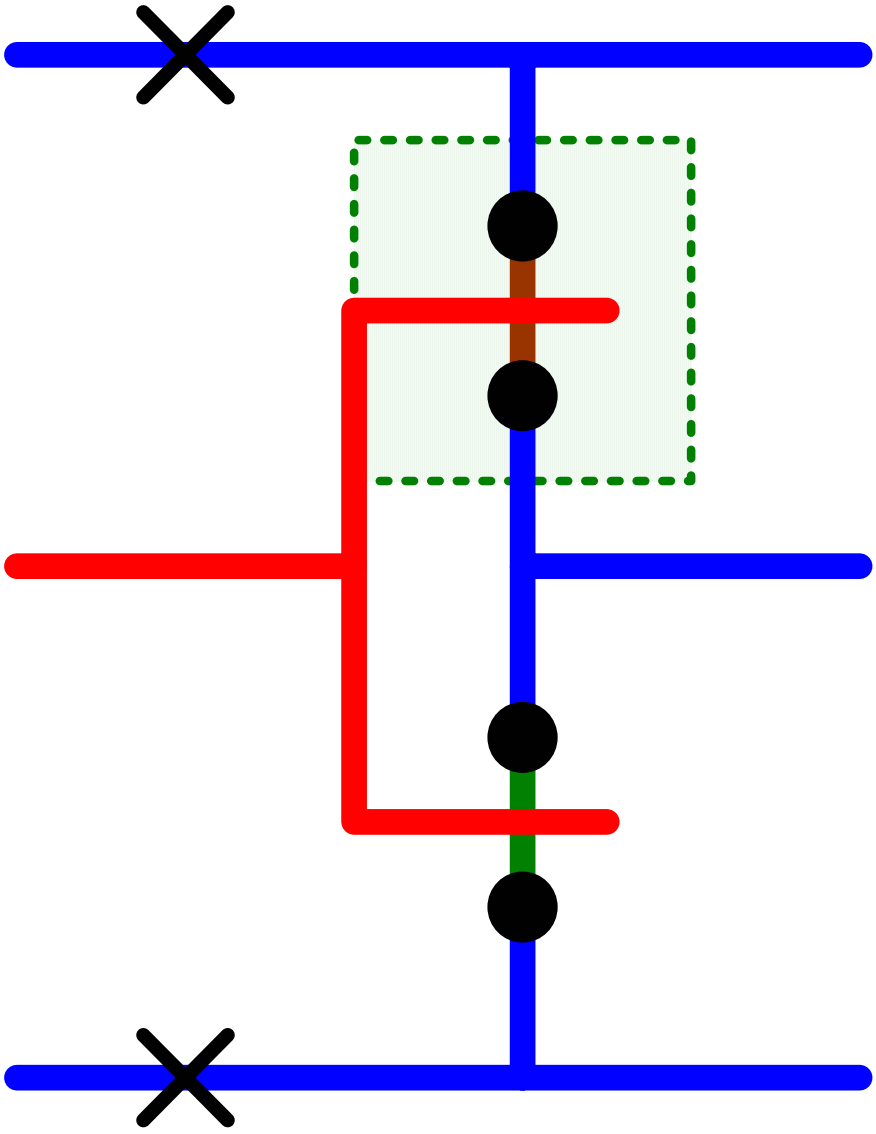
VLSI Coursework

Design and simulation of a 6-to-1 line multiplexer using the MicroWind package.



UNIVERSITY of ULSTER
school of electrical and mechanical engineering

COURSE: BEng (HONS) Electronic Systems
MODULE: EEE515J2 - ASICS & VLSI Design

BY: Colin K McCord

DATE: Thursday, 17 July 2003

TABLE OF CONTENTS

1.0. INTRODUCTION

The propose of this assignment is to use the MicroWind package to design, analyse and prove working a 6-to-1 line multiplexer using pass transistors and then CMOS transmission gates using the ES2 0.7µM 2-metal foundry. The validity of the design will be verified using the design rule checker (DRC) facility and a complete simulation of the operation of the circuit, showing that it operates in accordance with an appropriate Boolean expression for such a multiplexer.

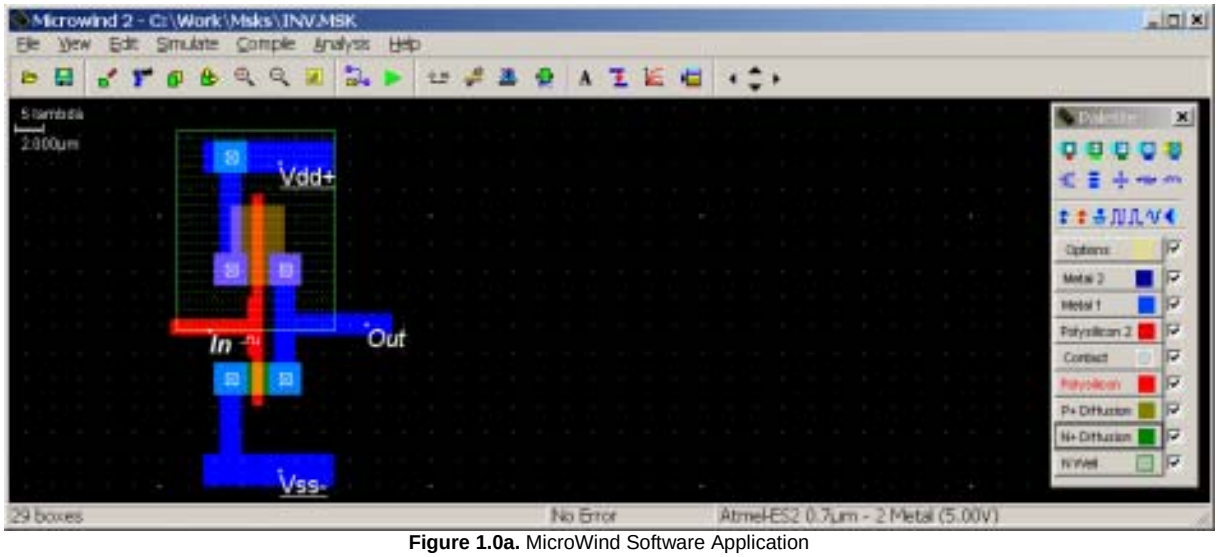


Figure 1.0a. MicroWind Software Application

"The MicroWind2 program allows the student to design and simulate an integrated circuit at physical description level. The package contains a library of common logic and analogue ICs to view and simulate. MicroWind2 includes all the commands for a mask editor as well as original tools never gathered before in a single module (2D and 3D process view, VERILOG compiler, tutorial on MOS devices). The student can gain access to circuit simulation by pressing one single key. The electric extraction of a circuit is automatically performed and the analogue simulator produces voltage and current curves immediately." [J1]

This report includes circuit / stick diagrams, screen dumps of MicroWind designs and other design details, as well as an English description on how the design works. The simulation results are included and explained by annotating the waveforms produced by the system. This report also includes some useful background information, related to VLSI and the design of the multiplexer.

The report is split into 8 sections for clarity (including this introduction): -

1. Introduction.
2. Background Information.
3. Multiplexers.
4. Design: 6-to-1 line multiplexer.
5. Simulation: 6-to-1 line multiplexer.
6. Conclusions.
7. References.
8. Appendixes.

In the conclusions the different implementations are compared and contrasted, taking account of different features and aspects of layout and performance. Reference has been made to extracted parameters and simulation performance data, using the MicroWind package facilities, within this comparison. It is also indicated what aspects of the design would be changed if the exercise was repeated from scratch.

The CD-ROM attached to appendix 5, contains the MicroWind mask files used in this report.

2.0. BACKGROUND INFORMATION

2.1. Boolean Algebra Laws and Rules

Boolean algebra uses many of the same laws as those of ordinary algebra. The OR function (X = A + B) is Boolean addition, and the AND function (X = AB) is Boolean multiplication. The following three laws are the same for Boolean algebra as they are for ordinary algebra: -

1. **Commutative law of addition and multiplication**: A + B = B + A, and AB = BA. These laws mean that the order of ORing and ANDing does not matter.

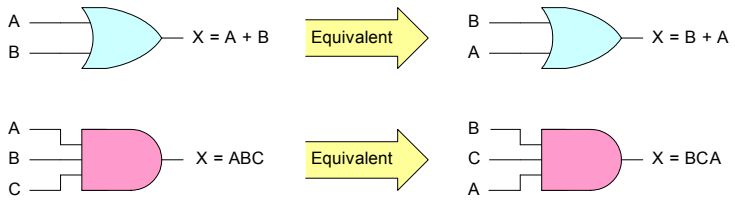


Figure 2.1a. Example of commutative law

2. **Associative law of addition and multiplication:** A + (B + C) = (A + B) + C, and A(BC) = (AB)C. These laws mean that the grouping of several variables ORed or ANDed together does not matter.

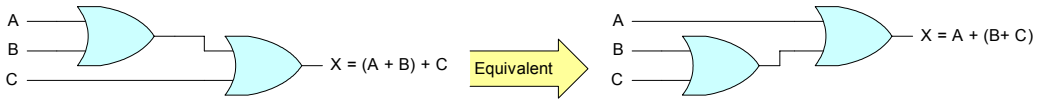


Figure 2.1b. Example of associative law

3. **Distributive law:** A(B + C) = AB + AC, and (A + B)(C + D) = AC + AD + BC +BD. This law shows methods for expanding an equation containing ORs and ANDs.

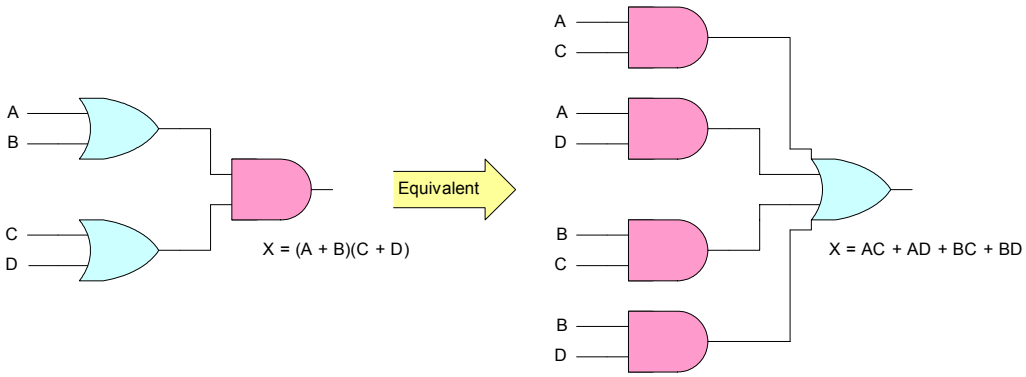


Figure 2.1c. Example of distributive law

These three laws hold true for any number of variables. For example, the commutative law can be applied to X = A + BC + D to form the equivalent equation X = BC + A + D. Besides the three basic laws, there are several rules concerning Boolean algebra, these rules allow for variables to be combined or eliminated forming simper equivalent circuits.

Rule 1: A.0 = 0
Rule 2: A.1 = A
Rule 3: A + 0 = A
Rule 4: A + 1 = 1
Rule 5: A.A = A
Rule 6: A+A = A
Rule 7: $A.\overline{A} = 0$
Rule 8: $A + \overline{A} = 1$
Rule 9: $\overline{\overline{A}} = A$
Rule 10: $A + \overline{A}B = A + B$
Rule 11: $\overline{A} + AB = \overline{A} + B$

2.2. Boolean Algebra Simplification

Often in the design and development of digital systems, a designer will start with simple logic rate requirements but add more and more complex gating, making the final design a complex combination of several gates, some having the same inputs.

At that point the designer must step back and review the combinational logic circuit that has been developed and see if there are ways of reducing the number of gates without changing the function of the circuit. If an equivalent circuit can be formed with fewer gates or fewer inputs, the cost of the circuit is reduced and its reliability is improved.

This process is called reduction or simplification of combinational logic circuits and is performed by using the laws and rules of Boolean algebra stated in the preceding section (2.1. Boolean algebra and rules).

The following example illustrates the use of Boolean algebra and presents some techniques for the simplification of logic circuits: -

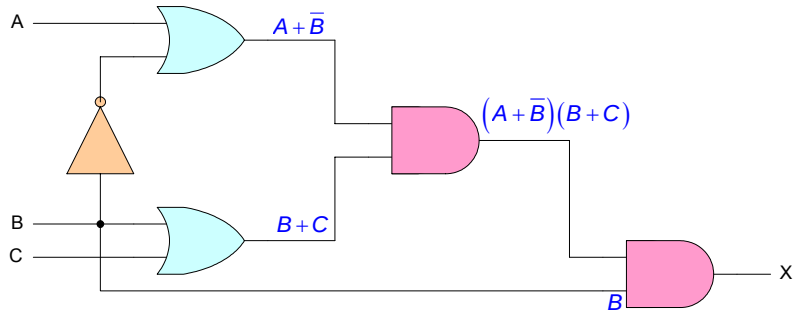


Figure 2.2a. Example logic circuit to be simplified.

The Boolean equation for *X* is: -

$$X = \left[\left(A+\overline{B}\right)(B+C)\right]B \qquad \{2.2.1\}$$

To simplify, first apply law 3: -

$$X = \left(AB + AC + \overline{B}B + \overline{B}C\right)B \qquad \{2.2.2\}$$

The $\overline{B}B$ term can be eliminated using rule 7 and then rule 3: -

$$X = \left(AB + AC + \overline{B}C\right)B \qquad \{2.2.3\}$$

Apply law 3 again: -

$$X = ABB + ACB + \overline{B}CB \qquad \{2.2.4\}$$

Apply law 1: -

$$X = ABB + ABC + \overline{B}BC \qquad \{2.2.5\}$$

Apply rules 5 and 7: -

$$X = AB + ABC + 0.C \qquad \{2.2.6\}$$

Apply Rule 1:

$$X = AB + ABC \qquad \{2.2.7\}$$

Factor an *AB* from both terms: -

$$X = AB(1 + C) \qquad \{2.2.8\}$$

Apply Rule 4 and then Rule 2: -

$$X = AB \leftarrow \text{Simplified equation} \qquad \{2.2.9\}$$

2.3. De Morgan's Theorem

To simplify circuits containing NANDs and NORs, a theorem developed by the mathematician Augustus De Morgan can be used. This theorem allows an expression having an inversion bar over two or more variables to be converted into an expression having inversion bars over single variables only.

In the form of an equation, De Morgan's theorem is stated as follows: -

$$\overline{A.B} = \overline{A} + \overline{B} \qquad \{2.3.1\} \qquad \overline{A + B} = \overline{A}.\overline{B} \qquad \{2.3.2\}$$

For three or more variables,

$$\overline{A.B.C} = \overline{A} + \overline{B} + \overline{C} \qquad \{2.3.3\} \qquad \overline{A + B + C} = \overline{A}.\overline{B}.\overline{C} \qquad \{2.3.4\}$$

Basically De Morgan's theorem is to break the bar over the variables and either change the AND to an OR, or change the OR to an AND.

To prove that this works let's apply the theorem to a NAND gate and then compare the truth table of the equivalent circuit to that of the original NAND gate (see figure 2.3a).

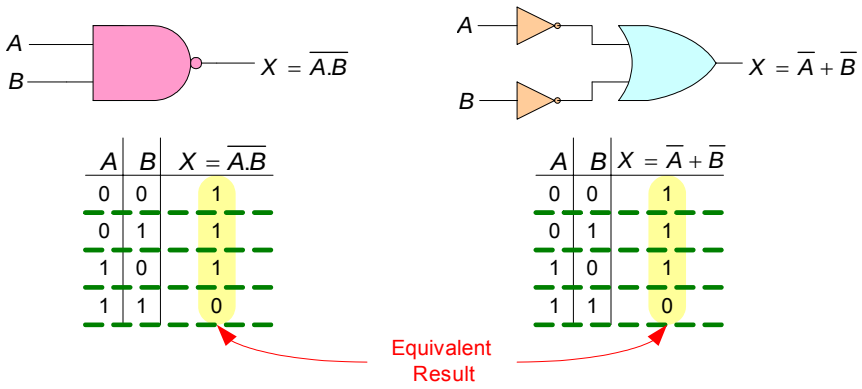


Figure 2.3a. De Morgan's theorem applied to NAND gate produce 2 identical truth tables.

2.4. Brief Summary of the History of the IC

1945: Transistor invented.
1950: The grown junction transistor.
1952: Single crystal silicon is fabricated.
1954: First commercial silicon transistor.
1954: Oxide masking process developed.
1954: First transistor radios.
1955: First field effect transistor.
1972: Intel 8008.
1973: Commercial BiCOMS ICs.
1973: Projection Printer Invented.
1974: First 4Kbit DRAM with 1T cell.
1974: Intel 8080
1976: 16Kbit DRAM introduced.
1978: Intel 8086/8088.

1958: Integrate circuit invented.
1959: Planar technology invented.
1960: Epitaxial deposition developed.
1960: First MOSFET fabricated.
1961: First commercial ICs.
1962: Transistor-transistor logic invented.
1963: First MOS IC.
1963: CMOS invented.
1964: First commercial contact printer.
1965: Moore's law.
1965: 100-bit shift register.
1966: 16-bit bipolar memory.
1966: First bipolar logic.
1966: Single transistor DRAM cell invented.
1967: NMOS disclosed.
1968: 64-bit bipolar array chip.
1969: BiCMOS invented.
1970: 1st NMOS IC.
1970: First commercial DRAM – 1Kbits.
1971: UVEPROM invented.
1971: Microprocessor invented.
1972: Digital signal processor invented.
1972: MOSFET Scaling.
1978: Step and repeat system invented.
1979: 64Kbit DRAM introduced.
1980: Modern DSP commercial available.
1980: IBM selects Intel 8088 for the PC.
1982: 256Kbit DRAM.
1982: Intel 80286.
1983: 1st CMOS DRAM.
1983: EEPROM invented.
1984: Flash memory invented.
1985: Commercial flash memory introduced.
1985: Intel 80386DX.
1986: ETOX style flash introduced.
1986: 1Mbit DRAM.
1988: 4Mbit DRAM.
1989: Intel 80486DXTM.
1991: 16Mbit DRAM.
1993: Intel Pentium
1994: 64Mbit DRAM.
1995: Intel Pentium Pro.
1997: Intel Pentium II.
1998: 256Mbit DRAM.
1999: Intel Pentium III.
2000: Intel Pentium IV.

For more detailed information on each event see [W1], the source of this material.

2.5. Introduction to Stick Diagrams

"Stick diagrams are useful for planning the layout and routing of integrated circuits. In a stick diagram, every line of a conducting material layer is represented by a line of a distinct colour. The width of a line is not important, as stick diagrams given only wiring and routing information." [B2]

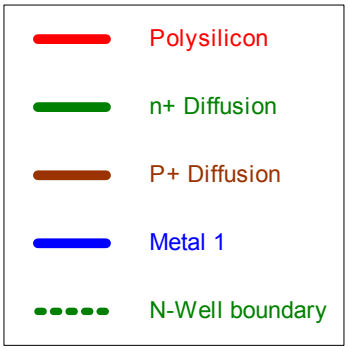


Figure 2.5a. Reduced Stick Diagram Key

Basically a VLSI circuit is a 3-dimensional set of patterned material layers. Stick diagrams provide a top view of the patterns. The colours are used to trace signal flow paths through the conducting layers in a complex integrated circuit.

A stick diagram is basically a schematic representation of a circuit at the physical design level.

Red (Polysilicon) crossed over Green (N+ Diffusion) represents an NMOS transistor (see figure 2.5b), while Red (Polysilicon) crossed over Brown (P+ Diffusion) represents an PMOS transistor (see figure 2.5c). Blue (Metal 1) may cross over green (N+ Diffusion), brown (P+ Diffusion) or red (Polysilicon) without a connection.

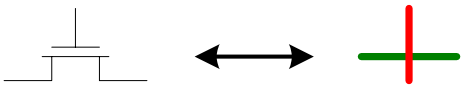


Figure 2.5b. Stick representation of an NMOS transistor

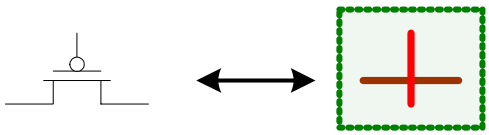


Figure 2.5c. Stick representation of an PMOS transistor

Connections between layers are specified by a via (represent by a black circle). Metal lines on different layers can cross one another, contacting these metal lines requires a via.

Stick diagrams can be drawn using a set of coloured pencils to aid in wiring of basic gates, or in routing interconnect lines on the chip. A VLSI layout can become complicated due to the large number of wires that need to be included. Since stick diagrams are easy to draw, they can be used to plan the wiring before accessing a CAD tool. Many hours of frustration can be avoided by planning ahead.

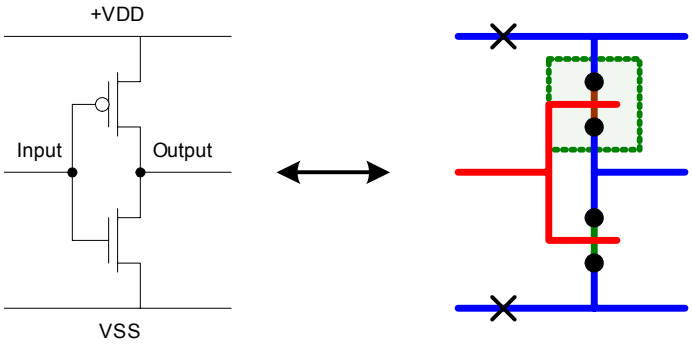


Figure 2.5d. Stick Diagram representation of a simple CMOS inverter

"Stick diagrams provide an easy approach to performing simple CMOS circuit layouts. Planning a physical design using stick diagrams before going to a CAD tool can save a lot of time and energy. " [B2]

2.6. Design Rules

"Design processes are aided by simple concepts such as stick and symbolic diagrams but the key element is a set of design rules. Design rules are the communication link between the designer specifying requirements and the fabricator who materialises them. Design rules are used to produce workable mask layouts from which the various layers in silicon will be formed or patterned." [B3]

The design rules used in this assignment (es207, see appendix 2) make use of a 'lambda-based' rule set. *"These rules are straightforward and relatively simple to apply. However, they are 'real' and chips can be fabricated from mask layouts using the lambda-based rule set. Tighter and faster designs will be realised if a fabricator's line is used to its full advantage and such rule generally particular not only to the fabricator but also to a specific technology"* [B3]

"Design rules are a set of geometrical specifications that dictate the design of the layout masks. A design rule set provides numerical values for minimum dimensions, line spacing, and other geometrical quantities that are derived from the limits of a specific processing line. The design rules must be followed to insure functional structures on the fabricated chip." [B2]

From more detailed information on design rules see reference [B3] pages 77 to 84 or/and [B2] pages 140 to 146.

2.7. Sheet Resistance

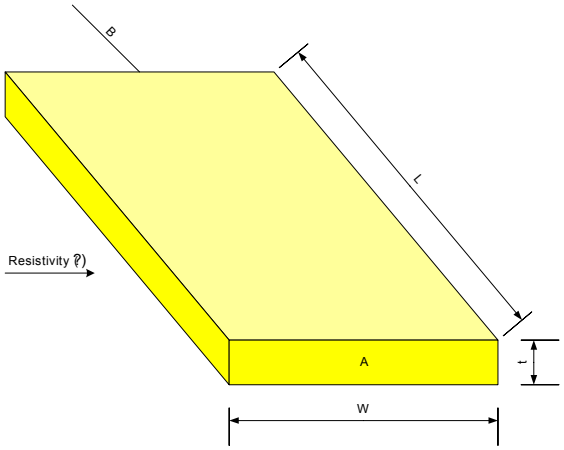


Figure 2.7a. Sheet resistance model

Cosider a uniform slab of conducting material of resistivity (ρ), of with (W), thickness (t), and length between face (L). Consider the resistance R_{AB} between the two opposite faces.

$$R_{AB} = \frac{\rho L}{A} \ \Omega , \qquad \{2.7.1\}$$

A = cross-section area, Thus

$$R_{AB} = \frac{\rho L}{tW} \ \Omega \qquad \{2.7.2\}$$

Now, consider the case in which L = W,

$$R_{AB} = \frac{\rho}{t} = R_S \qquad \{2.7.3\}$$

Where

R_S = ohms per square or sheet resistance.

"Note that R_S is completely independent of the area of the square; for example a 1µm per side square slab of material has exactly the same resistance as a 1cm per side square slab of the same material if the thickness is the same." [B3]

Typical sheet resistances R_S can be found in reference [B3] page 96, the source of this material.

2.8. Area Capacitances of Layers

Conducting layers are separated from the substrate and each other by insulating (dielectric) layers, and thus parallel plate capacitive effects must be present. For any layer (assuming that the dielectric thickness is known) the area capacitance can be calculated as follows: -

$$C = \frac{\varepsilon_0 \varepsilon_{ins} A}{D} \quad \text{F} \qquad \{2.8.1\}$$

A = area of plates, D = thickness of silicon dioxide (dielectric), ε_{ins} = relative permittivity of SiO_2 and ε_0 = permittivity of free space (8.85 x 10^{-14} F/cm). Normally layer area capacitances are expressed in pF/µm^2: -

$$C\left(\frac{\text{pF}}{\mu\text{m}^2}\right) = \frac{\varepsilon_0 \varepsilon_{ins}}{D} \frac{\text{F}}{\text{cm}^2} \times \frac{10^{12}\text{pF}}{\text{F}} \times \frac{\text{cm}^2}{10^8 \mu\text{m}^2} \qquad \{2.8.2\}$$

Typical values of area capacitance can be found in reference [B3] page 99, the source of this material.

2.9. The delay Unit

The time constant τ is directly link to capacitance and resistance, and effects the rise and fall times of logic signals. The VLSI design engineer must try and keep resistance and capacitance to a minimum, in order to reduce the time constant τ, hence reducing rise and fall times.

$\tau = RC$ {2.9.1} R = R_S (Sheet resistance) and C = C_g (Gate capacitance unit)

2.10. Choice of Layers

Frequently, in designing an arrangement to meet given specifications, there are several possible ways in which the requirements may be met, including the choice between the layers on which to route certain data and control signals.

- V_{DD} and V_{SS} should be distributed on metal layers to keep R_S low.
- Long length of polysilicon should be avoided because of the high R_S.
- Transistor resistance is normally higher than wiring resistance; hence there is no real danger of voltage divider effects between wiring and transistor resistances.
- Capacitive effects must also be considered. Signal line should have low values of R_S.

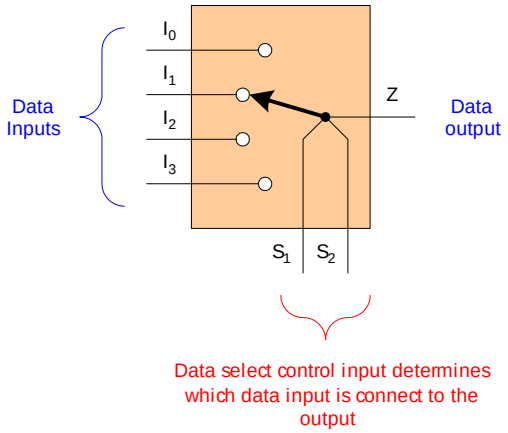
Layer	R	C	Comments
Metal	Low	Low	Good current capability without large voltage drop.
Polysilicon	High	Moderate	RC product is moderate; high IR drop.
Diffusion	Moderate	High	Moderate IR drop bit high C.

Figure 2.10a. Choice of layers, source [B3]

Note [B3] recommends that the maximum length (communication wire) of polysilicon should be 200λ and the maximum length of diffusion should be 20λ (taking account of peripheral and area capacitances).

3.0. MULTIPLEXERS

A multiplexer is a device capable of funnelling several data lines into a single line for transmission to another point. The multiplexer has two or more digital input signals connected to its input. Control signals are also input to tell which data-input line to select for transmission (data selection). Figure 3.0a illustrates the function of a multiplexer.



"Multiplexers are widely used and have many applications. They are also commonly available in a number of standard configurations in TTL and other logic families." [B3]

The multiplexer is also known as a data selector. Figure 3.0a shows that the data select control Inputs (S_1,S_0) are responsible for determining which data input (I_0 to I_3) is selected to be transmitted to the data-output line (Z). The S_1, S_0 inputs will be a binary code that corresponds to the selected data-input line. Figure 3.0b lists the truth table for input data selection.

$$Z = I_O.\overline{S_1}.\overline{S_0} + I_1.\overline{S_1}.S_0 + I_2.S_1.\overline{S_0} + I_3.S_1.S_0 \qquad \{3.0.1\}$$

Figure 3.0a. Functional diagram of a four-line multiplexer

S_1	S_0	Z
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Figure 3.0b. Truth table for a 4-to-1 line multiplexer

3.1. 4-to-1 Line Multiplexer using SSI Logic Gates

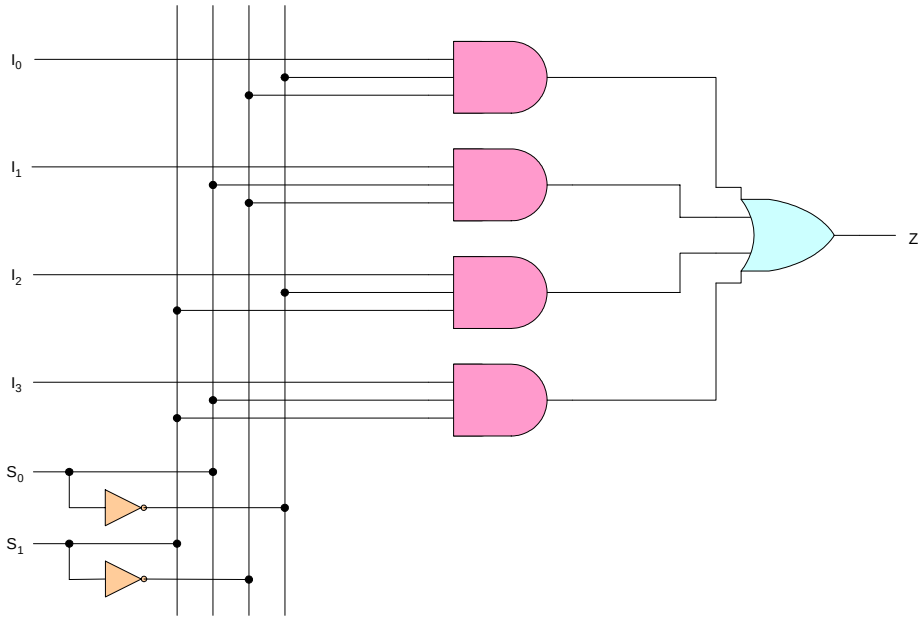


Figure 3.1a. Logic diagram for a four-line multiplexer

A four-line multiplexer built from SSI logic gates is shown in figure 3.1a. The control inputs (S_1,S_0) take care of enabling the correct AND gate to pass just one of the data inputs through to the output. It is possible to design the 4-to-1 multiplexer as specified in figure 3.1a at the integrated circuit physical level, but this solution is not optimal. For example figure 3.1b shows the mask file (MicroWind design file) of a 3 input AND gate (four required), notice that there are 4 NMOS and 4 PMOS transistors per gate. Figure 3.1c shows the mask file (MicroWind design file) of a 4 input OR gate, notice that there are 5 NMOS and 5 PMOS transistors per gate. Figure 3.1d shows the mask file (MicroWind design file) for a standard CMOS inverter which has 1 NMOS and 1 PMOS transistor.

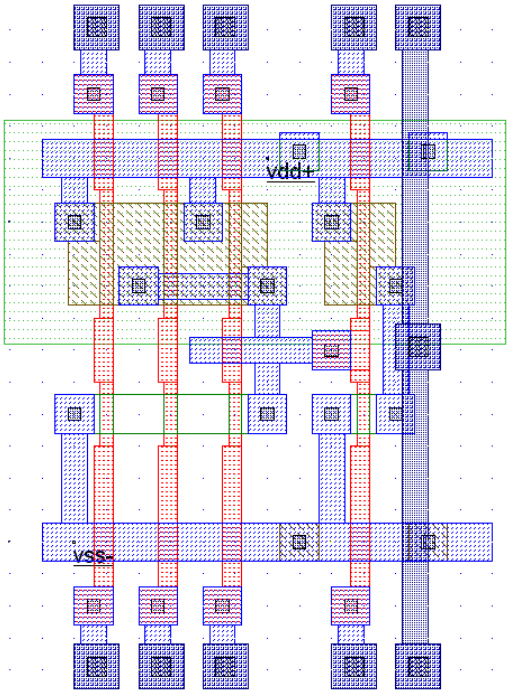


Figure 3.1b. 3 input AND gate.

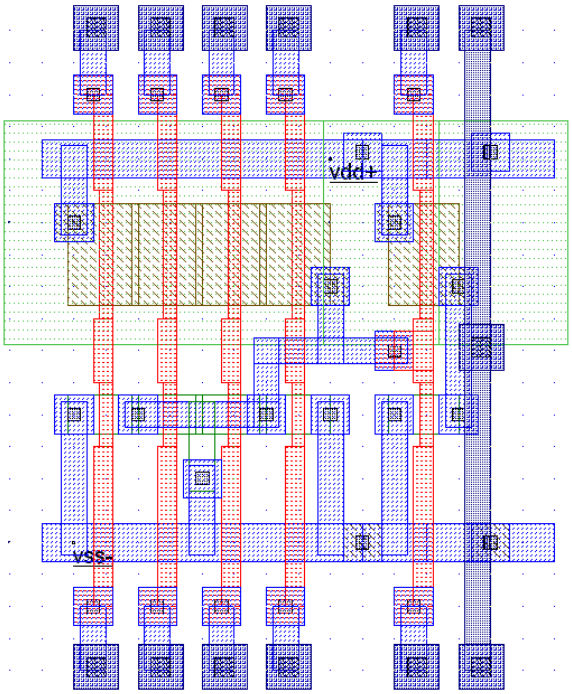


Figure 3.1c. 4 input OR gate.

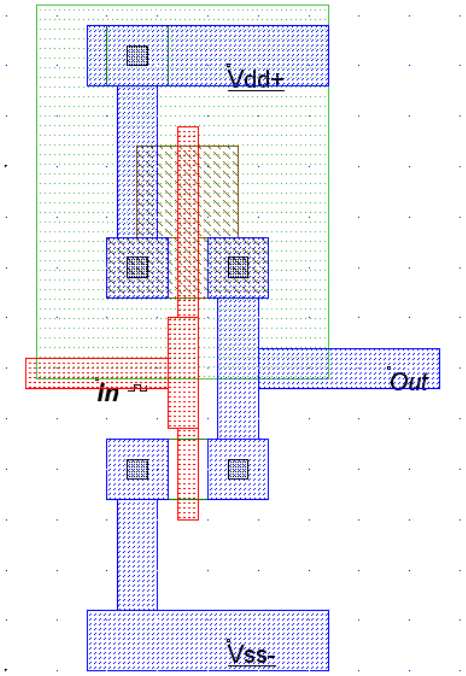


Figure 3.1d. Inverter

From Figure 3.1e it is clear that 50 transistors are required if the 4-to-1 multiplexer was designed using SSI logic gates, clearly there must be a more efficient method. Yes there is, the multiplexer can be designed using NMOS or PMOS pass transistors (requires only 12 transistors).

Logic Gate	No. Off	NMOS	PMOS
3 input AND	4	16	16
4 input OR	1	5	5
Inverter	2	4	4
Total		25	25
Total No. of Trans.		50	

Figure 3.1e. Number of transistors required

Notice that the output of both figure 3.1b and figure 3.1c appears to go through an inverter, hence an NAND / NOR gate is easier to create than an AND / OR gate, hence using De Morgan's theorem it is possible to redesign the 4-1 line multiplexer using only NAND / NOR gates. Reducing the total number of transistors, but it will still not be as efficient as the pass transistors solution.

3.2. 4-to-1 Line Multiplexer using NMOS Pass Transistors

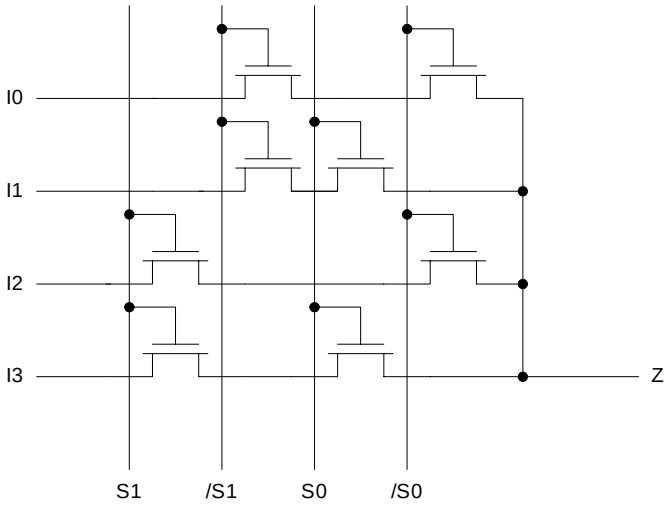


Figure 3.2a. Circuit diagram of a 4-to-1 line multiplexer using PMOS pass transistors.

