

# CHAPTER 1: MICROELECTRONIC TECHNOLOGY

Prof. C. Piguet, CSEM, Neuchâtel, Switzerland  
[christian.piguet@csem.ch](mailto:christian.piguet@csem.ch)

## Introduction

This course describes microelectronic technologies used to implement integrated circuits. In the design of embedded systems and Systems on Chip (SoC), one would be more and more capable of integrating all the system components on a single chip. It is therefore mandatory to have a good knowledge of what are these microelectronic technologies.

## CHAPTER 1. Layout and CMOS Design

### Abstract

This chapter will describe the design of integrated circuits while starting with microelectronic technologies and layout, i.e. the lowest level. Layout rules as well as fabrication process will be introduced, in order to be capable of deriving MOS delay models.

### Outline

1. Introduction to IC Technologies
2. Layout, Masks, Fabrication, Layout Rules
3. CMOS Design

### 1. Introduction to IC Technologies

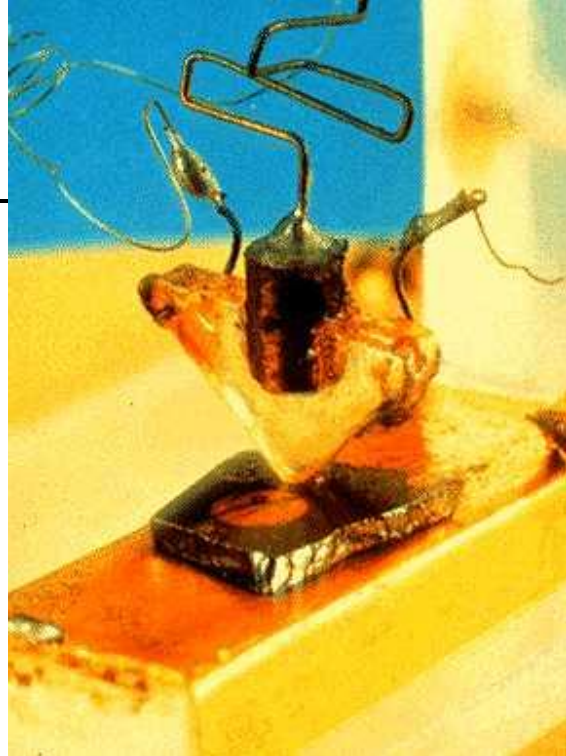
#### *The Transistor*

When Bell Telephone Laboratories announced on June 30, 1948 the invention of the transistor, the general press was almost indifferent. The New-York Times carried the news on the next to the last page of the paper, with four short paragraphs.

Inventors of the transistor are William Shockley, John Bardeen and Walter H. Brattain. They received the Nobel Prize in Physics in 1956 for the invention of transistor, on December 23, 1947.

Even technical journals were slow to appreciate the inherent possibilities of the transistor. To stir enthusiasm for the device, Bell Laboratories licensed it freely and publicized it extensively in seminars and papers. A free licence for the transistor! This means that nobody had understood what such a device was! Its direct concurrent was the vacuum tube, a strongly established commercial product. How the transistor was invented? Was such a device really needed? The answer is simple; the transistor had its origin in scientific theory rather than in technological developments.

From the beginning of the century, more and more studies have been performed on solid-state physics, metals and semiconductors. In 1935, a patent was issued to O. Heil for a field effect triode, although he was not able to explain how it worked. It is ironic that the concept of field effect transistors, so marvellously simple, provides practical implementations after the invention of the far more complex bipolar transistor.



1947: Bipolar Transistor

After the War World II, which interrupted many studies in semiconductor materials, new research programs have been decided. Bell Labs had such a program. The Bell Labs group, including the three inventors of the transistor, decided to limit their research to germanium and silicon, the simplest semiconductors. The group believed that the only explanation for the continued lack of understanding of semiconductors - despite intensive worldwide research - was the diffuse experimentation on so many different complex materials. Silicon and germanium, on the other hand, were elemental, simple; moreover, their atomic structure had revealed the same strong covalent bounding as a diamond, and thus their crystals tended to be striking free of defects.

This is a main point to realize working devices; keep them as simple as possible.

In 1958, the first field-effect transistor was working. It was called "Tecnitron" by its creator, S. Tszner, working in France. In 1960, JFET were commercially produced.

#### *European Invention of the « Transistron »*

It is a strange and unknown story, reported in Spectrum November 2005, p. 46 [1]. Europe missed the transistor, although it was also invented in Paris, completely independently of Bell Labs.

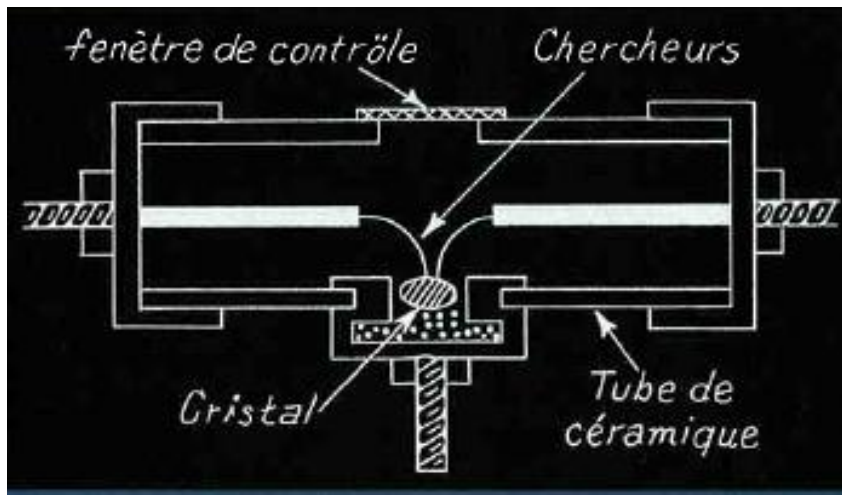
The "Transistron", very similar to the contact point Bell Labs transistor, was invented at the end of WW II in Paris, by two German scientists Herbert Mataré and Heinrich Welker. Both worked at Westinghouse subsidiary in Paris.

In 1948, a small radio receiver used this « transistron » (May 14, 1948). Four days later, French Secretary of PTT announced the invention of the "transistron", "brillante réalisation de la recherche française".

The "transistron" was made of two metal wires that contact a germanium silver material less than 100 micrometers apart. Another electrode contacts the other face of the germanium silver material. Early 1948, by varying the voltage on this electrode, Mataré noticed that he could influence the current through the other. They have to improve this first device, and in June 1948 they noticed a consistent amplification. It was a transistor! They apply for a French patent in August 13, 1948, after the Bell Labs revelation of the transistor. They produce some such devices in 1949 for the French PTT claiming they were better than the US devices (due to their slightly different fabrication process).

The figures below show this "transistron".

But French government and Westinghouse failed to capitalize on this "transistron". Nuclear physics was considered as more promising. So Mataré and Welker left Westinghouse and they move respectively to Intermetall and to Siemens. Mataré then goes to US in semiconductor industry.

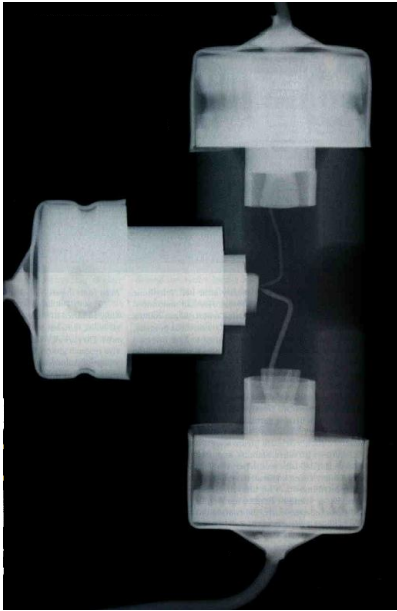


#### *Transistor Commercialization*

Engineers did not like transistors; they preferred tubes. The first market pull came from the hearing aids market, for which miniaturization was a must. The first transistorized hearing aid was announced by Sonotone in February 1953; it contained 5 transistors, but still required a pair of miniaturized tubes for the input and driver stages.

In the mid-fifties, several companies were designing transistors - Raytheon, General Electric, Sylvania, RCA - and this was mass production. A classic case of a successful latecomer to the transistor business was Texas Instruments. With a licence from Bells, TI turned to production by forming a semiconductor laboratory in 1953. It made its most significant contribution in 1954 with the first transistor made of silicon.

Such a business was very exciting! This soon created a rash of job-hopping. Some scientists who had trained in large, established companies abandoned them for more exciting opportunities. Among them was William Shockley, who left Bell Labs in 1954 to start its own company. It was mainly the reputation of its founder that enabled the Shockley Semiconductor Laboratory to attract some of the best talent in industry to Palo Alto, CA. Though the enterprise lacked business direction and ultimately faltered, it was the seed from which the entire Santa Clara area semiconductor industry - Silicon Valley - sprang.



The « transistron »

Young people, such as G. E. Moore and R. N. Noyce, joined Shockley Company. It was not a surprise when only two years later, they left with many colleagues, as they did not know under Shockley's leadership what their business was. Shockley was unable to focus on a product line. Moore and Noyce, with their colleagues, the "traitorous", as Shockley came to call them, set up in 1957 their new facilities in Palo Alto, as a separate venture of Fairchild. They decide to build the diffused-based transistor. Fairchild could not have made a better choice; the transistor-building process was easy, thickness of the base region was well controlled.

However, the main mark of Fairchild in the semiconductor industry was in 1959 its new planar technology. It was not announced before 1960, and the secret of its working was not revealed for several years. However, Control Data Corp's CDC 1604 Computer, introduced in 1960, comprised some 100'000 diodes and 25'000 transistors! The demand for the functions existed, but not the means to speed building the end equipment. A new technology was required, one that could accommodate the increasing complexity of the interconnections.

#### *The Integrated Circuit*

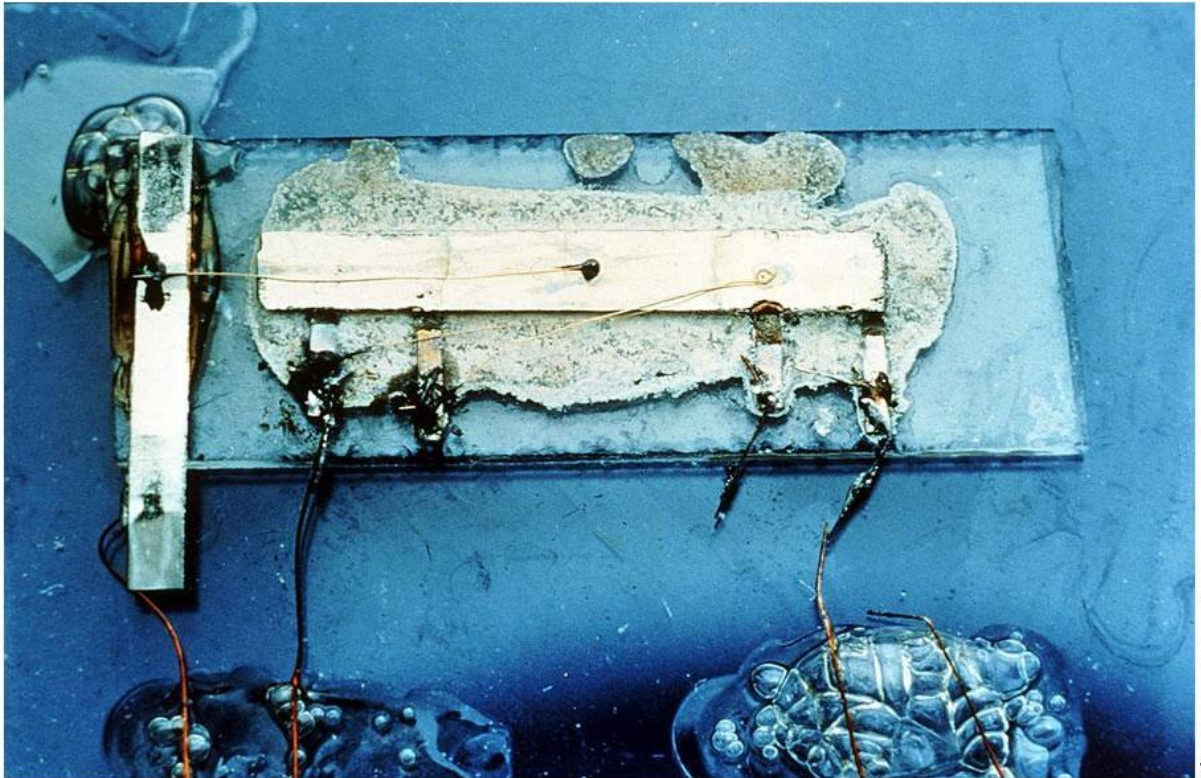
So pressing was the need for fabricating entire circuits in a single semiconductor block that had neither Jack Kilby nor Robert Noyce conceived the integrated circuit in 1959, someone else surely would have. One scientist with an eye in the future has proposed in 1952 the integrated-concept, i.e. to place in the same piece of silicon several transistors with their interconnections. But the invention in 1959 of the integrated circuit was only an answer to the cost of interconnecting discrete components into increasingly complex circuits.

In 1958, Kilby joined TI. During the summer vacations, as a new employee, he spent these weeks alone in the plant, thinking about a way to build entire circuits with semiconductors. In October 1958, Kilby began the design of a flip-flop to be built from scratch of monolithic germanium. The device was completed early in 1959, and revealed "as the most significant development by Texas Instrument since ... the commercial silicon transistor".

Oddly, though the announcement was reported widely in the press, it was greeted with inordinate scepticism. Many of the criticisms were true: the constraints of the integration meant that the performances of individual components of the circuit could not be optimized; design difficult to change; the yield, that would be little better than 10%.

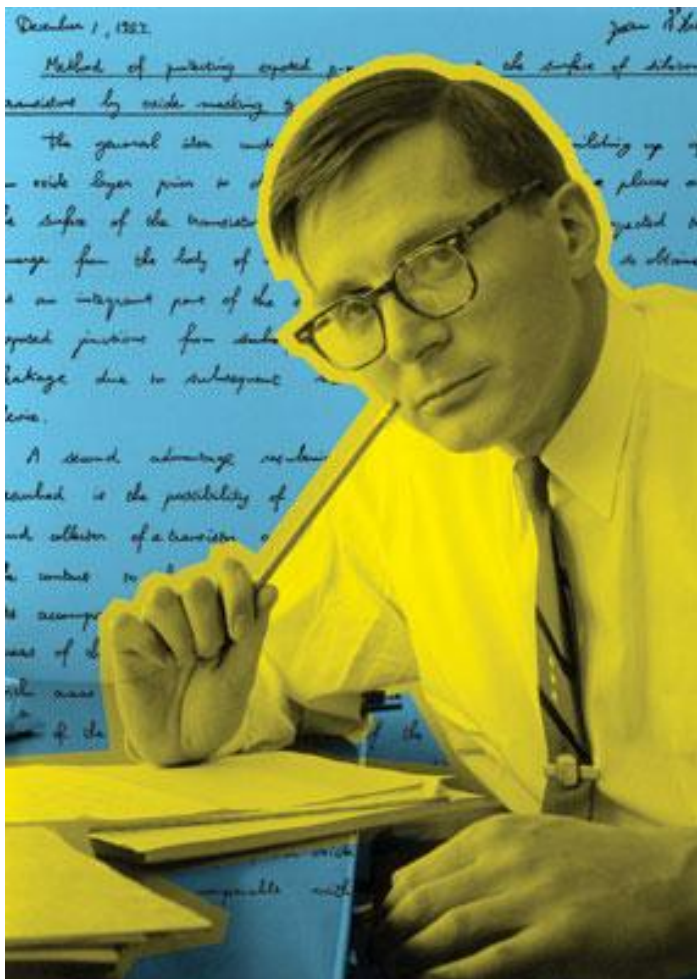


Noyce of Fairchild, after having heard of TI's announcement of the solid state circuit, shares his ideas with his colleagues of how to put several components on a single chip. By February 1960, Fairchild announced that they were manufacturing resistor-transistor logic RTL, an integrated circuit family named Micrologic. By the end of 1961, both TI and Fairchild were producing commercial integrated circuits in production quantities. Most of the production was for the military industry, such as the Minuteman missile for TI, and the National Aeronautics and Space Administration for Fairchild. Though they were simple devices, containing but a couple of logic gates per package, their price was about \$ 100 each in small quantities and perhaps half that in large volumes. The transistor-transistor logic TTL evolved from work done at several sources and was built in 1961 by TRW and in 1963 by Sylvania.

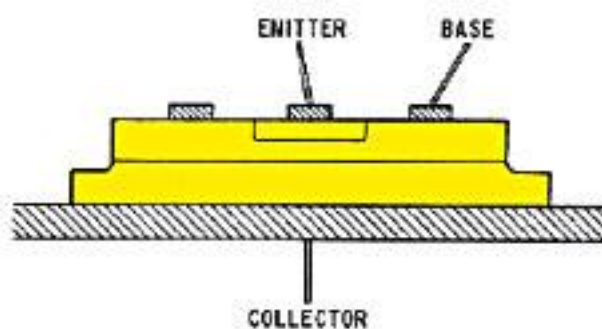


1958: Integrated Circuit. On 12 Sept. 1958, Kilby successfully demonstrated a **1.3MHz integrated RC oscillator**. Because TI had not yet mastered the art of diffusion in silicon, the first IC was built out of germanium bits. Bondwires interconnected the various components because Kilby had not solved that problem yet.

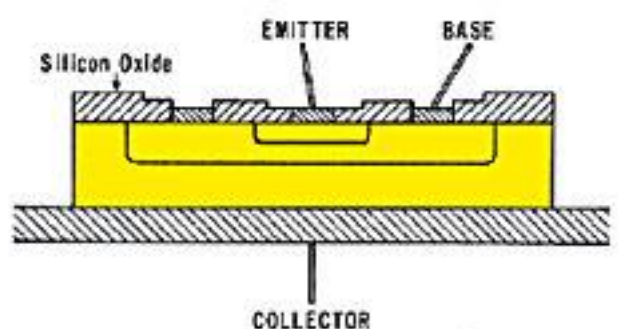
But the invention of the integrated circuit would not be possible without the outstanding contribution of a Swiss engineer, Jean Hoerni, who was the key guy in the group of "8" that have founded Fairchild. Jean Hoerni was a Swiss physicist of the « 8 » team of Fairchild that developed the planar technology. This technology allowed to invent the integrated circuit on 1<sup>st</sup> december, 1957. The idea of Hoerni is to fabricate a transistor while protecting it by putting a silicon oxide above the transistor. It was described in his Memo: « Methods of protecting exposed p-n junctions at the surface of silicon transistors by oxide masking techniques », 1<sup>st</sup> december 1957. These planar transistors were much more robust than the mesa ones. And sure, this technology allowed to use metallic wires deposited on top of the silicon oxide. It was therefore easy to interconnect the transistors by these metallic wires within the same process. This was extremely required to invent the integrated circuit



Jean Hoerni

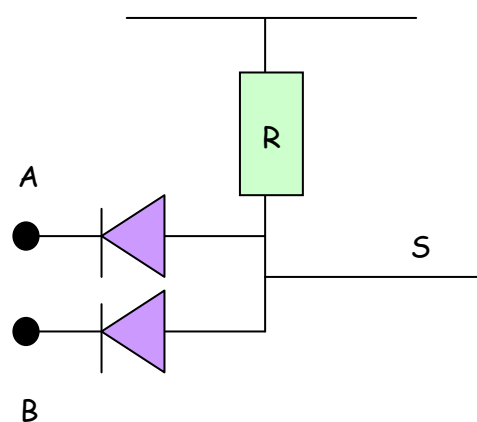


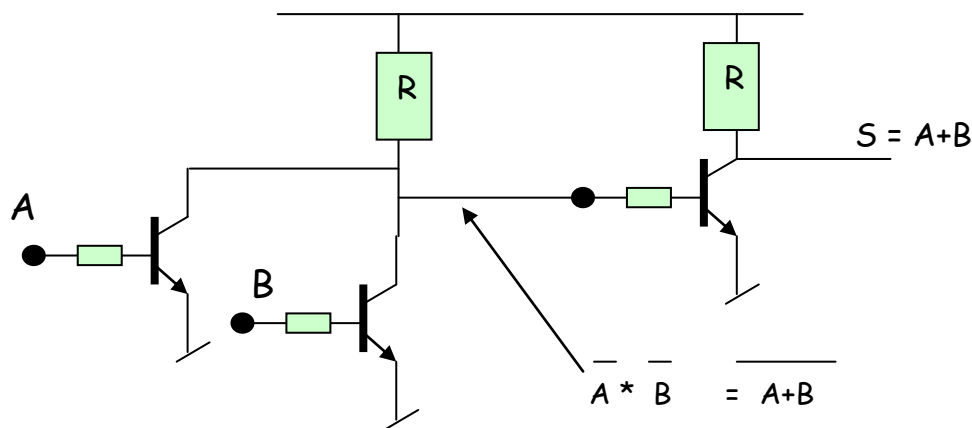
Mesa transistor on the left



Planar transistor on the right

The first logic gates were designed using diodes. Figure at right shows an AND gate. If either of the inputs is grounded, then that diode is on, and the output S is held low. It results in a permanent current if the output is low. The transistor-transistor logic TTL evolved from work done at several sources and was built in 1961 by TRW and in 1963 by Sylvania. Figure below shows a TTL NOR gate or a OR gate with an inverter.





