB can only be an output port
- All instructions are executed in 1 cycle. In reality, call procedures require 2 cycles.
- The Flash and EEPROM contents are not stored in the file.

### References

[8051] xxx

[Pic] 16f8x Data Sheet, Microchip, 1998 [www.microchip.com](http://www.microchip.com) <confirm>