A four-line multiplexer using NMOS pass transistors is shown in figure 3.2a. Basically there are two series transistors per data line, that's a total of 8 transistors (not including the 4 additional transistors required for the inverter circuits). Look closely at line I_0 notice that the two series transistors for that line are connected to $/S_1$ and $/S_0$, hence both transistors will switch ON and connect line I_0 to Z when $S_1 = 0$ and $S_0 = 0$.

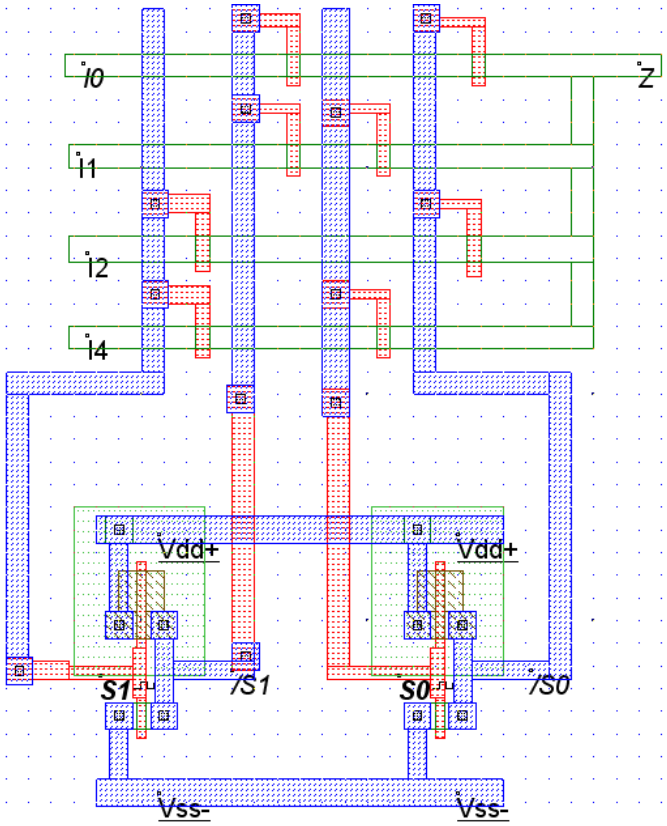


Figure 3.2b. 4-to-1 line multiplexer using NMOS pass transistors.

Figure 3.2b shows the mask file for the 4-to-1 line multiplexer (NMOS pass) clearly there are only 12 transistors and the circuit is much simpler than that the SSI logic design. But there is a problem with this design as NMOS transistors cannot pass a good logic '1', the circuit could be redesigned using PMOS pass

transistors, but PMOS transistors cannot pass a good logic '0'. Therefore the optimal solution is to use both PMOS and NMOS transistors in a transmission gate arrangement requiring a total of 20 transistors (10 NMOS, 10 PMOS, including inverter circuits), which is still more efficient than the SSI logic design.

Note the surface area of the design can be reduced by placing the NMOS pass transistors below the metal select lines. *"This practice is acceptable in this situation where a transistor gate is actually driven from and connected to the particular metal line which runs across it. This method of economising in area must be used with caution when locating transistors under metal layers to which they are not connected, and may not be acceptable when the underlying transistors are used as storage points to hold a charge and retain a logic level."* [B3]

3.3. 4-to-1 Line Multiplexer using a CMOS Transmission Process

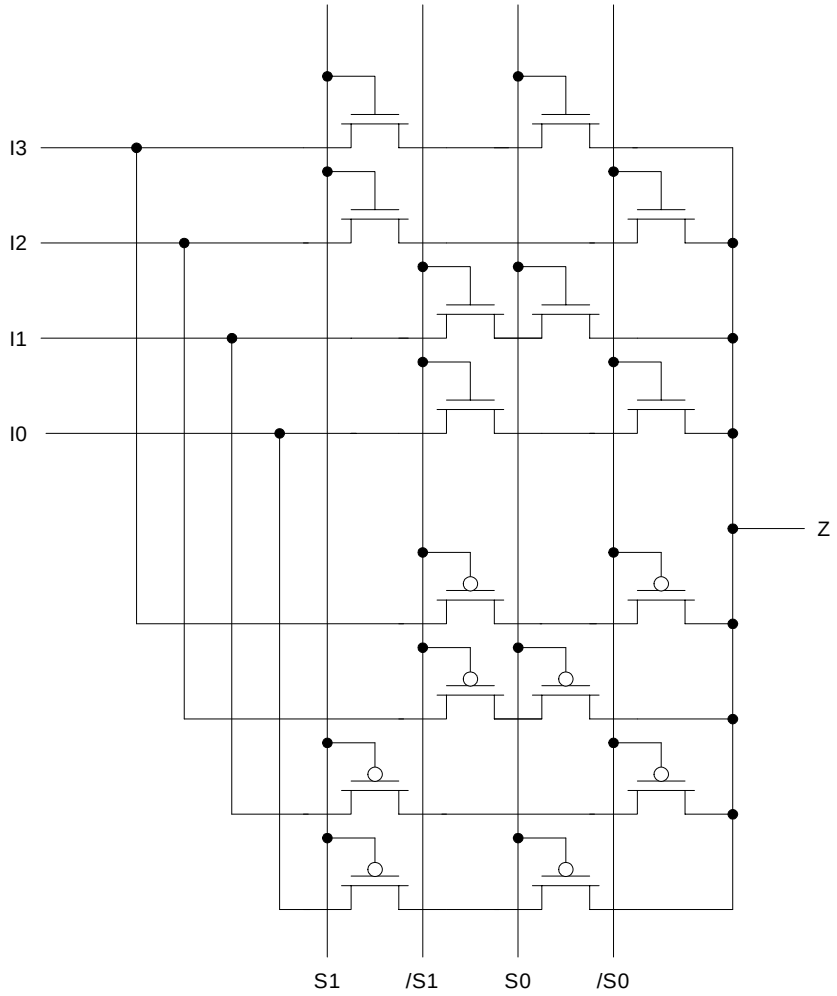


Figure 3.3a. Circuit diagram of a 4-to-1 line multiplexer using a CMOS transmission process.

A four-line multiplexer using a CMOS transmission process is shown in figure 3.3a. Basically there are two series NMOS transistors in parallel with two series PMOS transistors for every data line, that's a total of 16 transistors (not including the 4 additional transistors required for the inverter circuits).

Look closely at line I_0, notice that the two series NMOS transistors are connected to $/S_1$ and $/S_0$ and the two series PMOS transistors are connected to S_1 and S_0, hence all four transistors are ON when $S_1 = 0$ and $S_0 = 0$. The NMOS transistors will produce good logic '0's and the PMOS transistors will produce good logic '1's.

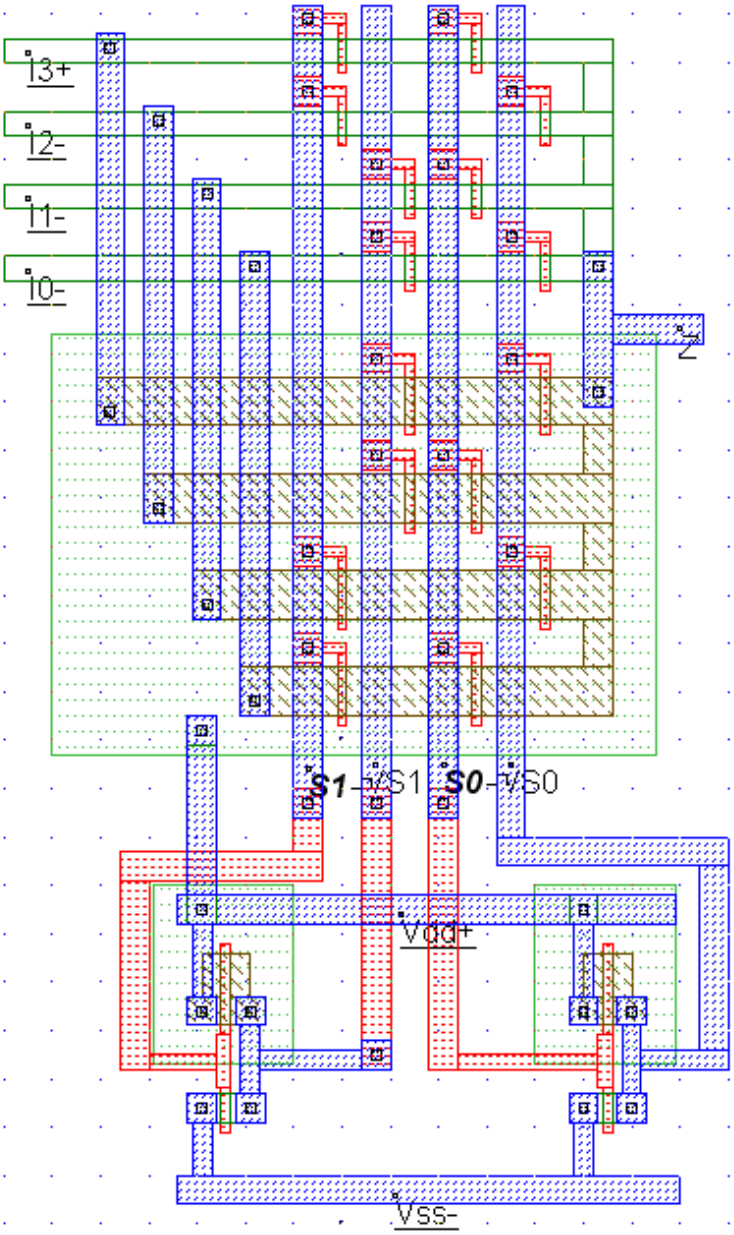


Figure 3.3b. 4-to-1 line multiplexer using a CMOS transmission process.

Figure 3.3b shows the mask file for the 4-to-1 line multiplexer (CMOS transmission process) clearly there are 20 transistors and the circuit is reasonably easy to follow. Notice that all of the NMOS and PMOS transistors are grouped together (this is good design practice), and one large N well is surrounding all of the PMOS transistors.

4.0. DESIGN: 6-TO-1 LINE MULTIPLEXER

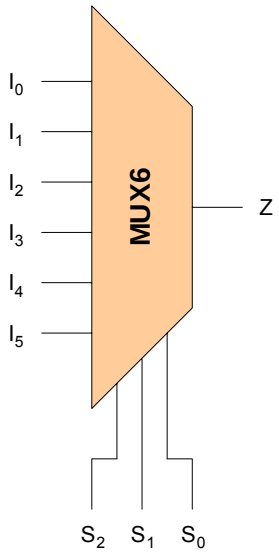


Figure 4.0a. Mux6 Symbol

The Boolean equation for a 6-to-1 line multiplexer is shown below (4.0.1), figure 4.0a shows the symbol and figure 4.0b shows the truth table. Normally the number of data lines a multiplexer has is a power of 2 (e.g. 2, 4, 8, 16, etc…), not in this case as 6 is not a power of 2 (non-standard mux). Therefore there will be two unused data line addresses (111 & 110).

$$Z = I_O.\overline{S_2}.\overline{S_1}.\overline{S_0} + I_1.\overline{S_2}.\overline{S_1}.S_0 + I_2.\overline{S_2}.S_1.\overline{S_0} + I_3.\overline{S_2}.S_1.S_0 + I_4.S_2.\overline{S_1}.\overline{S_0} + I_5.S_2.\overline{S_1}.S_0 \quad \{4.0.1\}$$

S_2	S_1	S_0	Z
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	NOT USED
1	1	1	NOT USED

Figure 4.0b. Truth table for 6-to-1 line multiplexer

These two unused data line addresses may cause problems. The smart design maybe to design an 8-to-1 line multiplexer and only use 6 of the 8 data lines. The problem with this is that more surface area is used, and if surface area is important the 6-to-1 solution would produce a more compact layout and maybe more suitable. But if the unused data line addresses are selected, the output Z is unpredictable, hence the designer making use of this chip must make sure not to use these unused data line addresses.

It has been decided to design a 6-to-1 line multiplexer with 2 unused data line addresses and not the 8-to-1. It is important to remember that S_2=1, S_1=1, S_0=1 and S_2=1, S_1=1, S_0=0 (No data line selected) will produce unpredictable outputs and should never be used, allow it makes the simulation more interesting.

4.1. Using NMOS Pass Transistors

From figure 4.1a it is clear that 3 series (e.g. AND gate) NMOS transistors are required for each data line, also notice that inverted inputs are required, hence the need for 3 CMOS invertors. 18 NMOS transistors are required for data selection, 3 NMOS and 3 PMOS transistors are required for the invertors; that's a total of 24 transistors (21 NMOS, 3 PMOS). Note the Boolean expression has not been simplified; it may be possible to simplify (using K Maps, De Morgan's, etc...) reducing the total number of transistors.

S_2	S_1	S_0	Z	NMOS Pass Transistors
0	0	0	I_0	$\overline{S_2}.\overline{S_1}.\overline{S_0}$
0	0	1	I_1	$\overline{S_2}.\overline{S_1}.S_0$
0	1	0	I_2	$\overline{S_2}.S_1.\overline{S_0}$
0	1	1	I_3	$\overline{S_2}.S_1.S_0$
1	0	0	I_4	$S_2.\overline{S_1}.\overline{S_0}$
1	0	1	I_5	$S_2.\overline{S_1}.S_0$

Figure 4.1a. Table specifying the NMOS transistors required for each data line.

Note: The unused data line addresses have been completely ignored. Theoretically if these addresses are used no data line is connected to the Z output, but their maybe some sort of output due to capacitive effects.

Using Figure 4.1a, the circuit diagram (figure 4.1b) can easily be sketched. For example look closely at line I_0 were the three series transistors for that line are connected to $/S_2./S_1./S_0$ as specified in figure 4.1a, hence line I_0 is connected to Z when S_2=0,S_1=0 & S_0=0. Note the additional inverter circuits are not shown in figure 4.1b.

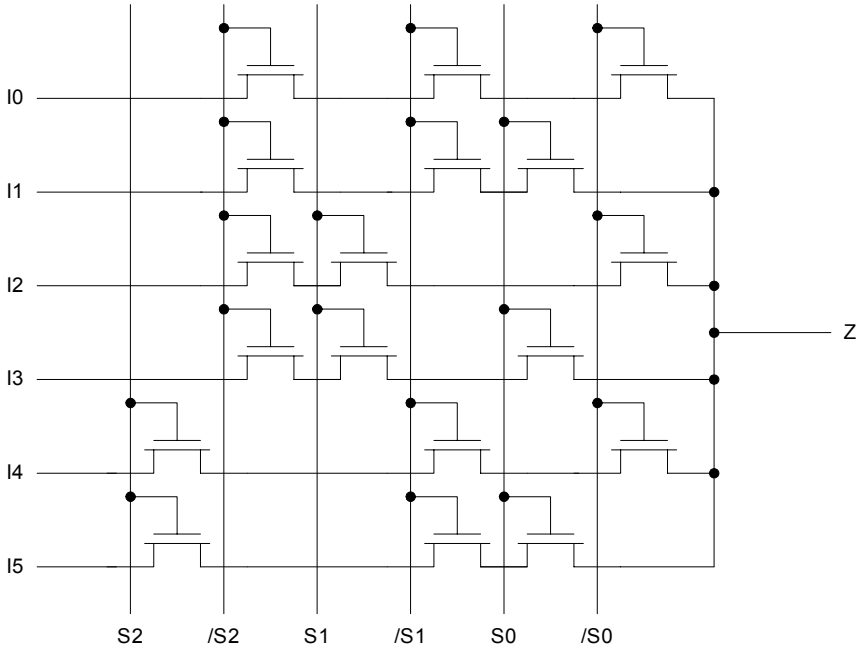


Figure 4.1b. NMOS circuit diagram.

Once the circuit diagram has been completed, the stick diagram can easily be drawn (figure 4.1c). Notice that the stick diagram (figure 4.1c) can be directly compared with the circuit diagram (figure 4.1b) as the layout of both is identical.

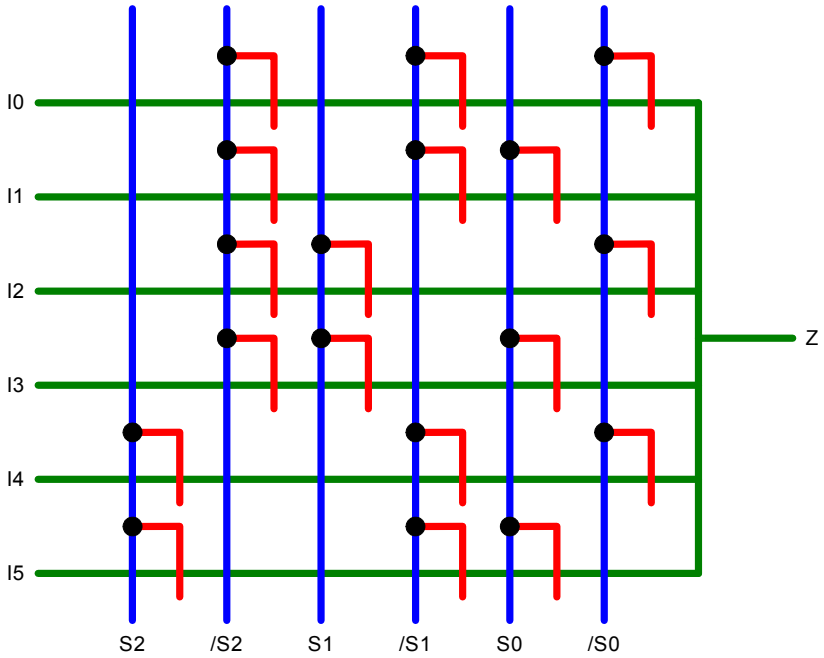


Figure 4.1c. NMOS stick diagram.

Once the stick diagram (figure 4.1c) was completed, the design was drawn in MicroWind (figure 4.1d) which has been verified by the design rule checker (foundry es207.rul selected). The mask file (MicroWind design file, see figure 4.1d) was designed for simplicity (easy to understand and describe) that can be directly compared with the stick diagram (figure 4.1c) and the circuit diagram (figure 4.1b). This means that the design may not be optimised for performance and compactness, allow time was taken making sure that transistor spacing was at a minimum; e.g. as close as the design checker would allow, without rueing the simplicity of the design.

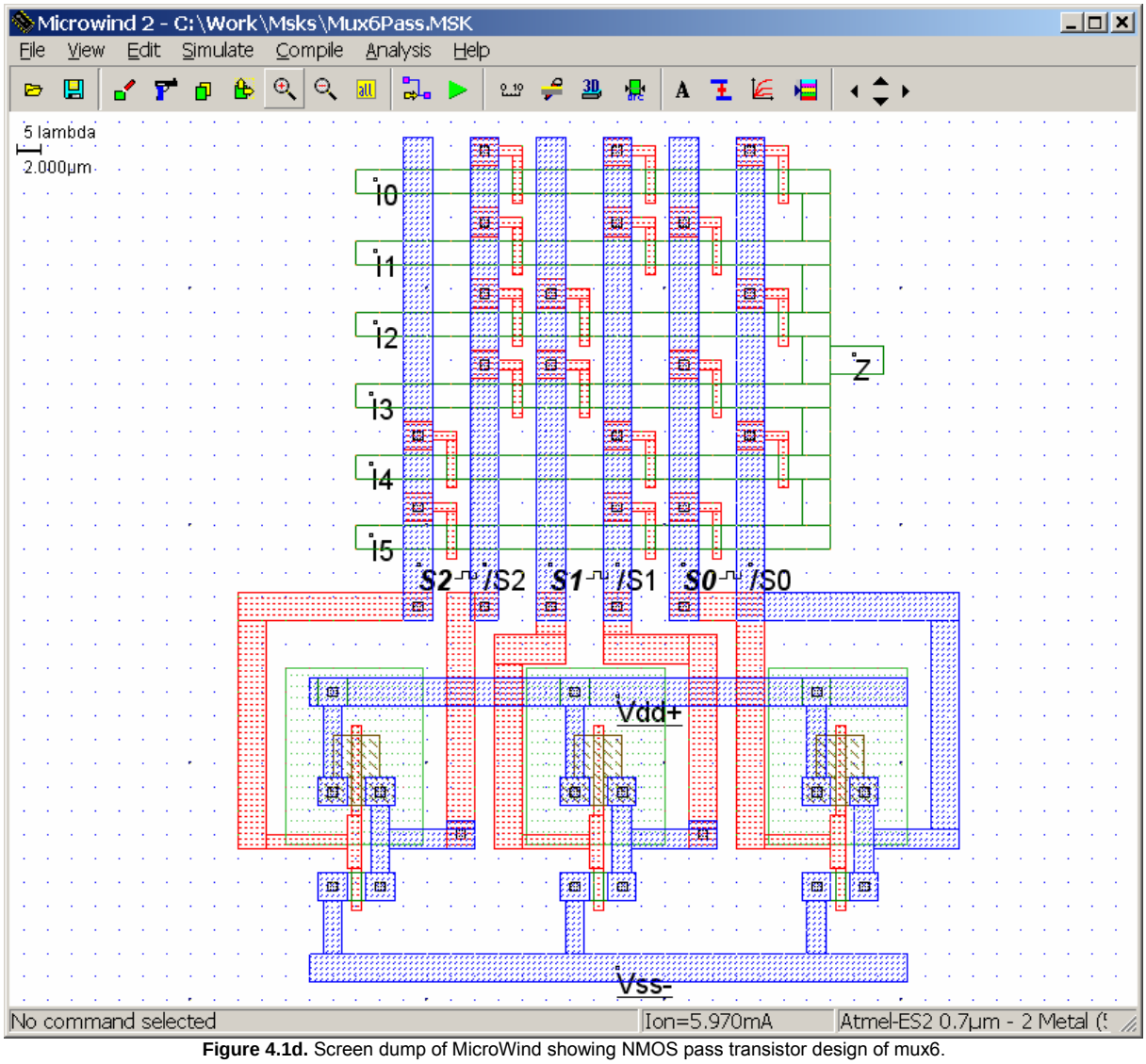


Figure 4.1d. Screen dump of MicroWind showing NMOS pass transistor design of mux6.

The design is extremely simple; basically there are three series NMOS transistors along each data line (e.g. all three transistors need to be on before the line is connected to Z). The three standard CMOS invertors, are used to invert the select lines, these select lines (6 including inverted lines) are connected to metal 1 which run vertically (parallel to each other) across N+ diffusion (data lines). A via (poly to metal 1) is used to connect the gate of each NMOS transistor to the appropriate select line as specified in figure 4.1a. The polysilicon crosses N+ diffusion (NMOS transistor); breaking current flow along the N+ diffusion (data line) until the transistor is switched ON.

Figures 4.1e to 4.1j shows the 2D view of the process for all of the data lines (I_0 to I_5). *"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 fabrication principles. A click of the mouse at the first point and the release of the mouse at the second point give the cross-section."* [J1]

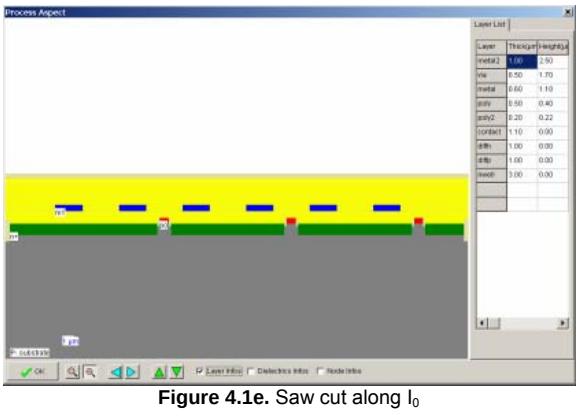


Figure 4.1e. Saw cut along I_0

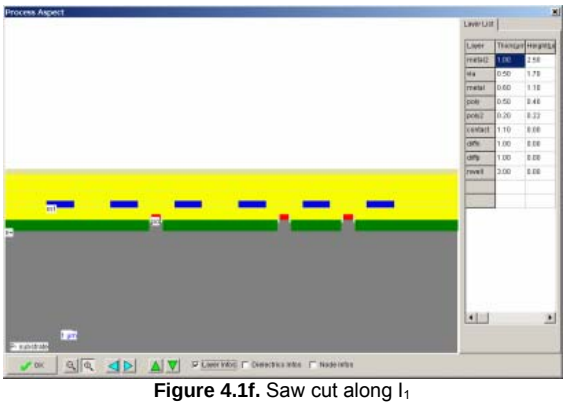


Figure 4.1f. Saw cut along I_1

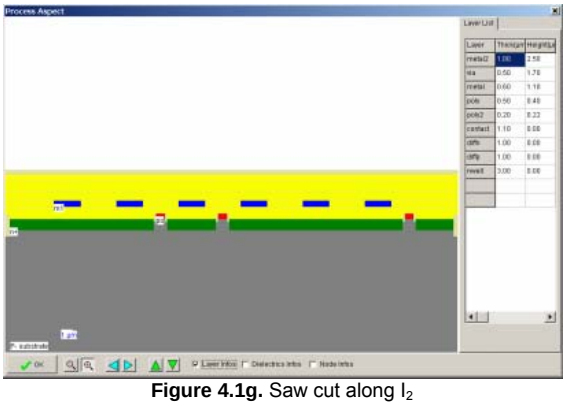


Figure 4.1g. Saw cut along I_2

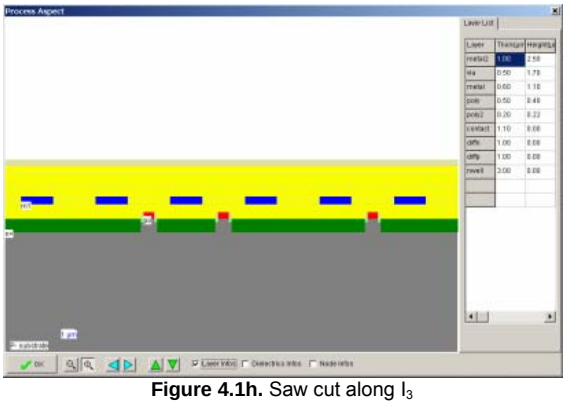


Figure 4.1h. Saw cut along I_3

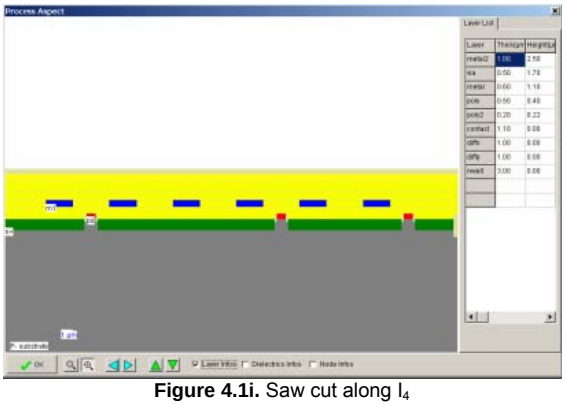


Figure 4.1i. Saw cut along I_4

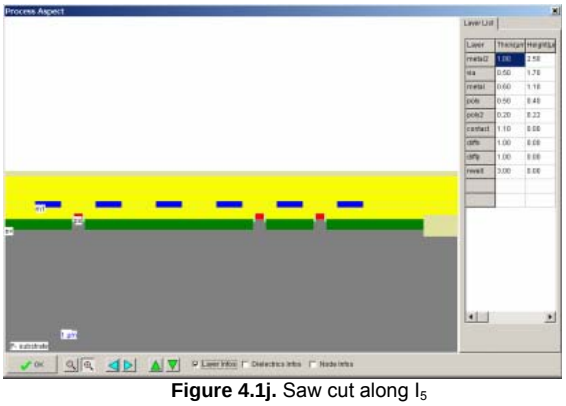


Figure 4.1j. Saw cut along I_5

Figures 4.1k to 4.1p shows the 2D view of the process for all of the select lines (/S0, S0, /S1, S1, /S2, S2).

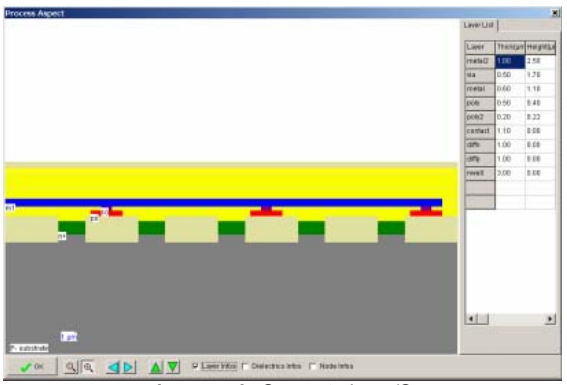


Figure 4.1k. Saw cut along $/S_0$

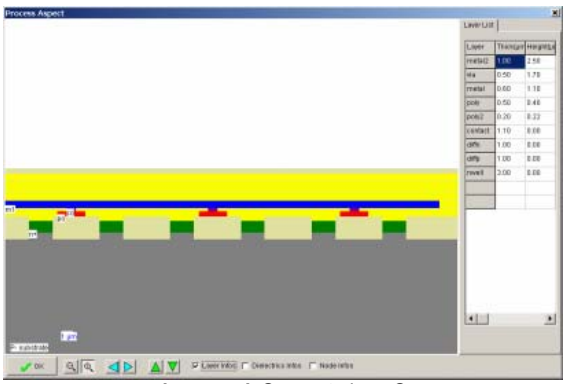


Figure 4.1l. Saw cut along S_0

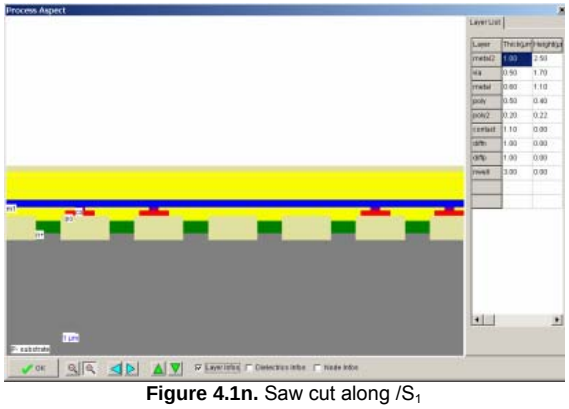


Figure 4.1n. Saw cut along /S_1

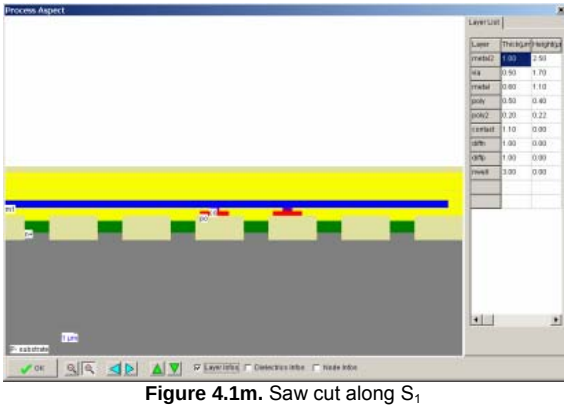


Figure 4.1m. Saw cut along S_1

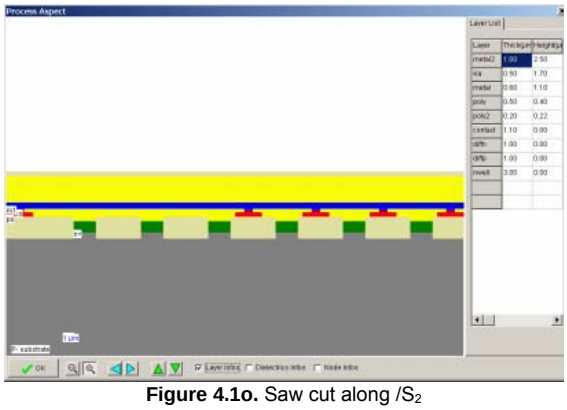


Figure 4.1o. Saw cut along /S_2

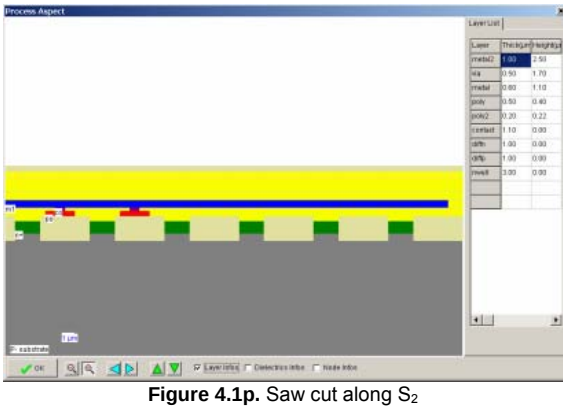


Figure 4.1p. Saw cut along S_2

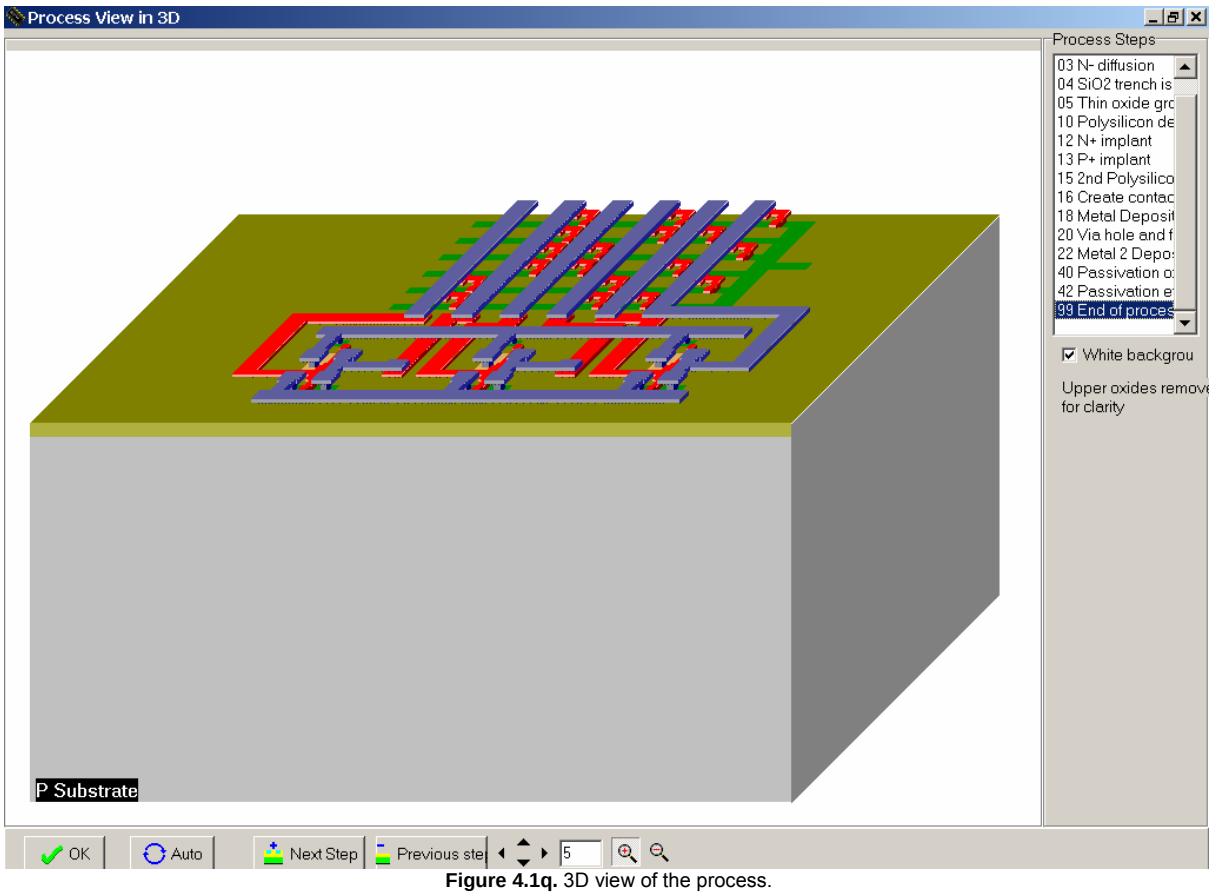


Figure 4.1q. 3D view of the process.

The simulation of the CMOS fabrication process is performed, step by step. On figure 4.1q, the picture represents the nMOS devices, pMOS devices, polysilicon and contacts, together with the metal layers stacked on the top of the active devices.