If  $A * B = 0$ , permanent current ! If  $R$  about 500 ohm,  
power is (5 V.) / 500 ohm = 50 mW per device!

### MOS Transistor

Even before Tszner in France has produced a junction field-effect transistor in 1958, many studies were under way in the U.S. on the possibilities of such a device. In 1959, RCA was working on FETs that could serve as logic for computers. But Wallmark of RCA, although he had a patent, never brought his concept to fruition.

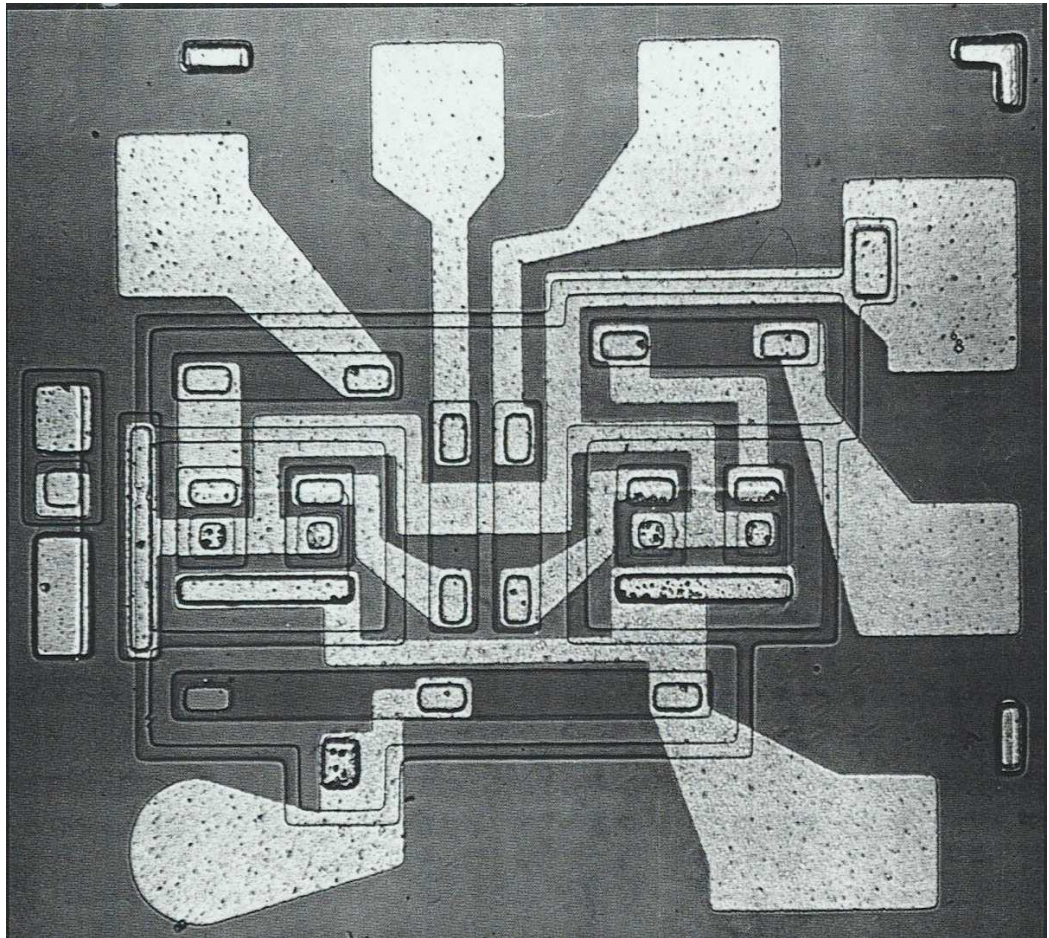
Two years later, Paul Weimer, another researcher of RCA, did. In 1962, RCA was fabricating multipurpose logic block comprising 16 MOS FETs on a single chip. By 1963, RCA had fabricated large arrays of several hundred MOS devices. They were however extremely sensitive to static charge, supply voltage was higher than those of bipolar transistors, and the speed was slower. Production was also plagued with oxide defects.

Few companies had the perseverance to see MOS circuits through. Fairchild abandoned the process after difficulties and returned its focus to bipolar circuit fabrication. Even RCA shifted its emphasis back to bipolar.

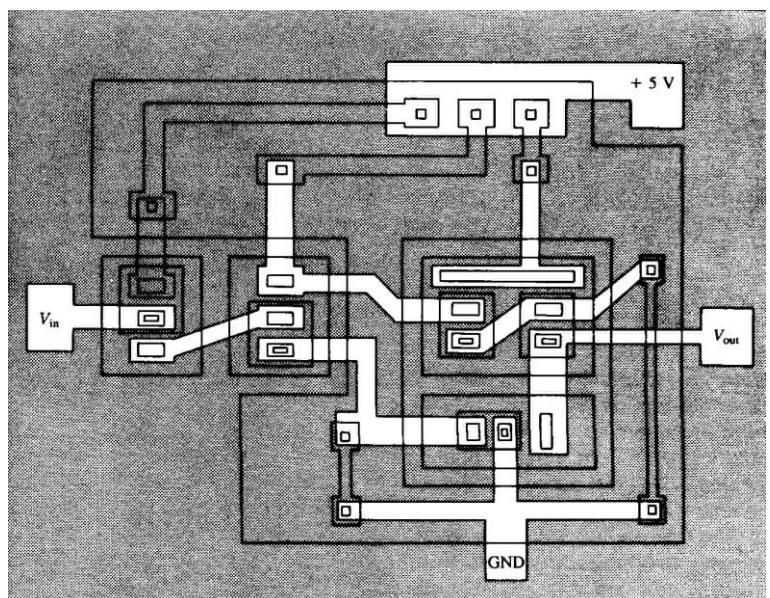
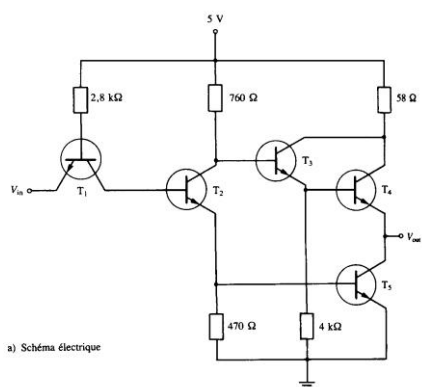
In mid-1965, only two companies were producing MOS ICs - General Microelectronics and General Instruments - the other companies took a wait and see stance. However, the market was growing, and in 1970, the total shipments exceeded \$ 100 million, surpassing nearly all predictions.

More than 25 companies were producing and selling ICs in 1965, with Fairchild being clearly the technology leader. Fairchild developed the idea to replace resistors by transistors in linear circuits. ROM memories appeared first in early 1967. A 64-bit RAM MOS memory was offered by Fairchild also in 1967.





1963: RTL Logic



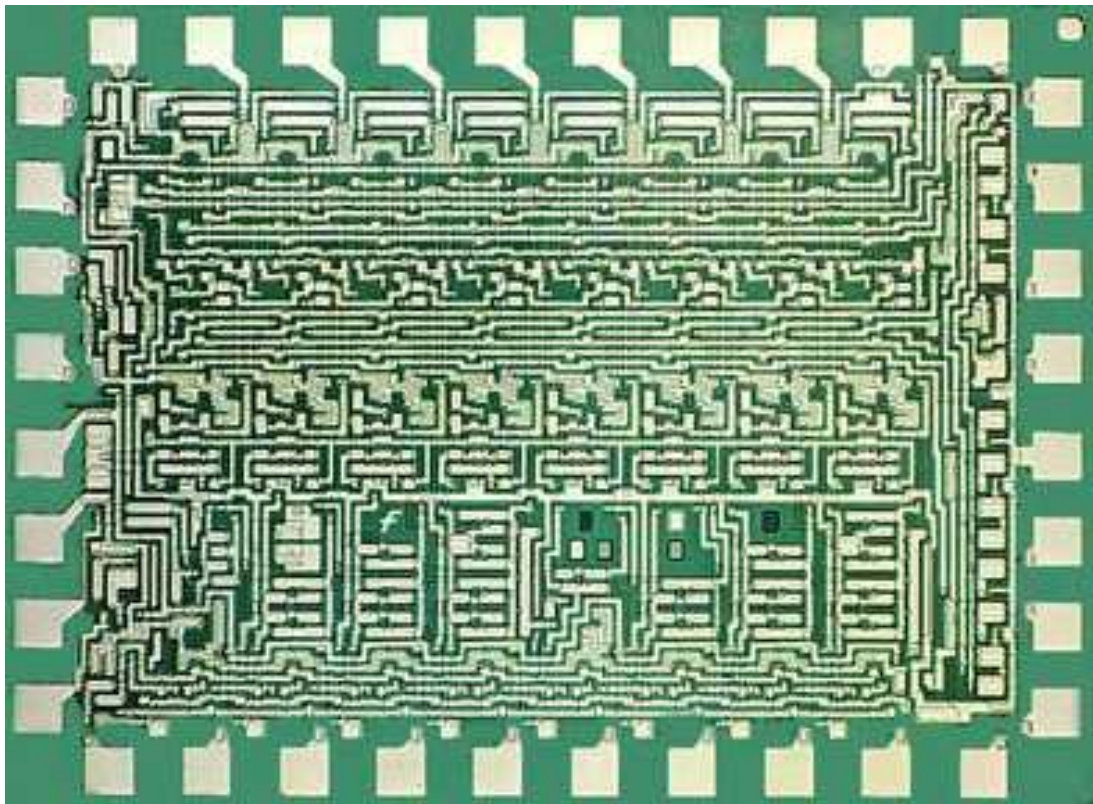
1961-1963: TTL Logic



This greatest period for the semiconductor industry was ended by a kind of game of musical chairs. Fairchild was stung by a mass exodus of executives. Charles E. Sporck, general manager of Fairchild, became President of National Semiconductor and took with him four key men. It was a bad period for Fairchild. John Carter resigned as chairman. Then Noyce, in line for presidency, resigned. At the same time, research and development director Gordon Moore left and formed with Noyce the Intel Corporation. C. Lester Hogan resigned from Motorola to take over as President Fairchild Camera. They were 13 startups in 1968 and another 8 in 1969 in the Silicon Valley. In 1969, ex-TI employees formed Mostek Corp. in Dallas.

Manufacturers of MOS devices found that they could give their transistors a great boost in speed by using N-channel rather than P-channel process. First Intel microprocessors [1.6] were integrated in P-channel technology. First EPROM memories were developed by Intel in 1972. In 1974, the 8080 from Intel was the first N-channel microprocessor. First VLSI text books such as the famous Mead/Conway presented only N-MOS realizations, and it was understood that N-MOS process would be the dominant technology for a very long time.

CMOS technology was only used for very special niches, such as electronic watches in the early seventies. However, ten years later, the move towards CMOS was achieved due to heat dissipation, electromigration and reliability problems. Today, the CMOS technology is clearly the dominant technology.



1967: MOS Logic

## References

- [1] Michael Riordan, "How Europe Missed the Transistor", IEEE Spectrum, November 2005, p. 46- 51.
- [2] C. Piguet, « Il y a 50 ans : l'invention du circuit intégré », FTFC'08, papier invité, Louvain-la-Neuve, Belgique, 26-28 mai 2008.

## 2. Layout, Masks, Fabrication, Layout Rules

Twenty-five years ago, a 1000 MOS integrated circuit was considered as a very big circuit. Development time was over one year. The circuit was designed, “simulated”, lay-out and verified by hand. It is therefore interesting to understand what is layout and what is layout design and verification today. Many CAD tools today allow the designer to design a digital chip without considering layout issues. However, one has to notice that the layout of standard cell are always designed by hand and that layout of most parts of very high performance microprocessors are designed by hand. Hand layout is always more performant than automatic layout [1.5].

### 2.1 Manual Layout Design

Layout design 25 years ago was performed by specialists that have to translate transistor schematics into layouts in a given technology. These layouts have to satisfy to many layout rules and were designed and coloured by hand at the 1000 :1 scale. Any error of 1 or 2 microns means that the complete drawing has to be restarted. Some years later, graphic editors on small computers have been used, but the work was still performed by specialists.

Today, the layout work has been divided in two very different tasks. For digital logic blocks, the designer has to use CAD tools (VHDL, logic synthesizers and place & route tools) that produce very large pieces of layout for any logic block. These CAD tools are based on standard cell libraries that has been designed by specialists and very well characterized in a given technology. The second task in layout design is precisely the design of such standard cell libraries as well as memories (ROM, RAM). This work is still performed by hand by layout specialists, and deep submicron technologies implies a much more complex work.

### 2.2 Masks and Layout

The design of layout masks (or layout) consists in defining all the geometries that define an integrated circuit. A layout is therefore a coloured drawing with a very large number of rectangles and polygons. This layout is then used to fabricate the masks that are used in the fabrication process of the integrated circuit itself. Layout specialists have to master the various layout rules that define the minimum distances between geometries. Productivities in manual layout design is about 4 to 6 transistors per day and per operator.

The technological process is a sequence of steps. Each major step requires a given mask that defines the geometries of the corresponding layer. Figure 1.0 shows the list of the masks used for a very simple CMOS 4 micron technology used at CSEM 20 years ago. This process is chosen here as an example as it is very simple and consequently more easy to explain

mask	color	comments
well (DP)	braun	defines p-well in which are located N-ch transistors
diffusion (OD)	black	defines zones diffused n+ or p+
poly (PS)	green	defines polysilicium wires polycrystallin and MOS gates
doping n+ (SP)	blue	defines zones in which transistors are doped n+
contact (CO)	purple	defines contacts between IN-PS or IN-OD
metal (IN)	red	defines metal lines

Figure 1.0 Six masks of a very simple 4 micron technology

The lithography process is used to deposit a layer (poly, metal) on the substrate (Fig. 1.1a). The mentioned layer is deposited on the substrate and covered by a photoresist. Then the mask is aligned and used to polymerize the photoresist by light through the transparent holes of the mask. Then the non-polymerized photoresist is removed by an given mixture. Another mixture is then used to remove the layer (poly, metal) where it is not protected by the polymerized photoresist. Finally the polymerized photoresist is removed to get the desired layer (Fig. 1.1a).

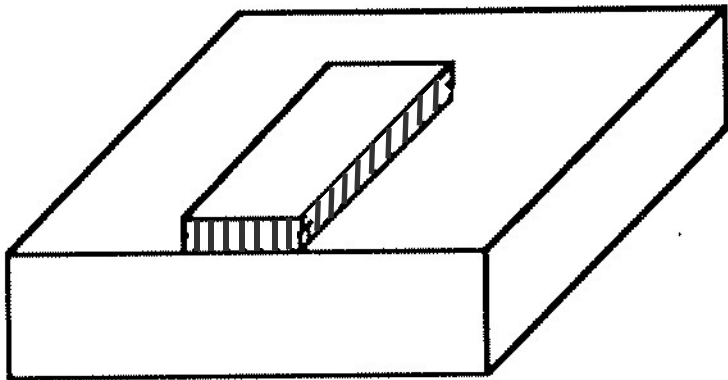


Figure 1.1a

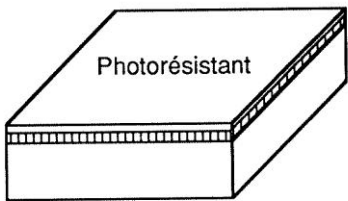


Fig. 1.52

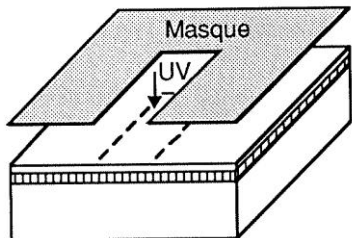


Fig. 1.53

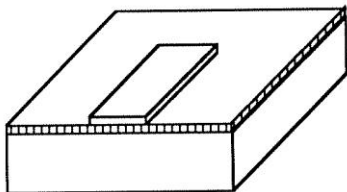


Fig. 1.54

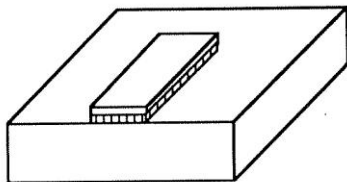


Fig. 1.55

Figure 1.1 Lithography



The fabrication process uses successively the 6 masks of Figure 1.1. Figure 1.2 shows the major steps:

- p-well design (Fig. 1.2a)
- creation of diffusions (Fig. 1.2b)
- definition of the poly-silicium MOS gates and wires (Fig. 1.2c)
- doping of the diffusions, n+ within the blue mask, and p+ outside (Fig. 1.2d)
- contacts opening (Fig. 1.2e)
- definition of the metal wires that are in aluminium (Fig. 1.2f)

One can understand that the colors of the various geometries refer to the colours of the corresponding masks. One can use other color codes [1.3]. Ref [1.1] describes the fabrication process.