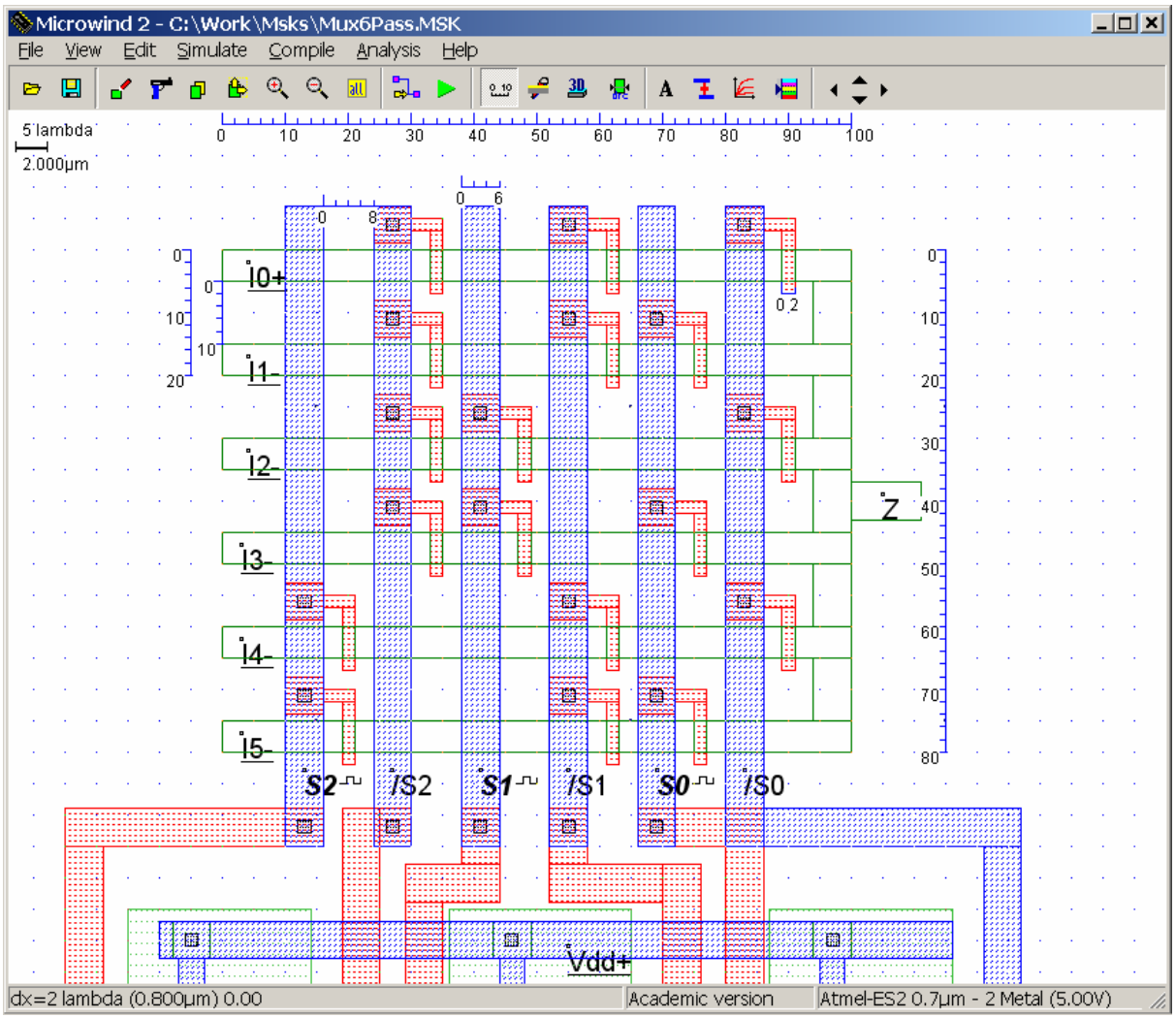


Figure 4.1r. Some useful measurements

4.1.1. PSPICE File

```
CIRCUIT C:\Work\Msks\Mux6Pass.MSK
*
* IC Technology: Atmel-ES2 0.7µm - 2 Metal
*
VDD 1 0 DC 5.00
VS1 26 0 PULSE(0.00 5.00 7.95N 0.05N 0.05N 7.95N 16.00N)
VS2 27 0 PULSE(0.00 5.00 3.95N 0.05N 0.05N 3.95N 8.00N)
VS0 28 0 PULSE(0.00 5.00 15.95N 0.05N 0.05N 15.95N 32.00N)
*
* List of nodes
* "/S0" corresponds to n°3
* "/S1" corresponds to n°4
* "/S2" corresponds to n°5
* "Z" corresponds to n°6
* "N8" corresponds to n°8
* "N9" corresponds to n°9
* "I0" corresponds to n°20
* "I1" corresponds to n°21
* "I2" corresponds to n°22
```

```
* "I3" corresponds to n°23
* "I4" corresponds to n°24
* "I5" corresponds to n°25
* "S1" corresponds to n°26
* "S2" corresponds to n°27
* "S0" corresponds to n°28
*
* MOS devices
MN1 0 5 17 0 N1  W= 2.00U L= 0.80U
MN2 0 5 16 0 N1  W= 2.00U L= 0.80U
MN3 0 5 15 0 N1  W= 2.00U L= 0.80U
MN4 1 5 14 0 N1  W= 2.00U L= 0.80U
MN5 17 26 13 0 N1  W= 2.00U L= 0.80U
MN6 16 26 12 0 N1  W= 2.00U L= 0.80U
MN7 0 26 4 0 N1  W= 2.40U L= 0.80U
MN8 19 4 11 0 N1  W= 2.00U L= 0.80U
MN9 18 4 10 0 N1  W= 2.00U L= 0.80U
MN10 15 4 9 0 N1  W= 2.00U L= 0.80U
MN11 14 4 8 0 N1  W= 2.00U L= 0.80U
MN12 11 28 6 0 N1  W= 2.00U L= 0.80U
MN13 13 28 6 0 N1  W= 2.00U L= 0.80U
MN14 9 28 6 0 N1  W= 2.00U L= 0.80U
MN15 10 3 6 0 N1  W= 2.00U L= 0.80U
MN16 12 3 6 0 N1  W= 2.00U L= 0.80U
MN17 8 3 6 0 N1  W= 2.00U L= 0.80U
MN18 0 28 3 0 N1  W= 2.40U L= 0.80U
MN19 0 27 18 0 N1  W= 2.00U L= 0.80U
MN20 0 27 19 0 N1  W= 2.00U L= 0.80U
MN21 0 27 5 0 N1  W= 2.40U L= 0.80U
MP1 1 28 3 1 P1  W= 6.00U L= 0.80U
MP2 1 26 4 1 P1  W= 6.00U L= 0.80U
MP3 1 27 5 1 P1  W= 6.00U L= 0.80U
*
C2 1 0 130.224fF
C3 3 0 44.956fF
C4 4 0 43.702fF
C5 5 0 42.703fF
C6 6 0 98.039fF
C8 8 0 14.660fF
C9 9 0  7.100fF
C10 10 0 14.660fF
C11 11 0  7.100fF
C12 12 0 22.220fF
C13 13 0 14.660fF
C14 14 0 14.660fF
C15 15 0 14.660fF
C16 16 0  7.100fF
C17 17 0  7.100fF
C18 18 0 22.220fF
C19 19 0 22.220fF
C20 1 0 18.440fF
C26 26 0 17.536fF
C27 27 0 19.648fF
C28 28 0 18.765fF
*
* n-MOS Model 3 :
* Standard
.MODEL N1 NMOS LEVEL=3 VTO=0.80 U0=0.060 TOX= 3.0E-9
+LD =-0.050U THETA=0.200 GAMMA=0.400
+PHI=0.700 KAPPA=0.010 VMAX=130.00K
+CGSO=200.0p CGDO=200.0p
*
* p-MOS Model 3:
* high speed pMOS
.MODEL P1 PMOS LEVEL=3 VTO=-1.10 U0=0.020 TOX= 3.0E-9
+LD =-0.050U THETA=0.200 GAMMA=0.400
+PHI=0.700 KAPPA=0.010 VMAX=100.00K
+CGSO=200.0p CGDO=200.0p
*
* Transient analysis
*
.TEMP 27.0
.TRAN 8.00PS 100.00N
.PROBE
.END
```

4.1.2. Calculation of Transistor Channel Resistance

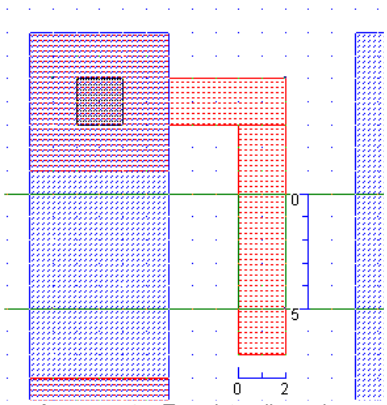


Figure 4.1.2a. Transistor dimensions

The simple n-type pass transistor shown in figure 4.1.2a has a channel length L = 5λ and a channel width W = 2λ. Therefore,

$$Z = \frac{L}{W} = \frac{5}{2} = 2.5 \qquad \{4.1.2.1\}$$

Thus, channel resistance

$$R = ZR_S \qquad \{4.1.2.2\}$$

Assuming R_S = 2 x 10^4 (from [B3], table 4-1, page 96), note this value is an approximation and may not be correct for the es207 foundry.

$$R = 2.5 \times 2 \times 10^4 = 50 \times 10^3 \Omega \qquad \{4.1.2.3\}$$

4.2. Using PMOS Pass Transistors

From figure 4.2a it is clear that 3 series (e.g. AND gate) PMOS transistors are required for each data line, also notice that the gate connections of the PMOS transistors are the inverse of the NMOS (figure 4.1a) pass transistors solution. The reason for this is that NMOS transistors are switched ON by a logic '1', while PMOS transistors are switched ON by a logic '0'. 18 PMOS transistors are required for data selection, 3 NMOS and 3 PMOS transistors are required for the invertors; that's a total of 24 transistors (3 NMOS, 21 PMOS).

S_2	S_1	S_0	Z	PMOS Pass Transistors
0	0	0	I_0	$S_2.S_1.S_0$
0	0	1	I_1	$S_2.S_1.\overline{S_0}$
0	1	0	I_2	$S_2.\overline{S_1}.S_0$
0	1	1	I_3	$S_2.\overline{S_1}.\overline{S_0}$
1	0	0	I_4	$\overline{S_2}.S_1.S_0$
1	0	1	I_5	$\overline{S_2}.S_1.\overline{S_0}$

Figure 4.2a. Table specifying the PMOS transistors required for each data line.

Note: The unused data line addresses have been completely ignored. Theoretically if these addresses are used no data line is connected to the Z output, but their maybe some sort of output due to capacitive effects.

Using Figure 4.2a, the circuit diagram (figure 4.2b) can easily be drawn. For example look closely at line I_0 were the three series transistors for that line are connected to $S_2.S_1.S_0$ as specified in figure 4.2a, hence line I_0 is connected to Z when S_2 = 0, S_1 = 0 & S_0 = 0. Note each data line has a unique combination of transistors, hence only one line is connected to Z at any one time, expect during periods of metastability (e.g. a transistor is switching OFF and another is switching ON, for a short period of time both transistors will be ON).

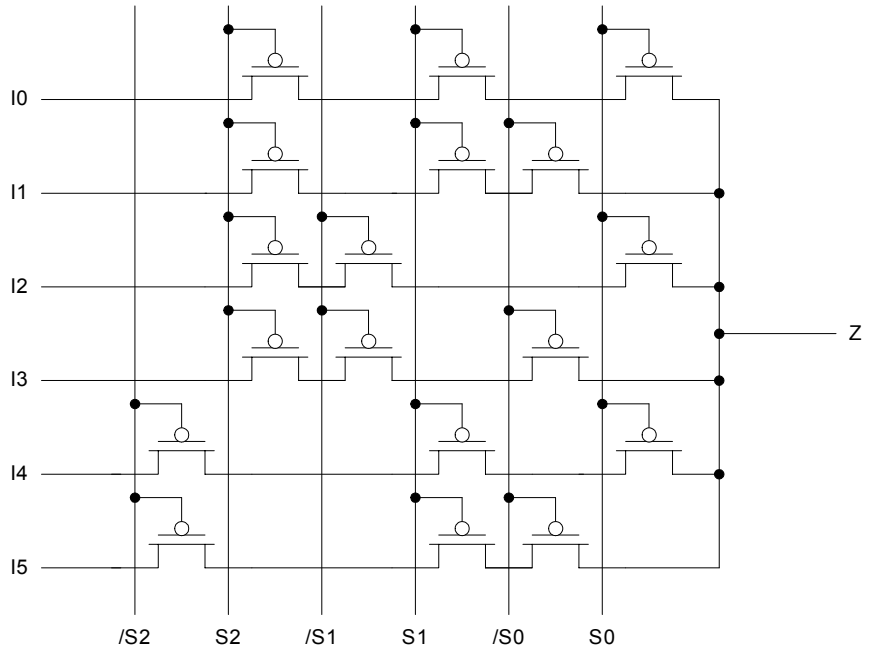


Figure 4.2b. PMOS circuit diagram.

Once the circuit diagram has been completed, the stick diagram can easily be drawn (figure 4.2c). Notice that the stick diagram (figure 4.1c) can be directly compared with the circuit diagram (figure 4.1b) as the layout of both is identical. Note that the PMOS stick diagram (figure 4.2c) is exactly the same as the NMOS stick diagram (figure 4.1c), expect that P+ diffusion was used and not N+, N well surrounding all of the transistors and the select line inputs have been inverted.

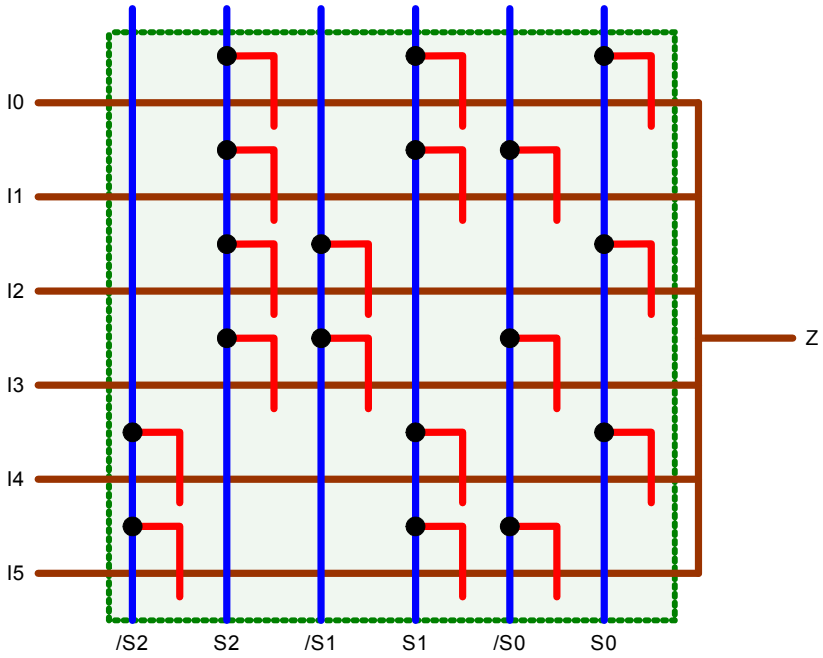


Figure 4.2c. PMOS stick diagram.

Once the stick diagram (figure 4.2c) was completed, the design was drawn in MicroWind (figure 4.2d) which was verified using the design rule checker (foundry es207.rul selected). The mask file (MicroWind design file) was designed for simplicity (easy to understand and describe) which can be directly compared with the stick diagram (figure 4.2c) and the circuit diagram (figure 4.2b). This means that the design may not be optimised for performance and compactness, allow time was taken making sure that transistor spacing was at a minimum; e.g. as close as the design check would allow, without rueing the simplicity of the design.

The design is extremely simple; basically there are three series PMOS transistors along each data line (e.g. all three transistors need to be on before the line is connected to Z). The three standard CMOS invertors, are used to invert the select lines, these select lines (6 including inverted lines) are connected to metal 1 which run vertically (parallel to each other) across P+ diffusion (data lines). A via (poly to metal 1) is used to connect the gate of each PMOS transistor to the appropriate select line as specified in figure 4.2a. The polysilicon crosses P+ diffusion (PMOS transistor); breaking current flow along the P+ diffusion (data line) until the transistor is switched ON. One large N well surrounds all of the PMOS pass transistors, instead of 18 separate small N wells around individual PMOS transistors as done for the PMOS devices in the inverter circuits. This solves routing problems as each N well requires a connection to +VDD.

The PMOS pass transistor solution (figure 4.2d) is similar to the NMOS pass transistor solution (figure 4.1d); hence a direct comparison can be made. One of the key differences is the width of the P+ diffusion (figure 4.2d) is double the width of the N+ diffusion (figure 4.1d), this is an important aspect of VLSI design and will be discussed in detail in chapter 6 (conclusions).

Figures 4.2e to 4.2j shows the 2D view of the process for all of the data lines (I_0 to I_5). *"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 fabrication principles. A click of the mouse at the first point and the release of the mouse at the second point give the cross-section."* [J1]

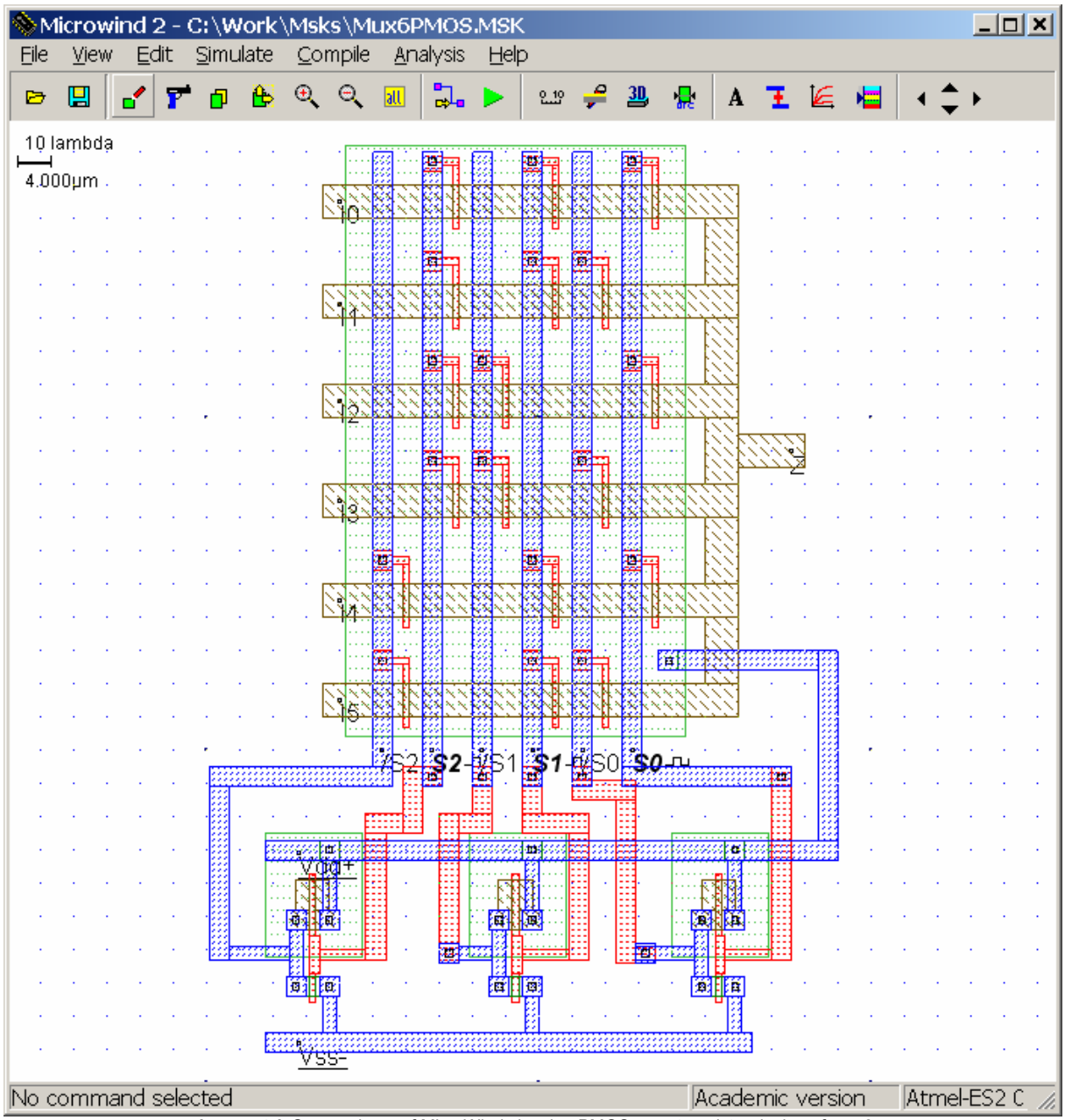


Figure 4.2d. Screen dump of MicroWind showing PMOS pass transistor design of mux6.

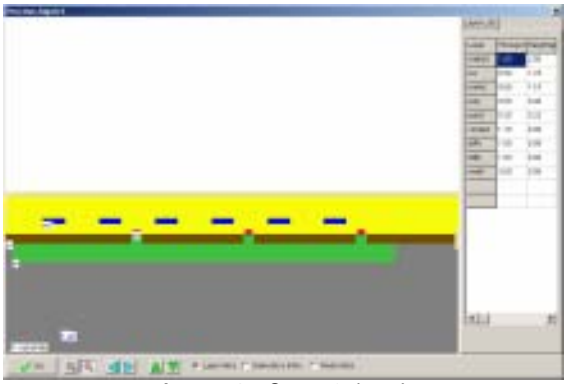


Figure 4.2e. Saw cut along I_0

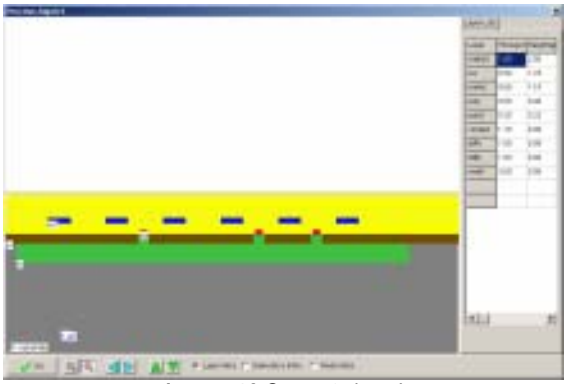


Figure 4.2f. Saw cut along I_1

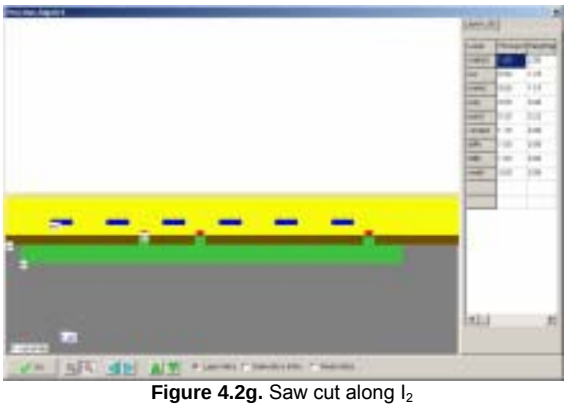


Figure 4.2g. Saw cut along I_2

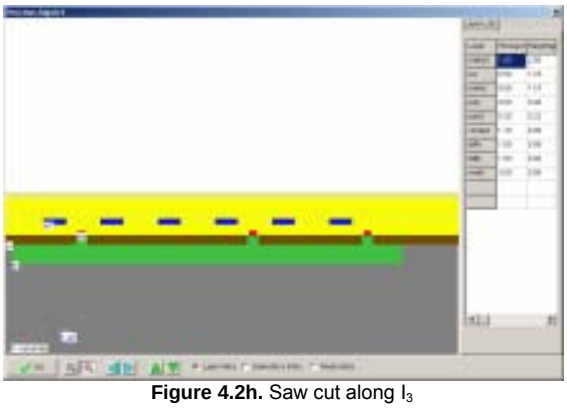


Figure 4.2h. Saw cut along I_3

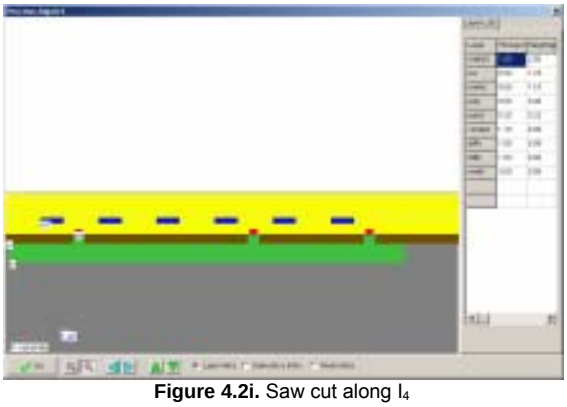


Figure 4.2i. Saw cut along I_4

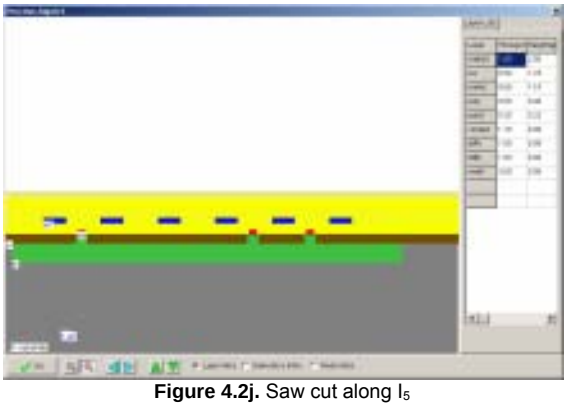


Figure 4.2j. Saw cut along I_5

Figures 4.2k to 4.1p show the 2D view of the process for all of the select lines (/S0, S0, /S1, S1, /S2, S2)

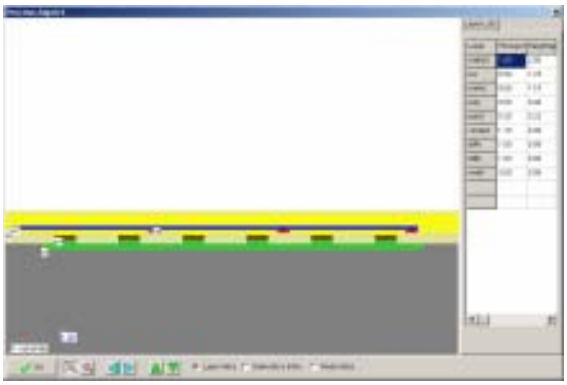


Figure 4.2k. Saw cut along S_0

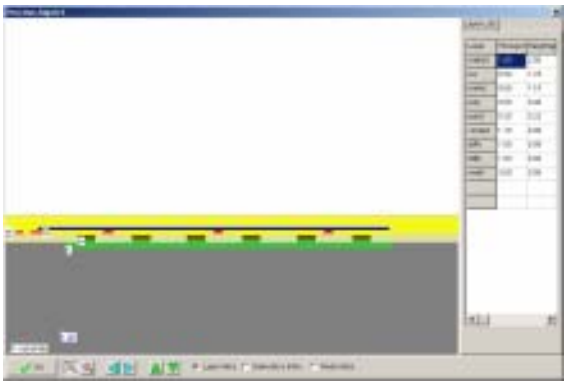


Figure 4.2l. Saw cut along $/S_0$

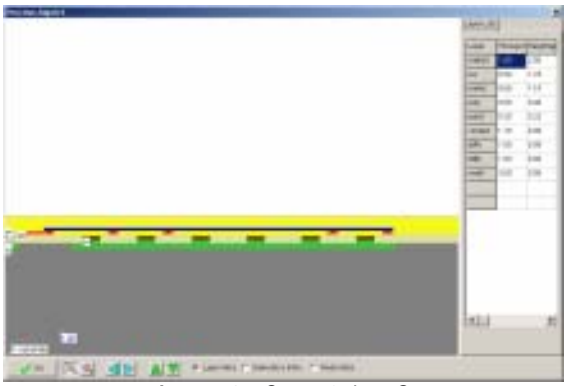


Figure 4.2n. Saw cut along S_1

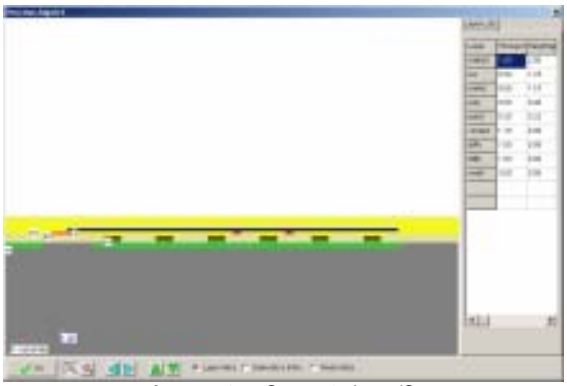


Figure 4.2m. Saw cut along $/S_1$

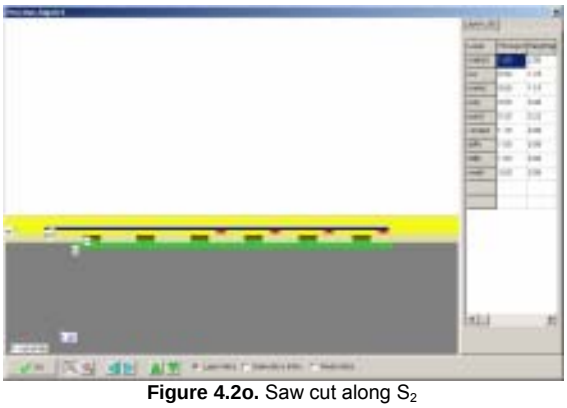


Figure 4.2o. Saw cut along S_2

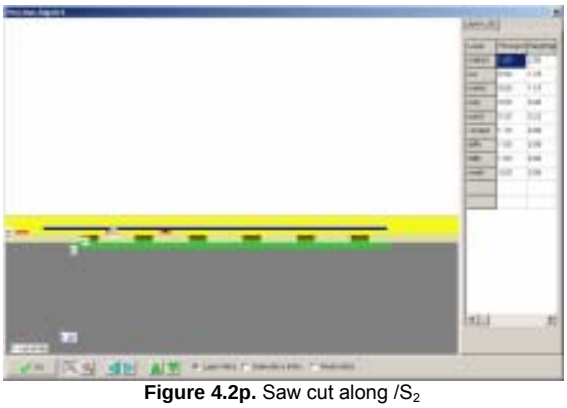


Figure 4.2p. Saw cut along $/S_2$

The simulation of the CMOS fabrication process is performed, step by step. On figure 4.2q, the picture represents the nMOS devices, pMOS devices, polysilicon and contacts, together with the metal layers stacked on the top of the active devices.

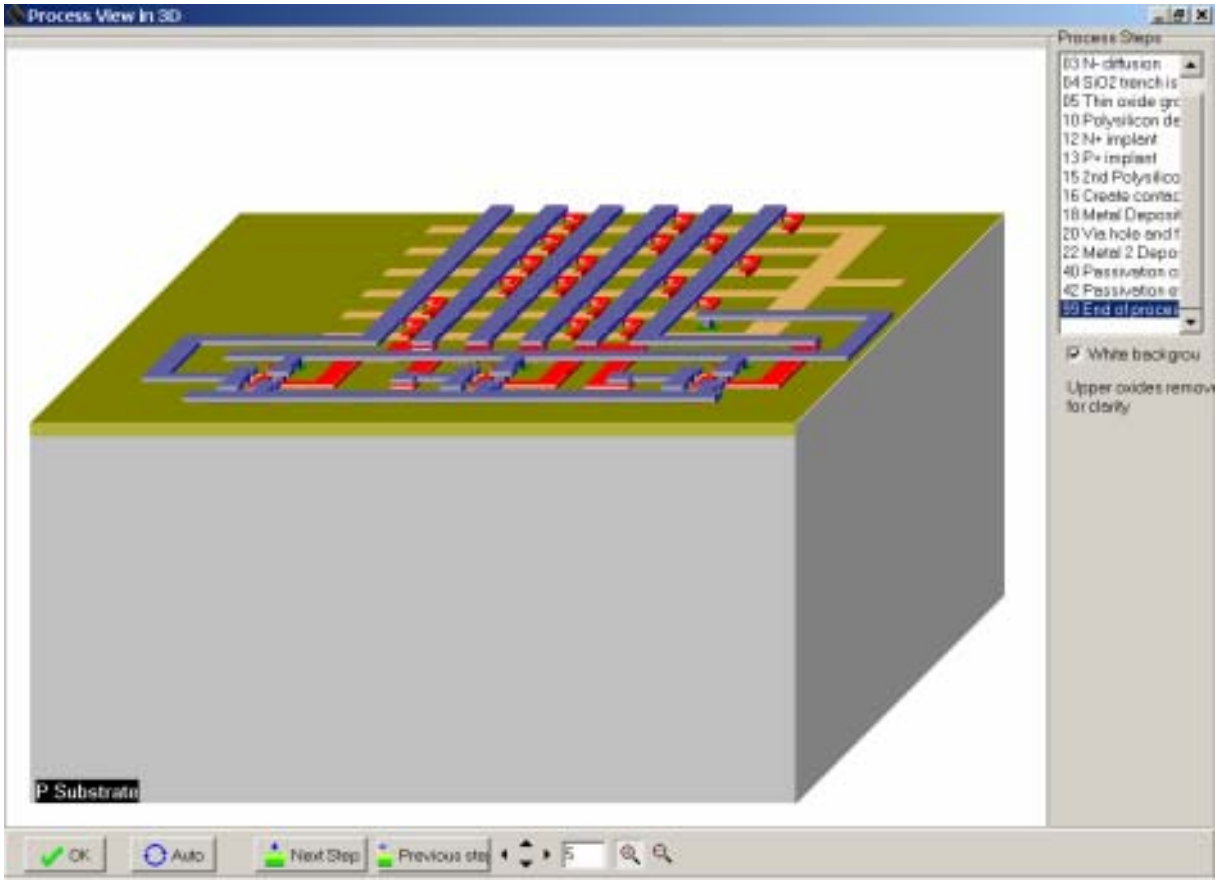


Figure 4.2q. 3D view of the process.

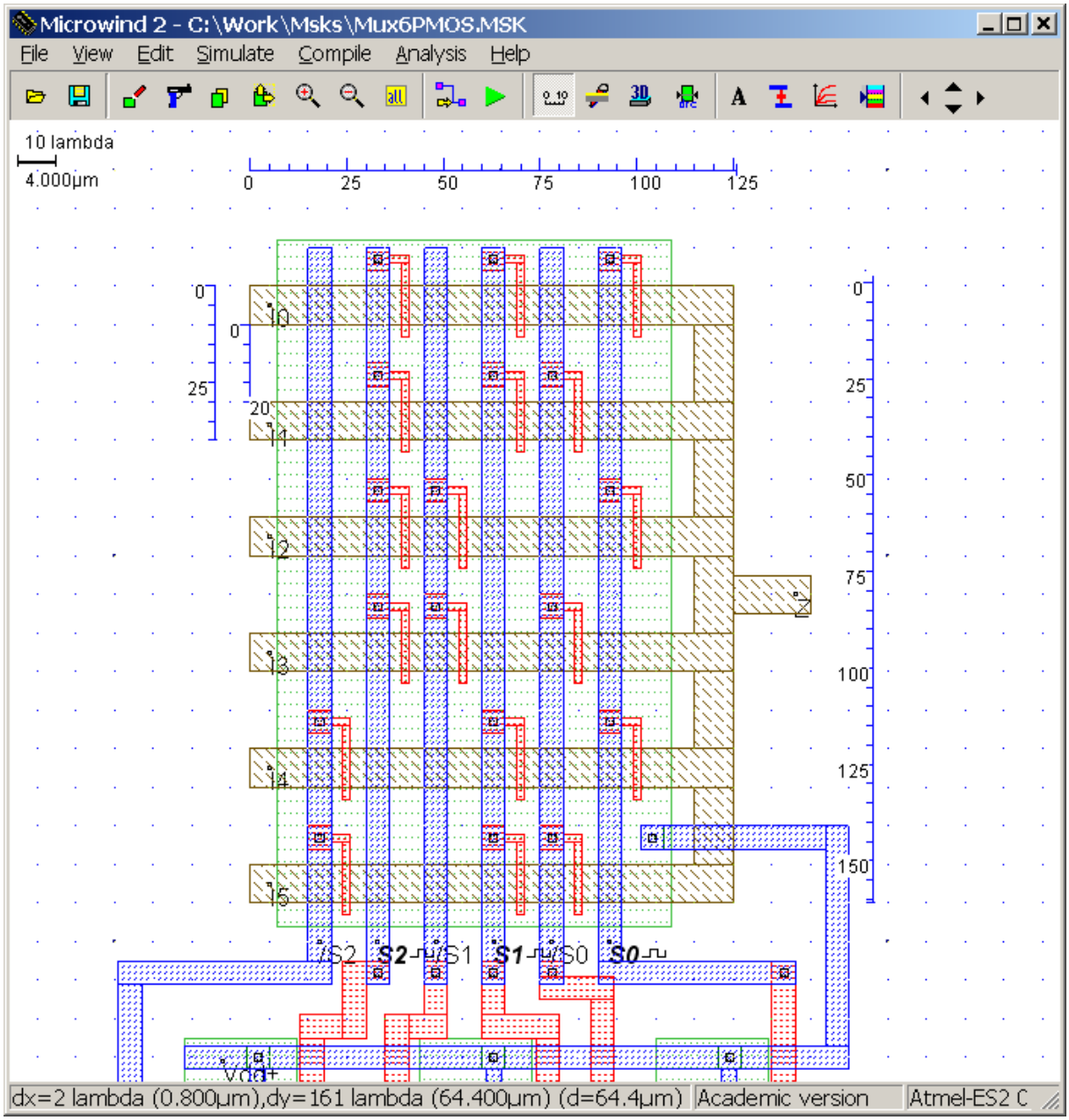


Figure 4.2r. Some useful measurements

4.2.1. PSPICE File

```
CIRCUIT C:\Work\Msks\Mux6PMOS.MSK
*
* IC Technology: Atmel-ES2 0.7µm - 2 Metal
*
VDD 1 0 DC 5.00
VS2 26 0 PULSE(0.00 5.00 63.95N 0.05N 0.05N 63.95N 128.00N)
VS1 27 0 PULSE(0.00 5.00 31.95N 0.05N 0.05N 31.95N 64.00N)
VS0 28 0 PULSE(0.00 5.00 15.95N 0.05N 0.05N 15.95N 32.00N)
*
* List of nodes
* "Z" corresponds to n°3
* "/S0" corresponds to n°4
* "N5" corresponds to n°5
* "N6" corresponds to n°6
```

```
* "N7" corresponds to n°7
* "N8" corresponds to n°8
* "N9" corresponds to n°9
* "/S1" corresponds to n°11
* "I0" corresponds to n°18
* "I1" corresponds to n°19
* "I2" corresponds to n°20
* "I3" corresponds to n°21
* "I4" corresponds to n°22
* "I5" corresponds to n°23
* "/S2" corresponds to n°24
* "S2" corresponds to n°26
* "S1" corresponds to n°27
* "S0" corresponds to n°28
*
* MOS devices
MN1 4 28 0 0 N1  W= 2.40U L= 0.80U
MN2 11 27 0 0 N1  W= 2.40U L= 0.80U
MN3 24 26 0 0 N1  W= 2.40U L= 0.80U
MP1 4 28 1 1 P1  W= 6.00U L= 0.80U
MP2 5 28 3 1 P1  W= 4.00U L= 0.80U
MP3 9 28 3 1 P1  W= 4.00U L= 0.80U
MP4 24 26 1 1 P1  W= 6.00U L= 0.80U
MP5 0 24 17 1 P1  W= 4.00U L= 0.80U
MP6 1 24 16 1 P1  W= 4.00U L= 0.80U
MP7 1 26 15 1 P1  W= 4.00U L= 0.80U
MP8 1 26 14 1 P1  W= 4.00U L= 0.80U
MP9 1 26 13 1 P1  W= 4.00U L= 0.80U
MP10 1 26 12 1 P1  W= 4.00U L= 0.80U
MP11 15 11 10 1 P1  W= 4.00U L= 0.80U
MP12 14 11 9 1 P1  W= 4.00U L= 0.80U
MP13 11 27 1 1 P1  W= 6.00U L= 0.80U
MP14 17 27 8 1 P1  W= 4.00U L= 0.80U
MP15 16 27 7 1 P1  W= 4.00U L= 0.80U
MP16 13 27 6 1 P1  W= 4.00U L= 0.80U
MP17 12 27 5 1 P1  W= 4.00U L= 0.80U
MP18 8 4 3 1 P1  W= 4.00U L= 0.80U
MP19 10 4 3 1 P1  W= 4.00U L= 0.80U
MP20 6 4 3 1 P1  W= 4.00U L= 0.80U
MP21 7 28 3 1 P1  W= 4.00U L= 0.80U
*
C2 1 0 439.117fF
C3 3 0 437.352fF
C4 4 0 52.175fF
C5 5 0 39.344fF
C6 6 0 20.024fF
C7 7 0 39.344fF
C8 8 0 20.024fF
C9 9 0 58.664fF
C10 10 0 39.344fF
C11 11 0 50.012fF
C12 12 0 39.344fF
C13 13 0 39.344fF
C14 14 0 20.024fF
C15 15 0 20.024fF
C16 16 0 58.664fF
C17 17 0 58.664fF
C18 1 0 53.512fF
C19 1 0 53.512fF
C20 1 0 53.512fF
C21 1 0 53.512fF
C22 1 0 34.192fF
C24 24 0 53.980fF
C26 26 0 29.043fF
C27 27 0 28.813fF
C28 28 0 30.886fF
*
* Crosstalk capacitance
*
Cx4_27 4 27 0.509fF
Cx4_28 4 28 0.584fF
Cx11_26 11 26 0.509fF
Cx11_27 11 27 0.509fF
Cx24_26 24 26 0.509fF
*
* n-MOS Model 3 :
* Standard
```

```
.MODEL N1 NMOS LEVEL=3 VTO=0.80 U0=0.060 TOX= 3.0E-9
+LD =-0.050U THETA=0.200 GAMMA=0.400
+PHI=0.700 KAPPA=0.010 VMAX=130.00K
+CGSO=200.0p CGDO=200.0p
*
* p-MOS Model 3:
* high speed pMOS
.MODEL P1 PMOS LEVEL=3 VTO=-1.10 U0=0.020 TOX= 3.0E-9
+LD =-0.050U THETA=0.200 GAMMA=0.400
+PHI=0.700 KAPPA=0.010 VMAX=100.00K
+CGSO=200.0p CGDO=200.0p
*
* Transient analysis
*
.TEMP 27.0
.TRAN 4.00PS 200.00N
.PROBE
.END
```

4.2.2. Calculation of Transistor Channel Resistance

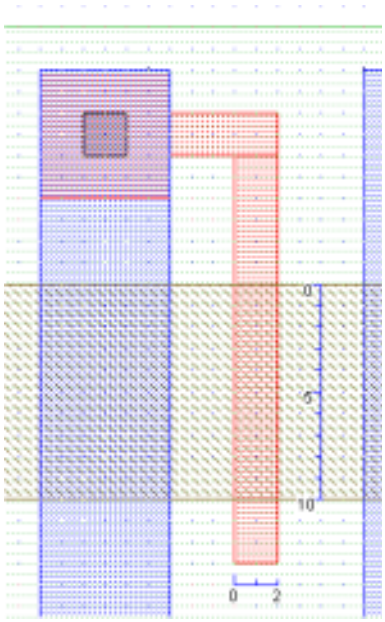


Figure 4.2.2a. Transistor dimensions

The simple p-type pass transistor shown in figure 4.2.2a has a channel length L = 10λ and a channel width W = 2λ. Therefore,

$$Z = \frac{L}{W} = \frac{10}{2} = 5 \qquad \{4.2.2.1\}$$

Thus, channel resistance

$$R = ZR_S \qquad \{4.2.2.2\}$$

Assuming $R_S = 4.5 \times 10^4$ (from [B3], table 4-1, page 96), note this value is an approximation and may not be correct for the es207 foundry.

$$R = 4.5 \times 5 \times 10^4 = 225 \times 10^3 \Omega \qquad \{4.2.2.3\}$$

4.3. Using CMOS Transmission Gates

From figure 4.3a it is clear that 3 series NMOS transistors in parallel with 3 series PMOS transistors are required for each data line. Note this arrangement is not a true transmission gate (see figure 4.3b) but a transmission process (see figure 4.3c), this technique simplifies the design making it easier to understand what is happening with all the befits of a transmission gate (e.g. good logic '1' and good logic '0'). Basically all that is required is to combine the NMOS pass transistor (4.1) design with the PMOS pass transistor design, (4.2) making a transmission process 6-to-1 line multiplexer.

S_2	S_1	S_0	Z	NMOS Pass Transistors	PMOS Pass Transistors
0	0	0	I_0	$\overline{S_2}.\overline{S_1}.\overline{S_0}$	$S_2.S_1.S_0$
0	0	1	I_1	$\overline{S_2}.\overline{S_1}.S_0$	$S_2.S_1.\overline{S_0}$
0	1	0	I_2	$\overline{S_2}.S_1.\overline{S_0}$	$S_2.\overline{S_1}.S_0$
0	1	1	I_3	$\overline{S_2}.S_1.S_0$	$S_2.\overline{S_1}.\overline{S_0}$
1	0	0	I_4	$S_2.\overline{S_1}.\overline{S_0}$	$\overline{S_2}.S_1.S_0$
1	0	1	I_5	$S_2.\overline{S_1}.S_0$	$\overline{S_2}.S_1.\overline{S_0}$

Figure 4.3a. Table specifying the NMOS & PMOS transistors required for each data line.

18 NMOS and 18 PMOS transistors are required for data selection, 3 NMOS and 3 PMOS transistors are required for the invertors; that's a total of 42 transistors (21 NMOS, 21 PMOS).

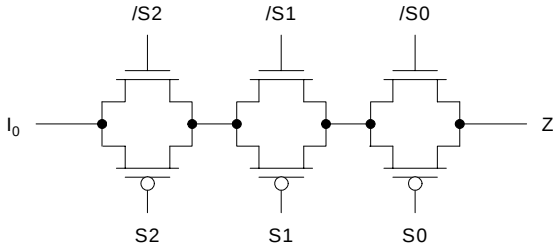


Figure 4.3b. Example transmission gate.

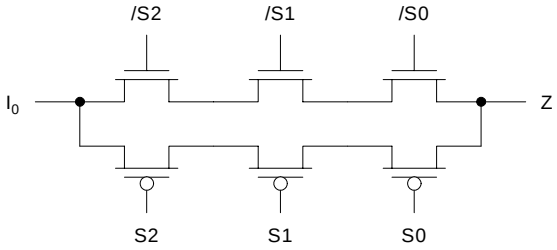


Figure 4.3c. Example transmission process.

Note: The unused data line addresses have been completely ignored. Theoretically if these addresses are used no data line is connected to the Z output, but their maybe some sort of output due to capacitive effects.

Using figure 4.3a, the circuit diagram (figure 4.3d) can easily be drawn. For example look closely at line I_0 were the three NMOS series transistors for that line are connected to $/S_2./S_1./S_0$ and are in parallel with three PMOS series transistors that are connected to $S_2.S_1.S_0$ as specified in figure 4.3a, hence line I_0 is connected to Z when $S_2 = 0$, $S_1 = 0$ & $S_0 = 0$. Note both the NMOS series transistors and PMOS series transistors are switched ON/OFF, hence the NMOS transistors will produce good logic '0's and the PMOS transistors will produce good logic '1's.

Note each data line has a unique combination of transistors, hence only one line is connected to Z at any one time, except during periods of metastability (e.g. a transistor is switching OFF and another is switching ON, for a short period of time both transistors will be ON).

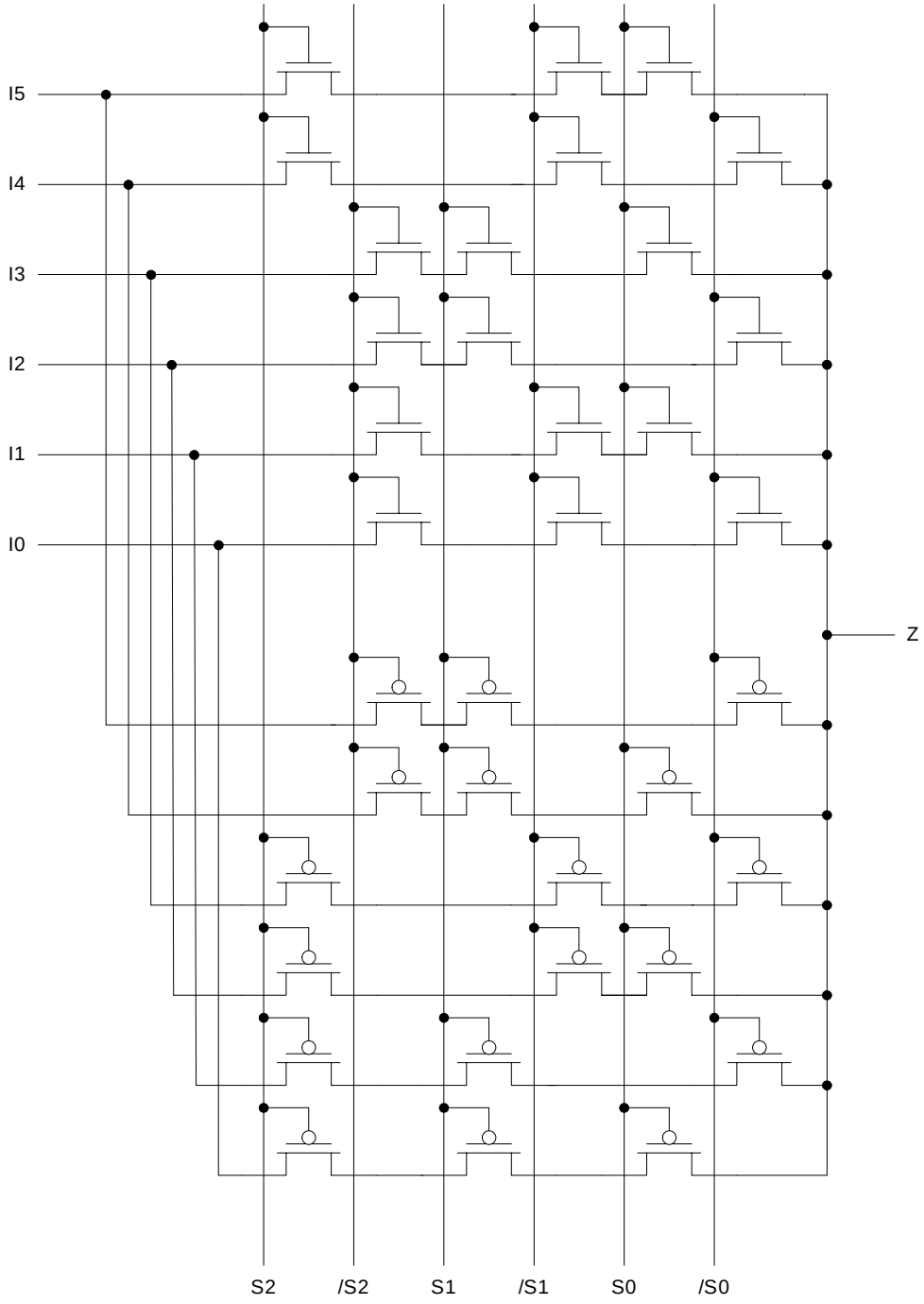


Figure 4.3d. CMOS transmission gate circuit diagram.

Once the circuit diagram has been completed, the sick diagram can easily be drawn (figure 4.3e). Notice that the stick diagram (figure 4.3e) can be directly compared with the circuit diagram (figure 4.3d) as the layout of both is identical.

Notice that the N and P type transistors are grouped together and there is one large N well surrounding all of the PMOS devices. If a true transmission gate arrangement was used and not a transmission process this grouping of N and P type transistors would be difficult to achieve due to routing problems. Also note that even if the N and P type transistor are not grouped there would still be routing problems, as each individual N well must be connect to +VDD.

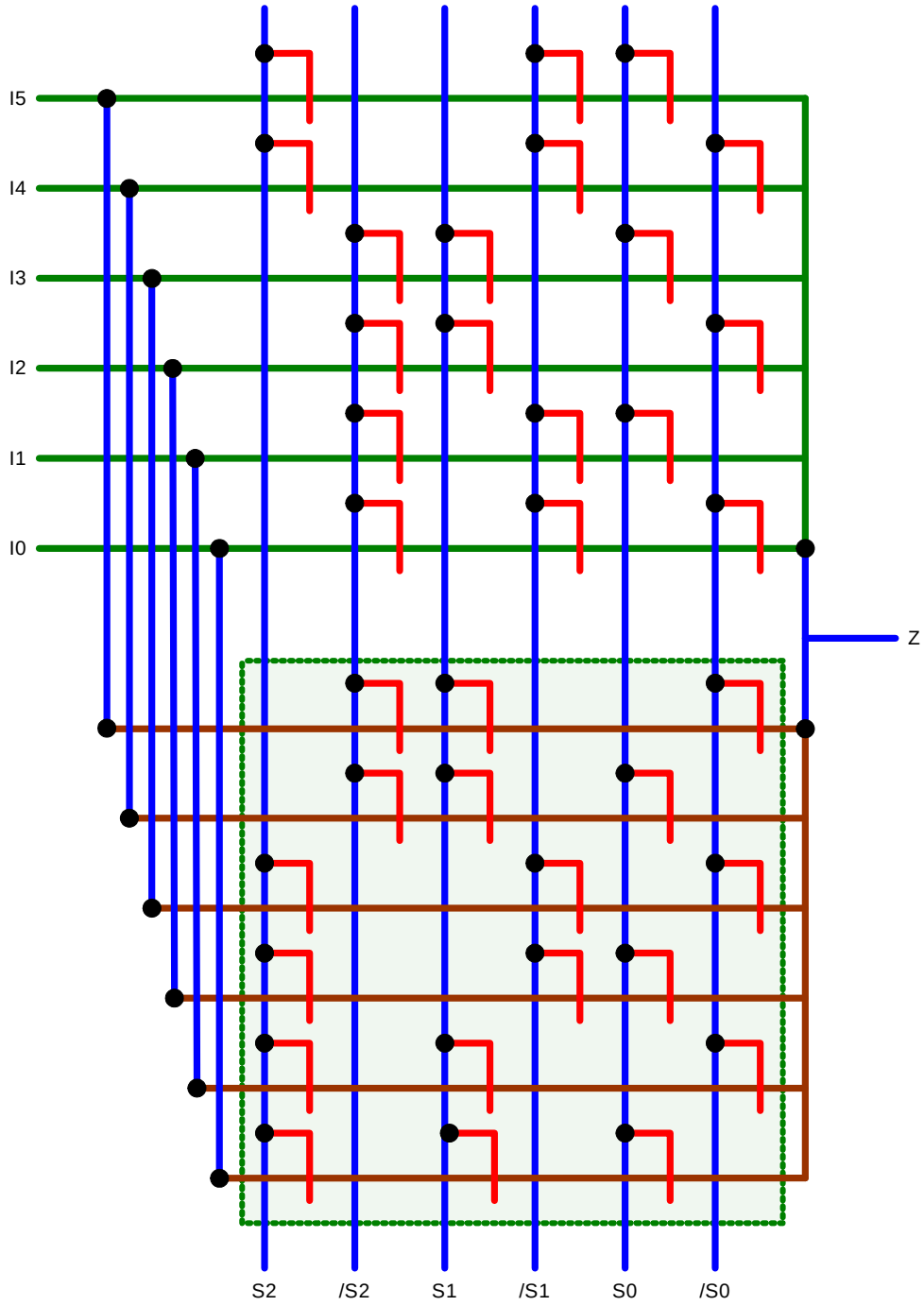


Figure 4.3e. CMOS transmission gate stick diagram.

Once the stick diagram (figure 4.3e) was completed, the design was drawn in MicroWind (figure 4.3f) which was verified using the design rule checker (foundry es207.rul selected). The mask file (MicroWind design file) was designed for simplicity (easy to understand and describe) which can be directly compared with the stick diagram (figure 4.3e) and the circuit diagram (figure 4.3d). This means that the design may not be optimised for performance and compactness, allow time was taken making sure that transistor spacing was at a minimum; e.g. as close as the design check would allow, without rueing the simplicity of the design.

The design is extremely simple; basically there are three series PMOS transistors in parallel with three series NMOS transistors along each data line (both NMOS and PMOS transistors are ON when the line is connected to Z).

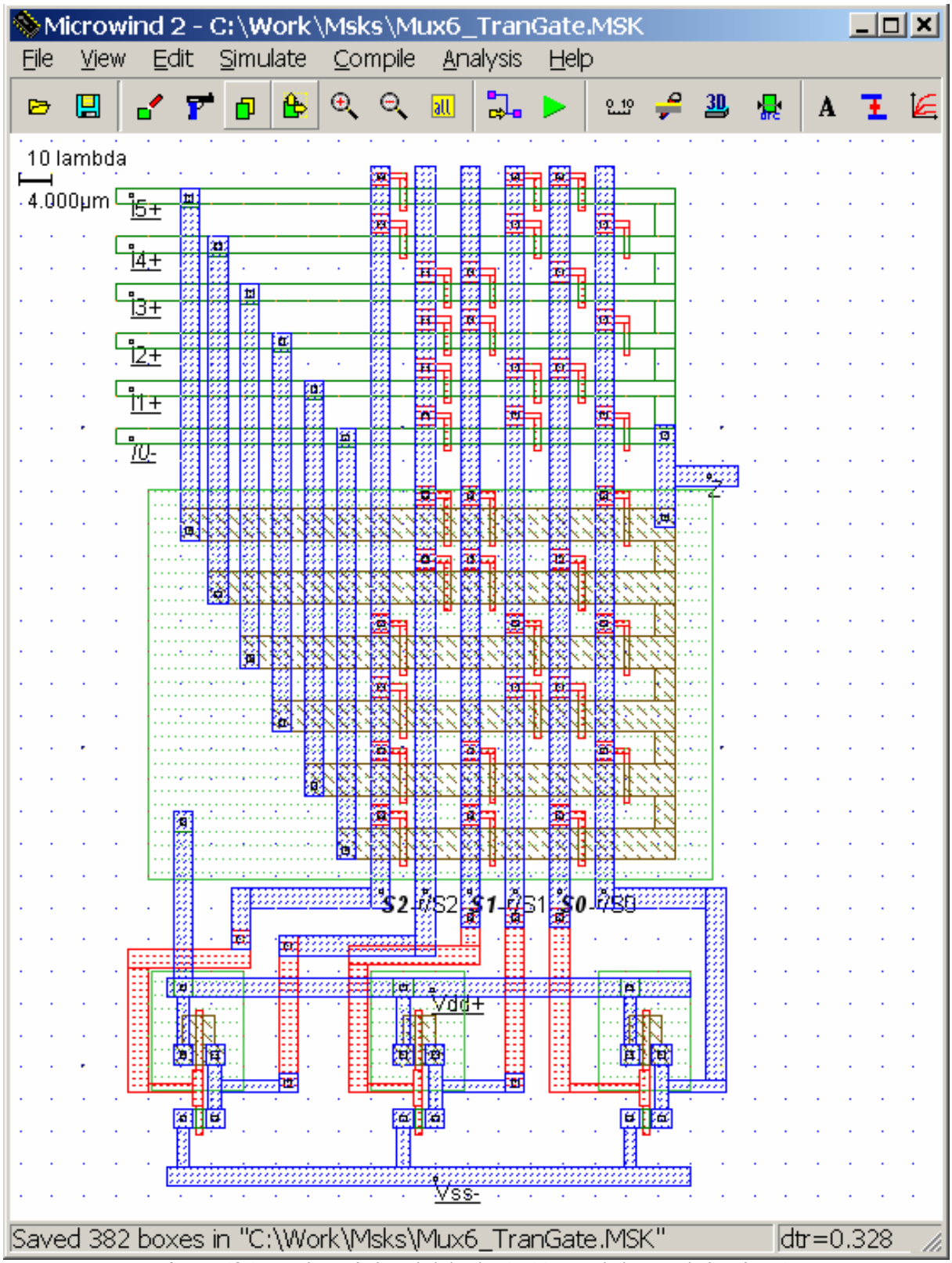


Figure 4.3f. Screen dump of MicroWind showing CMOS transmission gate design of mux6.

The three standard CMOS invertors, are used to invert the select lines, these select lines (6 including inverted lines) are connected to metal 1 which run vertically (parallel to each other) across P+ & N+ diffusion. A via (poly to metal 1) is used to connect the gate of each PMOS / NMOS transistor to the appropriate select

line as specified in figure 4.3a. The polysilicon crosses N+ diffusion for a NMOS transistor and P+ diffusion for a PMOS transistor. One large N well surrounds all of the PMOS pass transistors, instead of 18 separate small N wells around individual PMOS transistors as done for the PMOS devices in the inverter circuits. This solves routing problems as each N well requires a connection to +VDD.

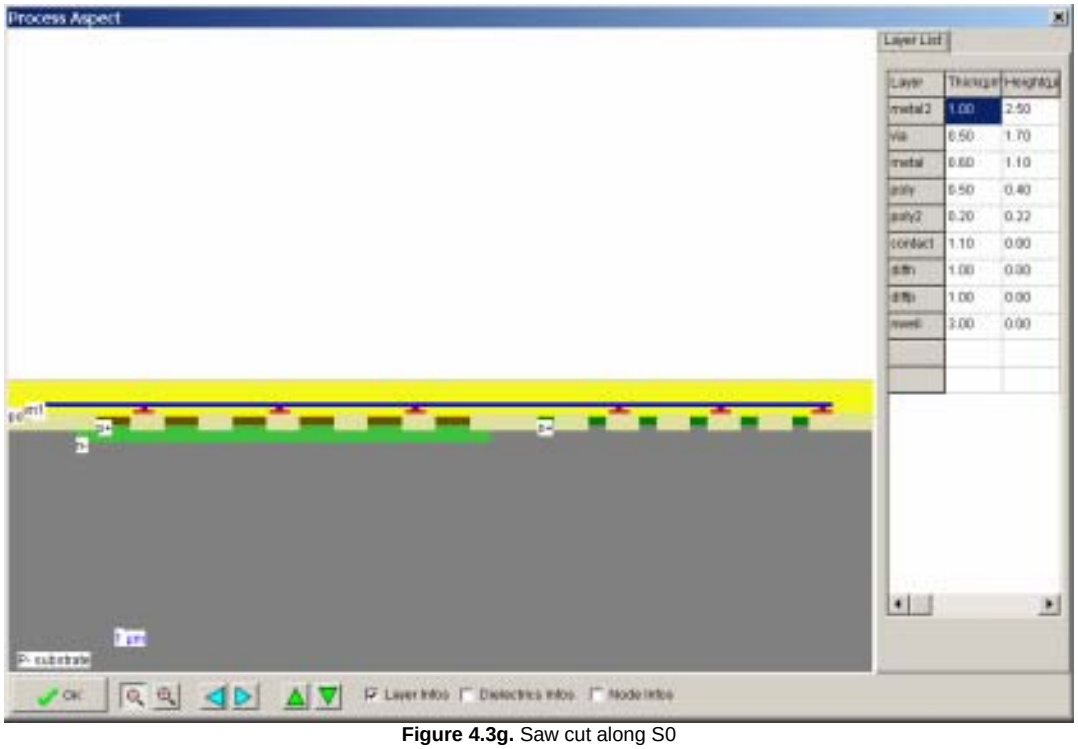


Figure 4.3g. Saw cut along S0

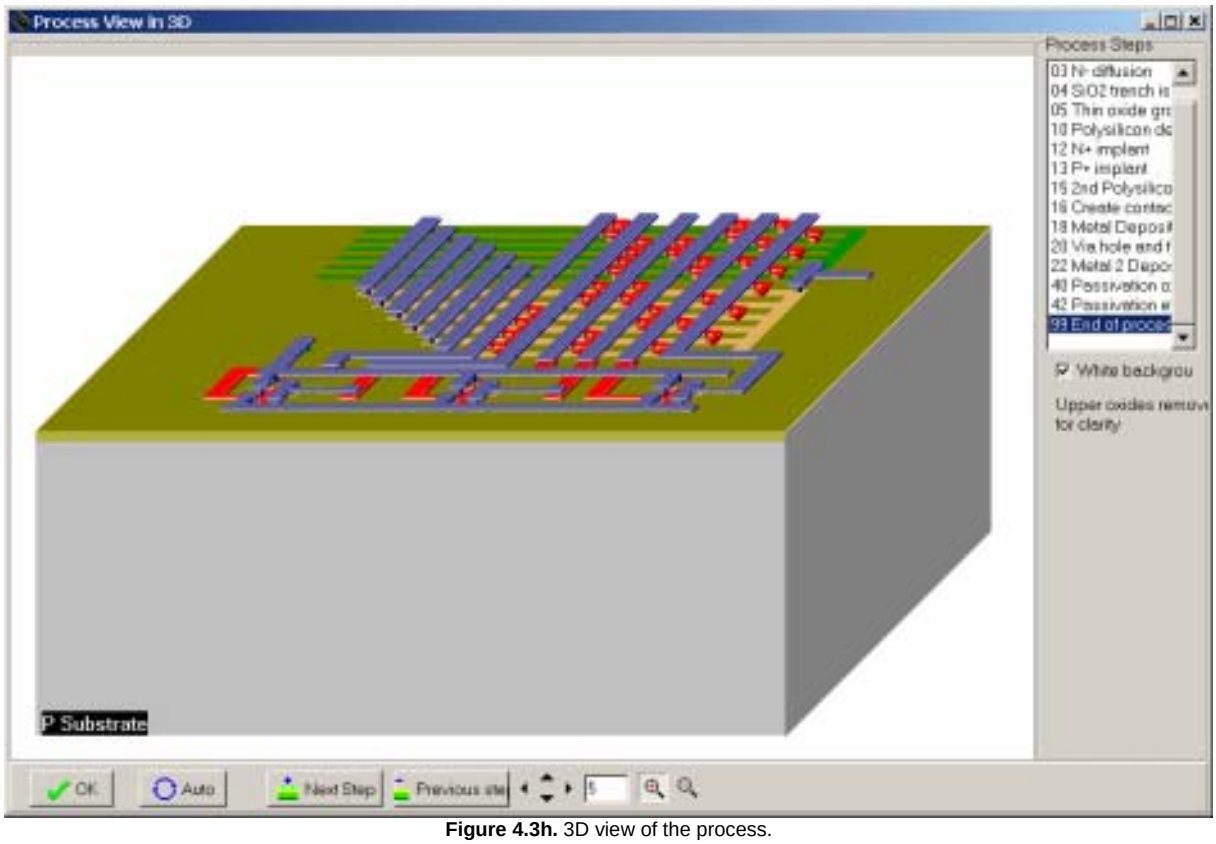


Figure 4.3h. 3D view of the process.

Figure 4.3g shows the 2D view of the process along S0 and figure 4.3h represents the NMOS devices, PMOS devices, polysilicon and contacts, together with the metal layers stacked on the top of the active devices.

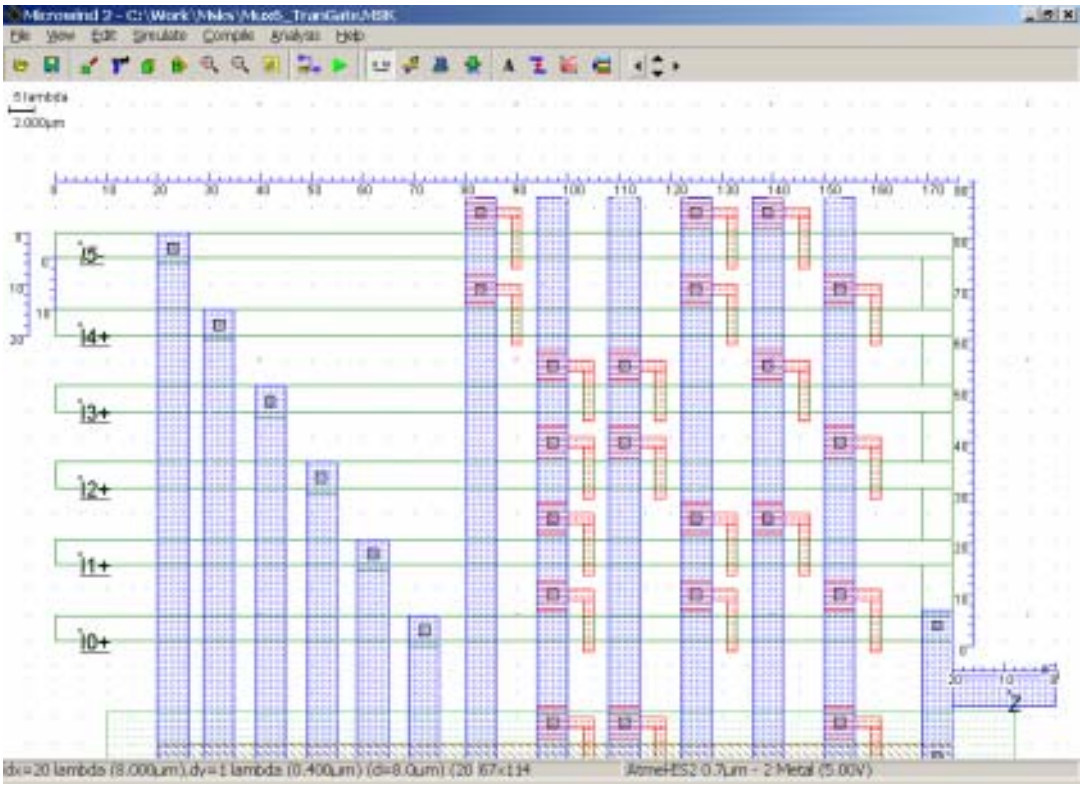


Figure 4.3i. Some useful measurements NMOS section.

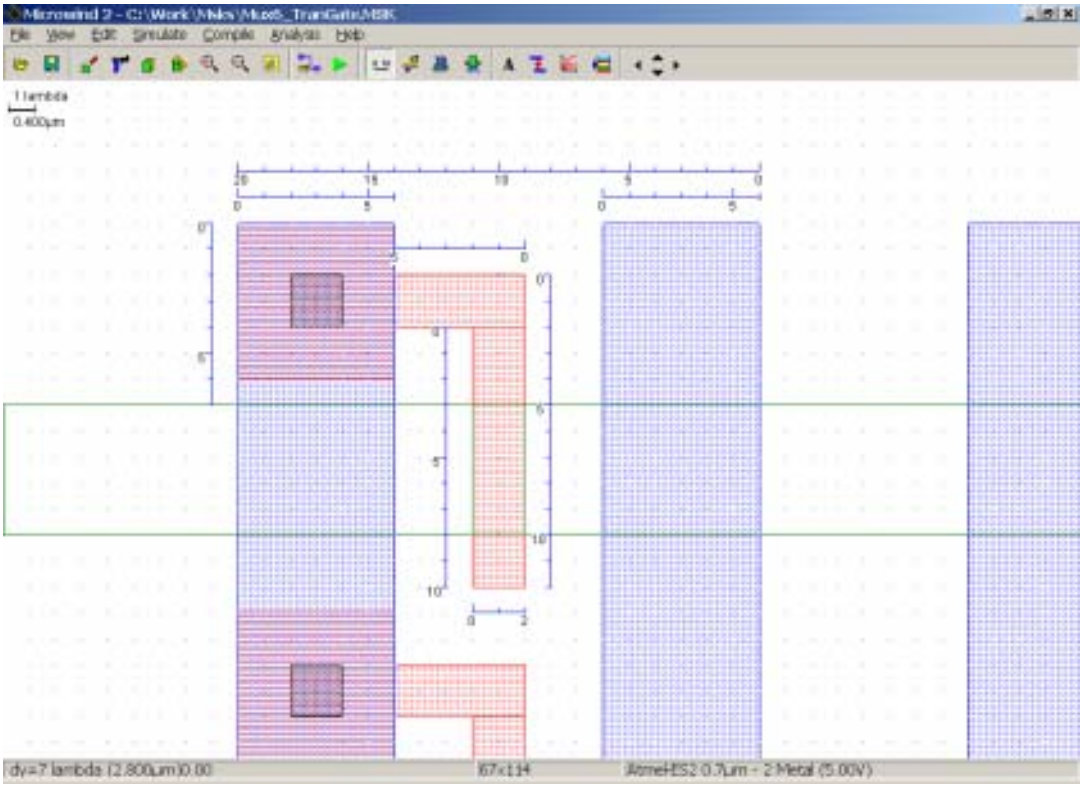


Figure 4.3j. Some useful measurements NMOS transistor.

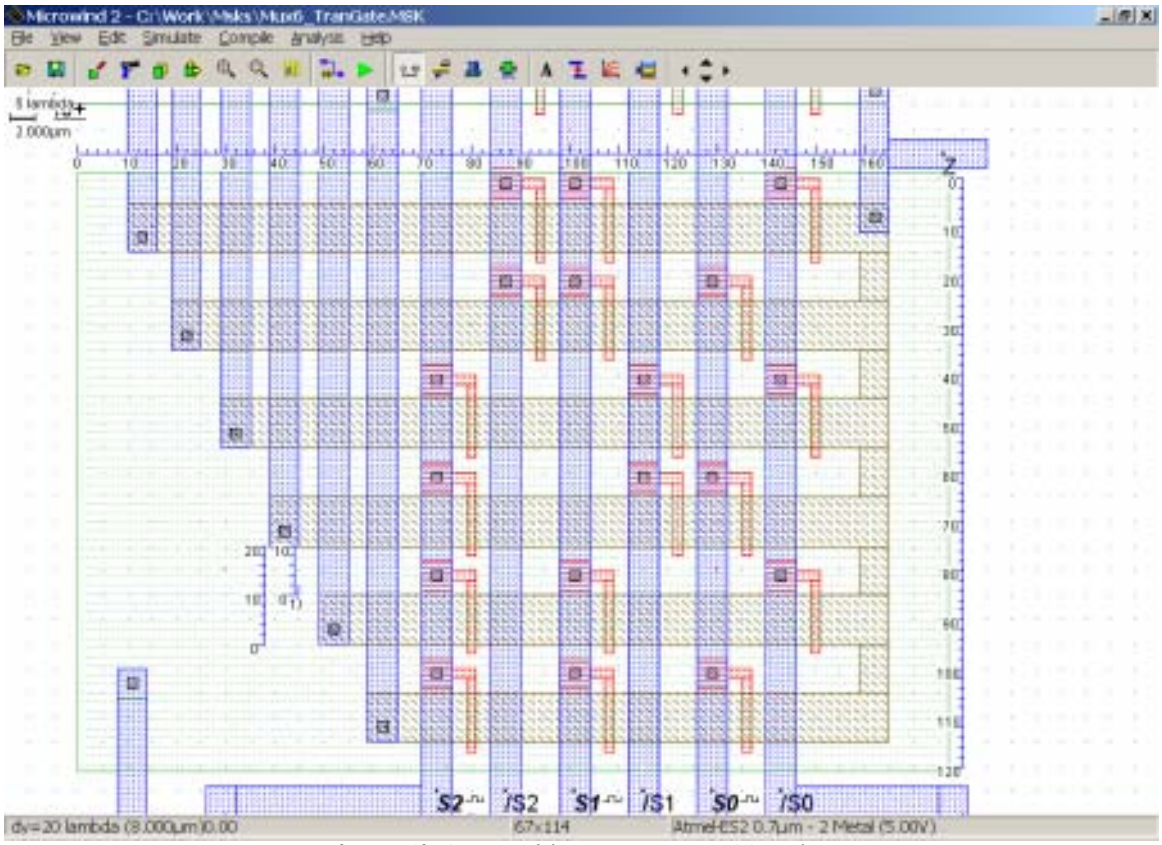


Figure 4.3k. Some useful measurements PMOS section.