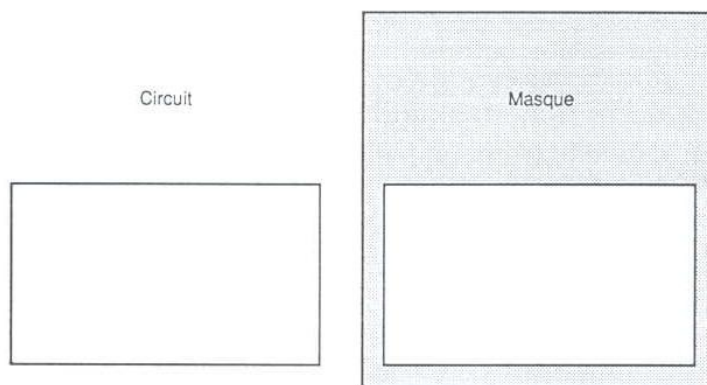


Figure 1.2a. P-well design

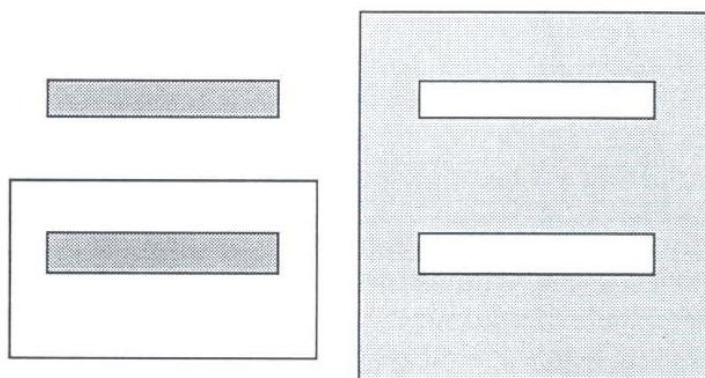


Figure 1.2b. Creation of diffusions

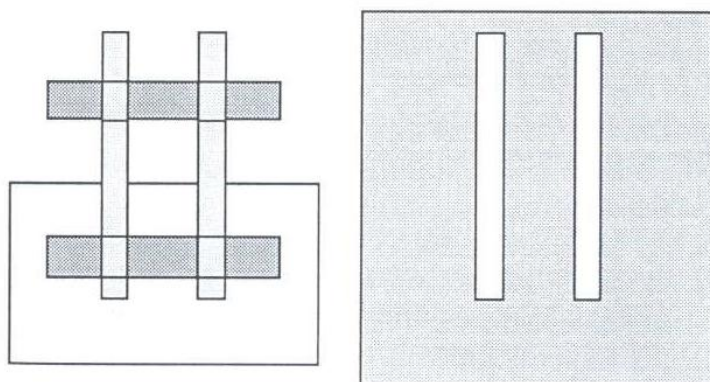


Figure 1.2c. Definition of the poly-silicium MOS gates and wires

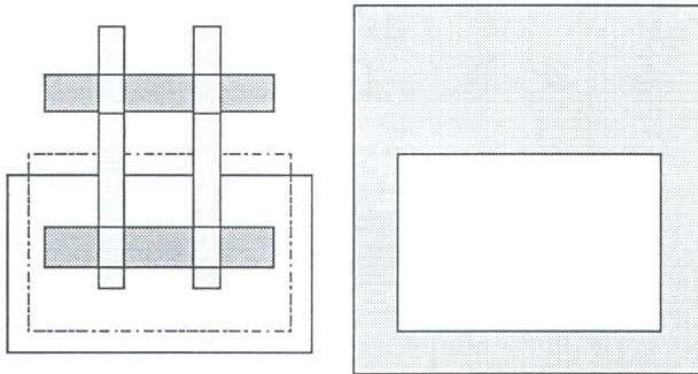


Figure 1.2d. Doping of the diffusions, n+ within the blue mask, and p+ outside

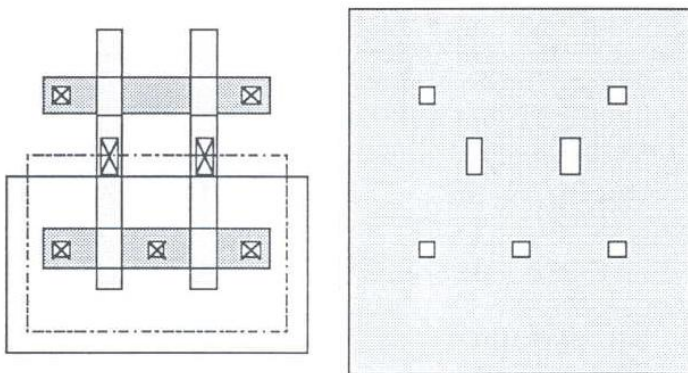


Figure 1.2e. Contacts opening

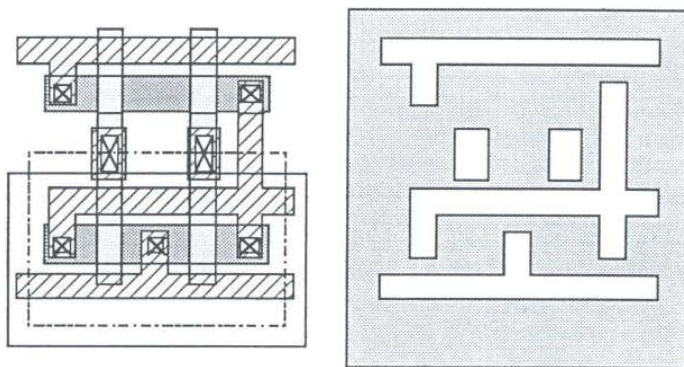


Figure 1.2f. Definition of the metal wires that are in aluminium

Figure 1.2 Major steps of a very simple fabrication process

### 2.3 Industrial Layout Rules

The fabrication process of integrated circuits has to align very precisely the successive masks on the chip itself. There are several alignment figures on the chip that are used for this alignment, but some misalignment is unavoidable. There are also some imperfections due to the process itself, i.e. chemical processes. It therefore necessary to design the layout with these problems in mind. It is why any foundry provides layout rules for its fabrication process that take into account all these imperfections as well as the performances of the production machines. The goal of industrial layout rules is to have the better packing density as well as the best yield (two contradictory goals). It explains why layout rules are more or less different from foundry to foundry even for the common  $0.25\ \mu\text{m}$  process. It results that it is not possible to use the layout for a given process for another process. Furthermore, the conversion process from one layout to another is tricky

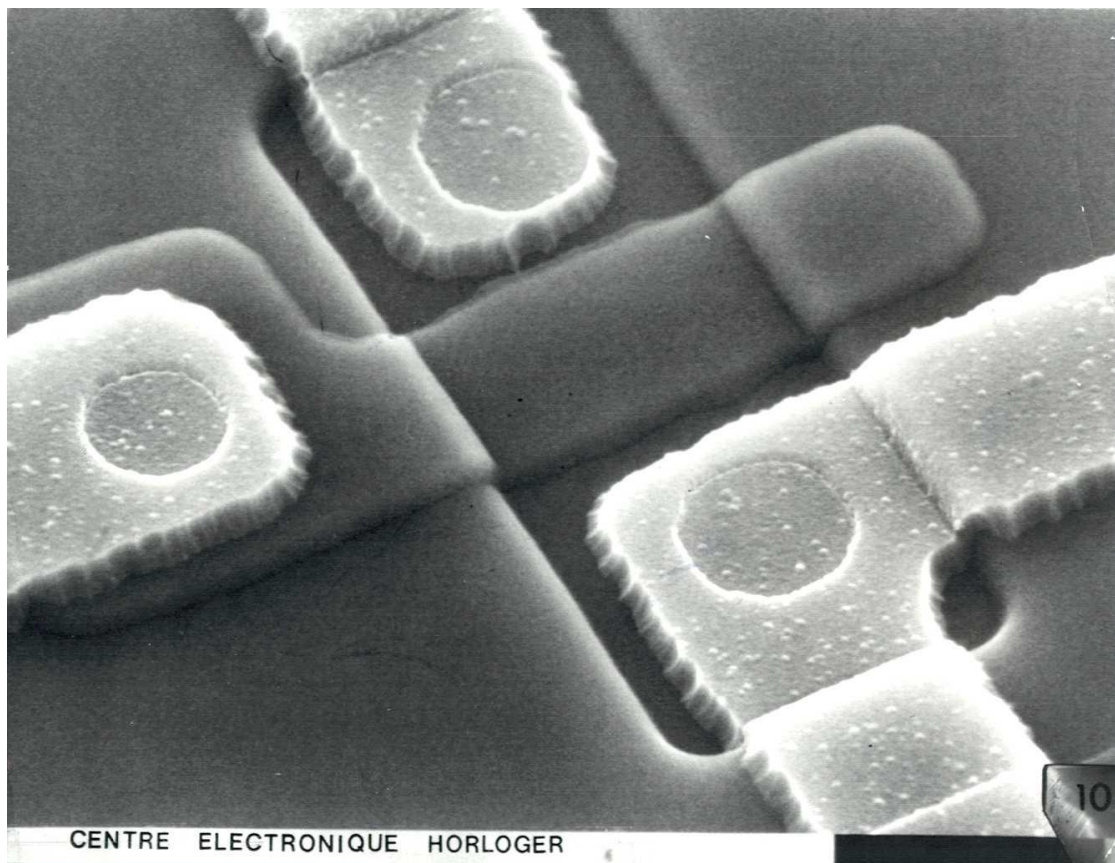


Figure 1.4 MOS transistor

Figure 1.3 shows the industrial layout rules of a very old CMOS  $4\mu$  process. Figure 1.4 shows a single MOS transistor designed in this process. One can see the diffusion defining a transistor with a polysilicium gate in the middle as well as two contacts and associated metal lines at both extremities.



MASKS	RULE	DISTANCE in $\mu$
well DP	1) width 2) length	6 16
diffusion OD	3) width 4) espace 5) distance to DP within DP 6) distance to DP outside DP	4 6 10 12
poly PS	7) width 8) espace 9) espace on a distance $\leq 4\mu$ 10) gate overlap outside DP 11) distance to OD 12) distance to OD outside OD without IN	4 4 2,5 3 3 2,5
dopage n+ SP	13) width 14) espace 15) distance to OD 16) distance to PS 17) distance to PS outside OD for for gate overlap 18) distance to PS for contact	6 6 3 2 0 1
contact CO	19) width 20) length 21) espace (on IN ) 22) distance to OD within OD 23) distance to OD outside OD 24) distance to PS in PS (if IN) 25) distance to PS in PS (if there is no overlap of IN ) 26) distance to PS outside PS 27) distance to SP (if no short-circuit) 28) dimension for short-circuit	3 3 à 12 6 2 3 1,5 0,5 3 4 3 * 8
métal IN	29) width 30) espace 31) overlap CO (at the opposite side of the overlap of IN ) 32) overlap CO (other cases )	4 4 1 0,5

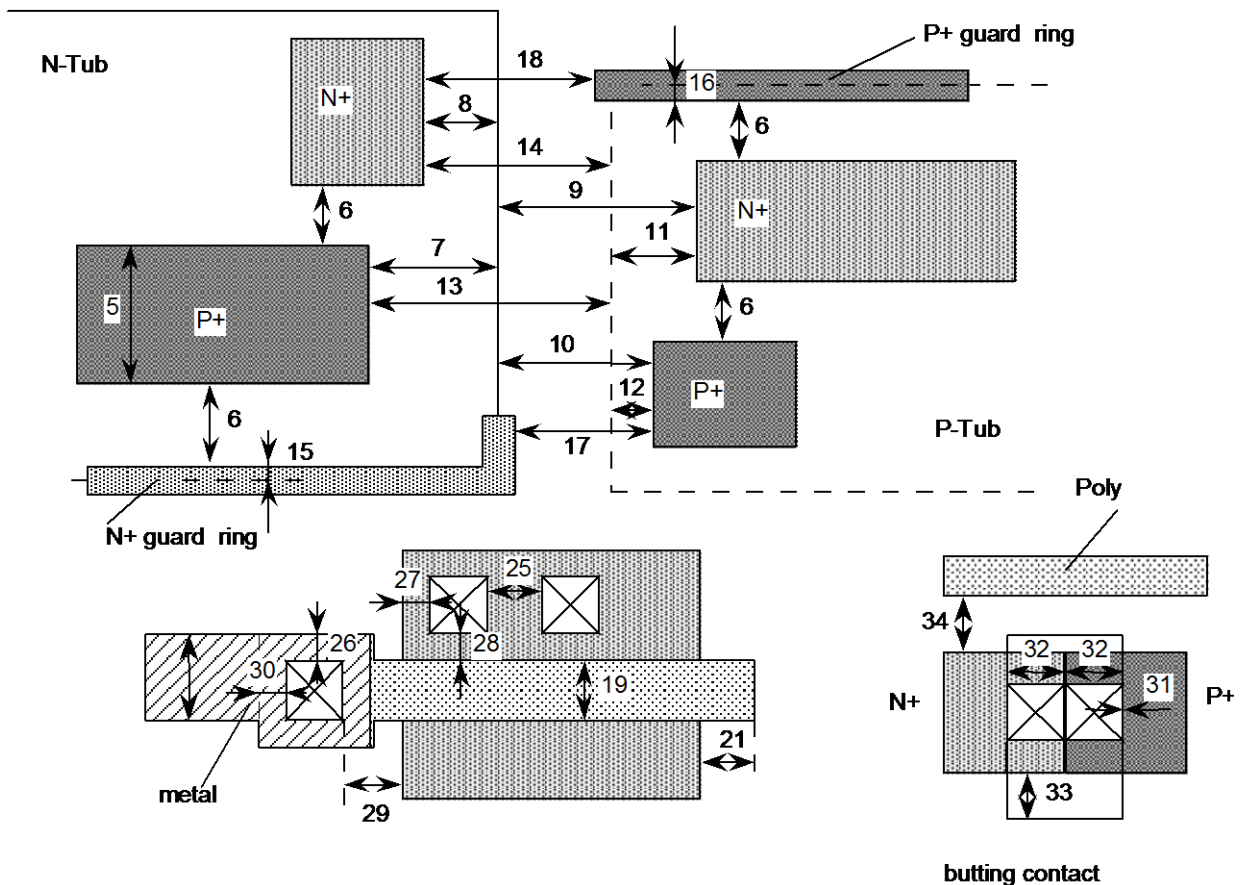
Figure 1.3 Layout Rules of the CMOS  $4\mu$  from CSEM.

Figure 1.5 shows the layout rules of the  $2\mu\text{m}$  process from VLSI Technology. It is also a quite old process, but one can see that the number of layout rules is largely increased. Note that layout rules of modern technologies ( $0.25$  to  $0.13\mu\text{m}$ ) are generally classified as confidential. These rules are given here to show that they are not easy to understand. Figure 1.6 shows a graphic representation of some of these rules that seem easier to understand.

In 90 nanometers technology node, there are 600 design rules!!

MASKS	RULE	DISTANCE in $\mu$
well N N-Tub CNW well P P-Tub CPW	1) width (min) 2) distance (well same voltage) 3) distance (well different voltage) 4) distance well N-Tub and P-Tub	4 4 ou 0 12 3
diffusion P+ CPD  diffusion N+ CND	5) width (min) 6) space (P+ to P+, N+ to N+, P+ to N+) 7) distance to P+ within N-Tub 8) distance to N+ within N-Tub 9) distance to N+ outside N-Tub 10) distance to P+ within P-Tub to N-Tub 11) distance to N+ within P-Tub 12) distance to P+ within P-tub 13) distance to P+ outside et to P-Tub 14) distance to N+ within N-Tub to P-Tub guard ring 15) 1/2 width guard ring N+ within N-Tub 16) 1/2 width guard ring P+ within P-Tub 17) distance N+ guard ring to P+ within P-Tub 18) distance P+ guard ring to N+ within N-Tub	2 3.5 5 3 7 5 4 2 8 6  1 0 4 4
poly CP	19) width (min) 20) space (min) 21) gate overlap 22) distance to diffusion	2 2.5 2 1
contact CC  Butting Contact	23) size (min) 24) size (max) 25) space 26) overlap poly CP on contact CC 27) overlap diffusion on contact CC 28) distance CC to gate CP 29) distance CC outside diff to diffusion 30) overlap métal 1 on contact CC 31) overlap mask butting CB to CC 32) distance mask butting CB center CC 33) distance mask butting CB diffusion 34) distance poly CP to diff butting contact 35) size min butting contact	2*2 5*5 3 1 1 1.5 2 1 0 2 0 3 2*6
métal 1 CM	36) width (min) 37) space (min)	2 3
Via CC2	38) via can be located on diffusion, poly et oxyto 39) size min 40) size max 41) space between via 42) overlap métal 1 or 2 on via 42) distance to via outside poly to poly CP 3) overlap of poly to via within poly 44) distance via-contact on diffusion 45) distance via-contact on poly	2*2 5*5 4 1 2 4 2 3
métal 2 CM2	46) width min 30) space min	3 4

Fig. 1.5 Layout Rules of the 2 $\mu$ m VLSI Technology

Figure 1.6 Layout Rules of the 2  $\mu\text{m}$  VLSI technology

## 2.4 The $\lambda$ layout rules

Universities were not interested to this industrial stuff, until some proposals from Universities were made to have a more synthetic view of layout rules. The first proposal was based on the  $\lambda$  parameter, which is a scale factor [1.3, 1.4]. The  $\lambda$  parameter is the maximum deviation of a fabricated geometry from the same geometry at the layout level in a given process (Fig. 1.7a). Depending on the value of the  $\lambda$  parameter, the layout of a given circuit can be described for several technologies (2 $\mu$ , 1 $\mu$ , 0.5 $\mu$  and below). This was interesting, as the technology independence was achievable.

The value  $2*\lambda$  is the minimum width of the transistor gates. A given technology is defined by this value  $2*\lambda$ : a 1 $\mu\text{m}$  technology is characterized by  $1\mu\text{m}=2*\lambda$ , et par consequently  $\lambda=0.5\mu\text{m}$ .

The set of layout rules is based on the following principles:

- when the misalignment of 2 masks results in a fatal error, one has to space the 2 geometries of  $2*\lambda$ .
- when the misalignment of 2 masks is not an issue, one has a space of  $1*\lambda$ .
- widths and spaces of wires are  $2*\lambda$ , only the metal wires do have  $3*\lambda$ .

Figure 1.7b shows the set of layout rules based on the  $\lambda$  parameter.

Any set of layout rules has to be memorized by a layout designer. The  $\lambda$ -based set of rules is therefore easy to memorize. Furthermore, they can be applied to any technology from any foundry. However, they cannot use the specific tricks that do present some specific technologies. Consequently, the resulting layouts are not very compact. It is why these rules were not used in industry, they were used



only by universities for test circuits. They are not used today, but their interest is in the understanding of what are and for what purpose layout rules do exist.

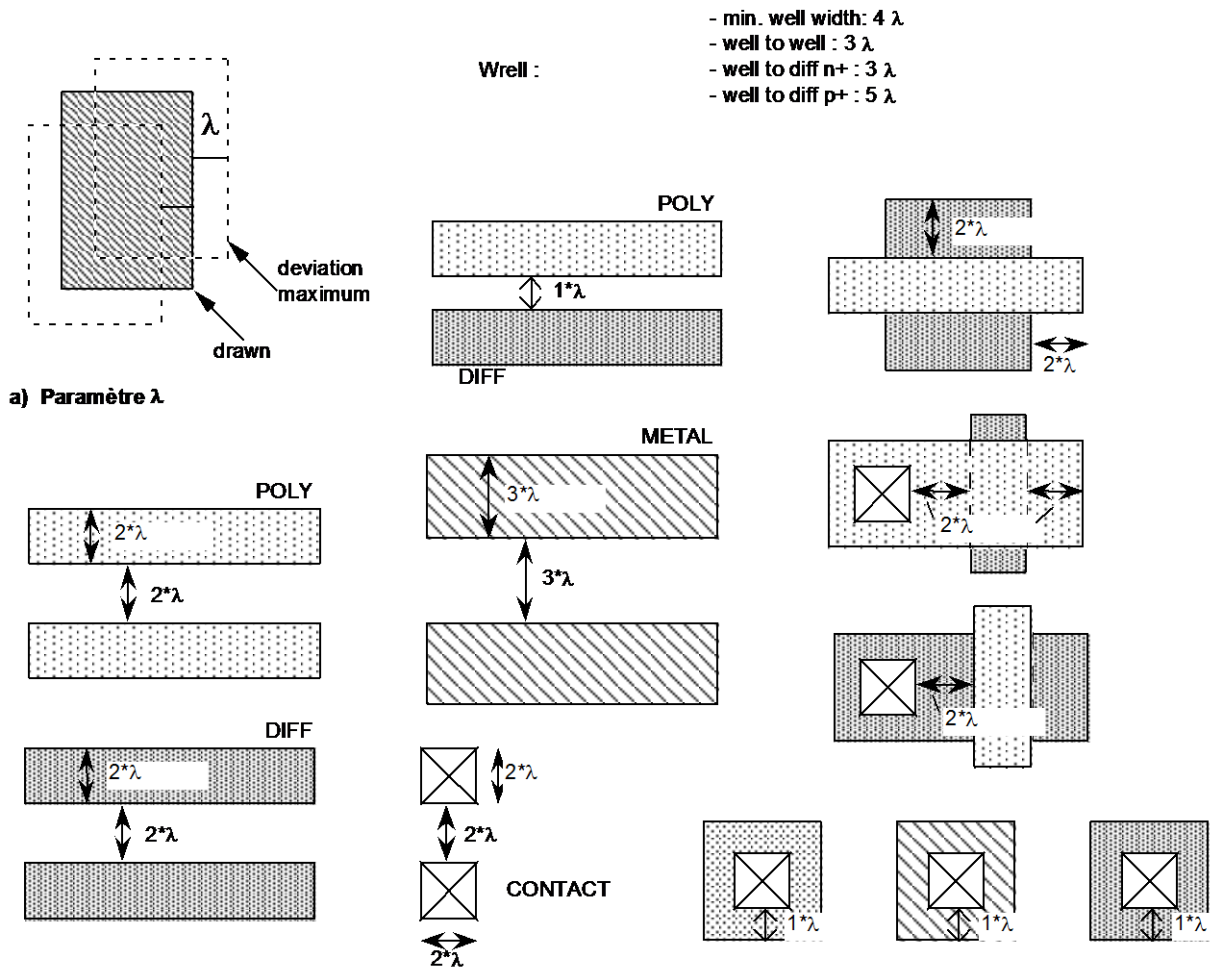


Figure 1.7  $\lambda$  layout rules

## 2.5 Geometrical Layout of a MOS transistor

Figure 1.8 shows the layout of a N-ch MOS transistor. This example requires the application of 22 over 32 layout rules of Figure 1.3 (including rules to well and to doping masks not shown on Figure 1.8). The diffusion rectangle is crossed by the poly-silicium gate that defines a transistor of width  $W$  and length  $L$ , with  $W/L=7/4\mu\text{m}$ .