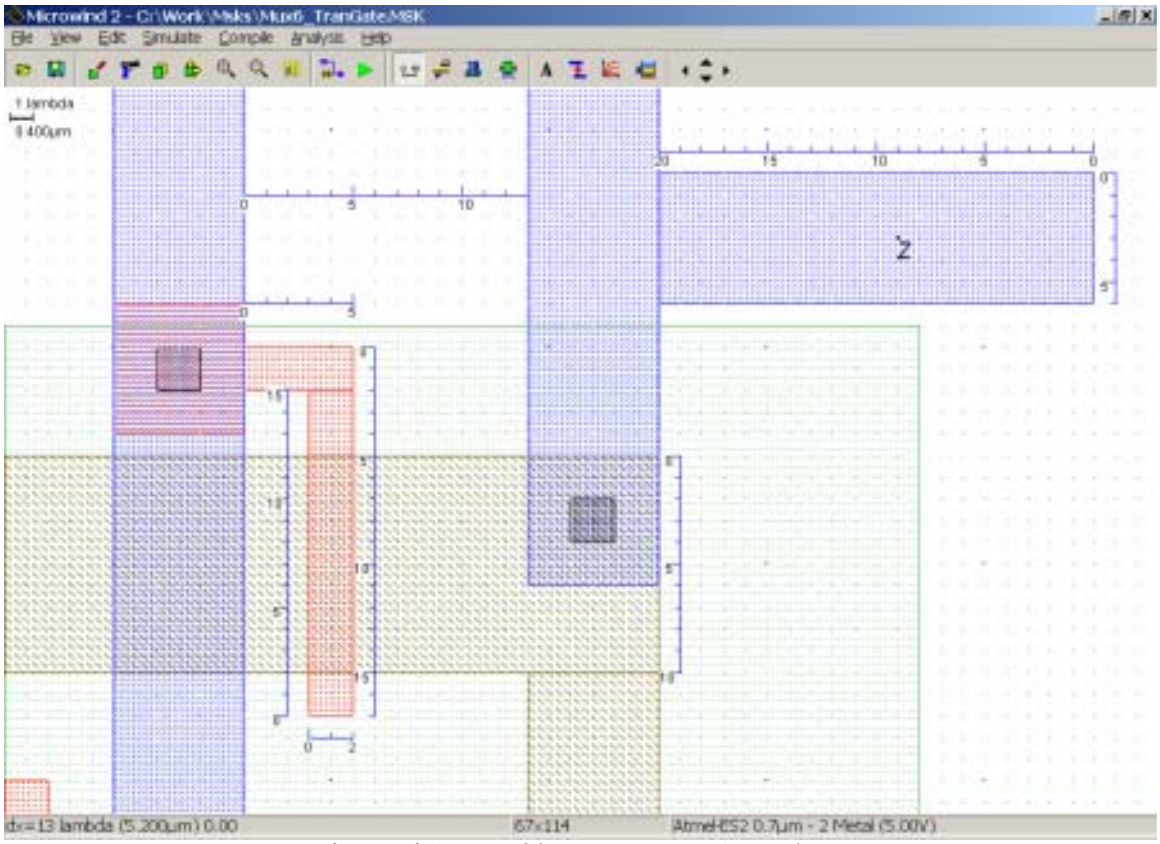


Figure 4.3l. Some useful measurements PMOS transistor.

4.3.1. PSPICE File

```
CIRCUIT C:\Work\Msks\Mux6_TranGate.MSK
*
* IC Technology: Atmel-ES2 0.7µm - 2 Metal
*
VDD 1 0 DC 5.00
VS0 38 0 PULSE(0.00 5.00 15.95N 0.05N 0.05N 15.95N 32.00N)
VS2 39 0 PULSE(0.00 5.00 63.95N 0.05N 0.05N 63.95N 128.00N)
VS1 40 0 PULSE(0.00 5.00 31.95N 0.05N 0.05N 31.95N 64.00N)
*
* List of nodes
* "Z" corresponds to n°3
* "/S0" corresponds to n°4
* "N5" corresponds to n°5
* "N6" corresponds to n°6
* "N7" corresponds to n°7
* "N8" corresponds to n°8
* "N9" corresponds to n°9
* "/S1" corresponds to n°13
* "I0" corresponds to n°18
* "I1" corresponds to n°19
* "I2" corresponds to n°20
* "I3" corresponds to n°21
* "I4" corresponds to n°22
* "/S2" corresponds to n°23
* "I5" corresponds to n°24
* "S0" corresponds to n°38
* "S2" corresponds to n°39
* "S1" corresponds to n°40
*
* MOS devices
MN1 0 38 4 0 N1  W= 2.40U L= 0.80U
MN2 27 4 3 0 N1  W= 2.00U L= 0.80U
MN3 31 4 3 0 N1  W= 2.00U L= 0.80U
MN4 29 4 3 0 N1  W= 2.00U L= 0.80U
MN5 26 38 3 0 N1  W= 2.00U L= 0.80U
MN6 30 38 3 0 N1  W= 2.00U L= 0.80U
MN7 28 38 3 0 N1  W= 2.00U L= 0.80U
MN8 36 13 26 0 N1  W= 2.00U L= 0.80U
MN9 37 13 27 0 N1  W= 2.00U L= 0.80U
MN10 34 13 28 0 N1  W= 2.00U L= 0.80U
MN11 35 13 29 0 N1  W= 2.00U L= 0.80U
MN12 32 40 30 0 N1  W= 2.00U L= 0.80U
MN13 33 40 31 0 N1  W= 2.00U L= 0.80U
MN14 1 23 32 0 N1  W= 2.00U L= 0.80U
MN15 1 23 33 0 N1  W= 2.00U L= 0.80U
MN16 1 23 34 0 N1  W= 2.00U L= 0.80U
MN17 1 23 35 0 N1  W= 2.00U L= 0.80U
MN18 0 40 13 0 N1  W= 2.40U L= 0.80U
MN19 0 39 36 0 N1  W= 2.00U L= 0.80U
MN20 1 39 37 0 N1  W= 2.00U L= 0.80U
MN21 0 39 23 0 N1  W= 2.40U L= 0.80U
MP1 1 38 4 1 P1  W= 6.00U L= 0.80U
MP2 7 4 3 1 P1  W= 4.00U L= 0.80U
MP3 5 4 3 1 P1  W= 4.00U L= 0.80U
MP4 9 4 3 1 P1  W= 4.00U L= 0.80U
MP5 8 38 3 1 P1  W= 4.00U L= 0.80U
MP6 6 38 3 1 P1  W= 4.00U L= 0.80U
MP7 10 38 3 1 P1  W= 4.00U L= 0.80U
MP8 14 13 5 1 P1  W= 4.00U L= 0.80U
MP9 15 13 6 1 P1  W= 4.00U L= 0.80U
MP10 11 40 7 1 P1  W= 4.00U L= 0.80U
MP11 12 40 8 1 P1  W= 4.00U L= 0.80U
MP12 16 40 9 1 P1  W= 4.00U L= 0.80U
MP13 17 40 10 1 P1  W= 4.00U L= 0.80U
MP14 0 23 11 1 P1  W= 4.00U L= 0.80U
MP15 1 23 12 1 P1  W= 4.00U L= 0.80U
MP16 1 40 13 1 P1  W= 6.00U L= 0.80U
MP17 1 39 14 1 P1  W= 4.00U L= 0.80U
MP18 1 39 15 1 P1  W= 4.00U L= 0.80U
MP19 1 39 16 1 P1  W= 4.00U L= 0.80U
MP20 1 39 17 1 P1  W= 4.00U L= 0.80U
MP21 1 39 23 1 P1  W= 6.00U L= 0.80U
*
```

```
C2 1 0 488.467fF
C3 3 0 348.691fF
C4 4 0 59.881fF
C5 5 0 36.768fF
C6 6 0 18.736fF
C7 7 0 54.800fF
C8 8 0 36.768fF
C9 9 0 54.800fF
C10 10 0 36.768fF
C11 11 0 18.736fF
C12 12 0 18.736fF
C13 13 0 57.257fF
C14 14 0 54.800fF
C15 15 0 54.800fF
C16 16 0 36.768fF
C17 17 0 36.768fF
C18 1 0 98.935fF
C19 1 0 111.367fF
C20 1 0 123.799fF
C21 1 0 136.231fF
C22 1 0 159.135fF
C23 23 0 61.276fF
C26 26 0  7.100fF
C27 27 0 14.660fF
C28 28 0  7.100fF
C29 29 0 14.660fF
C30 30 0 14.660fF
C31 31 0 22.220fF
C32 32 0  7.100fF
C33 33 0  7.100fF
C34 34 0 14.660fF
C35 35 0 14.660fF
C36 36 0 22.220fF
C37 37 0 22.220fF
C38 38 0 33.408fF
C39 39 0 40.102fF
C40 40 0 35.917fF
*
* n-MOS Model 3 :
* Standard
.MODEL N1 NMOS LEVEL=3 VTO=0.80 U0=0.060 TOX= 3.0E-9
+LD =-0.050U THETA=0.200 GAMMA=0.400
+PHI=0.700 KAPPA=0.010 VMAX=130.00K
+CGSO=200.0p CGDO=200.0p
*
* p-MOS Model 3:
* high speed pMOS
.MODEL P1 PMOS LEVEL=3 VTO=-1.10 U0=0.020 TOX= 3.0E-9
+LD =-0.050U THETA=0.200 GAMMA=0.400
+PHI=0.700 KAPPA=0.010 VMAX=100.00K
+CGSO=200.0p CGDO=200.0p
*
* Transient analysis
*
.TEMP 27.0
.TRAN 8.00PS 200.00N
.PROBE
.END
```

5.0. SIMULATION: 6-TO-1 LINE MULTIPLEXER

MicroWind simulation facilities are extremely powerful, simulation is based on spice parameters. There are a number of simulation options (see figure 5.0a), and many different ways of simulating circuit operation e.g. simulate on the layout, simulate graphically, etc.

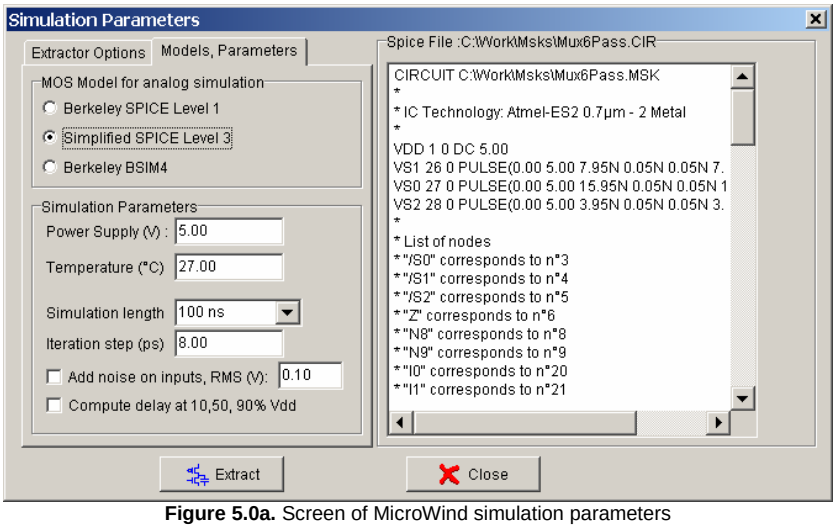


Figure 5.0a. Screen of MicroWind simulation parameters

This chapter contains complete simulation results along with a brief description for each of the 6-to1 line multiplexer designs shown in chapter 4. These simulations will demonstrate that the designs comply with the Boolean equation (4.0.1) for a 6-to1 multiplexer, the rise and fall time of waveforms, NMOS and PMOS transfer characteristics will also be looked at in detail.

5.1. Simulation of NMOS Pass Transistor design (mux6)

First setup clock inputs for the select lines S2..S0, so that they count up continuously in binary (000 → 111 → 000 etc…). Figures 5.1a to 5.1c show how the clock inputs were setup, notice that S1 is half the speed of S0 and S2 is half the speed of S1.

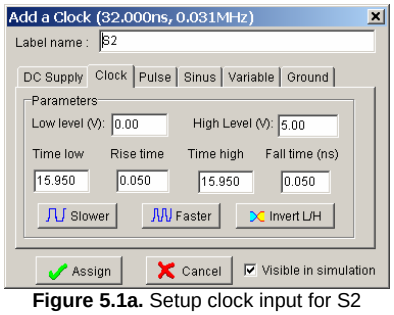


Figure 5.1a. Setup clock input for S2

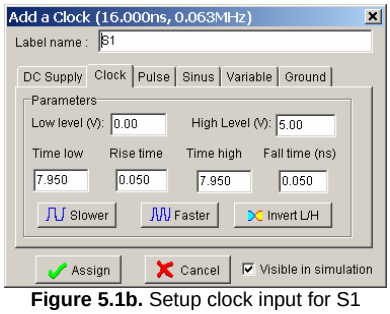


Figure 5.1b. Setup clock input for S1

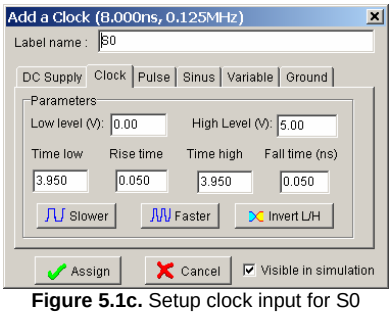


Figure 5.1c. Setup clock input for S0

All data lines (I5..I0) can now be easily tested. For example if I0 is connected to +VDD and all of the other data lines are connected to –VSS; the circuit can be graphically simulated showing that +VDD is connected to Z when S2 = 0, S1 = 0 and S0 = 0 (repeat until all data lines have been tested). The test can then be modified so that the data line being tested is set to –VSS and all others are set to +VDD, hence testing the multiplexer's zero passing abilities.

Figure 5.1d shows the simulated transfer characteristics of the NMOS pass transistors. This simulation is easily achieved in MicroWind by selecting [MOS characteristics] under the [Simulate] menu, using the mouse a transistor can be selected for simulation. Note all NMOS pass transistors in this design should have the same characteristics as they all have the same dimensions.

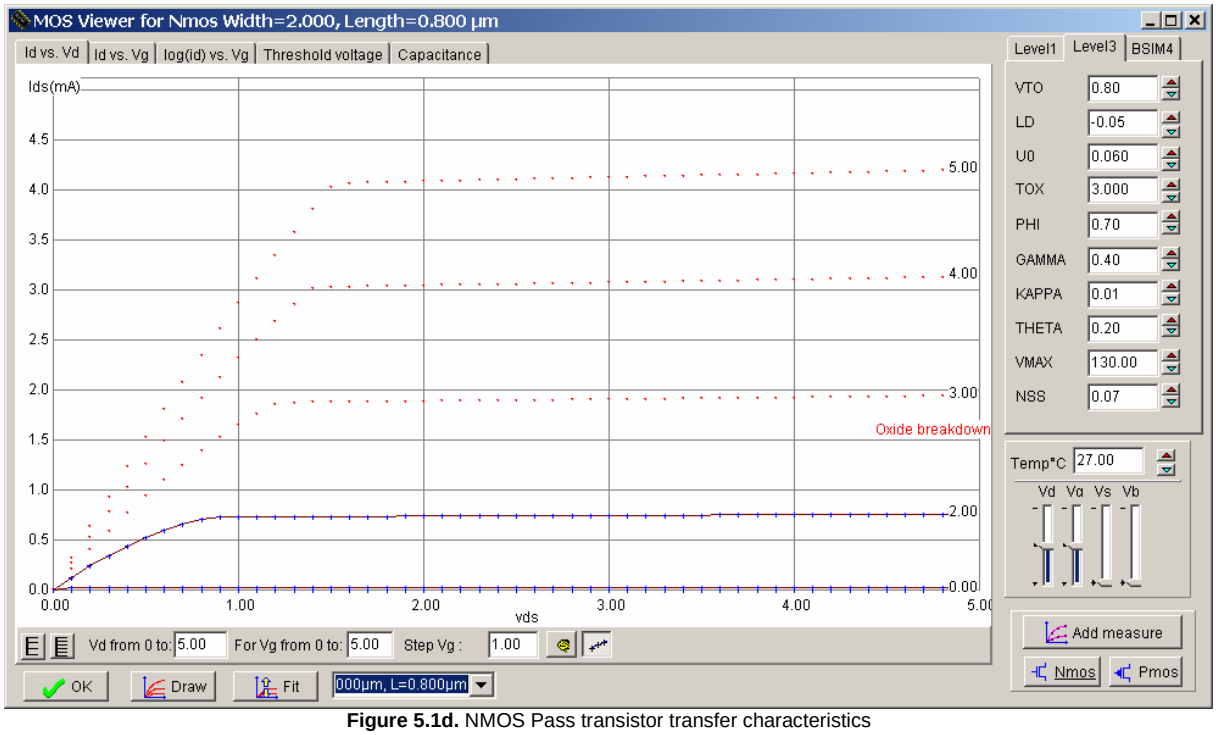


Figure 5.1d. NMOS Pass transistor transfer characteristics

5.1.1. Test Multiplexer's Operation and 5V Passing Ability

Figure 5.1.1a shows the simulation results of data line I0 connected to 5V and all others connected to 0V. Notice that the output Z is at a logic 1 when the select lines are at 000 as specified in the truth table figure 4.0b. Also notice that the logic '1' is only 3.483 volts and not 5 volts, proving that NMOS transistors are bad at passing logic '1's.

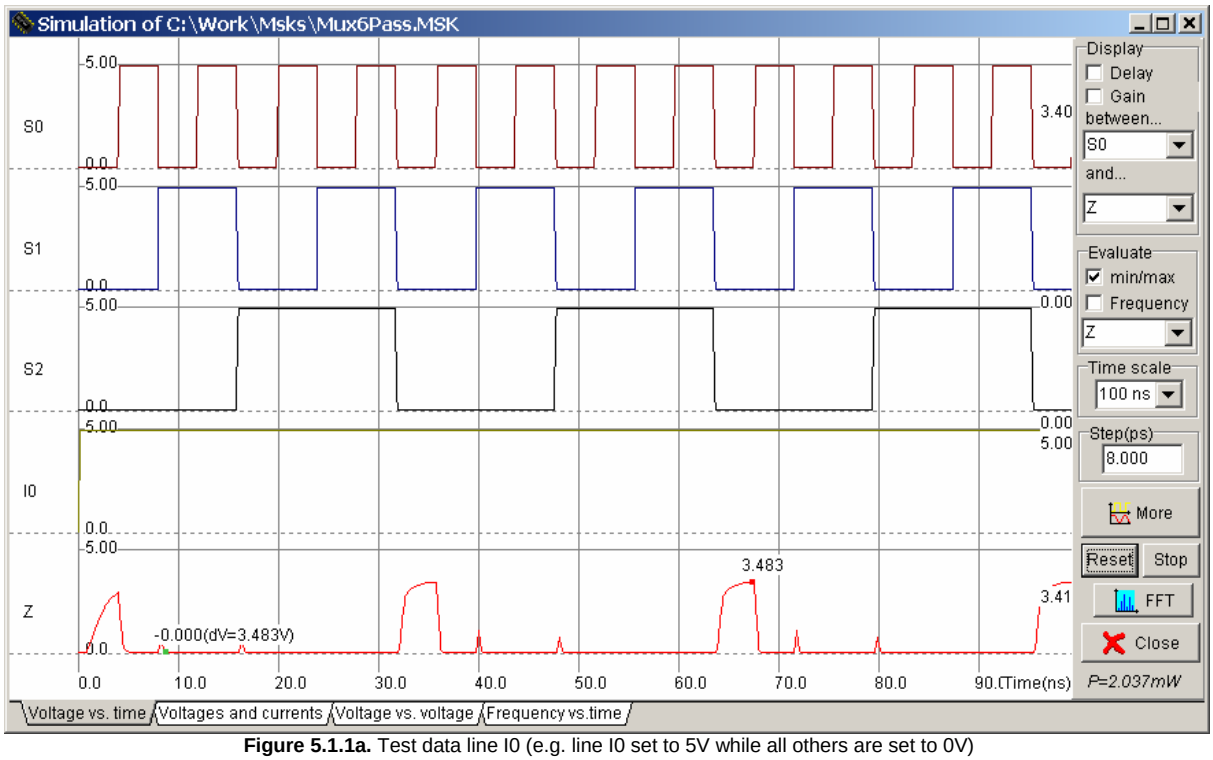


Figure 5.1.1a. Test data line I0 (e.g. line I0 set to 5V while all others are set to 0V)

Figure 5.1.1b shows the test results for data line I1, this time the output Z is at a logic '1' when the select lines are 001 as specified in the truth table figure 4.0b. Notice there are also a view glitches cause by metastability (transistors half ON / half OFF).

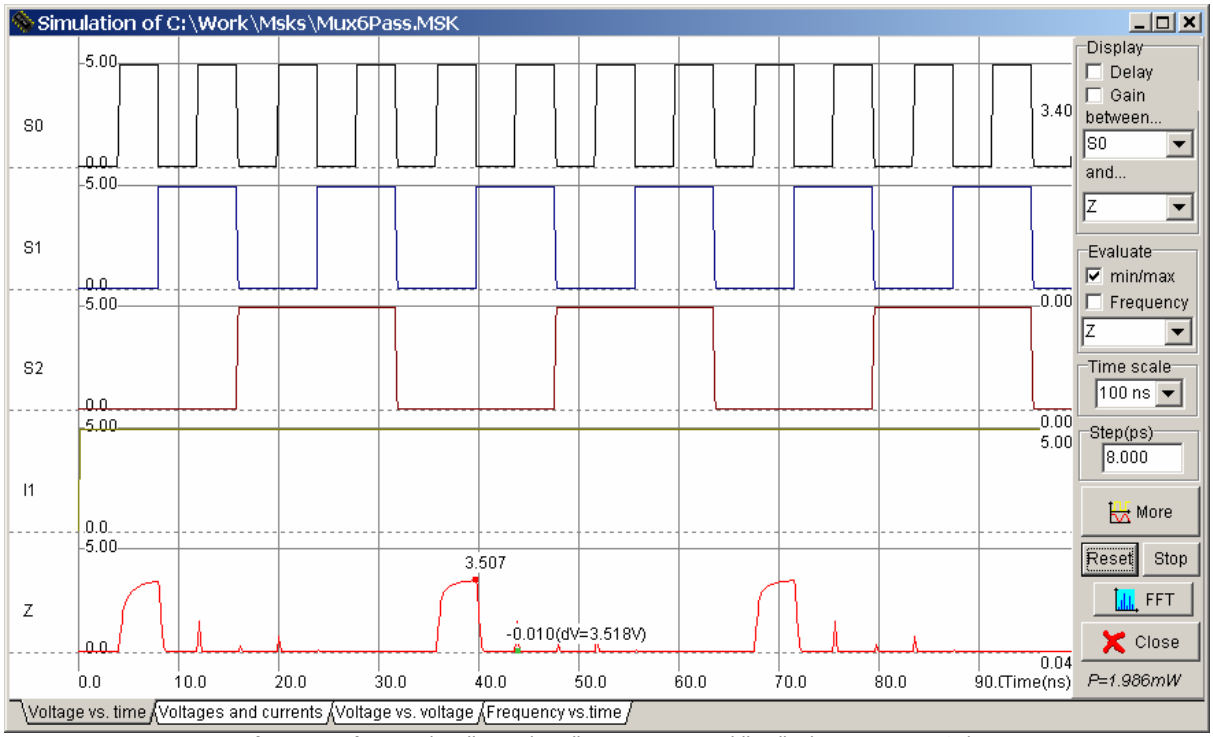


Figure 5.1.1b. Test data line I1 (e.g. line I1 set to 5V while all others are set to 0V)

Figure 5.1.1c shows the test results for data line I2, this time the output Z is at a logic '1' when the select lines are 010 as specified in the truth table figure 4.0b. Notice there is a small output when the unused select lines (110 & 111) are selected possibly due to capacitive effects.

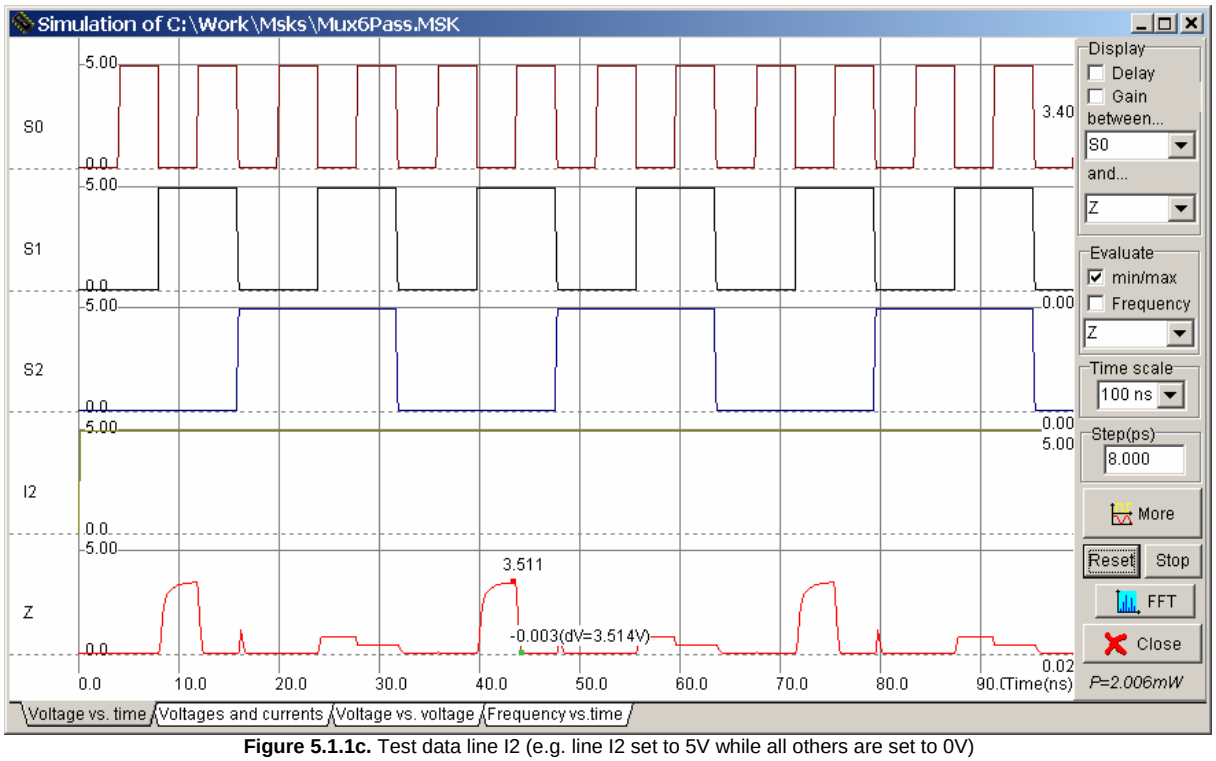


Figure 5.1.1c. Test data line I2 (e.g. line I2 set to 5V while all others are set to 0V)

Figure 5.1.1d shows the test results for data line I3, this time the output Z is at a logic '1' when the select lines are 011 as specified in the truth table figure 4.0b. Notice there is a small output when the unused select lines (110 & 111) are selected possibly due to capacitive effects.

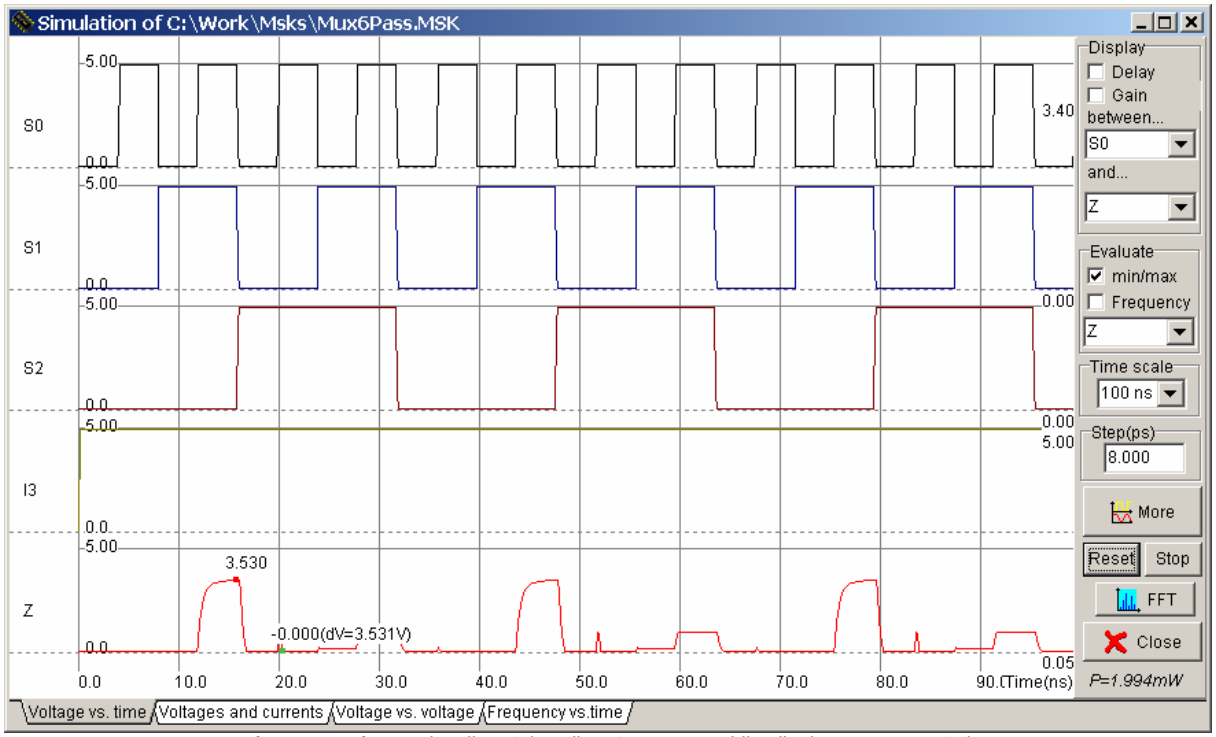


Figure 5.1.1d. Test data line I3 (e.g. line I3 set to 5V while all others are set to 0V)

Figure 5.1.1e shows the test results for data line I4, this time the output Z is at a logic '1' when the select lines are 100 as specified in the truth table figure 4.0b. Notice the output when the unused select lines (110 & 111) are selected has increased.

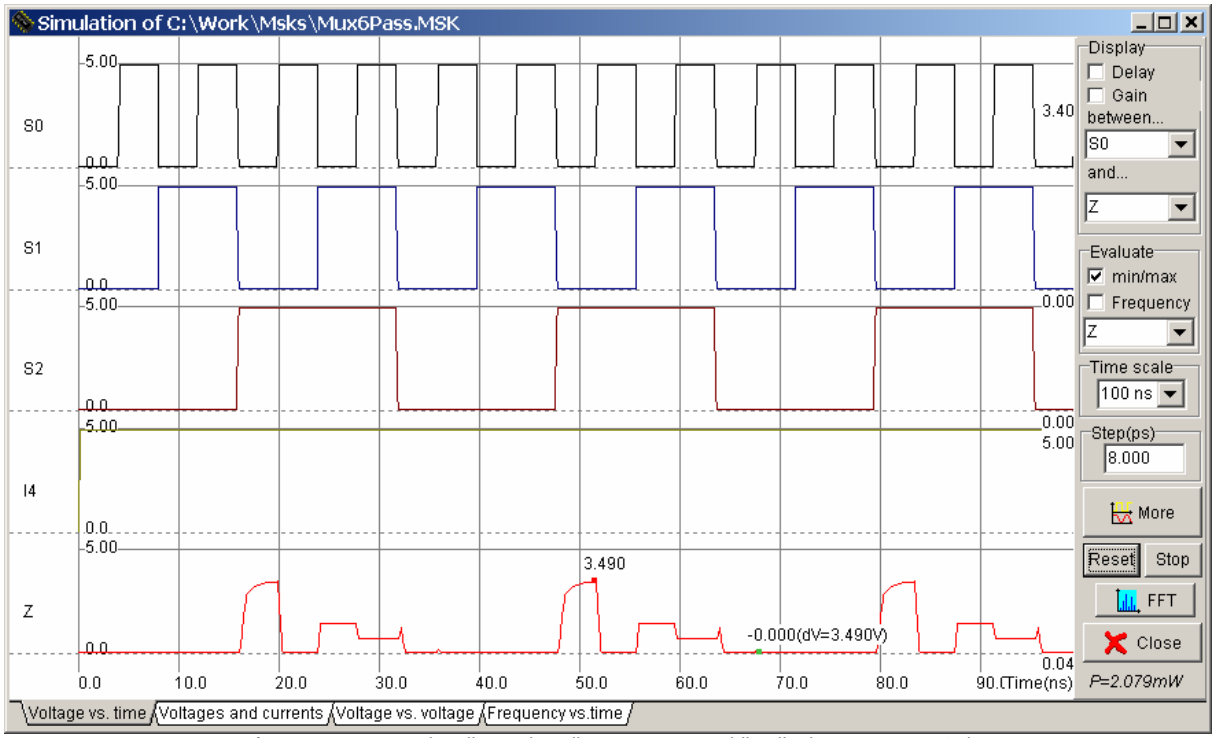


Figure 5.1.1e. Test data line I4 (e.g. line I4 set to 5V while all others are set to 0V)

Figure 5.1.1f shows the test results for data line I4, this time the output Z is at a logic '1' when the select lines are 101 as specified in the truth table figure 4.0b. Notice the output when the unused select lines (110 & 111) are selected has increased.

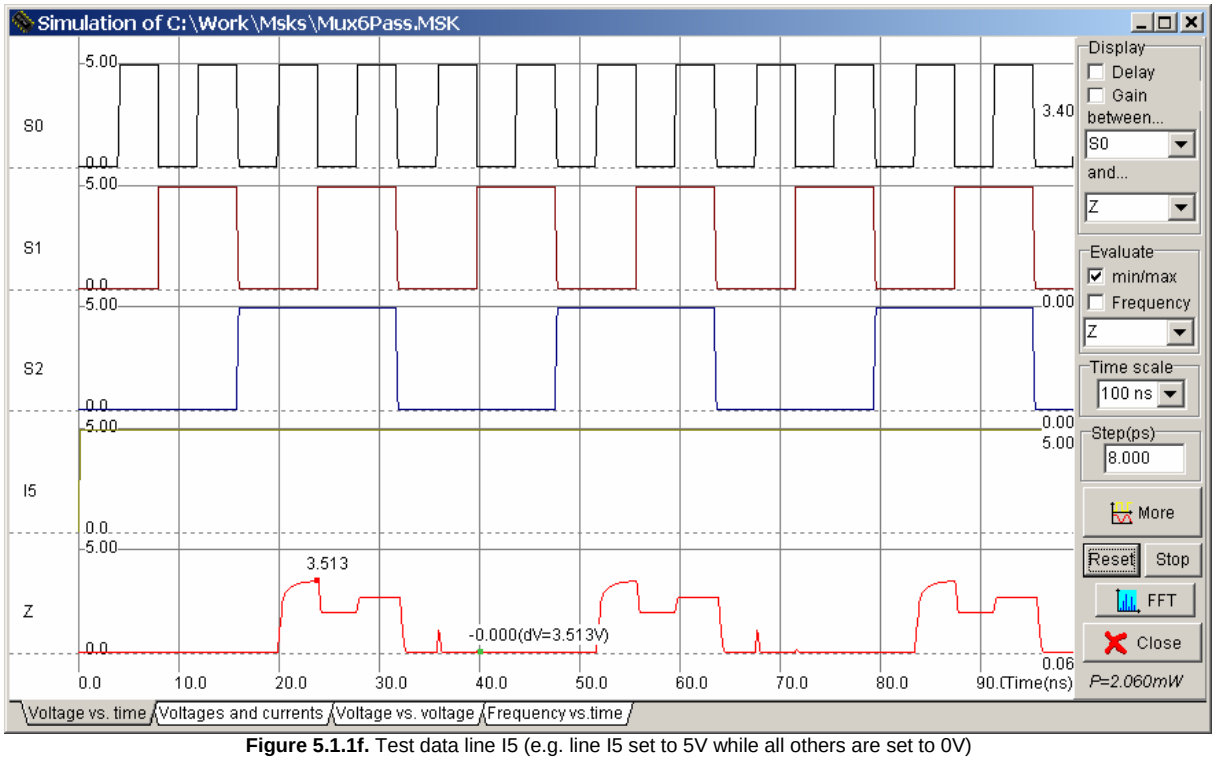


Figure 5.1.1f. Test data line I5 (e.g. line I5 set to 5V while all others are set to 0V)

Clearly state 110 and 111 is a problem; note these states should not be used. May be a good idea to use an 8-to-1 line multiplexer and don't use the last two states, or redesign the circuit to detect these unused states and connect the ground line to the output.