The poly-silicium gate overlap (rule 10) is quite large. In case of misalignment between diffusion (OD) and poly-silicium (PS) masks, it results in the fact that the transistor gate cannot be interrupted within the diffusion. A direct conducting path (short-circuit) between drain and source of the transistor is therefore avoided. Drain and source of the transistor are connected to metal wires through contacts as well as the poly-silicium gate. Metal overlaps around contacts are different depending on the direction, in order to increase the layout density.

From this example, one can see that layout design is a very complex task, as for a single transistor, one has to apply 22 rules over a total of 32. One has to check every rule while designing a single transistor, and several tens of thousands of transistors are required for a small integrated circuit. One can compare this work to software programming at the assembly level.

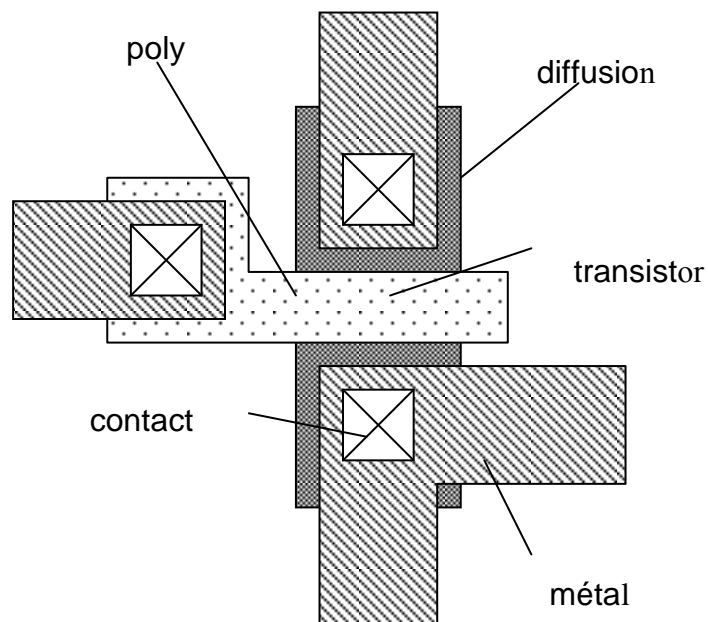


Fig. 1.8 MOS N-ch

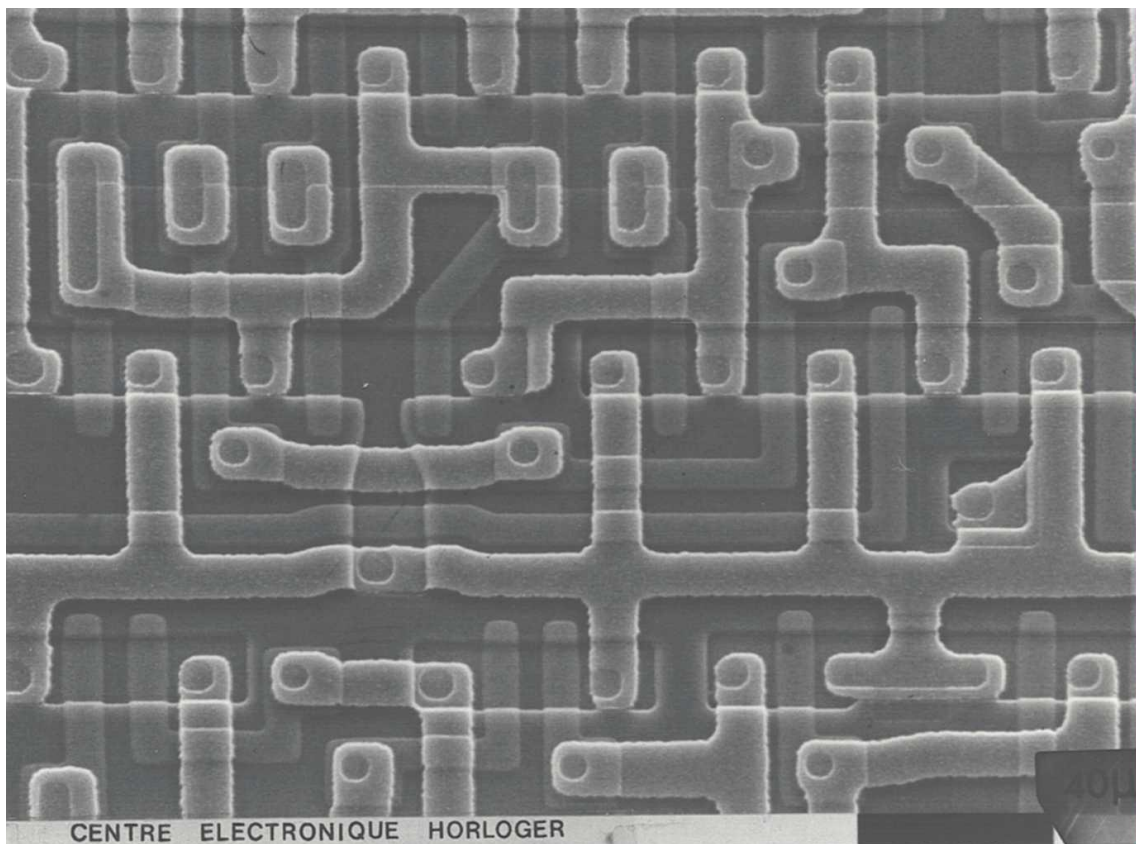


Fig. 1.9 Microphotograph of basic cells

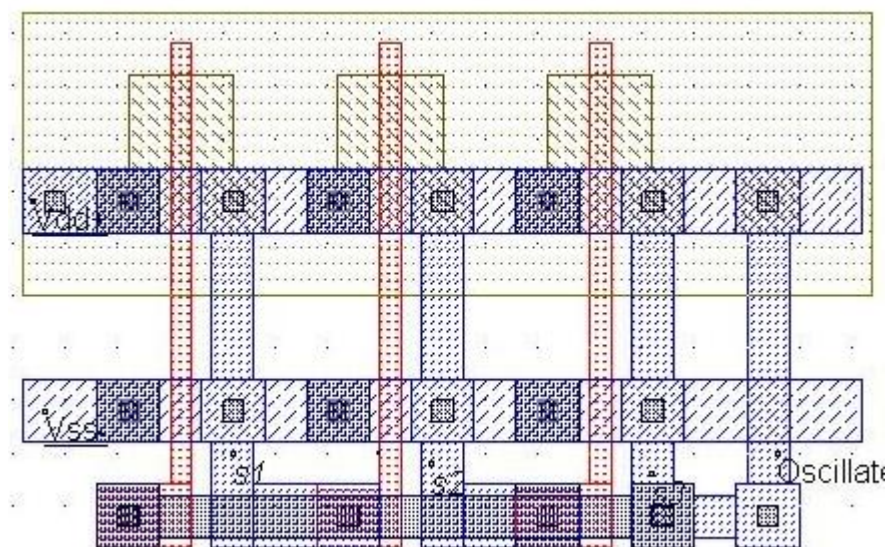
Figure 1.2 shows the design of a NOR gate layout. The first mask is the well (DP) defining the region in which are located N-ch MOS transistors. The second mask (OD) is the diffusion mask while the

third mask is the poly-silicium mask. One can notice the large distances between diffusion and well masks. The fourth mask is the doping mask (very similar to the well mask). This mask results in a n+ doping of the drains and sources of transistors. The transistor channels are not doped, as they are under the poly-silicium gate. But in this technology, the poly-silicium wires are also n+ doped. The dual doping mask is then used for the P-ch transistors that are p+ doped. It results that the poly-silicium wires are also doped p+ outside the well. It is why one has to short-circuit by a contact and a metal rectangle all the poly-silicium wires that cross the well, as shown for the fifth (contact) and sixth (metal) masks to kill the p+-n+ diodes. Figure 1.9 shows a microphotograph of some basic cells in this process.

### Layout in Deep Submicron Technologies

The layout design of standard cells in very deep submicron technologies is not so different than in micronic technologies. It is mainly due to the fact that standard cells layouts are using one (sometimes two) metal levels even if the technology provides 6 to 7 metal layers (that are very useful for the routing). Regarding low-power, the same goal has to be achieved, i.e. to reduce the parasitic capacitances, to avoid to have too many contacts in series for internal interconnections, a careful examination of the internal layout topology to avoid misplaced elements that increase the cell area and parasitic capacitances. Mainly rules of thumb are applied to get good layouts.

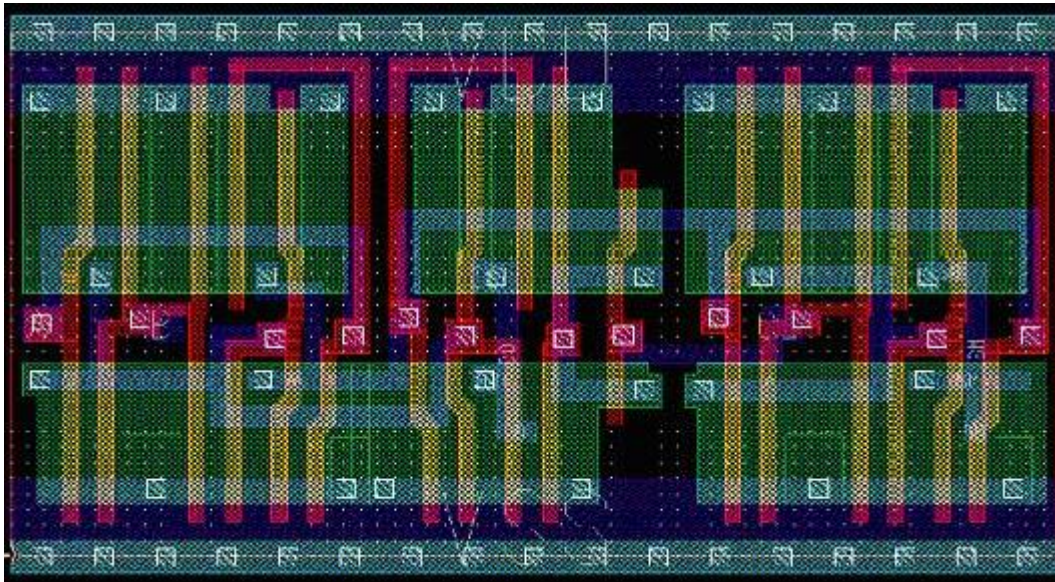
However, as mentioned above, the transistor area becomes relatively smaller than interconnections and contacts that have large overlaps. It gives a strong push to have transistor schematics that reduce the number of contacts. A main difference in deep submicron technologies is that cells can be connected inside the cells and not only on the top and bottom sides. Due to the large number of metal layers, cells can be abutted not only in rows, but the rows themselves are abutted, the Vss and Vdd supply lines are shared between two rows. In that way, there no more routing channels and the density is significantly increased. Furthermore, it gives more freedom to design not too small cells, as some room is necessary for internal connectors and too small cells could imply some congestion for the cell router itself.



Layout in 0.25  $\mu\text{m}$  of a standard cell (Microwind)

The above Figure shows a layout in 0.25  $\mu\text{m}$  provided by the PC tool Microwind largely used for education [1.15]. Three inverters in series are shown in a single layout. This tool provides design rules and transistor models down to 0.1  $\mu\text{m}$  [8.45]. The next Figure shows a 1-bit adder designed in 0.18  $\mu\text{m}$  for the CSEM standard cell library. Only one metal level (M1) is used inside the cell.





Layout in 0.18  $\mu\text{m}$  of a 1-bit adder (CSEM)

## 2.6 Description Language of Geometrical Layout

To manipulate and to communicate layout descriptions of any circuit, description languages have been proposed. Such languages are ascii or binary coded, and they describe rectangles, polygons and any layout geometry [1.2]. The goal of this Section is to present a very simple layout description language. In a system of geometrical coordinates X and Y, one can define a rectangle with 2 points (the two extremities) and therefore by the following instruction (Fig. 1.10a):

R x1 y1, x2 y2 ;

One can also define a polygon by a list of its coordinates and by duplicating the first point to indicate that the list is over (Fig. 1.10b):

P x1 y1, x2 y2, x3 y2, x3 y3, x1 y3, x1 y1 ;

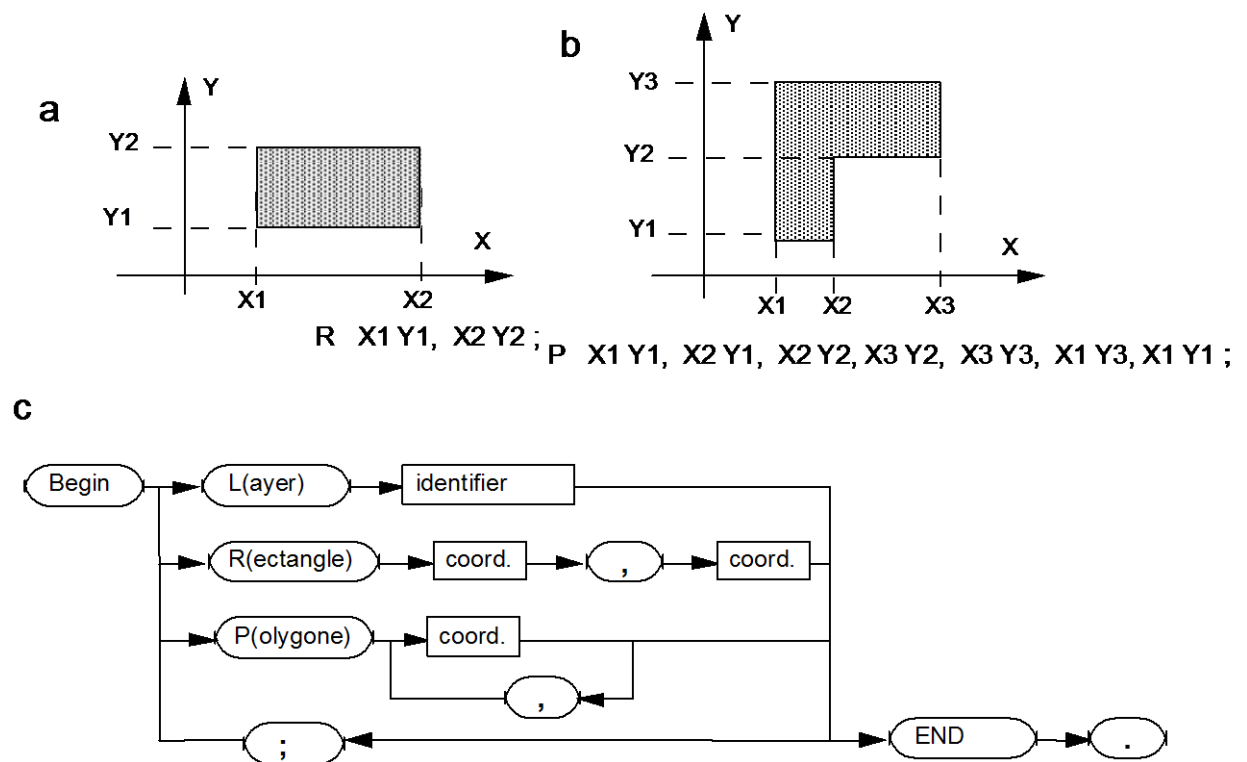


Figure 1.10. Description Language

Such a language has also to describe the mask type. All the geometries designed with the same mask are grouped together and the first instruction describes the mask type (key word L (for Layer) followed by the mask identifier). It means that all the following geometries have to be implemented with this mask:

L IN ; means that all the following geometries are in metal

This language is very simple, and industrial description languages are generally a little bit more sophisticated. The textual description of a single MOS transistor (similar to Figure 1.8) is the following:

```
BEGIN {N-ch MOS}
  L DP ;
  R 0 0, 40 27 ;
  L OD ;
  R 10 10, 30 17 ;
  L PS ;
  P 9.5 3.5, 22 3.5, 22 20, 18 20, 18 7.5, 9.5 7.5, 9.5 3.5 ;
  L SP ;
  R 5.5 0, 33 20 ;
  L CO ;
  R 11 4, 14 7 ;
  R 12 12, 15 15 ;
  R 25 12, 28 15 ;
  L IN ;
  R 0 3.5, 15 7.5 ;
  R 0 11.5, 16 15.5 ;
  R 24.5 0, 28.5 16 ;
END. {N-ch MOS}
```

## 2.7 Optical Tricks for Nanoscale Structures on ICs [1.7]

Several years ago, it was said that optical lithography, similar to photographic printing, used to print IC geometries from the various masks onto the silicon wafer (Fig. 1.1) should go down to 1 or 0.5 micron, but should fail for smaller geometries. It was due to the light wavelength that is too large to print geometries with smaller sizes than the wavelength. However, this prediction was wrong, as lithography is still used today for 130 to 90 nanometers technologies, and will be used as well for 65 nanometers!

Figure A shows how optical lithography is used to print very small geometries on to a silicon wafer. Some illuminator sends light through the mask and a lens that reduces the geometries (a factor of 4) to the wafer. The photoresist, polymerized by UV, can be removed, as it is soluble in acid, resulting in printing mask geometries on to the wafer. Regions unprotected by photoresist can then be also removed or etched by gases and finally the photoresist is removed itself. Several rounds are necessary for each layer and corresponding mask in the fabrication process.

So it seems difficult to generate 90 or 65 nanometers geometries with light wavelengths, which are today 193 nm (argon fluoride laser) and could reach 100 nm in the future. Sure to have smaller light wavelength is necessary, as well as nearly perfect lens (numerical aperture close to 1). However, some tricks are proposed today to improve significantly this process. It seems that these techniques could reach 1/8 of the light wavelength geometries, so about 10 nanometers, and some prototype transistors of 9 nm have been built.

Figure B shows how the mask is modified to result in much better geometries on the silicon wafer. It is called "Optical Proximity Correction" and its goal is to correct the distortions of the image as the light passes through the mask and lens. Such corrections are obviously performed by software (the designer still see a "correct" layout). Specifically, the corrections performed are different for a same geometry if it is isolated or if it is near to many other structures. Extremities of geometries, that do not receive enough light, are enlarged on the manipulated masks, to increase the amount of light and to have better printed extremities.

Another correction is necessary for many parallel structures due to the diffraction of light through the mask. The correction shown in Figure C is phase shifting of the light coming from two adjacent lines on the mask and they cancel each other in regions where no light is wanted. This phase shift is created on the mask by etching away some areas of the mask to achieve a 180 degrees phase shift.

Another correction, shown in Figure D, also for many parallel structures, consists in a double mask exposition, with some dipole element between the illuminator and the mask, when parallel structures change their directions. It allows the light to reach the wafer only in some angles due to diffraction and the goal is to match the angle of illumination and the angle of diffraction.



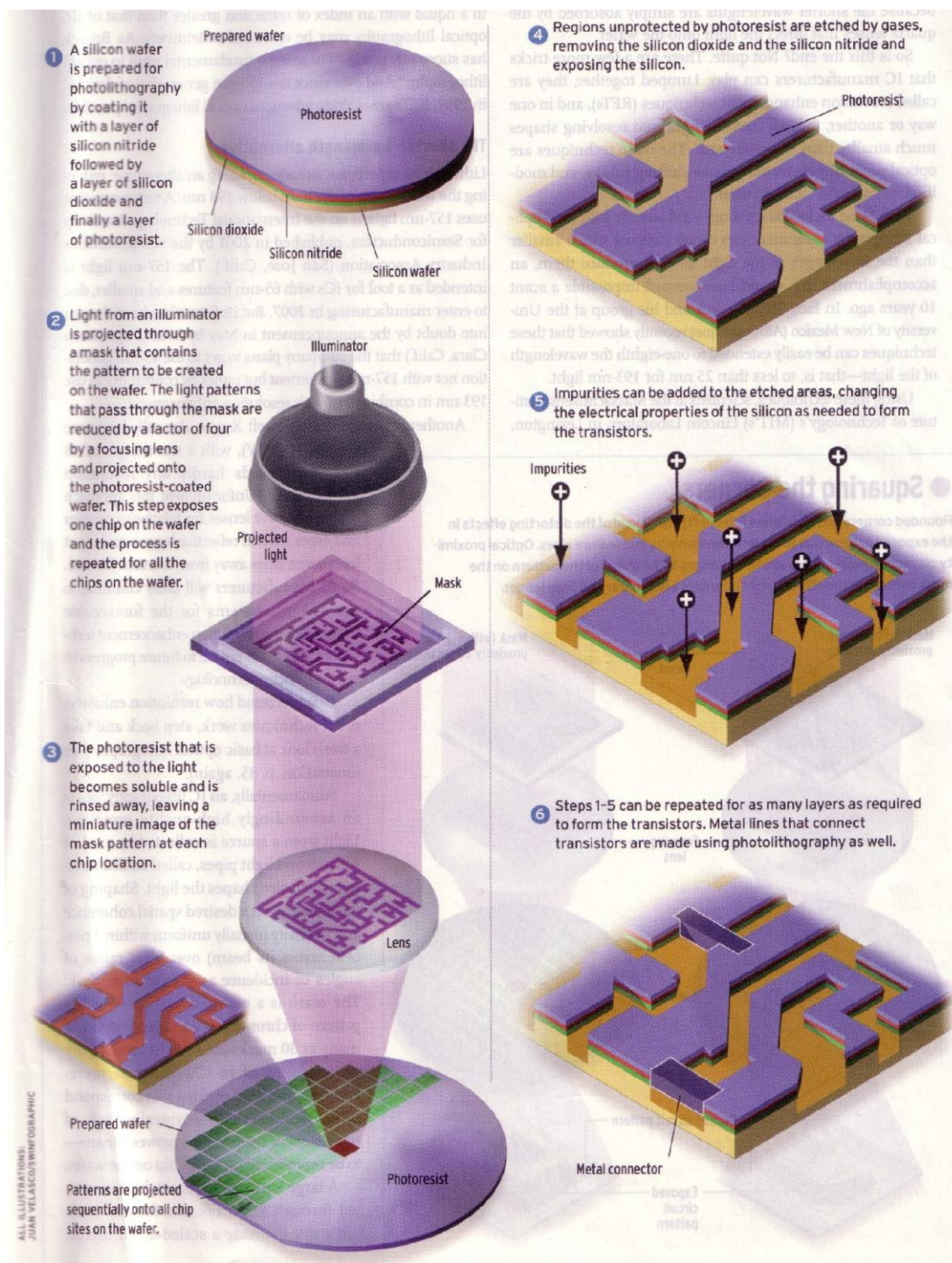


FIGURE A. Optical Lithography (Spectrum, September 2003)



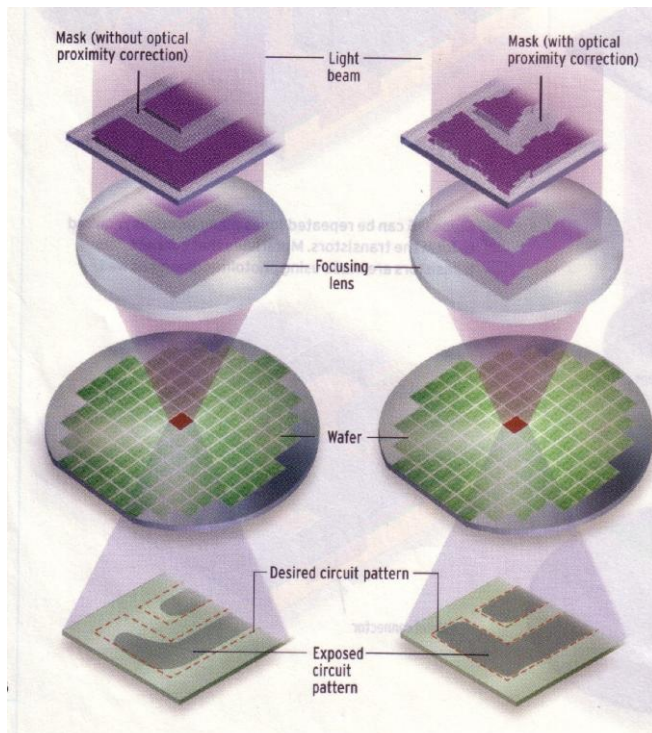


FIGURE B. Optical Proximity Correction (Spectrum, September 2003)

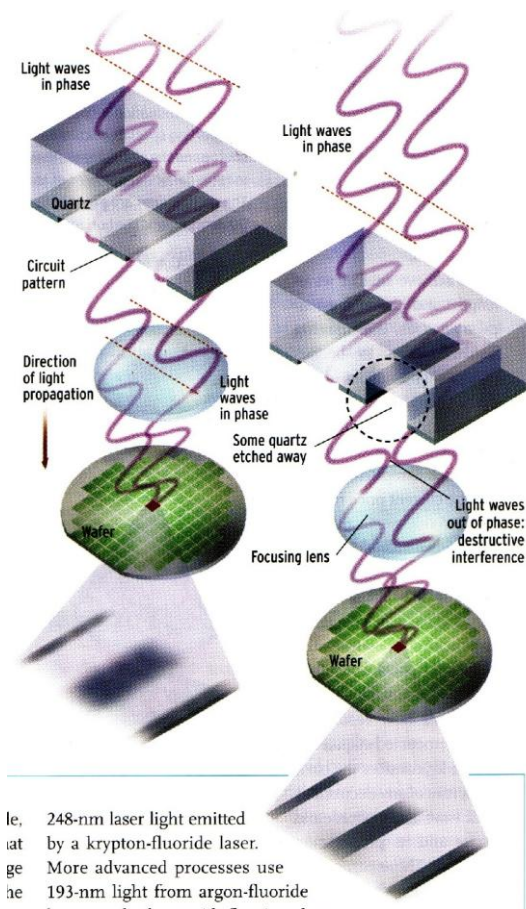


FIGURE C. Light Phase Shift by Mask Etching (Spectrum, September 2003)

## Sharpening Repetitive Patterns

Off-axis illumination improves the resolution of repetitive patterns by inserting special optical elements [inset, right] between the light source and the circuit mask. These elements direct the light onto the mask in particular patterns of angles. Here, a dipole element is used. The vertical lines are exposed first [left], the dipole mask is rotated 90 degrees, and the horizontal lines are exposed [right].

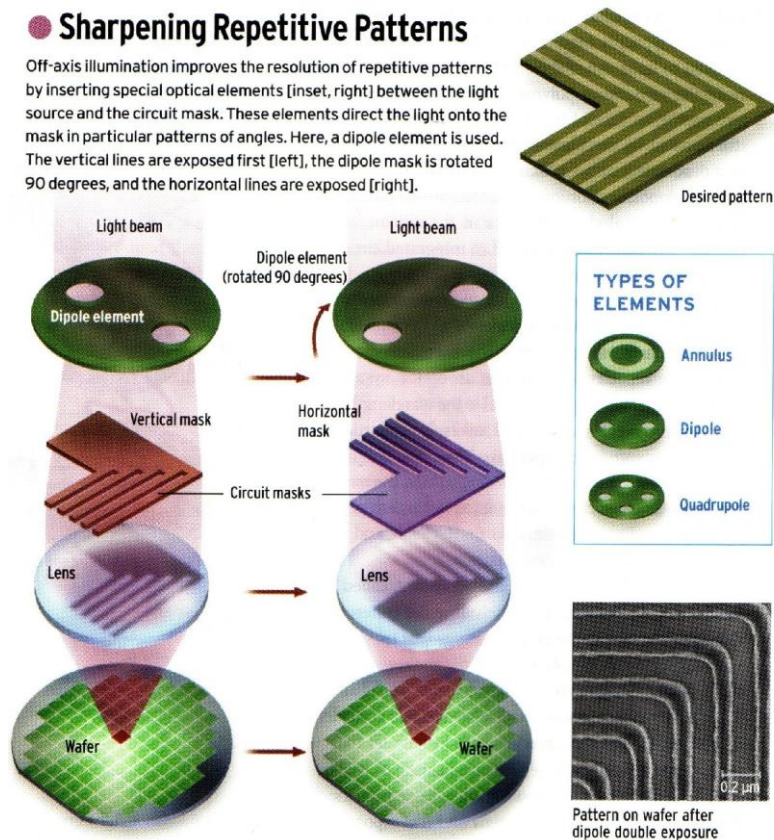


FIGURE D: Double Exposition ([1.7], Spectrum, September 2003)

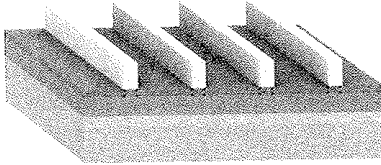
The last improvement in photolithography is double patterning. Instead of exposing the photoresist layer once under one mask, expose it twice. After the first exposition, one has geometries that are quite well separated. The second exposition constructs geometries between the first ones, as shown in Fig. E. So density is increased of a factor 2.

Double patterning is very difficult to implement (and cost is double cost, so a large increase). It is due to the fact that one has to repeat two times the process. A big difficulty is the possible misalignment of the two masks, which is a fatal error.

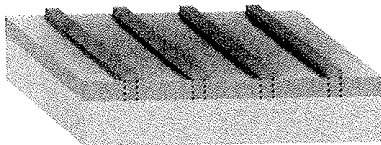
There are different techniques to achieve this double patterning, but they are all quite costly and difficult to implement practically. The joke is to say that it is the worse alternative due to its cost, but .... we do not have any other...

**LITHO-ETCH-LITHO-ETCH (LELE)**

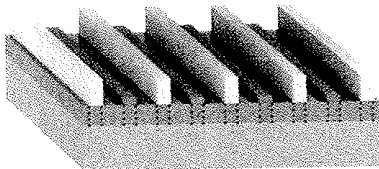
**1 Litho 1.** The first pattern [yellow] is exposed onto a hard mask.



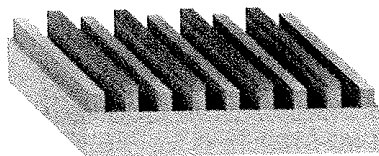
**2 Etch 1.** The first pattern is etched into the hard mask [brown].



**3 Litho 2.** A second pattern [yellow] is exposed onto silicon [blue], doubling pattern density.



**4 Etch 2.** The final, double-density pattern is engraved into the silicon.



**5 Wash.** The remaining mask is washed away.

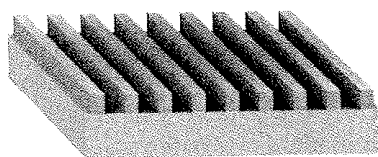


FIGURE E : Double Patterning (Spectrum, November 2008)

**2.8 OPC and DFM (Optical Proximity Correction and Design For Manufacturing)**

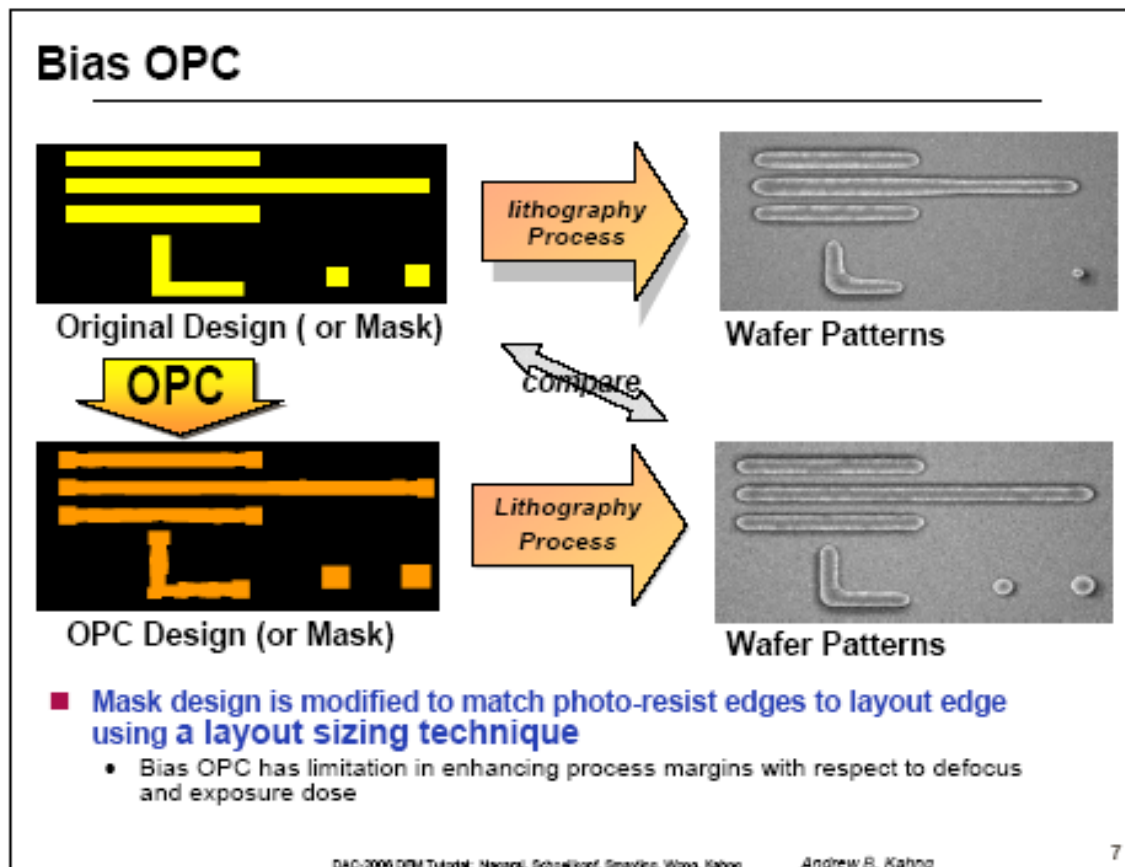
As mentioned above, minimum features size of circuit patterns becomes significantly smaller than the lithographic wavelength. So on chip, we have the following effects compared to the layout:

- Corner rounding
- Line-end shortening
- Line width shrinking

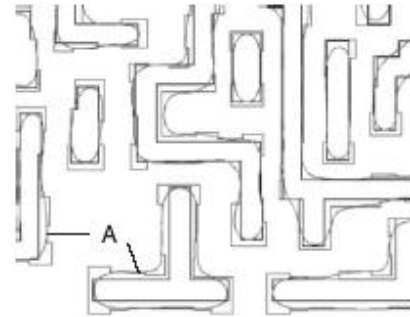
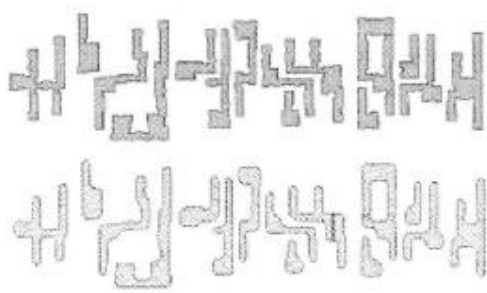
So OPC (Optical Proximity Correction) is used to modify the masks in such a way that on chip, one has geometries that are similar to the original layout. Line-ends are, for instance, enlarged as well as some line widths.

Figure below described first the original layout. If used as such, many geometries are not printed on silicon. So the mask is modified according to many OPC rules and one can see that most geometries of the original layout are printed on silicon.

Another very important domain is DFM, i.e. Design For Manufacturing. The idea is to adapt the layout style to the OPC rules. For instance, a more regular layout can be fabricated more easily, so logic blocks such as regular PLA or ROM memories, or some kind of regular layout, could be used







Here, quite regular layout  
With only horizontal metal lines and  
vertical poly and diffusion lines

## Références.

- [1.1] C. Piguet, A. Stauffer, J. Zahnd "Conception des circuits ASIC numériques CMOS", Dunod, Bordas, Paris, 1990
- [1.2] R. F. Ayres "VLSI silicon compilation and the art of automatic microchip design" Prentice-Hall, Inc., Englewood Cliffs, N.J., 1985.
- [1.3] C. Mead, L. Conway "Introduction to VLSI Systems" Addison Wesley, 1980.
- [1.4] T. W. Griswald "Portable Design Rules for Bulk CMOS" VLSI Design, Sept-Oct. 1982, pp. 62-67.
- [1.5] C. Piguet, A. Stauffer "Synthèse de circuits ASIC. Des choix méthodologiques aux applications industrielles" Dunod, Bordas, Paris, 1990.
- [1.6] C. Piguet, H. Hügli, « Une brève Histoire du calcul », Presses Polytechniques et Universitaires Romandes PPUR, février 2004
- [1.7] F. Schellenberg, "A Little Light Magic", IEEE Spectrum, pp. 34-39, September 2003

## Problems

### Layout Tools:

The following problems consist to design layout in a given technology. There are two ways:

- 1) To use a graphic editor (PC Corel Draw, Word, Illustrator, Canvas) with a given grid to design several cells (INV, NOR, NAND, others) according to the design rules of the technologies described at the beginning of the Chapter 1 and based on  $\lambda$  parameter (with  $\lambda = 1$  grid unit). Use colours (blue for metal, green for diffusion, red for poly, black for contact (or a cross), dark for well).
- 2) To use MicroWind2 from the French school INSA. There is a DRC (Design Rule Checker) embedded in the graphic editor, so you are sure to design a correct layout. Furthermore, you can extract your layout and to simulate it with an analog simulator.

It can be freely charged from the Web at the address:

[www.microwind.org](http://www.microwind.org)

The commercial site for Microwind is [www.microwind.net](http://www.microwind.net).

There are two different tools (see annexe for the front page):

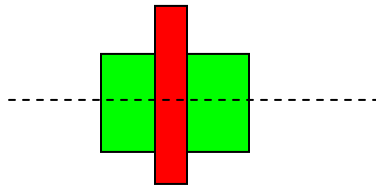
- 1) MicroWind, Layout Editor and Simulator
- 2) Dsch2, a logical editor and simulator (gates and transistors)

We will use here the first tool MicroWind, but you can use the second tool Dsch2 for schematics in transistors and gates.

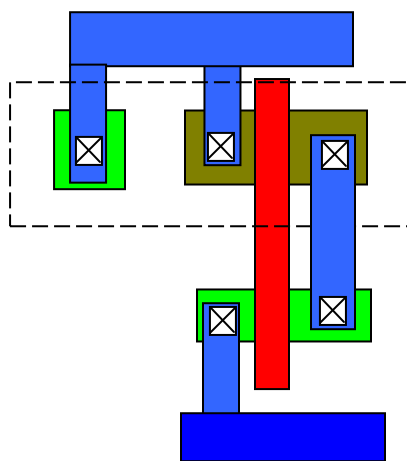
### Problem 1.

This is a very easy problem, just to design a MOS transistor while using MicroWind. You have:

- to start MicroWind2, you have the window shown in the third figure of the annexe
- to choose a technology (menu file, item "select a fondry")
- to use the palette, design a polysilicon line and a N+ dissusion rectangle. You see the lambda dimensions in the bottom message of the window.



- to use menu simulate, item "process section in 2D", click a dotted line like in previous figure, and you will see the process section
- to use menu simulate, item "MOS characteristics" or the "Simulate MOS characteristics" icon, and you will see the MOS  $I_d=f(V_d, V_g)$ . You can change the model (model 1, model 3), you can also change the parameters of the model.
- MODEL 1 is a very simple model (only valid for channel length of  $> 10 \mu\text{m}$ ) with:  
Saturated mode:  $I_{ds} = \frac{1}{2} \cdot K_P \cdot (W/L) \cdot (V_{gs} - V_t)^2$
- You can change the parameters,  $V_t$ , temperature, you can choose another larger or bigger lenth MOS, you can choose another technology
- MODEL 3 is a more sophisticated model, check the big differences while using various technologies.
- To use the menu simulate item "process steps in 3D" and go through the "next steps" button to see how is constructed a MOS transistor
- To simulate the MOS transistor. You have to apply some input or clock signals to the inputs, for instance a clock to the gate and another one to the source of the transistor. You have to click on the clock icon (third row of the palette) and place the icon on the gate and on the source. You have a window in which you can define the period, for instance, a clock 2 times faster for the source. You have also to observe the output, i.e. the drain, by placing the "visible node" icon on the drain. Save the layout before simulation
- Simulate by using the simulate menu, item "run simulation". You can change the clock period by clicking on clock1 or clock2, and also modifying the clock names. On the simulation window, you can see more by using the "more" button.
- To redo the exercice for a P-ch MOS



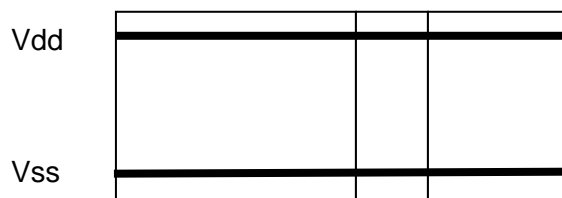
### Problem 2.