5.1.2. Test Multiplexer's Operation and 0V Passing Ability

Figures 5.1.2a to 5.1.2f shows the simulation results of passing zeroes through the multiplexer. Clearly NMOS pass transistors are good at passing logic '0's.

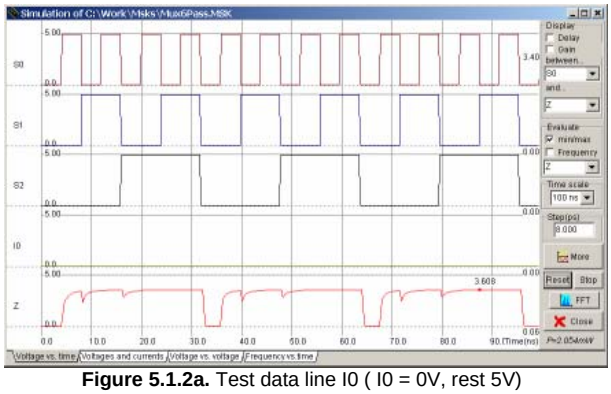


Figure 5.1.2a. Test data line I0 (I0 = 0V, rest 5V)

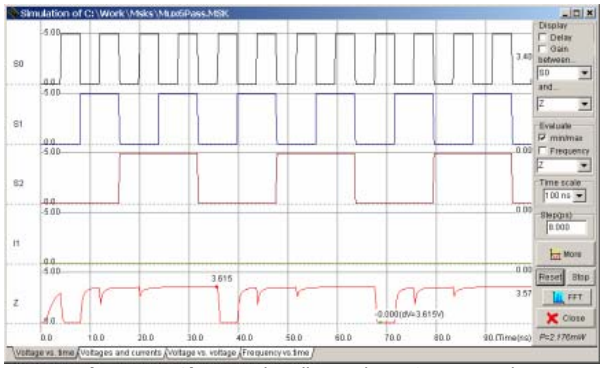


Figure 5.1.2b. Test data line I1 (I1 = 0V, rest 5V)

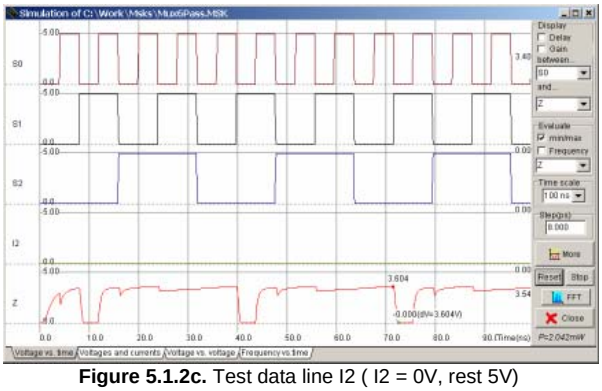


Figure 5.1.2c. Test data line I2 (I2 = 0V, rest 5V)

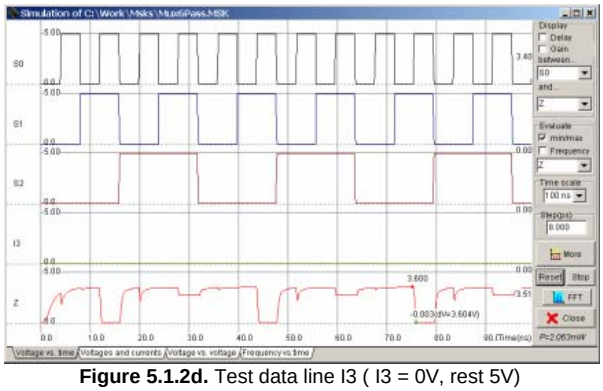


Figure 5.1.2d. Test data line I3 (I3 = 0V, rest 5V)

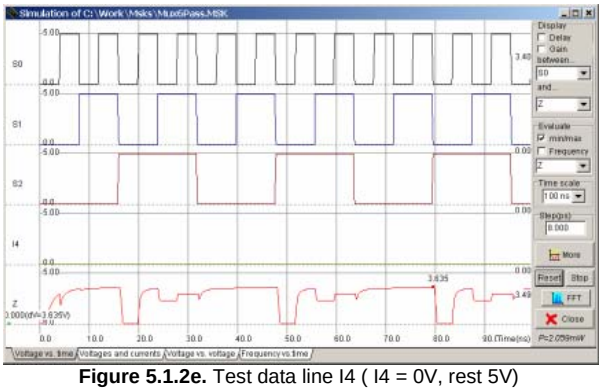


Figure 5.1.2e. Test data line I4 (I4 = 0V, rest 5V)

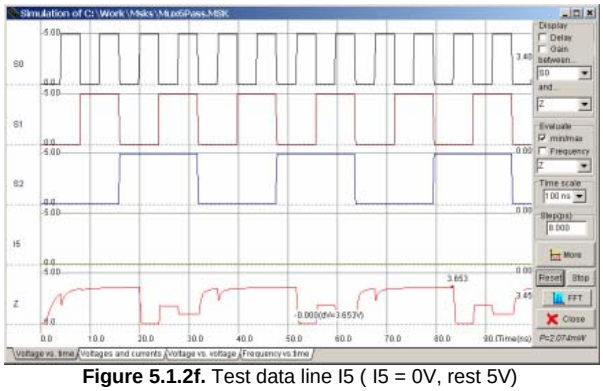


Figure 5.1.2f. Test data line I5 (I5 = 0V, rest 5V)

5.1.3. Simulate on Layout

Another useful MicroWind simulation feature is the ability to simulate on the layout, figure 5.1.3a shows a simulation carried out on the layout with S2 = 0, S1 = 0, S0 = 0, data line I0 connect to 5V and all others connected to 0V. Clearly as expected line I0 is connected to Z, as all three series transistors are ON.

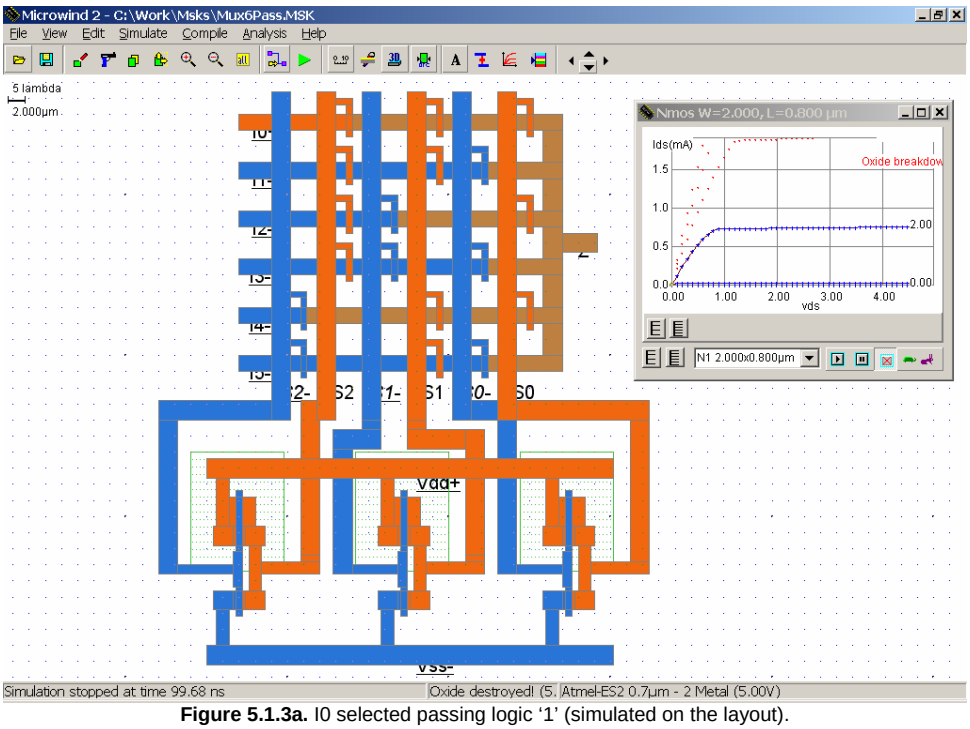


Figure 5.1.3a. I0 selected passing logic '1' (simulated on the layout).

Figure 5.1.3b shows a simulation carried out on the layout with S2 = 0, S1 = 0, S0 = 0, data line I0 connect to 0V and all others connected to 5V. Clearly as expected line I0 is connected to Z, as all three series transistors are ON.

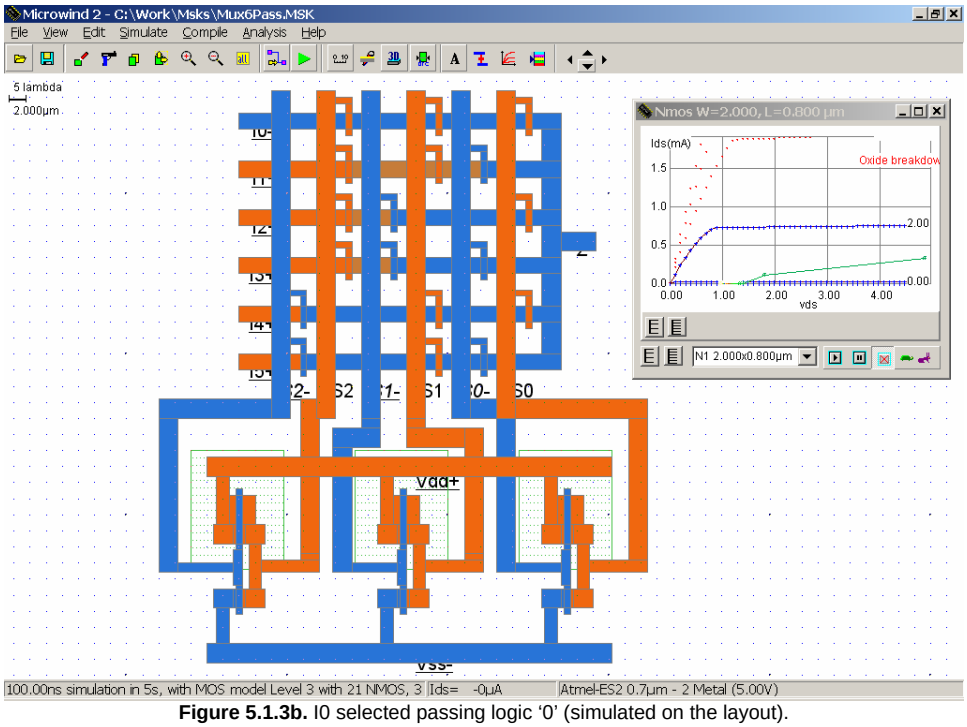


Figure 5.1.3b. I0 selected passing logic '0' (simulated on the layout).

5.1.4. Rise and Fall Time

Figure 5.1.4a shows the multiplexer switching rise and fall time. The rise time is about 808ps and the fall time is about 140ps.

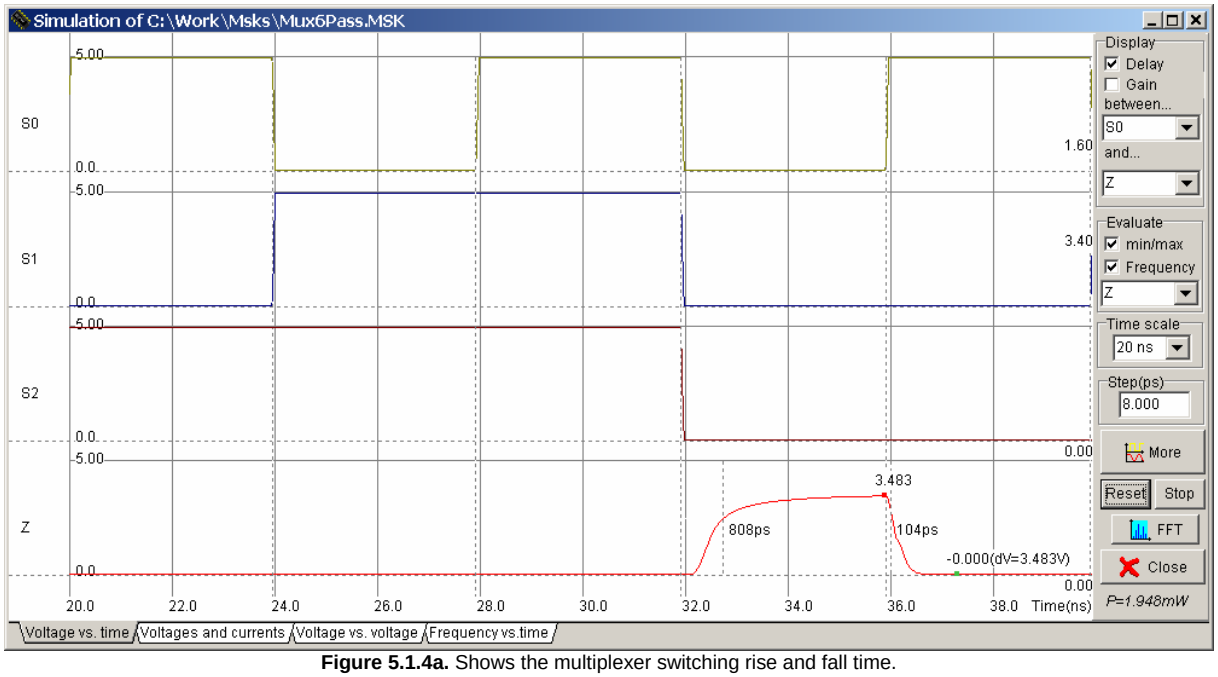


Figure 5.1.4a. Shows the multiplexer switching rise and fall time.

Figure 5.1.4b shows the rise and fall time of data line I0. The rise time is about 968ps and the fall time is about 264ps.

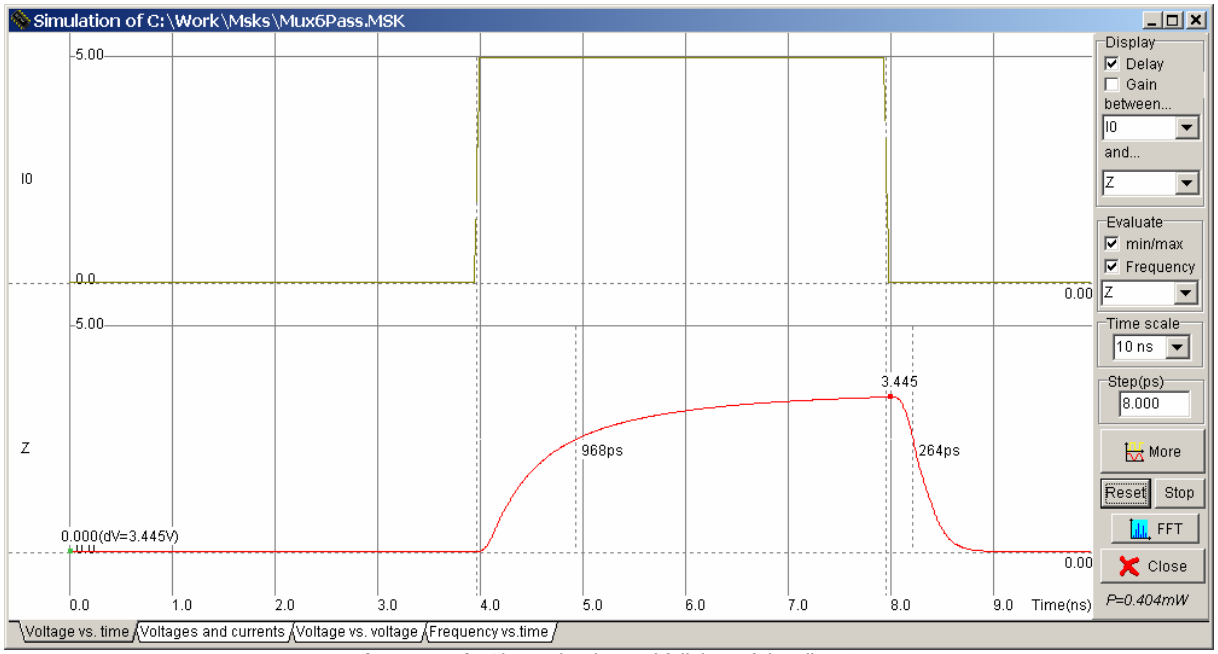


Figure 5.1.4b. Shows the rise and fall time of data line I0.

5.2. Simulation of PMOS Pass Transistor Design (mux6)

Figure 5.2a shows the simulated transfer characteristics of the PMOS pass transistors. This simulation is easily achieved in MicroWind by selecting [MOS characteristics] under the [Simulate] menu, using the mouse a transistor can be selected for simulation. Note all PMOS pass transistors in this design should have the same characteristics as they all have the same dimensions.

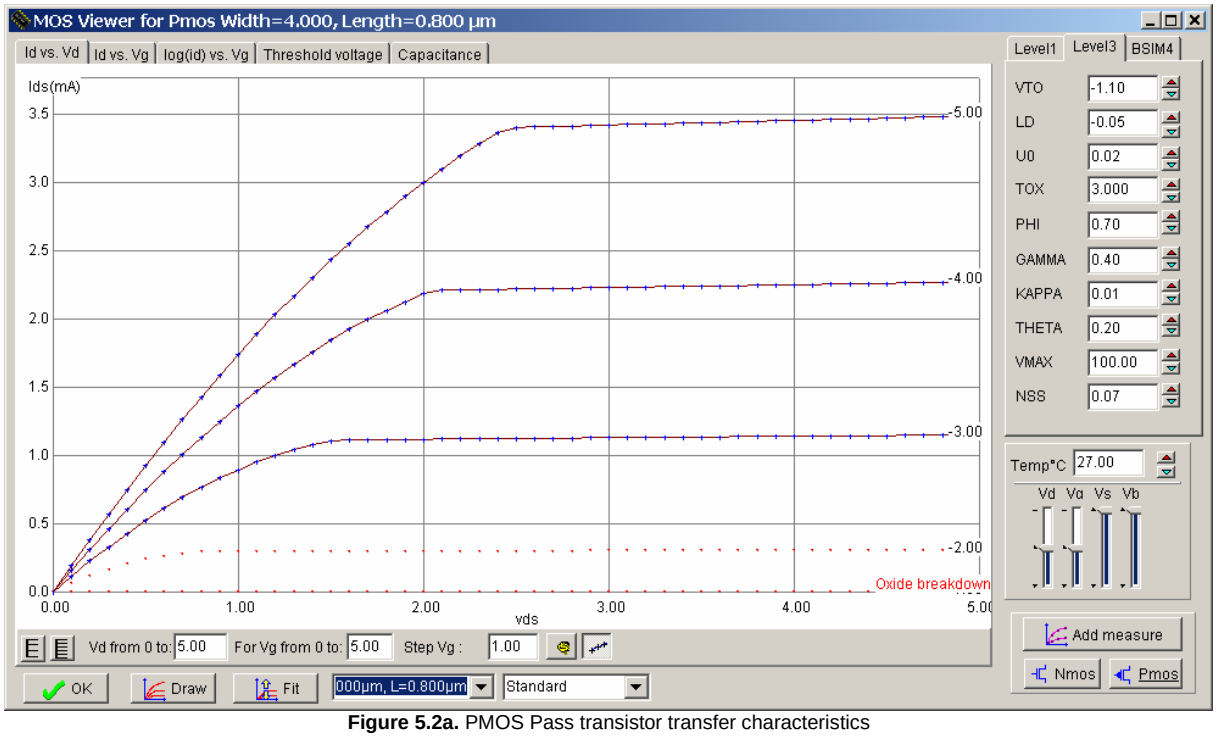


Figure 5.2a. PMOS Pass transistor transfer characteristics

5.2.1. Test Multiplexer's Operation and 5V Passing Ability

Setup clock inputs for the select line S2..S0, so that they count up continuously in binary (000 → 111 → 000 etc…) as was done for the simulation of the NMOS pass transistor design (see figures 5.1a to 5.1c). All data lines can now be easily tested, using the same technique used for the NMOS design.

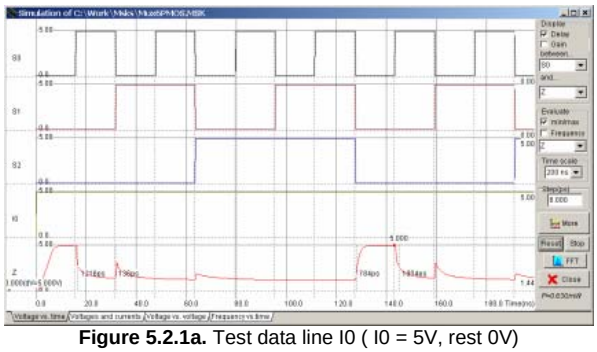


Figure 5.2.1a. Test data line I0 (I0 = 5V, rest 0V)

Figure 5.2.1b. Test data line I1 (I1 = 5V, rest 0V)

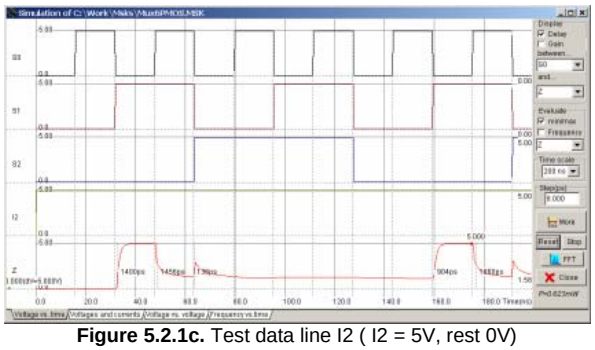


Figure 5.2.1c. Test data line I2 (I2 = 5V, rest 0V)

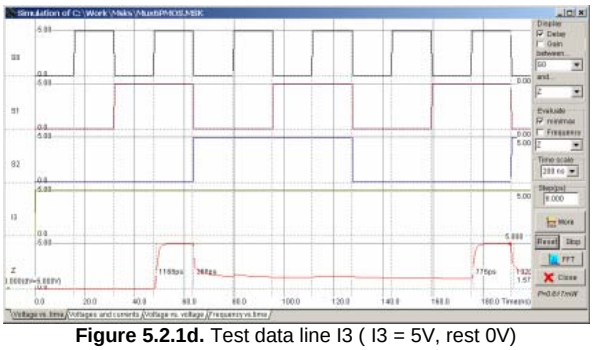


Figure 5.2.1d. Test data line I3 (I3 = 5V, rest 0V)

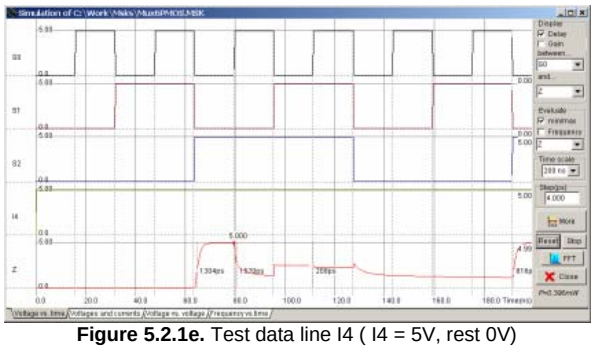


Figure 5.2.1e. Test data line I4 (I4 = 5V, rest 0V)

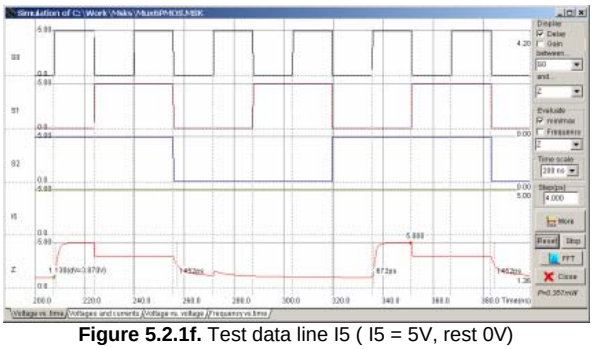


Figure 5.2.1f. Test data line I5 (I5 = 5V, rest 0V)

Figures 5.2.1a to 5.2.1f clearly demonstrate the operation of the multiplexer, notice that PMOS transistors are good at passing logic '1's, and bad at passing logic '0's which is the opposite to NMOS transistors. The unused select line addresses (110 & 111) produced logic '1' when testing data line I4 and I5 (see figure 5.2.1e and 5.2.1f), even through no data line is connected to Z during this period, this is properly due to capacitive effects.

5.2.2. Test Multiplexer's Operation and 0V Passing Ability

Figures 5.2.2a to 5.2.2f show the simulation results of passing zeros through the multiplexer. Clearly PMOS pass transistors are bad at passing '0's. The unused select line address (110 & 111) produced logic '0' when testing data line I4 and I5 (see figure 5.2.2e and 5.2.2f), even through no data line is connected to Z during this period, this is properly due to capacitive effects. Clearly these capacitive effects seem to be directly linked to line I4 and line I5.

Figure 5.2.2a. Test data line I0 (I0 = 0V, rest 5V)

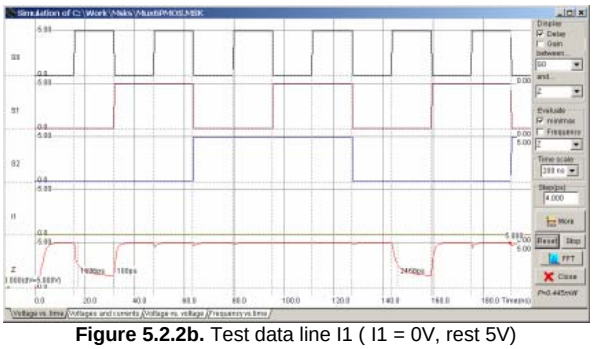


Figure 5.2.2b. Test data line I1 (I1 = 0V, rest 5V)

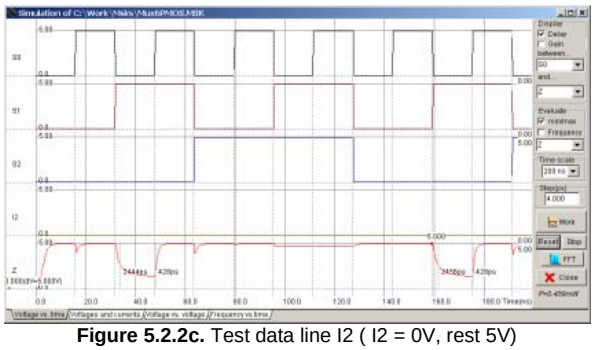


Figure 5.2.2c. Test data line I2 (I2 = 0V, rest 5V)

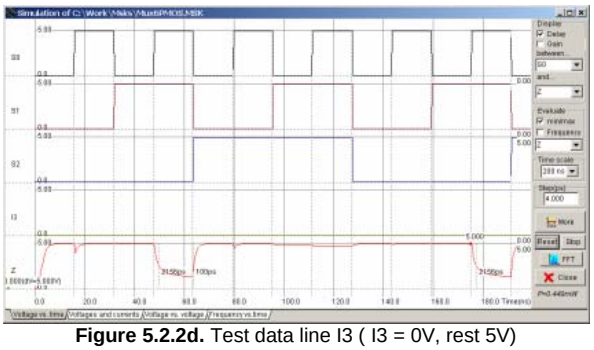


Figure 5.2.2d. Test data line I3 (I3 = 0V, rest 5V)

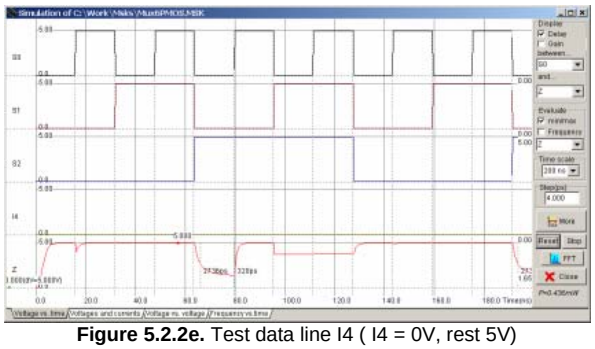


Figure 5.2.2e. Test data line I4 (I4 = 0V, rest 5V)

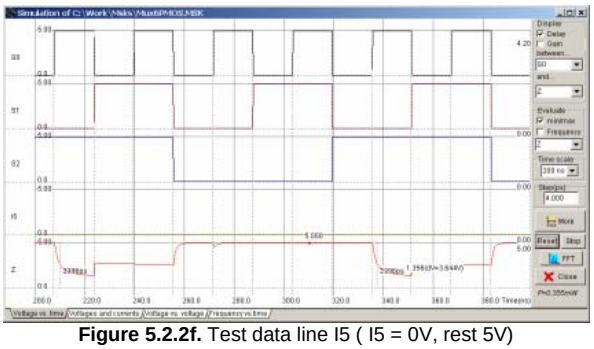


Figure 5.2.2f. Test data line I5 (I5 = 0V, rest 5V)

5.2.3. Simulate on Layout

Figure 5.2.3a shows a simulation carried out on the layout with S2 = 0, S1 = 0, S0 = 0 (line I0 selected), data line connected to 5V and all other data lines connected to 0V. Clearly as expect line I0 is connected to Z, as all three series transistors are ON (active low).

Figure 5.2.3b shows another simulation carried out on the layout with S2 = 0, S1 = 0, S0 = 0 (line I0 selected), data line I0 connected to 0V and all other data lines connected to 5V. Clearly as expected line I0 is connected Z, as all three series transistors (active low) are ON.

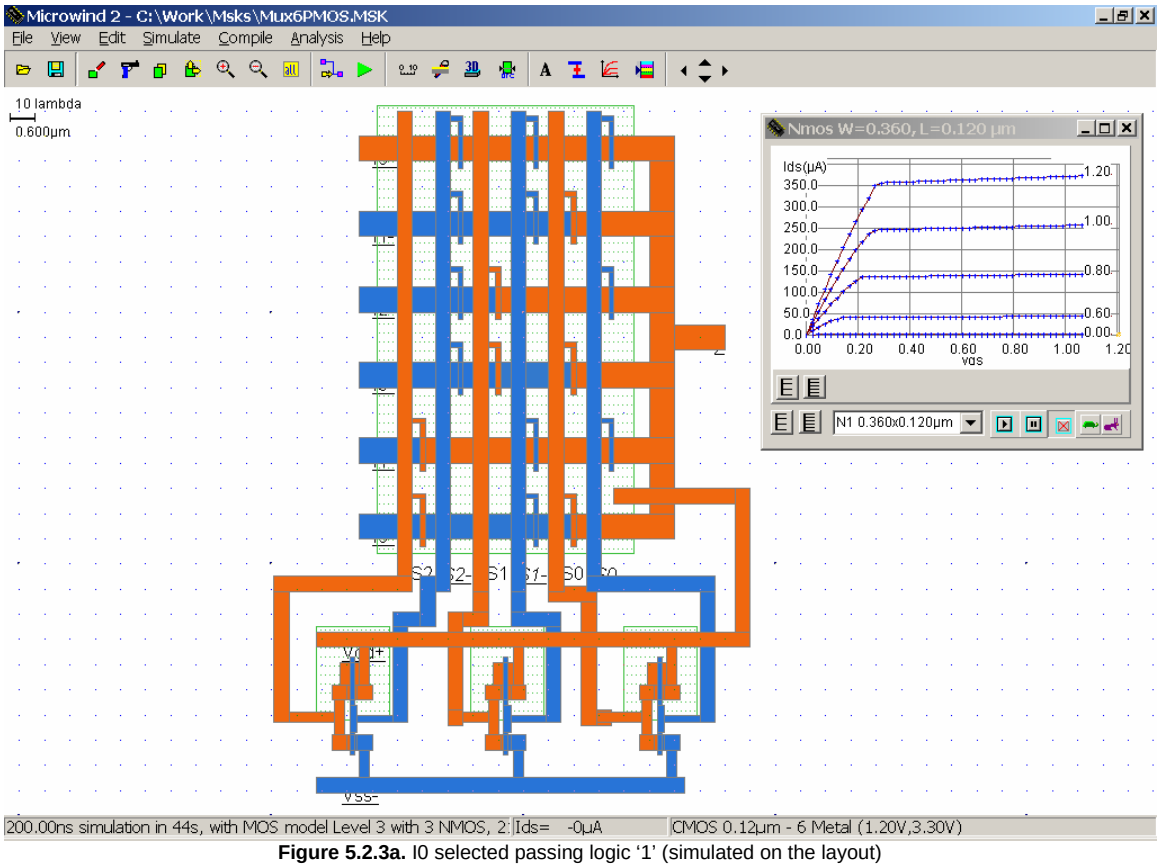


Figure 5.2.3a. I0 selected passing logic '1' (simulated on the layout)

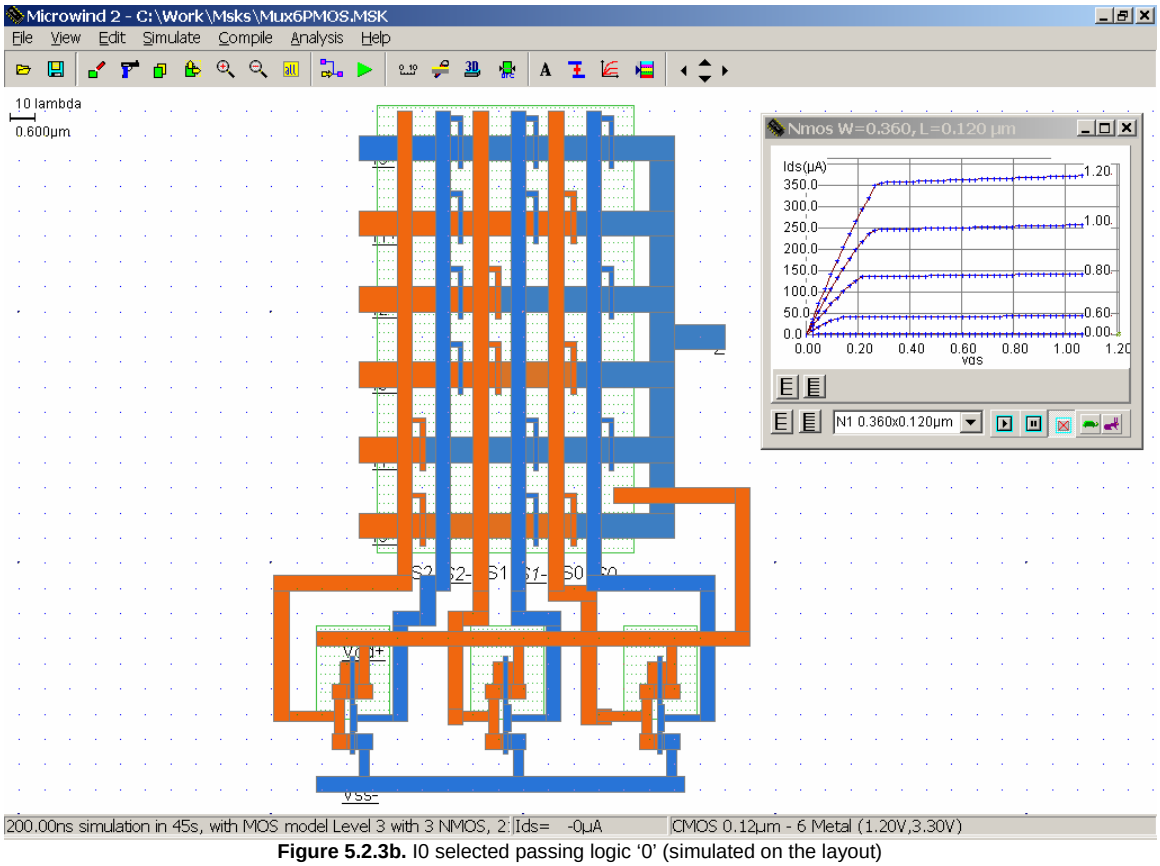


Figure 5.2.3b. I0 selected passing logic '0' (simulated on the layout)

5.2.4. Rise and Fall Time

Figure 5.2.4a shows the multiplexer switching rise and fall time. The rise time is about 728ps and the fall time is about 1944ps.