You have to design an inverter layout with Microwind and to simulate it. Such a layout is shown below. There is a n-well (dotted line) in which P-ch transistors are located (be careful that it is a n-well (the only well provided in Microwind) and **NOT** a p-well as it is the case in Figure 1.2. It is why it is used for P-ch transistors). There are two different diffusions in green and a vertical polysilicon in red. The top blue metal line is Vdd, the bottom blue line is Vss- The output is the third blue metal line.

There is at top left a N+ green diffusion in the N-well connected through a blue metal line to Vdd. This so-called butting contact is used to fix the voltage of the n-well, otherwise the well is floating (even Microwind says that it is dangerous if the well is floating). So a N+ diffusion in a N-well material results in a good contact, and if the N+ diffusion is connected to Vdd, it means the n-well is at the Vdd potential (what is required for P-ch transistors).

### Problem 3.

To use a graphic editor to design several cells (INV, NOR, NAND, others) according to the design rules of the technologies described at the beginning of the Chapter 1 and based on  $\lambda$  parameter (with  $\lambda = 1$  grid unit). Use colours (blue for metal, green for diffusion, red for poly, black for contact (or a cross), dark for well). Use a concept in which cells have the same height and can be abutted with the inputs/outputs on the two non-abutted sides of the cell. ). Compare the cell area in similar technologies ( $\lambda=1 \mu\text{m}$  in  $2 \mu\text{m}$  process or  $\lambda=2 \mu\text{m}$  in  $4 \mu\text{m}$  process).



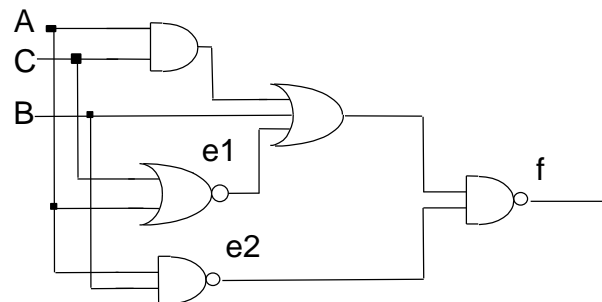
You can design and simulate these cell layouts with using minimal rules (it works fine in simulation!). But more advanced exercises consist to design these cell layouts while using the minimal layout rules of the chosen technology in order to have minimal silicon areas. For deep submicron technologies, you will probably see that minimal layout rules are different that the  $\lambda$  rules described in Section 2.4.

### Problem 4.

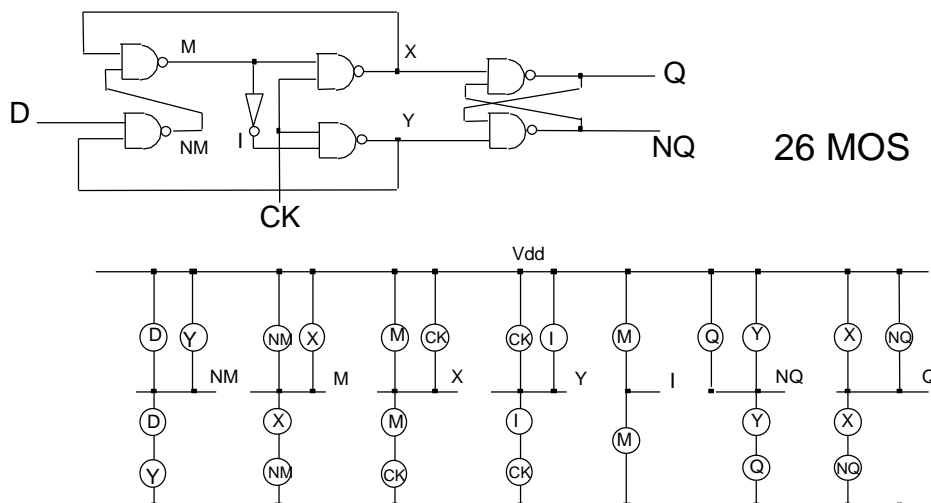
An interesting exercise is to look at the layouts of the Microwind database, and to try to read these layouts to establish the transistor schematic. You could also see how, for complex layouts, the different cells are systematically interconnected.

### Problem 5.

Use the very small library designed in problem 1 to implement with inv, nor and nand gates the logical function  $f$  defined below. Use Boolean transformations (de Morgan) to get a gate schematic using only the cell in your library. Use the "group" function for each cell to copy/paste your cells in the better way and interconnect them with metal/poly wires to achieve the required function. Try several cell placements to try to improve the layout density.

**Problem 6.**

Use the very small library designed in Problem 1 to implement with INV and NAND gates the D-Flip-Flop shown below. Generally D-Flip-Flop is designed within one standard cell, with all interconnections inside the cell. One can be convinced that the layout design is therefore much more difficult.

**Problem 7.**

Look at the layout database of MicroWind, select 2 or 3 layouts and try to perform some reverse engineering to get the transistor schematics from the layout.

As easy layouts to analyze, you have:

TGate.MSK, xor2.MSK, invSmeSize.MSK, Inverter.MSK, inv3.MSK

As more complex but quite interesting layouts, you have:

Ram1.MSK, fadd.MSK, D latch.MSK, buffer.MSK

**Annexe: Web page of MicroWind**

*Welcome to Microwind2 homepage*



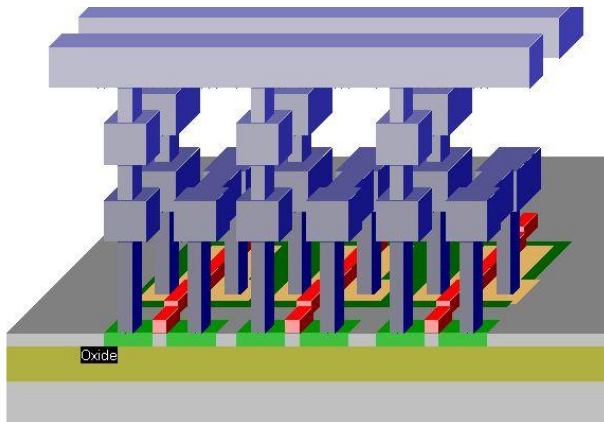
More about  
Microwind2:

[What is Microwind Editor](#)  
[3D View](#)  
[2D View](#)  
[Analog Simulation](#)  
[Mos Models](#)

More about  
Dsch2:

[Logic editor](#)  
[Logic simulator](#)

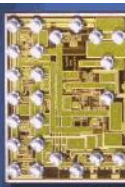
Partners  
[Matra S&I](#)  
[STInfinion](#)



## Introducing CMOS circuit design on PC

Introduction à la conception CMOS sur PC

Freeware Windows 95,98 or NT



[Belles images microelectron](#)

[Beautiful ima microelectron](#)

- [Charger gratuitement MICROWIND2 \(700K zip\)](#) (v 07 May 2001)
- Charger gratuitement [Dsch2 \(500K zip\)](#) [Schématique](#) 12 June 2001
- [Documentations et Planches PPT en ligne](#)
- [Présentation des logiciels](#)

- [Load freeware MICROWIND2\(700K zip\)](#) (v 07 M
- [Load freeware DSCH2 \(Schematics\)](#) 12 June 200
- [Load documentation and PPT slides](#)
- [Presentation of the software](#)

International links : [Motorola](#) | [Atmel](#) | [Europractice](#) | [Sematech](#) | [IEEE](#) | [BSIM](#) Berkeley | [Medea](#) | [Mesdie](#)  
[\[Retour\]](#) [\[Back\]](#) Update 12 Jun 01

## Présentation/ Presentation

Microwind2 est un logiciel gratuit sur PC pour la conception et simulation *Microwind2 is a friendly and free PC tool for designing and simulating microelectronic circuits at layout level. The tool contains full editing facilities (Copy, cut, past, duplicate, move), 2D and 3D views of the circuit, MOS characteristics, 2D cross section, 3D process viewer, and an on-line analog simulator.*

DSCH2 est le logiciel compagnon de Microwind2, pour la conception de circuits logiques. A l'aide de primitives, un circuit hiérarchique peut être conçu et simulé. Des symboles interactifs facilitent la simulation. La simulation logique inclut les retards de commutation et l'évaluation du courant consommé. *DSCH2 is the companion software for logic design. Based on primitives, a hierarchical circuit is built and simulated. Interactive symbols are used to friendly simulation, which includes delay and power consumption evaluation.*

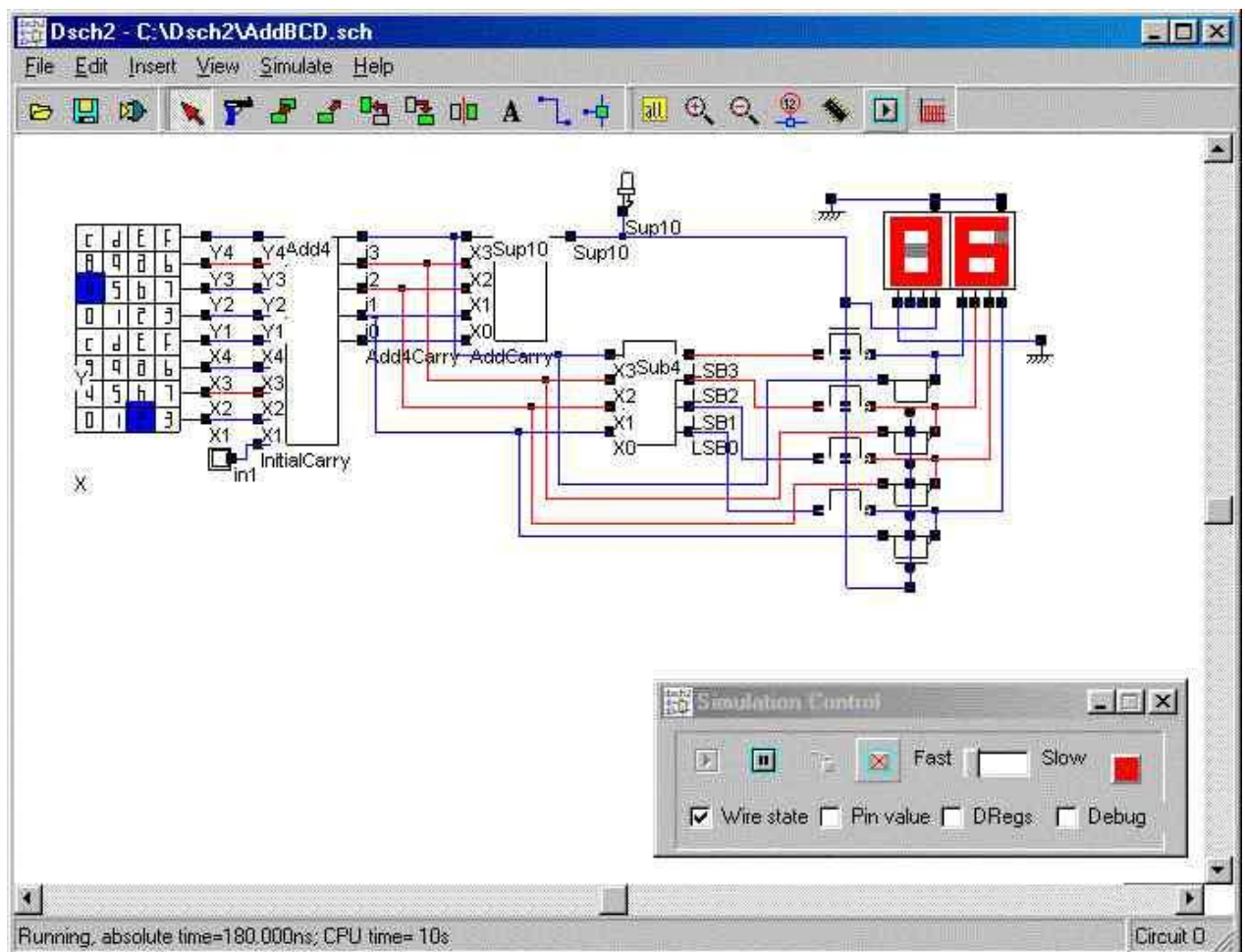
- Gratiiciel Windows 95,98 ou NT, 486, 16Mo Minimum
- [Charger gratuitement MICROWIND2](#)
- Charger gratuitement [Dsch2](#)
- Freeware Windows 95,98 ou NT, 486, 16Mo Minimum
- [Load freeware MICROWIND2 now](#)
- Load [Dsch2](#)

[\[Haut\]/\[Top\]](#)

## Editeur et simulateur logique/Logic editor and simulator

DSCH2 est le logiciel compagnon de Microwind2, pour la conception de circuits logiques. A l'aide de primitives, un circuit hiérarchique peut être conçu et simulé. Des symboles interactifs comme les claviers, leds, et afficheurs facilitent la simulation. La simulation logique inclut les retards de commutation et l'évaluation du courant consommé.

*DSCH2 is the companion software for logic design. Based on primitives, a hierarchical circuit is built and simulated. Interactive symbols such as keyboard, led and displays are used for friendly simulation, which includes delay and power consumption evaluation.*

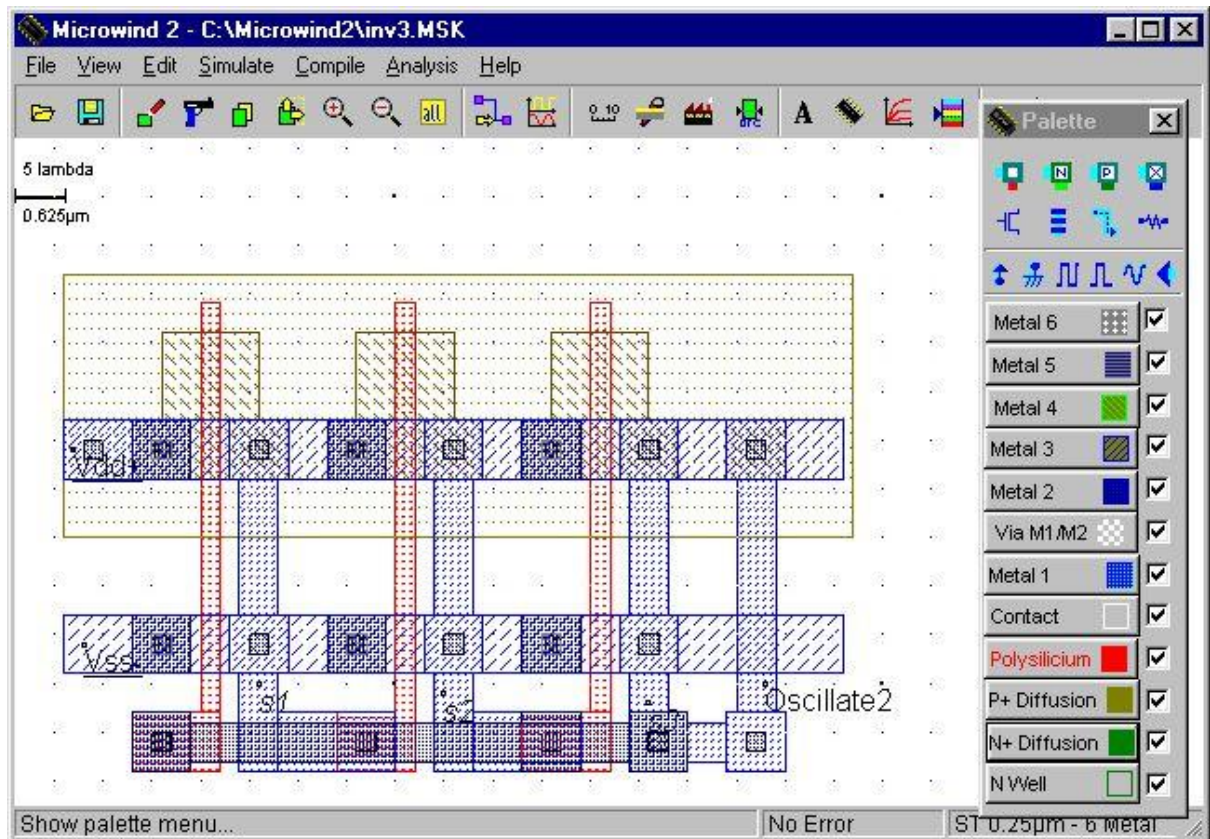


[\[Haut\]](#)/[\[Top\]](#)

***Microwind2 Editor***

L'écran principal de Microwind2 est donné ci-dessous. Vous pouvez charger un dessin de masques, l'éditer (cut, paste, copie, matrices, etc...), insérer des contacts, des transistors, des plots, par un simple clic à la souris.

*This is the main screen of Microwind2. You may edit boxes layers. You may cut, past, duplicate, generate matrix of layers. You may use the editor to insert contacts, MOS devices, pads, complex components with a single click.*

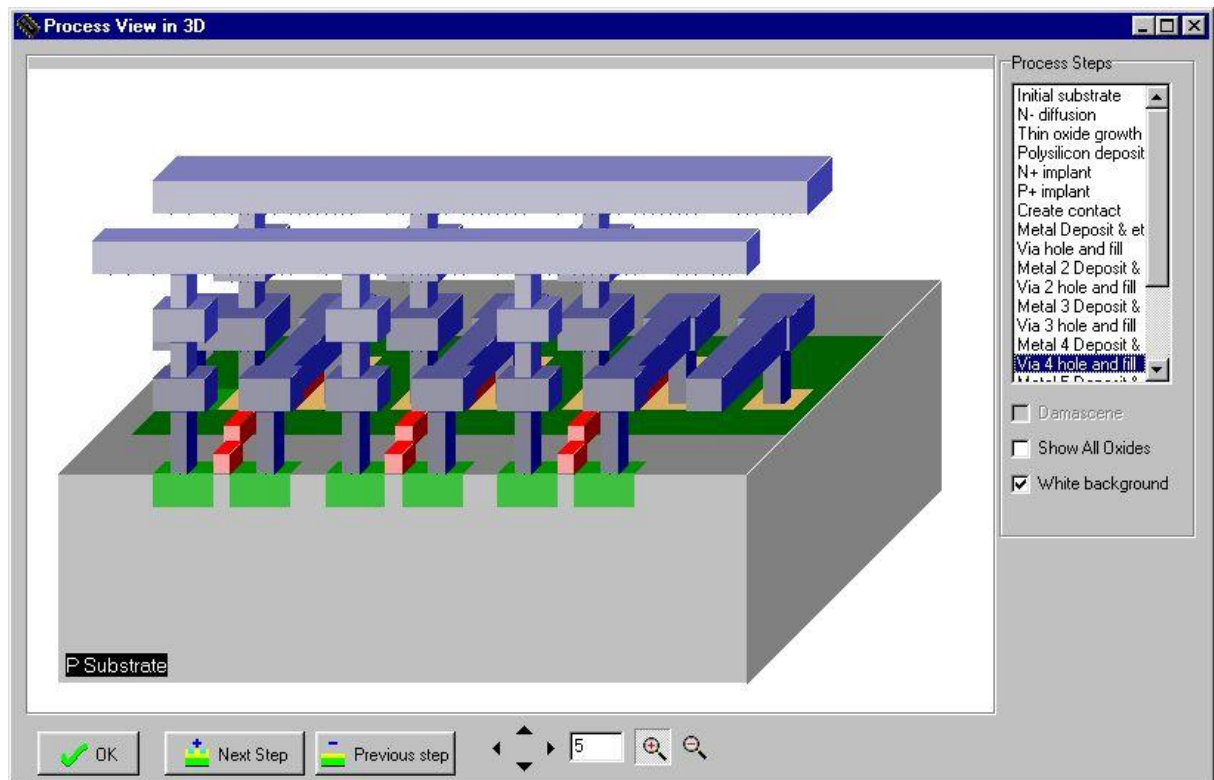


[\[Haut\]/\[Top\]](#)

## Microwind2 3D viewer

Vous n'enseignerez plus la technologie sub-micronique comme avant. Sans doute LA commande qui fait le renom de ce logiciel. La vue 3D, étapes par étapes, permet de voir la fabrication du circuit dessiné, en particulier l'effet d'auto-alignement, les contacts et les interconnexions. Possibilités de zoom, de décalage en X, Y et Z.

*You will never teach deep-submicron technology like before. to see the step-by-step fabrication of any portion of layout. contacts and metallizations are created. See the self-aligning polysilicon gate is fabricated. Zoom or shift the drawing at X, Y and Z.*



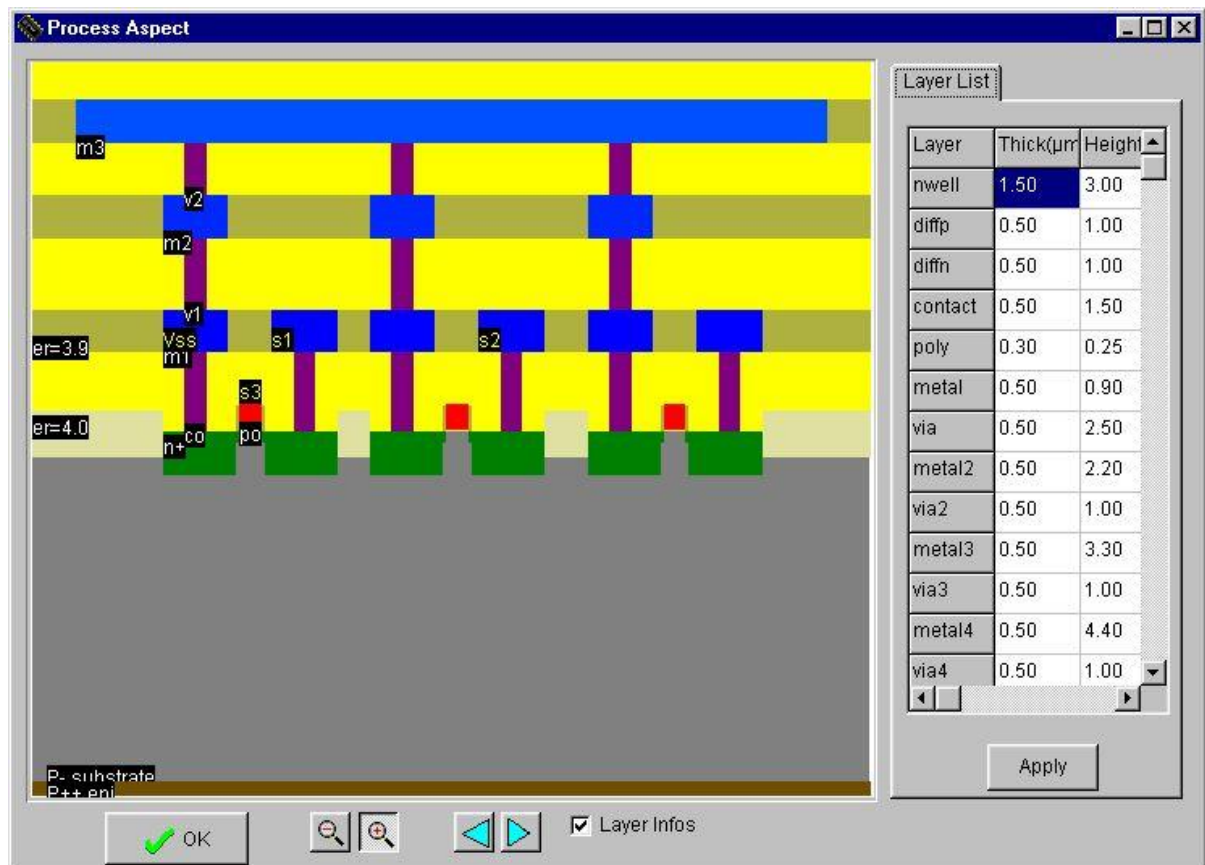
[\[Haut\]](#)/[\[Top\]](#)

## *Microwind2 2D viewer*

Une commande très intéressante pour comprendre les structures d'oxyde, les matériaux à faible permittivité (Low K) et la passivation. Faites un zoom sur l'oxyde de grille ultra fin, sur les structures laterales du MOS (LDD). Les signaux électriques (Vss, Vdd, horloges) sont reportées sur la vue 2D.

*Again a valuable tool for understanding the oxide structure (Low K) and high K (SiO<sub>2</sub>) sandwich, and passivation. Zoom on the MOS lateral drain diffusion structure. See the VDD*



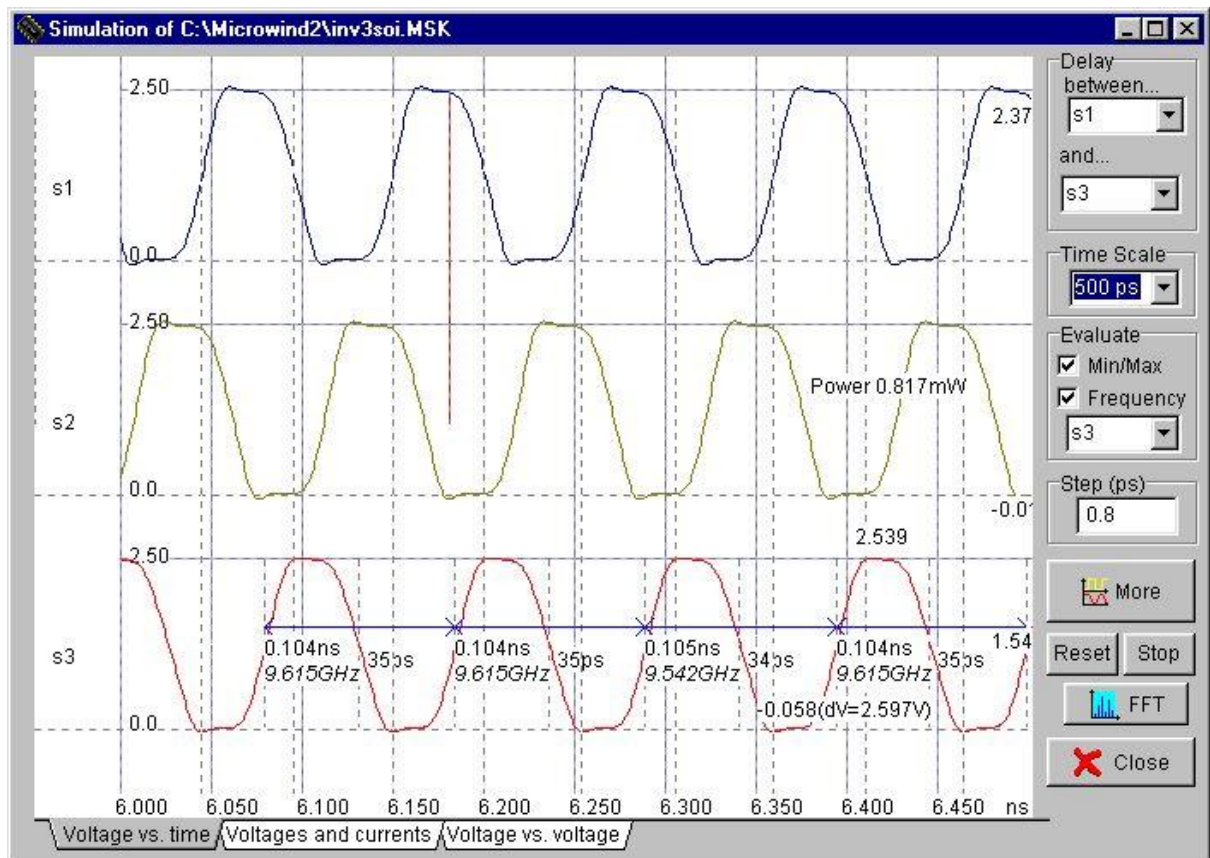


[\[Haut\]/\[Top\]](#)

## *Microwind2 analog simulator*

Pas de simulateur externe requis. L'extracteur/simulateur analogique est inclus et complètement intégré. Basé sur les modèles 1, 3 ou 9, le simulateur calcule rapidement les tensions et courants en fonction du temps, avec des calculs intuitifs de fréquence, de délai, de min/max. Même la puissance est calculée. La simulation est configurée pour une large plage de technologies: 1.2 micron jusqu'à 0.07 micron (J'ai bien écrit 0.07).

*No SPICE or external simulator is needed. The analog simulator uses MOS model 1,3 or MOS Model 9, features fast time-domain simulation, current estimation, with very intuitive post processing: frequency, delay estimation. Even power estimation is on-screen. The simulation is performed on a wide range of technologies: 1.2micron down to 0.07micron (I did say 0.07).*

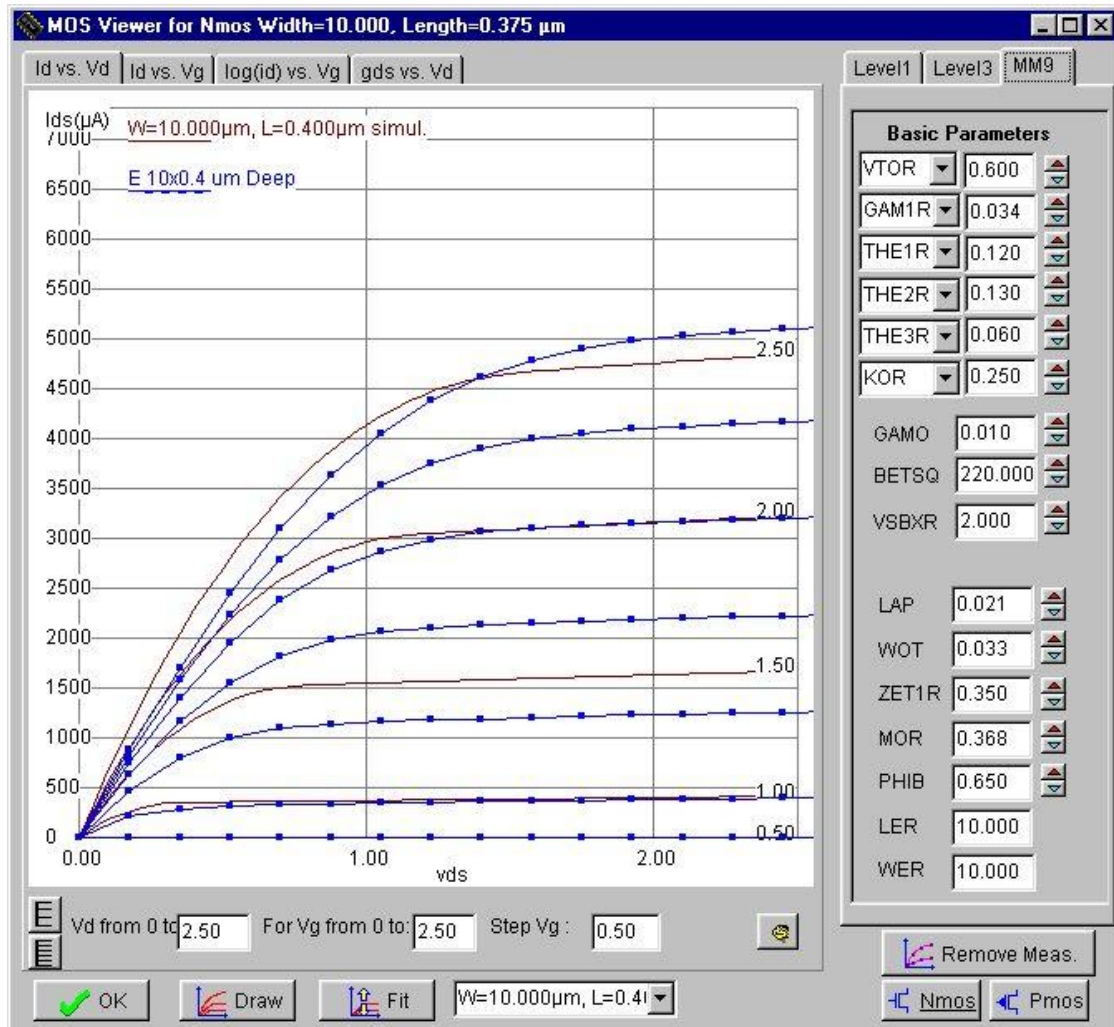


[\[Haut\]](#)/[\[Top\]](#)

## *Microwind2 tutorial on MOS devices*

Un écran irremplaçable pour comprendre les caractéristiques des MOS, avec une interface que vos étudiants vont apprécier. Changez les paramètres du modèle pour caler les courbes sur les mesures que NOUS avons faites en 0.35, 0.25 et 0.18 micron avec ST Microelectronics. Dans le [manuel](#), un tutorial, et beaucoup d'information sur les modèles de MOS ainsi que la signification de leurs paramètres.

In english: tutorial to understand MOS models. One can modify the MOS model parameters to fit to the curves proposed for 0.35, 0.25 and 0.18 micron of ST Microelectronics. In the book, you will find a tutorial as well as information on MOS models with the meaning of their parameters.



### 3. CMOS DESIGN

#### 3.1. Synthesis Method: CMOS Gate Synthesis

To design a CMOS gate, one has to derive the *symmetrical logical equation* while using the Karnaugh maps of the logic function. One has to notice that the designer has to perform twice the logic simplification, i.e. one synthesis for the N-ch network while considering blocks of '0' and one synthesis for the P-ch network while considering blocks of '1'. From the *symmetrical logical equation*, one can derive the two structural expressions and therefore the transistor schematic of the considered CMOS gate.

This method, i.e. to perform twice the Boolean simplification for each N-ch and P-ch networks, is called « separated simplification » method, as one has to consider in a first step the blocks of '0' in the Karnaugh map to generate the N-ch network and in a second step the blocks of '1' of the Karnaugh maps to generate the P-ch network. Figure 1.11a shows the synthesis of the simplest CMOS gate (the inverter) and Figure 1.11b the synthesis of a NAND gate with 2 inputs.

The « separated simplification » method is not the only one available. However, the « separated simplification » method provides several advantages like the following:

- possibility to have the two structural expressions as sums of products
- the transistor networks can therefore be implemented as branches of serial transistors connected between the output and Vdd or Vss
- a more regular layout
- very simple timing models, i.e. to charge or discharge the output capacitance through one branch
- a better testability

Goal: to generate the 2 structural expressions  $z_N$  and  $z_P$  from a truth table or Karnaugh maps while using the symetrial logic equation

Separated Simplification Method:

- to take the '0' blocks for the N-ch network
- to take the '1' blocks for the P-ch network

Example : inverter

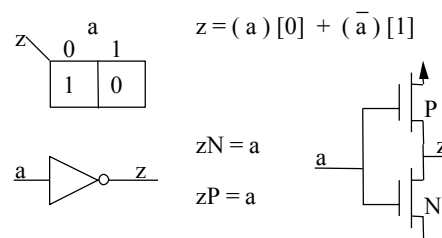


Figure 1.11a Synthesis of an inverter with the « separated simplification method



Other example : NAND Gate

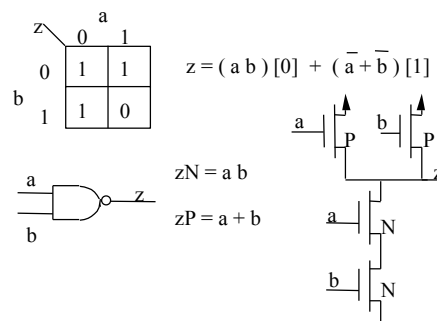


Figure 1.11b Synthesis of a NAND gate with the separated simplification method

Figure 1.12 shows the synthesis of a complex CMOS gate while using the proposed separated simplification method.

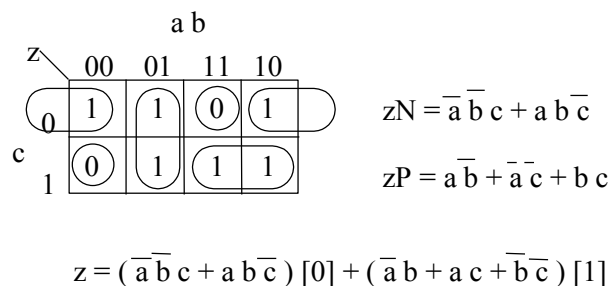


Figure 1.12 Complex CMOS gate generated by the separated simplification method

Figure 1.13 shows the design of a CMOS complex gate. The separated simplification method is used for the branch-based implementation (at the right of the first picture) while the schematic at the left is a conventional implementation with bridges between serial transistors. These bridges have to be implemented in layout with contacts and metal layers, resulting in wasted silicon area, larger parasitic capacitances and higher power consumption. On the right of the picture, Karnaugh maps are used to generate the N-ch network (blocks of "0") and the P-ch network (blocks of "1").

The branch-based schematic is more regular and the layout is easier to design. It will be a key point in the future as regular layout will be mandatory in sub 65 nm technologies. However, this example shows also the drawback of this technique: the branch-based N-ch contains more transistors (9) than the other (6). It is sometimes the case for complex logic gates. However, for simple gates, only a few supplementary transistors have to be used in branch-based logic.

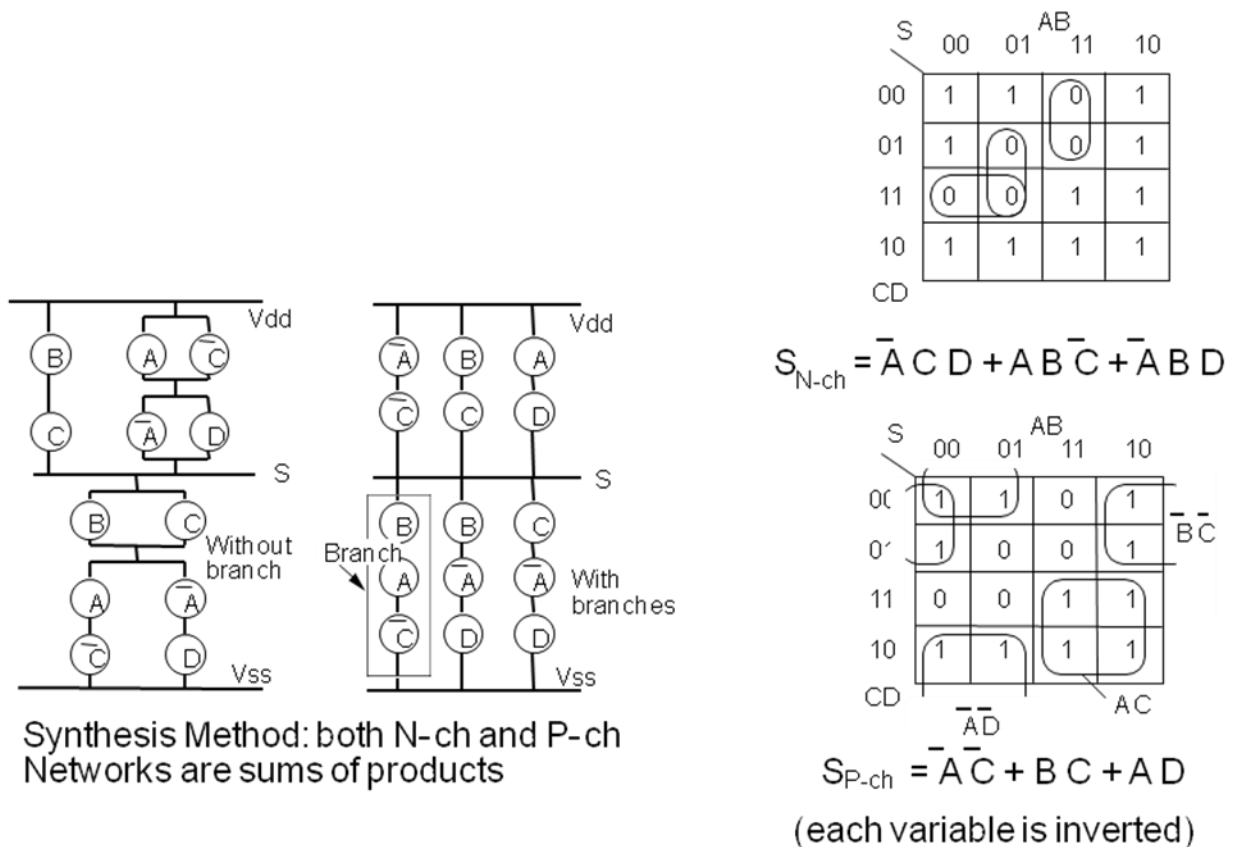


Figure 1.13 Complex CMOS gate generated by the two design methods

### 3.2. Circuits with transmission MOS or Gates

The same basic method can be applied to circuits with transmission MOS or gates. The best choice to explain this is to start from a transmission gate-based schematic in order to see how cubes appear on the corresponding Karnaugh map. Figure 1.14 shows a 2:1 multiplexer constructed with two transmission gates. The basic explanation goes from top to bottom, but the proposed synthesis method can be seen from bottom to top of Figure 1.14.