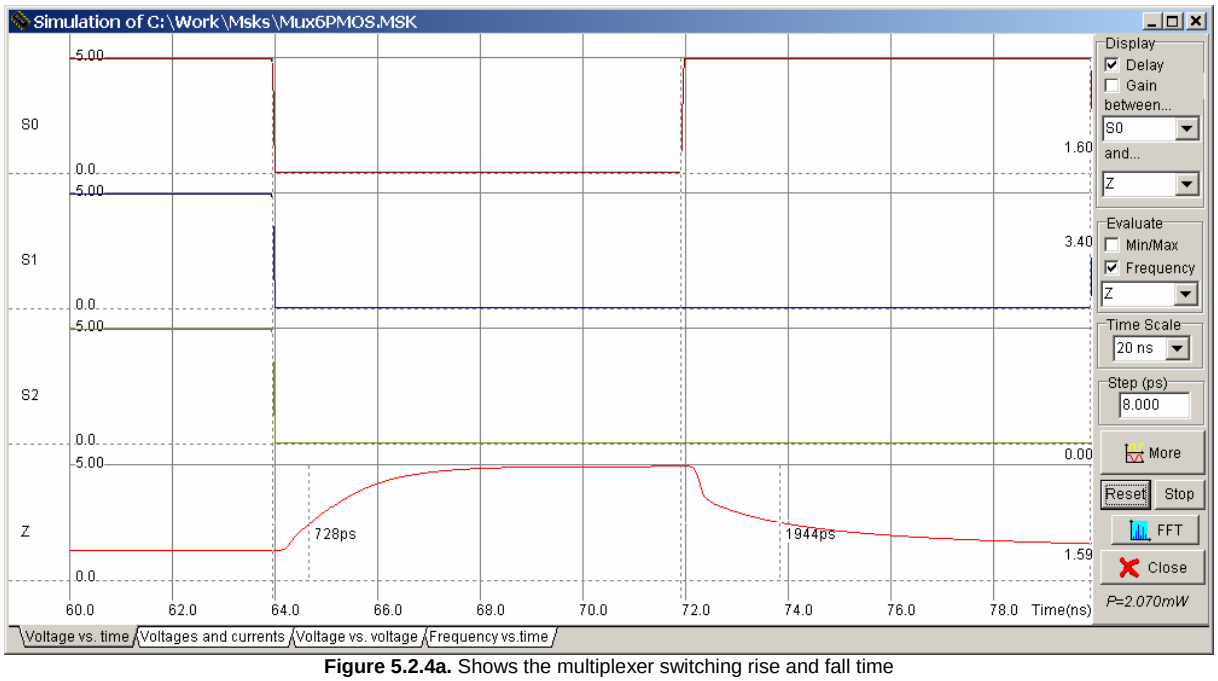


Figure 5.2.4a. Shows the multiplexer switching rise and fall time

Figure 5.2.4b shows the rise and fall time of data line I0. The rise time is about 576ps and the fall time is about 2680ps.

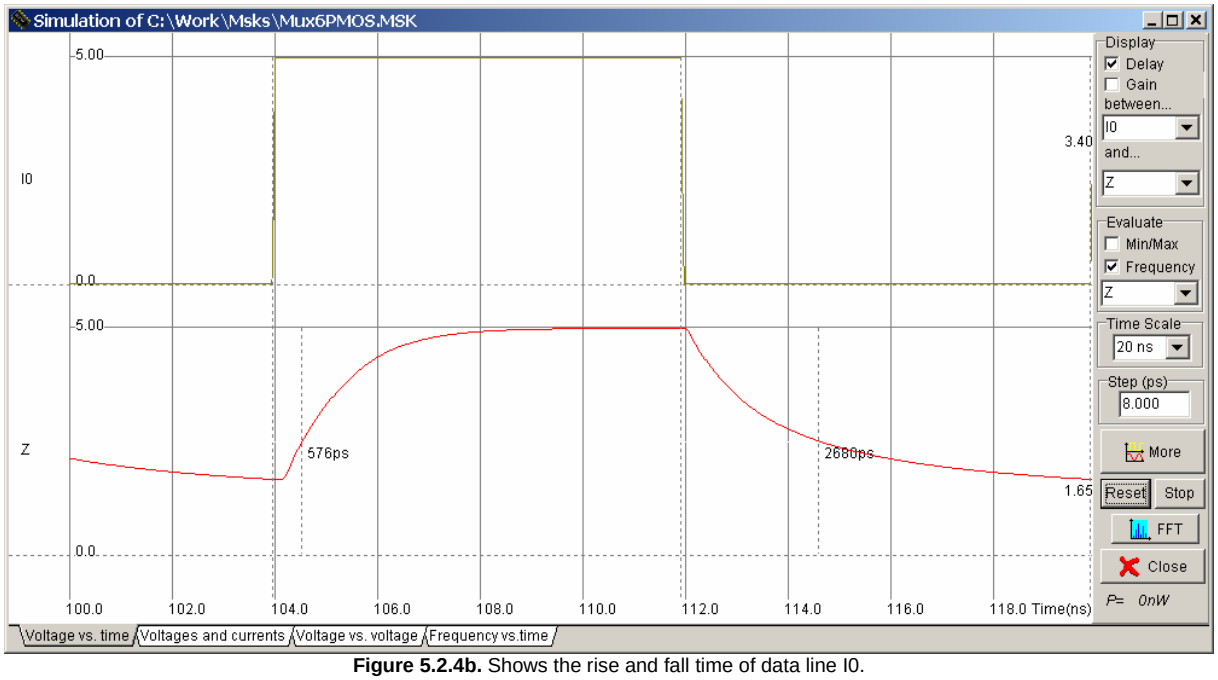


Figure 5.2.4b. Shows the rise and fall time of data line I0.

5.3. Simulation of CMOS Transmission Gate Design (mux6)

NMOS transistor dimensions are the same as that in the NMOS pass transistor design, hence the NMOS transistor transfer characteristics of both are identically (see figure 5.1d for NMOS transfer characteristics). PMOS transistor dimensions are the same as that in the PMOS pass transistor design, hence the PMOS transistor transfer characteristics of both are identically (see figure 5.2a for PMOS transfer characteristics).

5.3.1. Test Multiplexer's Operation and 5V Passing Ability

Setup clock inputs for the select lines S2...S0, so that they count up continuously in binary (000 → 111 → 000 etc…) as was done for the simulation of both the NMOS & PMOS pass transistor designs (see figures 5.1a to 5.1c). All data lines can now be easily tested, using the same technique used for both the NMOS & PMOS designs.

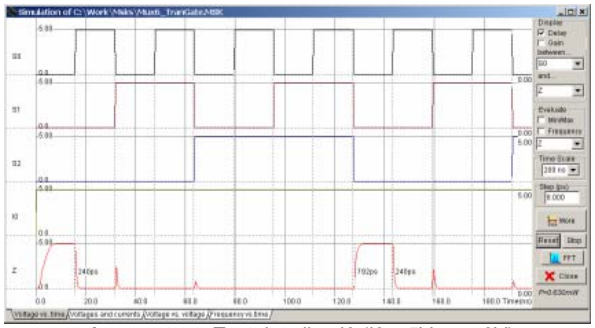


Figure 5.3.1a. Test data line I0 (I0 = 5V, rest 0V)

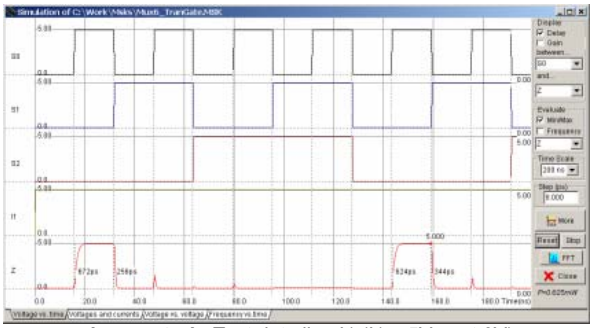


Figure 5.3.1b. Test data line I1 (I1 = 5V, rest 0V)

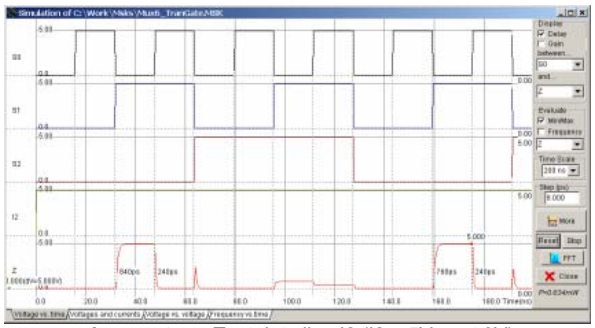


Figure 5.3.1c. Test data line I2 (I2 = 5V, rest 0V)

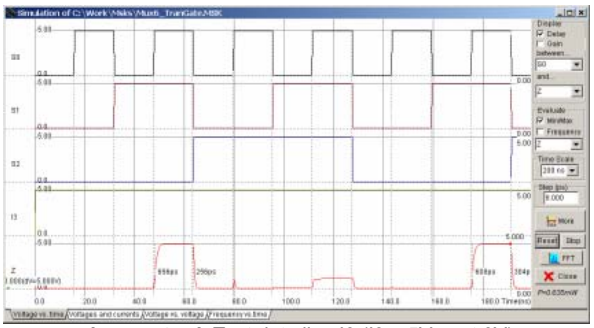


Figure 5.3.1d. Test data line I3 (I3 = 5V, rest 0V)

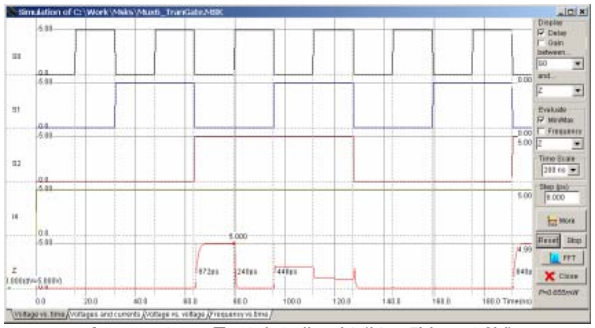


Figure 5.3.1e. Test data line I4 (I4 = 5V, rest 0V)

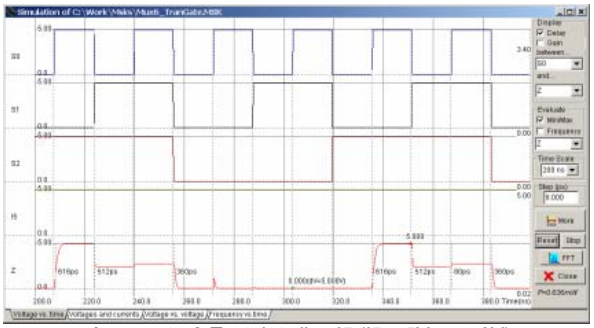


Figure 5.3.1f. Test data line I5 (I5 = 5V, rest 0V)

Figures 5.3.1a to 5.3.1f clearly demonstrate the operation of the multiplexer. Notice that the CMOS transmission gate arrangement produces good logic '1's and good logic '0's. The unused select line addresses (110 & 111) produce a bad logic '1' when testing line I4 and I5 (see figures 5.3.1e and 5.3.1f), even through no data line is connected to Z during this period, this is properly due to capacitive effects. There are also a view metastability (transistor half ON / haft OFF) glitches, for example look at figure 5.3.1a were there is a small glitch at 30nS, this occurs when S1 goes from LOW to HIGH and S0 goes from HIGH

to LOW, clearly for a short period of time both are LOW, hence I0 is connected to Z for a short period of time (as S2 is also LOW at this period in time).

5.3.2. Test Multiplexer's Operation and 0V Passing Ability

Figures 5.3.2a to 5.3.2f show the simulation results of passing zeros through the multiplexer. Clearly the CMOS transmission gate arrangement is good at passing logic '1's and logic '0's. The unused select line addresses (110 & 111) produced a bad logic '1' when testing data line I4 and I5 (see figure 5.3.2e and 5.3.2f), even through no data line is connected to Z during this period, this is properly due to capacitive effects.

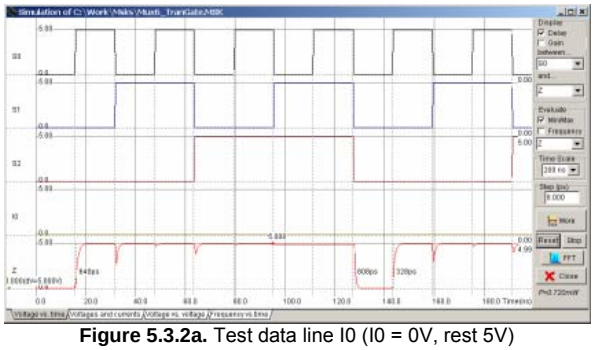


Figure 5.3.2a. Test data line I0 (I0 = 0V, rest 5V)

Figure 5.3.2b. Test data line I1 (I1 = 0V, rest 5V)

Figure 5.3.2c. Test data line I2 (I2 = 0V, rest 5V)

Figure 5.3.2d. Test data line I3 (I3 = 0V, rest 5V)

Figure 5.3.2e. Test data line I4 (I4 = 0V, rest 5V)

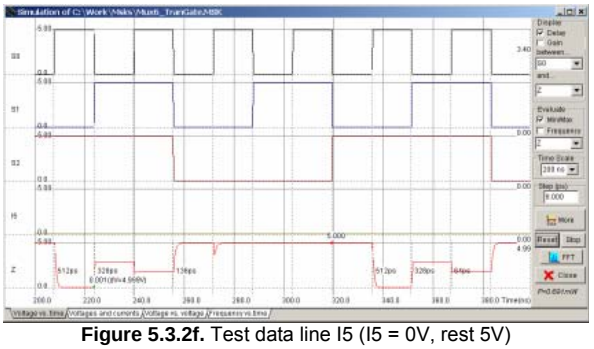


Figure 5.3.2f. Test data line I5 (I5 = 0V, rest 5V)

5.3.3. Fully Test Multiplexer's Operation in one Simple Simulation

The select lines cycle through all possible values as before, but this time each data line is set at a different voltage; I0 = 0V, I1 = 1V, I2 = 2V, I3 = 3V, I4 = 4V and I5 = 5V. Ignoring the unused select line addresses (110 & 111) figure 5.3.3a clearly demonstrates the full operation of the multiplexer. Clearly Z = 0V at 000, Z = 1V at 001, Z = 2V at 010, Z = 3V at 011, Z = 4V at 100 and Z = 5V at 101, proving the operation of the multiplexer.

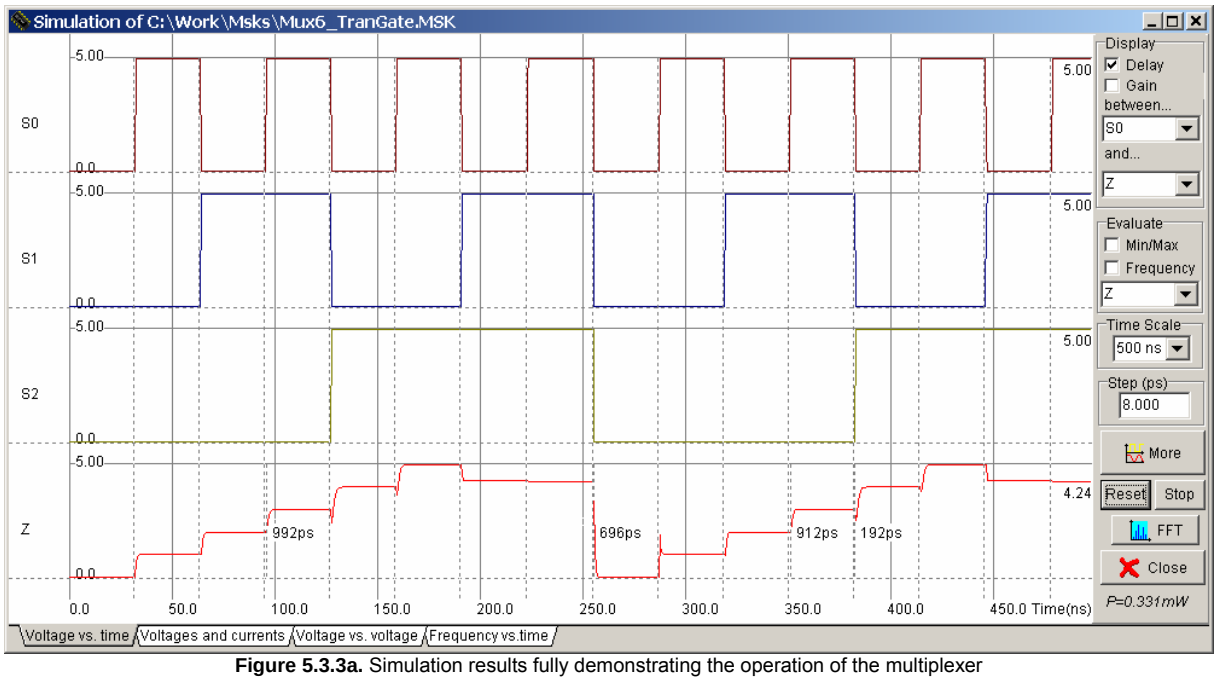


Figure 5.3.3a. Simulation results fully demonstrating the operation of the multiplexer

5.3.4. Simulate on Layout

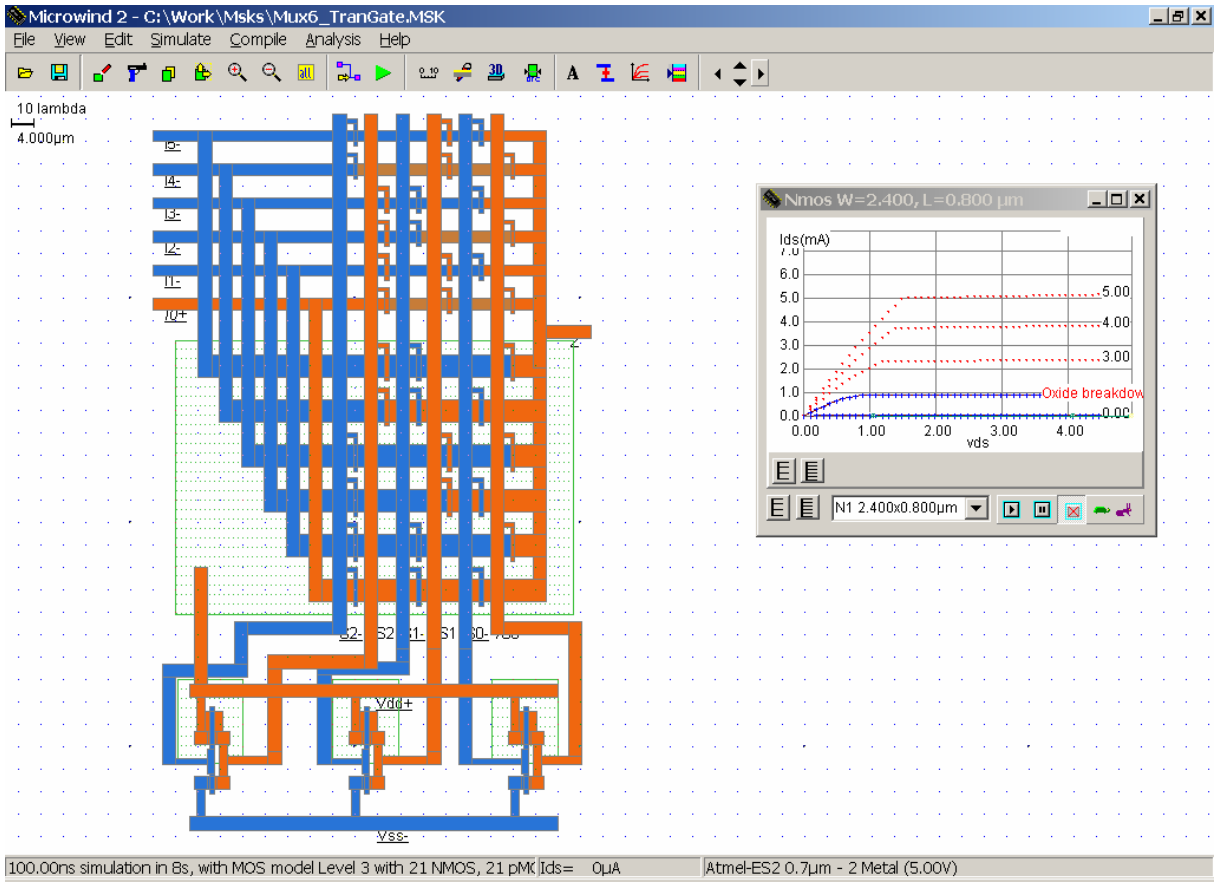


Figure 5.3.4a I0 selected passing logic '1' (simulated on the layout)

Figure 5.2.4a shows a simulation carried out on the layout with S2 = 0, S1 = 0, S0 = 0 (line I0 selected), data line connected to 5V and all other data lines connected to 0V. Notice that the three series NMOS transistors connected to line I0 are ON (active HIGH) and the three series PMOS transistors are also ON (active LOW). It is also clear that the PMOS transistors are being used to produce the good logic '1's; because the shading between the NMOS transistors indicates a bad '1', and the shading between the PMOS transistors indicate that a good '1' is being passed to Z.

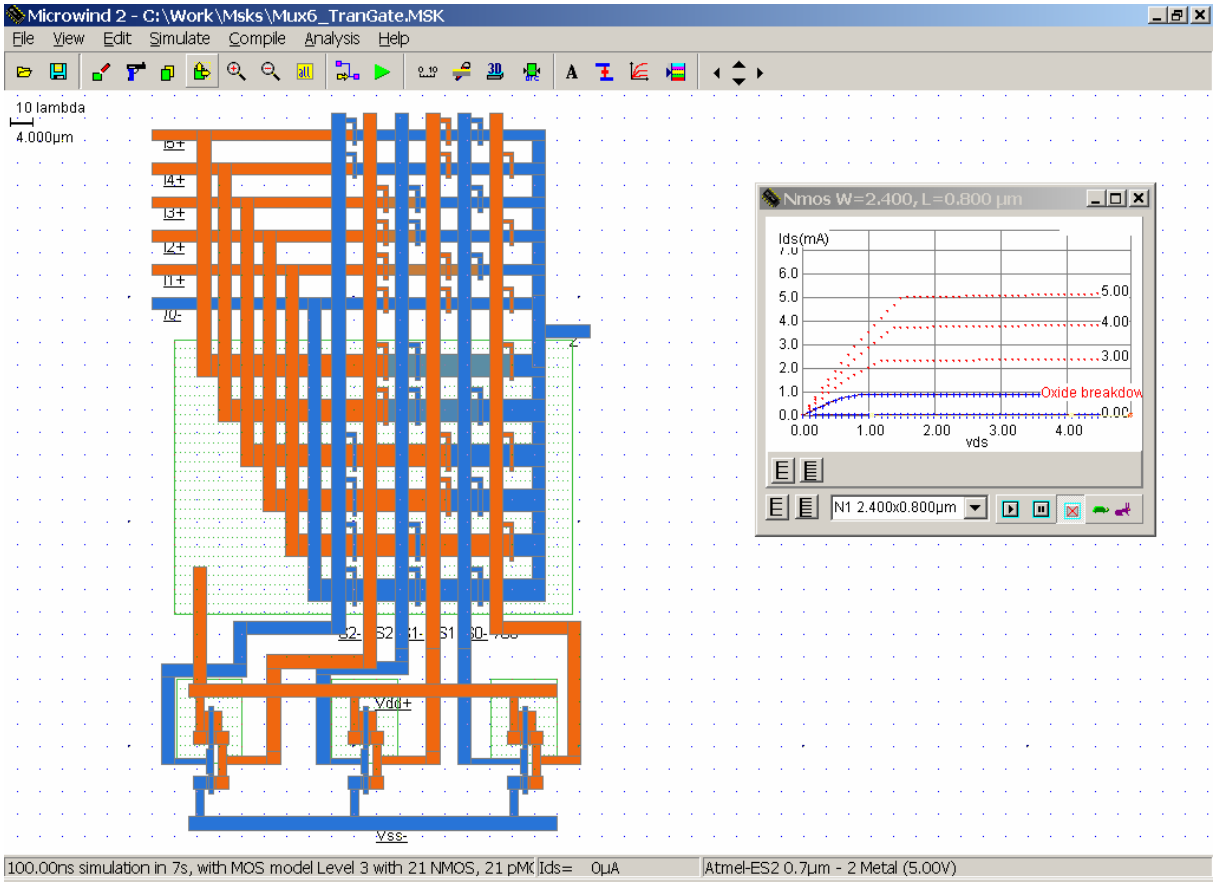


Figure 5.3.4b I0 selected passing logic '0' (simulated on the layout)

Figure 5.3.4b shows a simulation carried out on the layout with S2 = 0, S1 = 0, S0 = 0 (line I0 selected), data line connected to 0V and all other data lines connected to 5V.Clearly as expected line I0 is connected to Z, as the three series NMOS transistors are ON (active HIGH) and the three series PMOS transistors that are in parallel with the NMOS transistors are also ON (active LOW). NMOS transistors are used to generate a good logic '0', but unfortunately this cannot be easily seen here has the shading between the PMOS transistors looks the same as that of the ground rail (blue), but clearly this is the case as this was proven on other simulations.

Look closely at figure 5.3.4b, it shows something else happening that's very interesting; look at the PMOS transistors connected to line I5. It looks like the first PMOS transistor is leaking as there is a different shade of blue between the first PMOS and the last PMOS transistor (note the two end transistors are OFF). Also Notice that there appears to be no NMOS leakage, as the shading between the transistors is the same as the ground line.

5.3.5. Rise and Fall Time

Figure 5.3.5a shows the multiplexer switching rise and fall time. The rise time is about 800ps and the fall time is about 240ps.

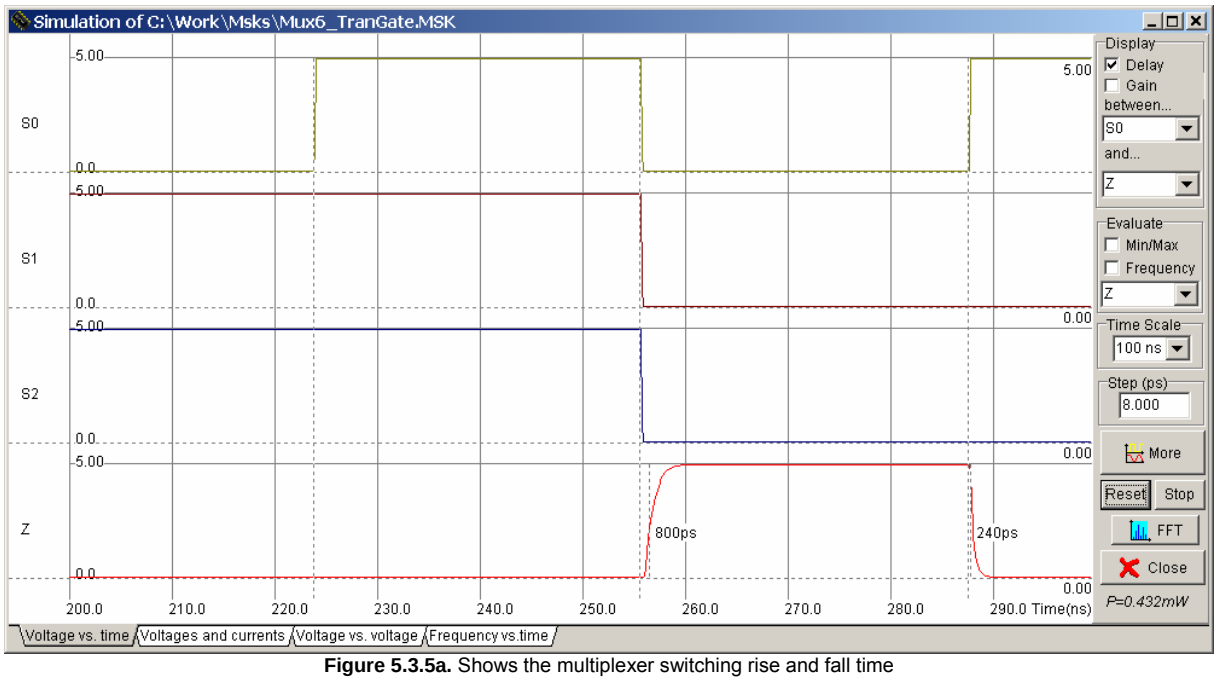


Figure 5.3.5a. Shows the multiplexer switching rise and fall time

Figure 5.3.5b shows the rise and fall time of data line I0. The rise time is about 656ps and the fall time is about 616ps.

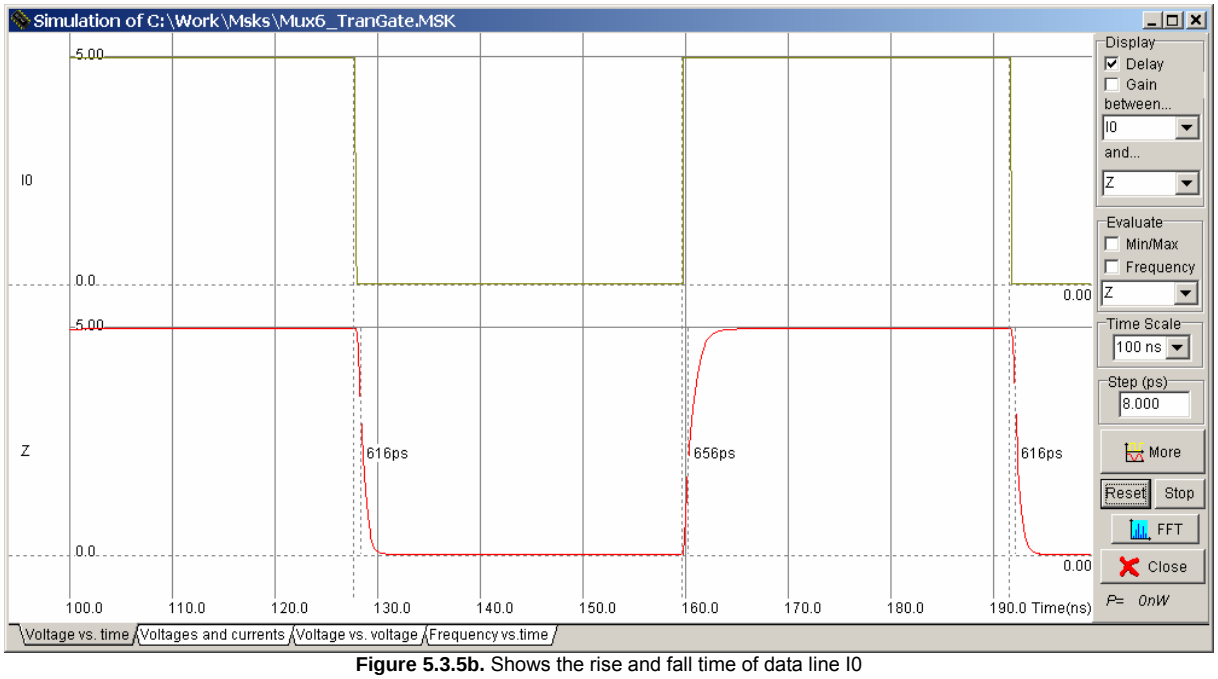


Figure 5.3.5b. Shows the rise and fall time of data line I0

5.3.6. Line Resistance and Capacitance

By double clicking on any line, the line will be highlighted and a useful dialog appears (called "Navigator"). This dialog contains extremely use information. Figure 5.3.6a shows some useful information about line S0, the capacitance of the line if 70.17fF, line resistance is 2.332 kΩ, line inductance is 0.05 nH and line length is 231 µm. One of the key things to notice is that the resistance of the polysilicon is 2329Ω, while the metal resistance is only 3Ω. Clearly this is a key design aspect, and it is good design practice to have short lengths of polysilicon because of the high resistance. Notice line S0 is connect to the inverter using a length of polysilicon, clearly line resistance would be decreased if metal 2 was used instead.

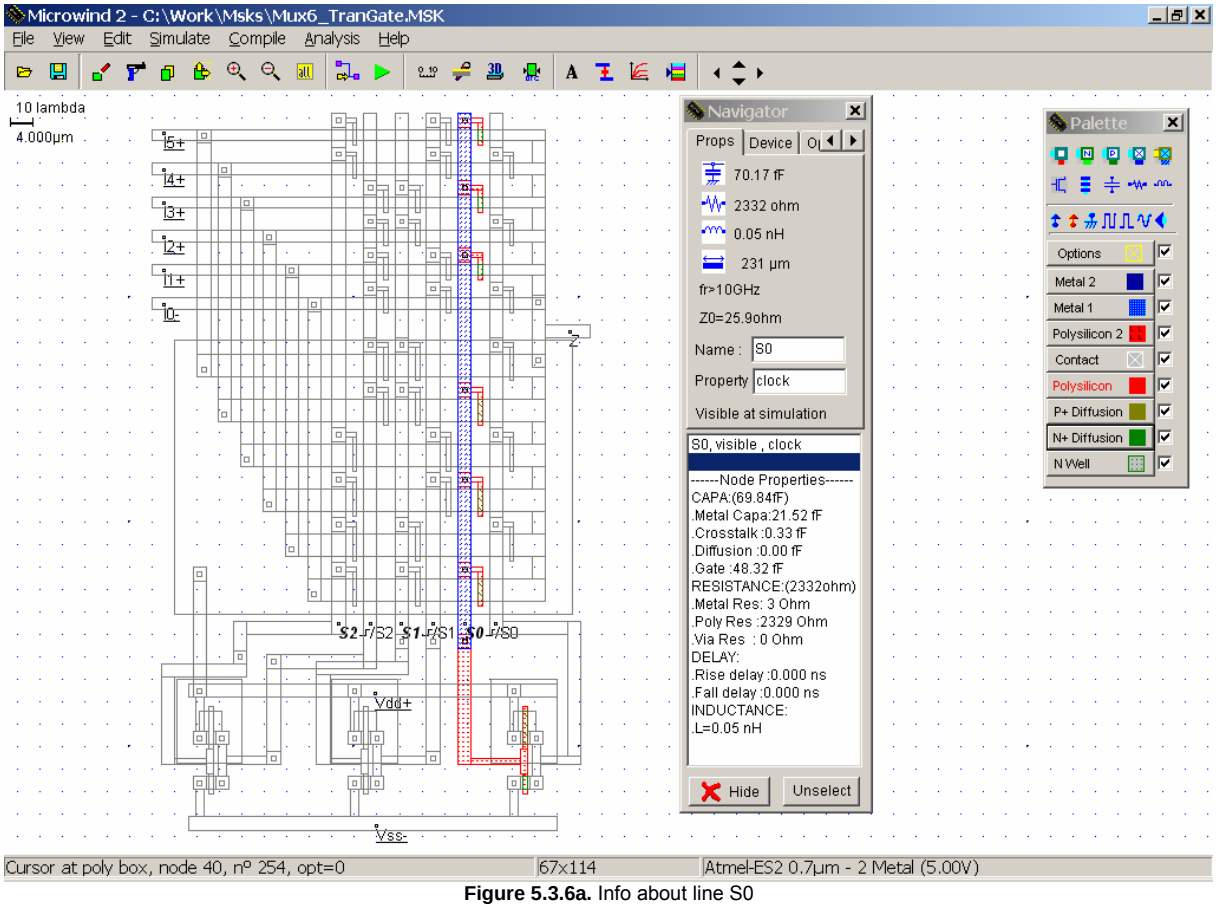


Figure 5.3.6a. Info about line S0

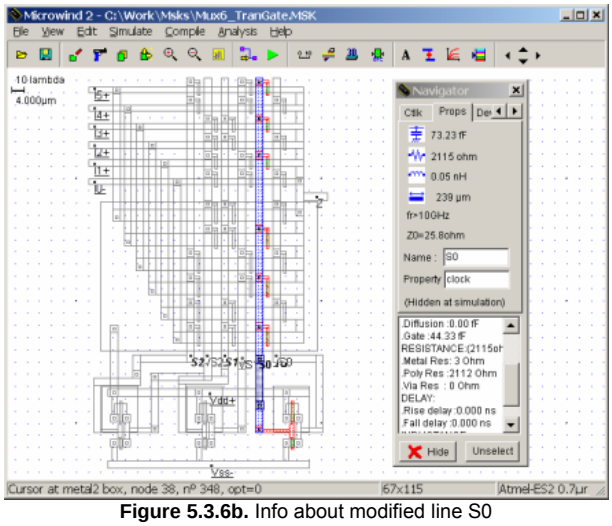


Figure 5.3.6b. Info about modified line S0

In Figure 5.4.6b line S0 has been modified by replacing the polysilicon with metal 2. Notice that line resistance is now only 2115Ω instead of 2332Ω that's a reduction of 217Ω (9.3%). Clearly it makes since to redesign all of the design in this report so that long lengths of polysilicon can be avoided.

Also notice that line capacitance has increased from 70.17 fF to 73.23 fF, that's an increase of 3.06 fF (4.2%).

6.0. CONCLUSIONS

The purpose of this assignment was to design and simulate a 6-to-1 line multiplexer using three different implementations; NMOS pass transistors, PMOS pass transistors and CMOS transmission gates. Clearly this objective has been achieved successfully, as it has been proven using simulation that each design works, although further development is required as there are limitations and flaws.