Figure 1.14a shows the conventional 2:1 multiplexer that connects one of the two inputs b or c to the output depending on the value of the control input a. It is constructed with 2 transmission gates, one of them being conducting and the other 'off'. Such a multiplexer is generally depicted as shown by Figure 1.14b with CMOS complementary transmission gates.

However, it is possible to draw the same multiplexer as shown in Figure 1.14c while using P-ch transistors in the top region and N-ch transistors in the bottom region. This drawing style is derived from gate-matrix layout [2.26]. This schematic (Fig. 1.14c) is based on branches that are connected between the output w and an input variable (and not Vss or Vdd as it is the case in pure static logic style).

It is therefore easy to derive the structural expression of the N-ch and P-ch networks:

$$wN = (a) [c] + (\text{NON}(a)) [b]$$

$$wP = (\text{NON}(a)) [c] + (a) [b]$$

By using  $w = C0 + C1$ , with  $C0(a) = wN(a)$  and  $C1(a) = wP(\text{NON}(a))$ , one has:

$$w = (a) [c] + (\text{NON}(a)) [b] \text{ as symmetrical equation}$$

The expressions inside the parenthesis represent the cube borders as shown in Figure 1.14d. The variables inside the brackets represent the content of the cubes. Cubes not only contains '0' and '1' (i.e. [0] and [1], as it is the case for pure static logic), but also a set of '0' and '1'. They are arranged in such a manner that the content of a cube is dependent on a single input variable (i.e. [b] or [c]).

In the synthesis procedure, one has to search for such cubes on the Karnaugh map. A symmetrical equation can then be derived and *each transmission term (i.e. with [b] or [c]) provides a contribution to both N-ch and P-ch structural expressions.*

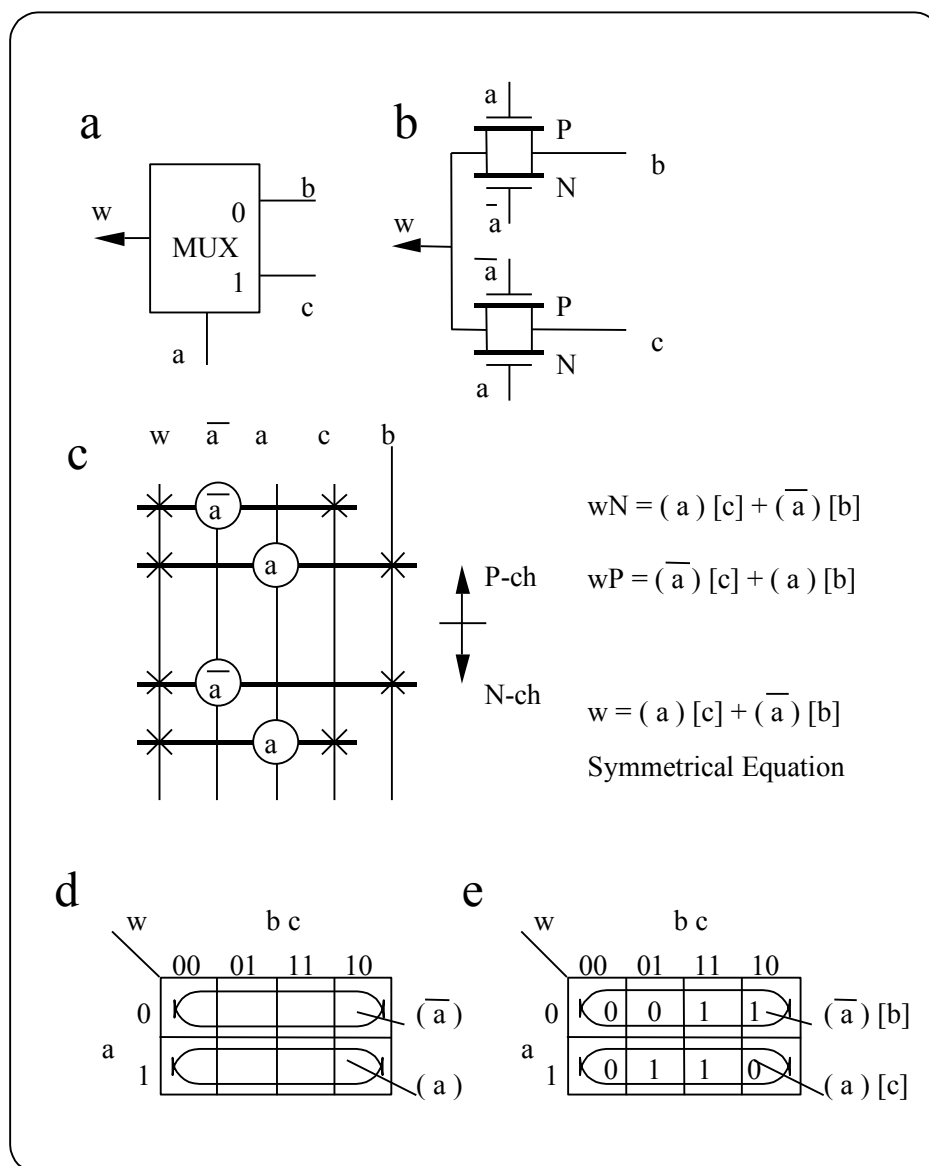


Figure 1.14 Analysis and Synthesis of a 2:1 multiplexer with transmission gates

Figure 1.15 shows another example for which one cube is a conventional cube with [0] and the other cubes are transmission cubes. This means that both cube types can be synthesized for the same circuit that has some branches connected to Vss (and Vdd) and other branches connected to input variables. This illustrates the fact that the proposed method is powerful.

Figure 1.16 shows Karnaugh maps of a 1-bit incrementer. The cells are connected together to have a N-bit incrementer by connecting the output Co of the preceding cell to the carry-in Cin of the next cell. The Karnaugh maps describe  $A=A+1$ , i.e. the arithmetic addition of Cin and a:

$0 + 0 = 0$ , carry 0  
 $0 + 1 = 1$ , carry 0  
 $1 + 0 = 1$ , carry 0  
 $1 + 1 = 0$ , carry 1

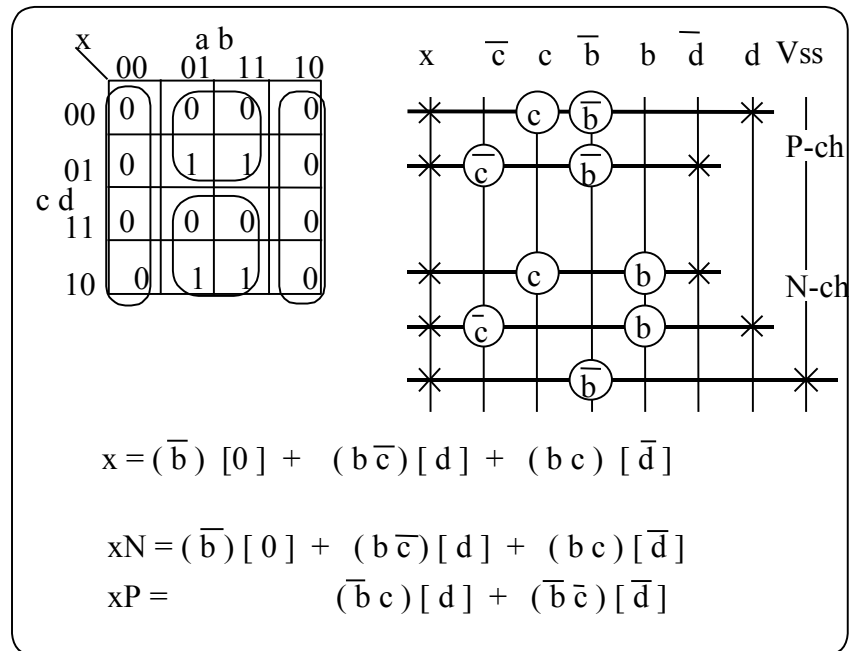


Figure 1.15 Synthesis of a complex CMOS gate with transmission gates

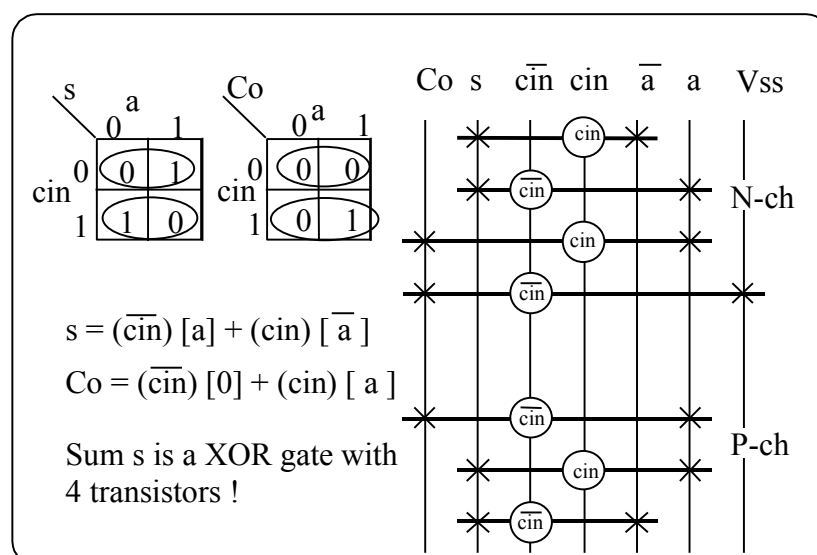


Figure 1.16 Incrementer

The sum  $s$  is XOR(Cin,  $a$ ) while the carry-out  $Co$  is AND(Cin,  $a$ ). The implementation of Figure 1.16 is based on transmission branches (other implementations do exist). One has to notice that the XOR



gate for the sum  $s$  is implemented with only 4 transistors! One has to take care of the carry propagation: if  $C_{in}$  is propagated to  $C_o$  through a single transmission MOS or gate (called Manchester Carry Chain), one has a distributed RC chain. The carry propagation is therefore very slow, as it is proportional to the number at square of transmission MOS or gate.

### 3.3. Exercises

The goal of these exercises is to design some logic functions in CMOS technologies, i.e. with MOS transistors. The tool Microwind (Dschr) can then be used to design and simulate these logic functions. Sections 2 and 3 present the design methodologies and Section 4 the Microwind tool.

#### *MicroWind*

The following problems have to be solved by a manual design, and then the resulting circuits have to be simulated with a very simple logic or transistor "simulator". This very simple simulator is the analog simulator MicroWind and the gate and transistor simulator Dschr2 from the French school INSA.

It can be freely charged from the Web at the address:

[www.microwind.org](http://www.microwind.org)

The commercial site for Microwind is [www.microwind.net](http://www.microwind.net).

There are two different tools (see § 2.23 for the front page):

- 3) MicroWind, Layout Editor and Simulator
- 4) Dschr2, a logical editor and simulator (gates and transistors)

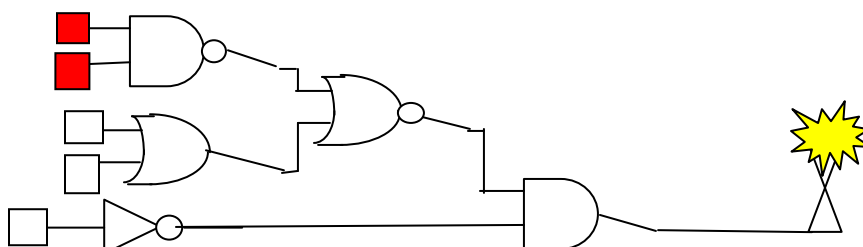
In this §, we will use the second tool Dschr2. The following three examples show how to use Dschr2 to "simulate" logic gates, transistor networks and Finite-State Machines.

#### *How to use Dschr 2.0:*

- 1) Click on Dschr2.exe
- 2) You have a menu bar as well as Library symbols
- 3) Drag library symbols (gates, transistors) to the location you want. The mouse pointer on the symbols in the library box indicates what the symbols mean.
- 4) Use the connect icon in the menu bar to draw connections. Point the start point and go to the other point while pressing the mouse button.
- 5) Drag the buttons in the library box (top left) to the inputs
- 6) Drag the LED light (second row, middle) to the output
- 7) Zoom out in the menu bar to see the complete schematic
- 8) You are ready to simulate
- 9) Take the simulate menu in the menu bar and start the simulation
- 10) You can press any input button to "1" (it becomes red) and try all the input combinations. The output LED light becomes eventually on.
- 11) Stop the simulation in the dialog box (4<sup>th</sup> icon in the first row)
- 12) The last icon in the menu bar (extreme right) allows you to display the timing diagram
- 13) You can use the dialog box at top left, red arrows to see different parts of the timing diagram. Maximum and average current are indicated, as well as power consumption.

You can save your circuit (circuit.sch), but with, if you click on the icon, Windows 2000 tries to open an Outlook schedule file. You have first to open Dschr2.exe and then to use the "open file" menu.

#### *Experience a)* With logic gates



This circuit is in fact not coming from Dsch2. Gates are represented as boxes with the following signs, & for AND, >=1 for OR, 1 for inverter .

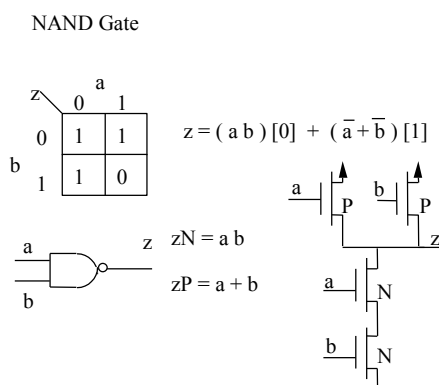
The simulated circuit can be translated in Verilog:

```
// DSCH 2.2a , Flat Verilog
// 23.07.2001 17:03:26
// Z:\Cours 2001\ALaRI\Cours 2001\ALaRI-un.sch

module EPFL-un(in1,in2,in4,in5,in6,out1);
  input in1,in2,in4,in5,in6;
  output out1;
  wire i0w1,i0w2,i0w3,i0w4,i0w5,i0w6;
  or or21(i0w3,in4,in5);
  nand nand21(i0w4,in2,in1);
  not not11(i0w5,in6);
  nor nor21(i0w6,i0w4,i0w3);
  and and21(out1,i0w5,i0w6);
endmodule

// Simulation parameters
// in1 CLK 10 10
// in2 CLK 20 20
// in4 CLK 30 30
// in5 CLK 40 40
// in6 CLK 50 50
```

### Experience b) With transistors



You also can add input buttons and the LED light at the output. You also can generate a Verilog file.

```
// DSCH 2.2a , Flat Verilog
// 23.07.2001 18:12:00
// example

module example( in1,in2,out1);
  input in1,in2;
  output out1;
  wire i0w1;
  pmos pmos1(out1,vdd,in1);
  pmos pmos2(out1,vdd,in2);
  nmos nmos1(out1,i0w1,in1);
```

```

nmos nmos2(i0w1,vss,in2);
endmodule

// Simulation parameters
// in1 CLK 10 10
// in2 CLK 20 20

```

### 3.4. PROBLEMS

**Problem 1:** to implement in CMOS the logic function  $z = f(a, b, c)$ :

<i>a</i>	<i>b</i>	<i>c</i>	<i>z</i>
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

One has the choice between several implementations, i.e. branch-based or with transmission gates

Verify the logic function you have designed (following experience a), then verify the MOS transistor schematic (following experience b).

**Problem 2 :** Design of the 1-bit arithmetic function  $S=A+1$ , while taking into account the carry-in  $C_i$  and generating a carry-out  $C_o$ . The truth table is shown in Figure 1.16, but one has to choose another implementation than the one proposed in Figure 1.16.

Verify the logic function you have designed (following experience a), then verify the MOS transistor schematic (following experience b).

Then try to connect four bits to get a 4-bit incrementer (with logic functions following experience a)) and verify that your incrementer increments correctly.

**Problem 3 :** Design of the MOS schematic of the following logic function:

$$S = (B + C) * (\overline{A} * D + A * \overline{C})$$

It is better to design first the truth table, then to choose the simplification method to design the MOS schematic of this function.

Verify the MOS transistor schematic (following experience b).

**Problem 4 :** Design of the 1-bit logic function that is a comparator of the two inputs  $A_i$  and  $B_i$ , with the following carry signals :

- carry  $C=1$  if  $A > B$  and 0 otherwise
- carry  $D=1$  if  $A < B$  and 0 otherwise

Each 1-bit comparator has two data inputs and two carry-in signals  $C_i$  and  $D_i$ . It generates two carry-out signals  $C_o$  and  $D_o$  that are propagated to the next 1-bit comparator cell. One has to decide if the comparison starts with LSB or MSB bit for a N-bit comparator. One has to search for the 1-bit comparator truth table and then to decide which simplification method to use.

Verify the logic function you have designed (following experience a).

**Problem 5 :** Design of the logic function given by the following specifications : an airline company has to hire some people at the following conditions :

- to be single, male and Swiss
- to be single, Swiss and less than 25 years old
- to be a woman, single and foreigner
- to be a man and less than 25 years old
- to be single and more than 25 years old

Verify the logic function you have designed (following experience a), then verify the MOS transistor schematic (following experience b).

**Problem 6:** Design of the logic function that determines the number of days within a month. The month encoding is performed in the BCD code like the following:

- M4 for the tens
- M3, M2, M1 and M0 for the units

These 5 variables are the inputs of a logic function that produces the number of days within a month: 30 days, 31 days and February.

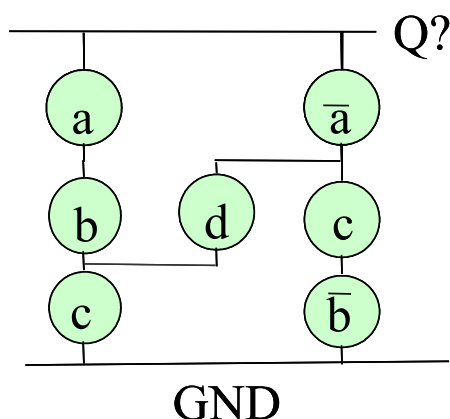
Verify the logic function you have designed (following experience a).

**Problem 7:** In the following Boolean equation, try to guess the required number of transistors, then try to implement it in transistors using various methods.

- Topological dual
- Branch-based
- Transmission gates

$$\text{Equation: } G = A (B + \overline{C}) + BC$$

**Problem 8:** Only the N.ch network of the following CMOS gate is shown. Design the P-ch network with the topological dual method, then give the Boolean equation and try to re-design the gate with less transistors.





**Problem 9: XOR gate**

A XOR gate can be designed using the conventional method using the Karnaugh map. The resulting equation is then:

$$S = \text{not}(A) * B + A * \text{not}(B)$$

This equation can be re-written as:

$$S = (\text{not}(A) + \text{not}(B)) * (A + B) \quad \text{with } D = \text{not}(A + B)$$

Design the MOS schematic of this XOR gate, simulate it, and try to give some advantages of this implementation.

**Problem 10: AND gate**

An AND gate is generally designed as a NAND gate followed by an inverter. Another implementation is to use transmission transistors. Try to design such an implementation:

- AND Karnaugh map
- Simplification with 2 cubes, one of them containing 0 and 1
- MOS schematic
- Simulation
- Try to give some pros and cons of this implementation

**Problem 11: N-MOS versus CMOS**

Starting from the N-MOS circuit below, design the CMOS circuit (using dual topological method or using the Karnaugh map) and simulate both of them. Explain how work the two circuits and give the pros and cons of both implementations.