Obviously Boolean algebra laws and rules, make it possible to describe logic circuits as an expression. Boolean algebra simplification is extremely useful to the design engineer as it allows for the reduction of the number of logic gates, clearly there are many other ways to simplify logic expressions, one of the most popular is 'k' maps, by putting loops around adjacent cells a reduced logic expression can easily be found.

It is also clear that every combinational logic circuit can be formed entirely from NAND or NOR gates, this is achieved using De Morgan's Theorem, e.g. break the line and change the sign. The main reason why a designer would want to make their combinational logic circuit out for entirely NAND gates is to reduce cost. For example the 7400 TTL IC has four 2 input NAND gates, say the original circuit used two AND gates and an OR gate, this would require two ICs while if the circuit was converted to NAND gates using De Morgan's theorem only one 7400 TTL IC is required. But the purpose of this assignment is to design a customized IC (VLSI design) not to use commercially available ICs; hence it is not as important to design a circuit using a single type of logic gate, allow NAND gates are preferred to AND gates because of the lower transistor count. In fact quite often VLSI designed circuits are not made up of logic gates because this is wasteful of space and transistors, the simplest and most efficient (low transistor count) way of designing a circuit is to use pass transistors (in series AND, in parallel OR), but pass transistor cannot pass a good logic '1' and a good logic '0' hence CMOS transmission gates are normally used instead.

VLSI technologies are advancing at an incredible pace; transistors are getting smaller and smaller for example in 1988 1.2 million transistors were squeezed into one IC, but now it is possible to produce up-to 110 million transistors in a single IC, that's an increase of nearly 100 times in just 14 years. In fact it is expected for the transistor count to double every 2 years, for example in 1999 it was only possible to squeeze 42 millions transistors into a single IC, 2 years later it was possible to squeeze 110 million. Obviously modern PC microprocessor design is at the forefront of VLSI design; companies like AMD and Intel are the world leaders. There latest chips may be a view years behind in terms of the transistor count (AMD latest XP processor has 37.5 Million transistors see appendix 4), but in terms of performance there at the forefront of VLSI technology running at clock speeds above 2GHz, that just a view years ago was through to be impossible without using liquid nitrogen to cool the chip.

Clearly stick diagrams are a useful tool for the VLSI design engineer; they are easy to draw using a set of coloured pencils, different colours representing different conducting material layers. The usefulness of stick diagrams has been demonstrated several times in this report as stick diagrams were used as a guild to design the mask file for all of the designs in this report.

Undoubtedly design processes are aided by simple concepts such as stick and symbolic diagrams but the key element is a set of design rules. Lambda-based rule sets like that used in this assignment are extremely useful as the mask design file contains no real dimensions (not measured in μm), just dimensions with respect to lambda. Hence when new foundries become available and more transistors can be squared into a single IC, the existing design can be easy converted to the new foundry (value of lambda would have changed). But changes may have to be made to the design to accommodate the design rule checker (makes sure the circuit will work) as the new foundry may have different spacing specifications of lambda (probably smaller hence would not be a problem), for example minimum spacing between the metal1 for foundry es207 is 3λ, while for foundry cmos012 the minimum is 4λ.

Obviously the resistance and capacitance of data lines are one of the main concerns of VLSI design. Clearly both affect the rise and fall time of logic signals, as the time constant is a product of resistance and capacitance. If there is a large rise and fall time, this will limit the maximum speed the circuit can operated at. Resistance also affects the I_2R losses (heat loss), which give CPU manufactures like Intel and AMD problems relating on how to cool there processors. For many years it was thought diamonds were extremely good insulators (this is true), but recently it was discovered that diamonds could be used as a semiconductor that was must better than silicon (reduced electrical resistance, higher resistance to heat and perhaps a higher transistor count is possible), it is likely that diamond semiconductor technology may be at the forefront

of VLSI technology sometime in the future, but because diamonds are expense when compared to the cost of silicon (mostly sand), these new diamond semiconductor ICs will properly be used for specialised applications (NASA, MOD, fighter jets, super computers, etc…) and not for every household item that silicon is used for today.

It is clear that multiplexers are widely used and have many applications (including CPU design); they have the ability of funnelling several data lines into a single line for transmission to another point. Basic multiplexer design of (4-to-1 line) was shown in chapter 3 of this report. Evidently a multiplexer can be designed using SSI logic gates, but this method is inefficient for VLSI design because of the high transistor count (e.g. 50 transistors for 4-to-1). Clearly designing a multiplexer using NMOS or PMOS pass transistors is the most efficient VLSI design (only 12 transistors required for mux4 including CMOS inverter), but this method has the drawback of producing bad logic '1's (NMOS) or bad logic '0's (PMOS) and therefore is unpractical as the multiplexer will probably be used to drive other logic circuits (e.g. bad fan out). The optimal solution is to use a transmission gate arrangement, that is made up of both NMOS and PMOS transistors and produces a good logic '1' and '0', allow the transistor count is double that of the pass transistor solution.

The 6-to-1 line multiplexer is non-standard, clearly it is the norm for the number of data lines a multiplexer has to be a power of 2 (e.g. 2, 4, 8, 16), hence making use of all the possible combinations of the select lines. It was decided to strictly stick to the design specification and to design a 6-to-1 line multiplexer; hence since three select lines are required there were 2 unused select addresses (110 & 111), theoretically when these unused addresses are selected, none of the data lines are connected to the Z output, but because (N+ and P+ diffusion) has high line capacitance a logic '1' may appear at the output until the capacitance discharges, depending on the logic level of the pervious data line to be connected. For example if the pervious data line was at logic '1' then line 111 is selected (unused address) all data lines are disconnected from the output but there remain a charge producing logic '1'. Note this charge will dissipate in a short period of time depending on the load resistance of the device connected to the output Z.

The NMOS pass transistor design of the 6-to-1 line multiplexer, is simple, there are only 24 transistors in the design. The PMOS pass transistor design is exactly the same except, that it uses PMOS transistors instead of NMOS, width of diffusion channels doubled (P instead of N) and an N Well surrounding all of the PMOS transistors. Both layouts were designed for simplicity and not performance and space, clearly there are a view flaws in the design, long lengths of polysilicon was used to connect some of the inverter lines to the select lines, this should have been avoided because of the high resistance, also long lengths of N and P diffusion were used as data lines, this should have been avoided because of the high capacitance of the diffusion layers. Therefore before any simulations have taken place it is obvious that the circuit will have high rise and fall times as the time constant is a product of resistance and capacitance, and may only be suitable for low speed applications.

Detailed simulations were carried out on both the NMOS and PMOS designs (direct comparison can be made), clearly as expected NMOS transistors were good at passing logic '0's but bad at passing logic '1's (about 3.5V not 5V), while PMOS transistors were good at passing logic '1's but bad at passing logic '0's (about 1.5V not 0V). Note these bad values of logic '1's and '0's are just inside the range of a logic '1' and '0', but there is a reason why logic circuits have such a large range (e.g. noise margin). Hence the bad logic '1's and '0's would have reduced noise immunity, and would have trouble driving multipliable devices (fan out).

Clearly the operation of the NMOS & PMOS pass transistor multiplexers were fully simulated and proven to operate correctly. But notice during simulation there were small glitches occurring throughout all of the simulation results, clearly this is due to metastability cause when a transistor is switching OFF and another is switching ON, e.g. for a short period of time both will be ON simultaneously switching in a data line to output Z. Also as expected an output was monitored at Z when the unused select lines were selected, this output was maximum when the pervious state was at logic '1', proving that the diffusion tracks are acting as an capacitor storing charge.

The CMOS transmission gate design is not made up of transmission gates, but is a transmission process that should give the same results as if it was designed using transmission gates. Clearly the design is simpler than it would be if it was made using transmission gates, as routing would be more difficult, allow layer metal 2 could be used for easy routing, but adds complexity to the design. Detailed simulation was also carried out and the design was proven to work, the key thing to note is that a good logic '1' and a good logic '0' was produced, making the transmission processes design the more practical of the three. Because it has a greater noise immunity and can drive many more devices (good fan out) than the NMOS / PMOS designs.

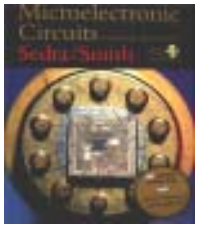
The rise and fall time of all three implementations were measured. Clearly the NMOS pass transistor design has a fast fall time (264ps) but a slow rise time (968ps) while the PMOS pass transistor design has a fast rise time (576ps) and a very slow fall time (2680ps). Notice that the CMOS transmission gate design has the best of both with almost equal rise (656ps) and fall times (616ps), but not has fast as the NMOS fall time and the PMOS rise time. These rise and fall times are due to line capacitance and resistance, replacing the data line with metal 1 (instead of diffusion) and only have a small amount of diffusion to form each transistor would reduce data line capacitance and resistance, hence decreasing fall and rise times for a higher performance multiplexer. An experiment showing that the line resistance of the polysilicon is high was carried out (result shown in chapter 5.3.6). The Spice files were extracted and included in chapter 4 for each of the design; clearly these spice files are extremely useful. Notice that capacitance values are included and can be used for detailed circuit analyses.

The mask files designed in this assignment were manually designed; clearly this is a slow and boring process that is not practical in the real world. Rarely does designer designs at such a low level, a software program is normally used to automatically convert circuit diagram to the physical level of an IC chip. This is demonstrated in appendix 3 using program Dsch2 to generate a Verilog file, which can be compiled in MicroWind. Although MicroWind is extremely powerful it is not at the leading edge of technology, it is a academic program design to allow students to learn how to make simple VLSI circuits, it might be possible to manufacture a chip using the MASK files generated by MicroWind but it is not a commercial product. Appendix 1 shows an extremely powerful commercial package used for VLSI design.

Clearly a 6-to-1 line multiplexer using the MicroWind software package was successfully designed, analyses and proved. The performance can be improved by replacing long lengths of high resistance polysilicon and long lengths of high capacitance diffusion with metal. The compactness of the area can be improved by placing the transistors below the metal select lines, this practice is acceptable in this situation where a transistor gate is actually driven from and connected to the particular metal line which runs across it (use with caution).

7.0. REFERENCES

Text Books

[B1] 

Microelectronic Circuits – Fourth Edition.

By S. Sedra and Kenneth C. Smith
Publisher: Oxford university press
ISBN: 0-19-511690-9
1998

[B2] 

Introduction to VLSI Circuits and Systems

By John P. Uyemura
Publisher: John Wiley & Sons
ISBN: 0-471-12704-3
2002

[B3] 

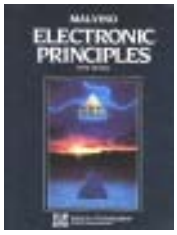
Basic VLSI Design

By Douglas A. Puncknell and Kamran Eshraghian
Publisher: Prentice Hall
ISBN: 0-13-079153-9
UUJLIB: 621.395PUC
1994

[B4] 

Analog and Digital Electronics

By Peter H. Beards
Publisher: Prentice Hall
ISBN: 0-13-571753-1
1996

[B5] 

Electronic Principles

By Albert Paul Malvino
Publisher: McGraw-Hill
ISBN: 0-07-113480-8
1993

Journals

[J1] Microwind & Dsch Version 2 - User Manual (February 2002 by Etienne Sicard).

Web Sites

[W1] http://www.icknowledge.com (IC Knowledge is a company providing knowledge resources to the semiconductor industry)

[W2] http://vlsi.wpi.edu/cds/examples/layout.html (Cadence design tools tutorial by VLSI CAD laboratory Worcester Polytechnic institute)

[W3] http://www.cadence.com/ (Cadence design tools official website)

[W4] http://intrage.insa-tlse.fr/~etienne/Microwind/ (MicroWind official website, free download)

[W5] http://www.hardware-unlimited.com/reviews/axp1800/ (Review of the Athlon XP 1800+ Processor, October 20, 2001, Copyright © 2000, 3D-Unlimited Network).

[W6] http://www.AMD.com (AMD official website).

A1. POWERFUL COMMERICAL PACKAGE

Cadence custom IC design tools (composer schematic editor, virtuoso layout editor, simulation environment, extraction utilities for design verification and extraction. device level placer for automated generation of physical mask layout from circuit schematics), see [W2] and [W3] for details.

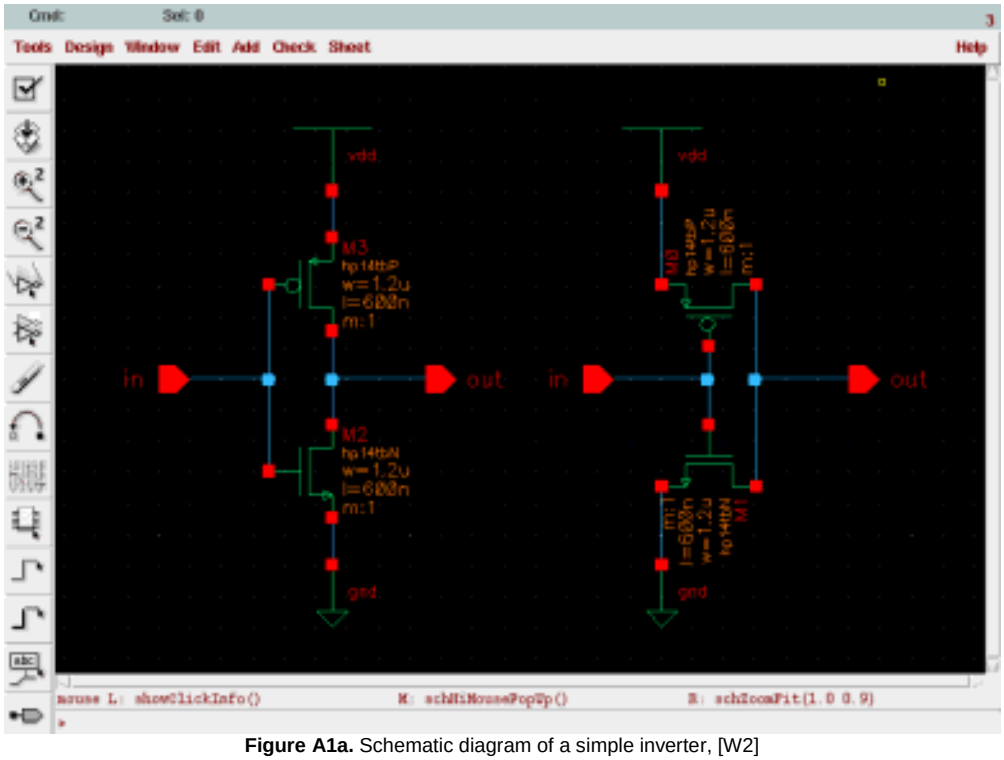


Figure A1a. Schematic diagram of a simple inverter, [W2]

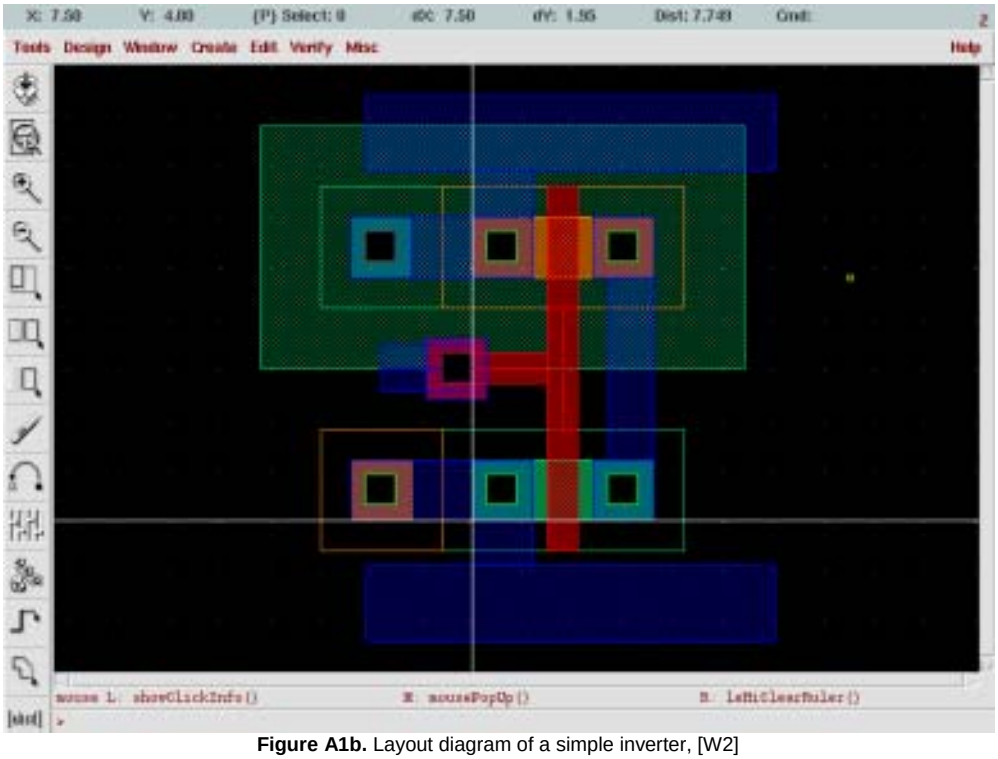
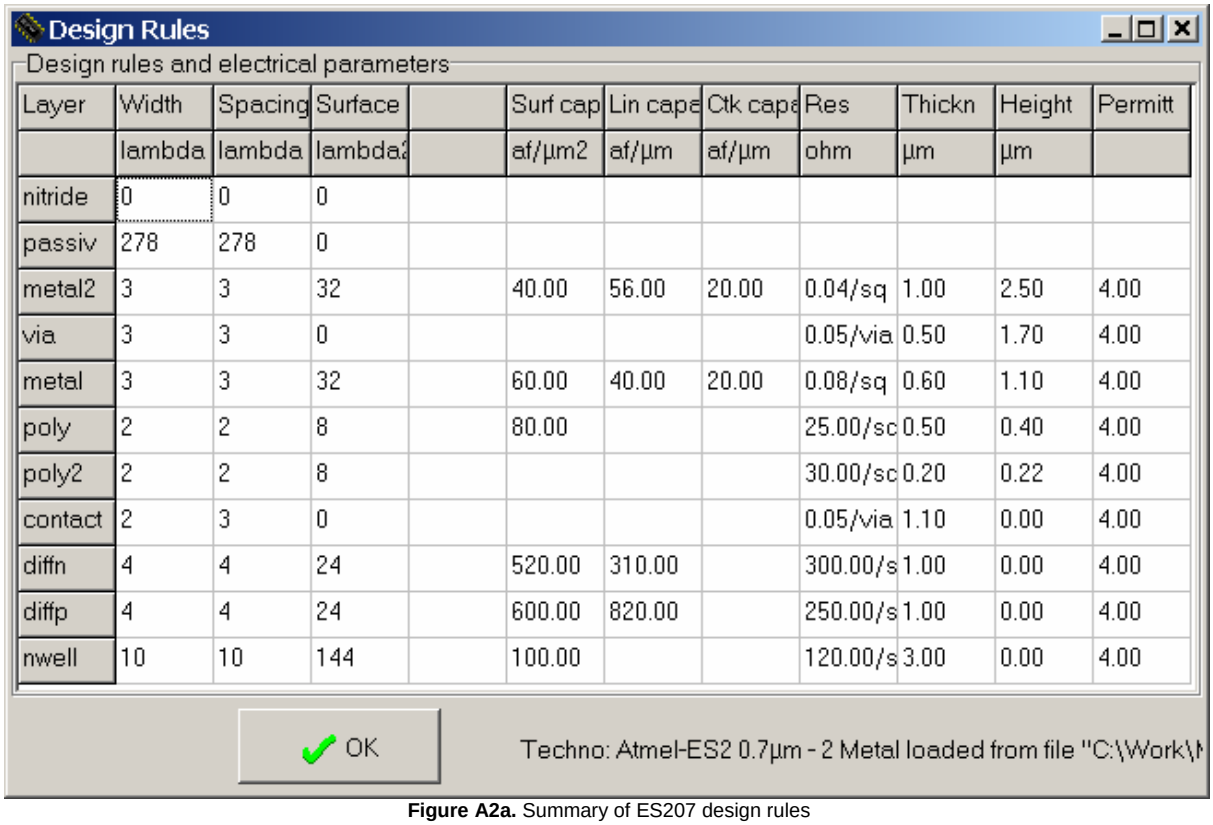


Figure A1b. Layout diagram of a simple inverter, [W2]

A2. ES207 DESIGN RULE FILE

The es207 design rule file was used for all designs and simulations carried out in this report, figure A2a contains a summary of the design rules.



Layer	Width	Spacing	Surface		Surf cap	Lin capa	Ctk capa	Res	Thickn	Height	Permitt
	lambda	lambda	lambda2		af/µm2	af/µm	af/µm	ohm	µm	µm	
nitride	0	0	0								
passiv	278	278	0								
metal2	3	3	32		40.00	56.00	20.00	0.04/sq	1.00	2.50	4.00
via	3	3	0					0.05/via	0.50	1.70	4.00
metal	3	3	32		60.00	40.00	20.00	0.08/sq	0.60	1.10	4.00
poly	2	2	8		80.00			25.00/sc	0.50	0.40	4.00
poly2	2	2	8					30.00/sc	0.20	0.22	4.00
contact	2	3	0					0.05/via	1.10	0.00	4.00
diffn	4	4	24		520.00	310.00		300.00/s	1.00	0.00	4.00
diffp	4	4	24		600.00	820.00		250.00/s	1.00	0.00	4.00
nwell	10	10	144		100.00			120.00/s	3.00	0.00	4.00

Figure A2a. Summary of ES207 design rules

Complete listing of ES207.rul: -

```
MICROWIND v1.00
*
* Rule file for
* Atmel ES2 0.7µm
* CMOS 2-metal
*
* 02 Dec 97 by Etienne Sicard
* 04 Dec 97 : tune ctk capas
* 05 Dec 97 : Add Cgs,Cgd
* 24 Jan 98 : Modify Capa
* 19 May 98 : add contact res
* 29 Oct 98 : diff thickness
* 13 Avr 00 : thickness locos
*
NAME Atmel-ES2 0.7µm - 2 Metal
*
lambda = 0.4     (Lambda is set to half the gate size)
metalLayers = 2  (Number of metal layers : 5)
*
* Design rules associated to each layer
*
* Well (1)
r101 = 10    (well width)
r102 = 10    (well spacing)
*
* Diffusion (12,14)
*
r201 = 4     (diffusion width)
r202 = 4     (diffusion spacing)
r203 = 6     (border of nwell on diffp)
```

```
r204 = 6     (nwell to next diffn)
r205 = 4     (diffn to diffp)
* Poly (11)
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 (16)
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 (17)
r501 = 3    (metal width)
r502 = 3    (metal spacing)
* Via (18)
r601 = 3    (Via width)
r602 = 3    (Spacing)
r603 = 3    (To unrelated contact)
r604 = 2    (border of metal&metal2)
*   metal 2 (19)
r701 = 3    (Metal 2 width)
r702 = 3    (spacing)
*
* Pads (Passiv is 20)
*
rp01 = 278 (Pad width)
rp02 = 278  (Pad spacing)
rp03 = 13   (Border of Via for passivation )
rp04 = 13   (Border of metals)
rp05 = 63   (to unrelated active areas)
*
*
* Thickness of layers
*
thdn = 1.0
thdp = 1.0
thnw = 3.0
thsti = 2.0
hesti = -1.6
thpoly = 0.5
hepoly = 0.4
thme = 0.6
heme = 1.1
thm2 = 1.0
hem2 = 2.5
thpass = 1.0
hepass = 3.0
thnit = 0.5
henit = 4.0
*
* Resistance (ohm / square)
*
repo = 25
reco = 0.05
reme = 0.075
revi = 0.05
rem2 = 0.040
*
* Parasitic capacitances
*
cpoOxyde = 2300 (Surface capacitance Poly/Thin oxyde aF/µm2)
cpobody = 80   (Poly/Body)
cmebody = 60
cmelineic = 40   (aF/µm)
cmepoly = 58
cm2body = 40
cm2lineic = 56  (aF/µm)
cm2poly = 30
cm2metal = 60
*
* Crosstalk
*
```

```
cmextk = 20      (Lineic capacitance for crosstalk coupling in aF/µm/µm)
cm2xtk = 20      (C is computed using Cx=cmextk*l/spacing)
*
* Junction capacitances
*
cdnpwell = 520   (n+/psub)
cdpnwell = 600  (p+/nwell)
cnwell = 100    (nwell/psub)
cpwell = 100    (pwell/nsub)
cldn = 310      (Lineic capacitance N+/P- aF/µm)
cldp = 820      (Idem for P+/N-)
*
* Nmos Model 3 parameters
*
NMOS
l3vto = 0.8
l3vmax = 130e3
l3gamma = 0.4
l3theta = 0.2
l3kappa = 0.01
l3phi = 0.7
l3ld = -0.05
l3kp = 135e-6
l3nss = 0.07
l3cgd = 200
*
* Pmos Model 3
*
PMOS
l3vto = -1.1
l3vmax = 100e3
l3gamma = 0.4
l3theta = 0.2
l3kappa = 0.01
l3phi = 0.7
l3ld = -0.05
l3kp = 47e-6
l3nss = 0.07
l3cgd = 200
*
* MicroWind simulation parameters
*
deltaT = 8.0e-12   (Minimum simulation interval dT)
vdd = 5.0
temperature = 27
*
* CIF&GDS2
* MicroWind name, Cif name, Gds2 n°, overetch for final translation
*
cif nwell CNWI 1 0.0
cif aarea CTOX 2 1.0
cif poly CPOL 11 0.0
cif diffn CNPI 12 1.0
cif diffp CPPI 14 1.0
cif contact CCON 16 0.1
cif metal CME1 17 0.0
cif via CVIA 18 -0.1
cif metal2 CME2 19 0.0
cif passiv CPAS 20 0.0
cif text text 0 0.0
*
* End atmel-ES2 0.7µm
*
```

A3. AUTOMATIC GENERATION OF THE MASK FILE

The mask files for the 6-to-1 line multipliers in this assignment were manually designed; clearly this is a slow and boring process that is not practical in the real world. Rarely does a designer design at such a low level, software programs can automatically convert circuit diagrams to the physical level of an IC chip.

This automatic generation of the mask file, will be demonstrated using the DSCH2 (Version 2.4c) program. This program is made by Etienne Sicard (same person who made MicroWind); it is freeware and extremely powerful, allow there are many bugs within the program, manly related to the user-interface and visual display.

"The DSCH2 program is a logic editor and simulator. DSCH2 is used to validate the architecture of the logic circuit before the microelectronics design is started. DSCH2 provides a user-friendly environment for hierarchical logic design, and simulation with delay analysis, which allows the design and validation of complex logic structures. A key innovative feature is the possibility to estimate the power consumption of the circuit. Some techniques for low power design are described in the manual." [J1]

Basically a circuit is drawn in DSCH2 using library components (e.g. AND, OR, NOT, NMOS trans, PMOS, trans etc…). Once the circuit has been drawn all that is required is to generate a Verilog file which can be compiled in MicroWind and used to generate the mask file.

A3.1. Automatic Generation of the Mask File for an Inverter

Draw the inverter circuit in DSCH2 and then generate the Verilog file.

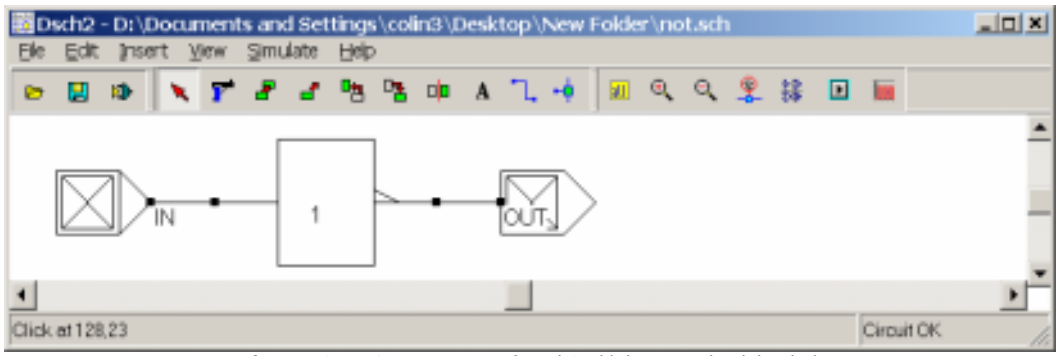


Figure A3.1a. Screen Dump of Dsch2 with inverter circuit loaded

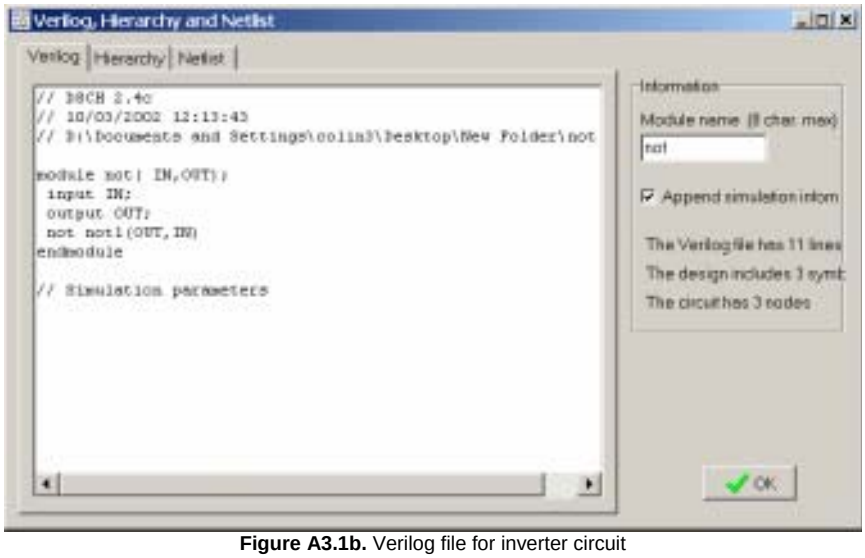


Figure A3.1b. Verilog file for inverter circuit

Compile Verilog file in MicroWind to generate mask file.

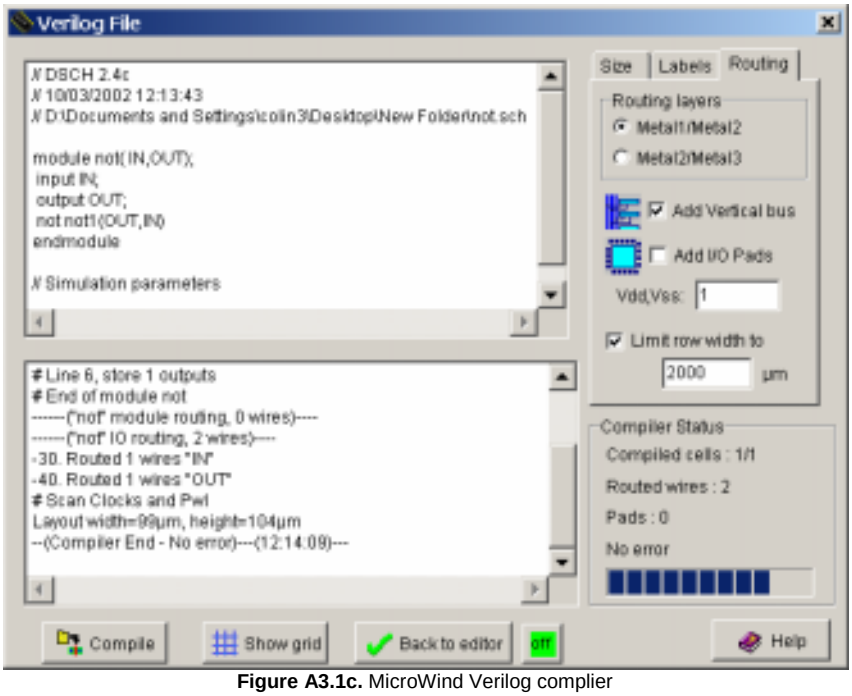


Figure A3.1c. MicroWind Verilog complier

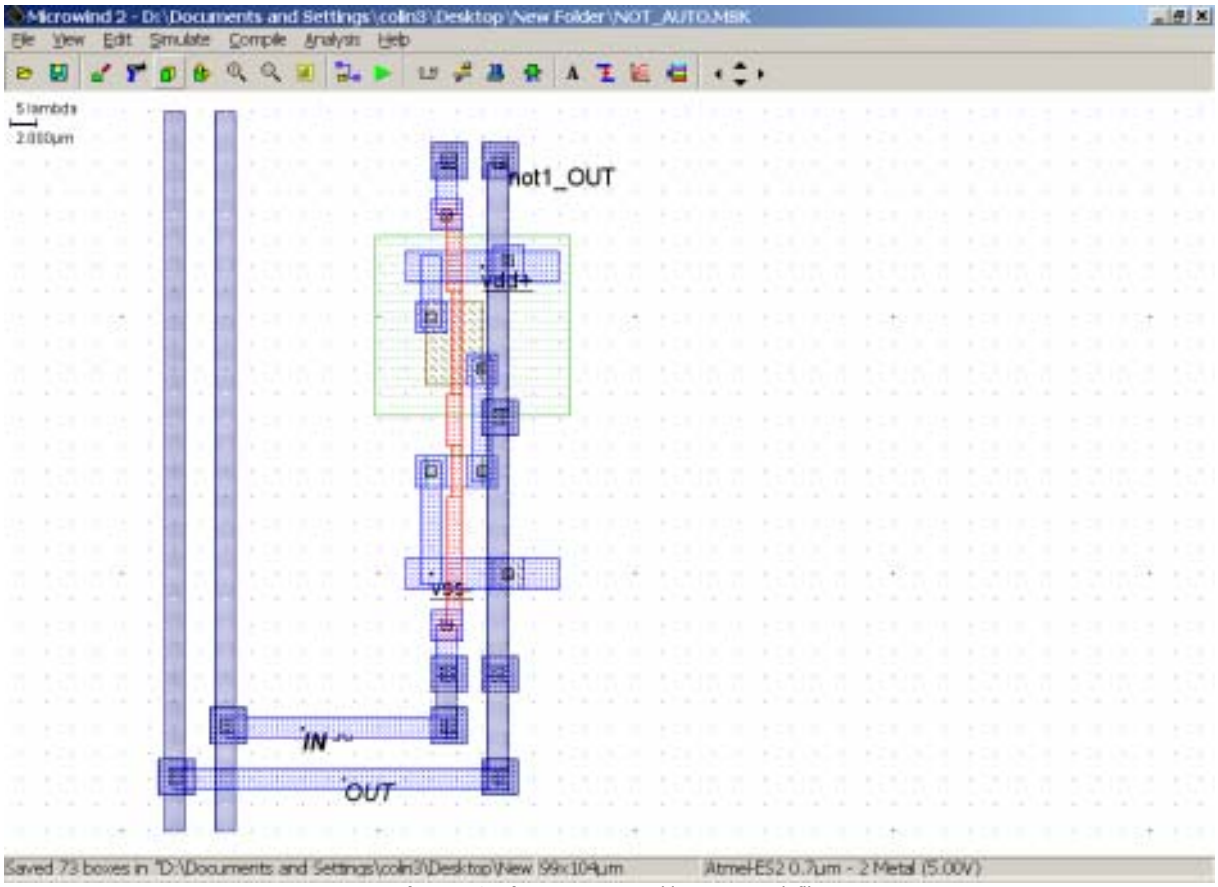


Figure A3.1d. Auto generated inverter mask file

Clearly figure A3.1d shows the classic CMOS inverter, but it is not as pretty as the manually design one.

A3.2. Automatic Generation of the Mask File for a 2-to-1 line Multiplexer

Draw the mux2 circuit in DSCH2 then generate the Verilog file and compile in MicroWind.

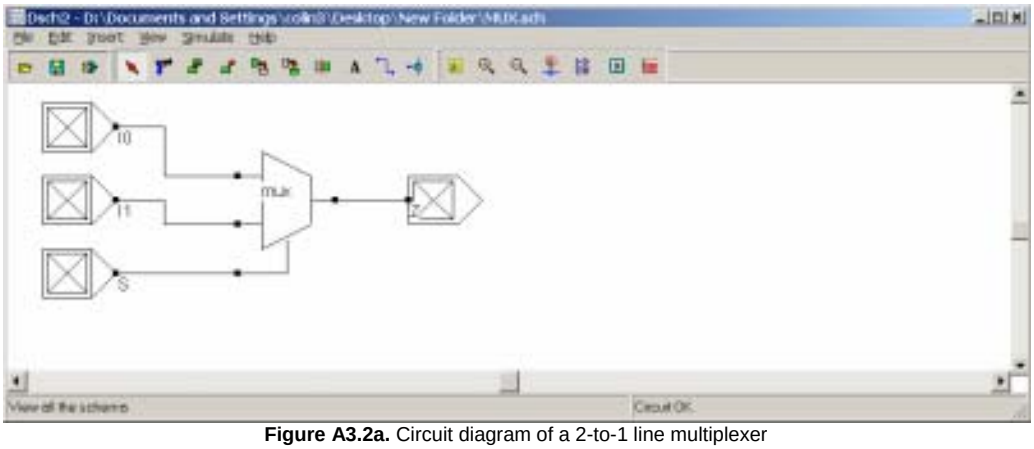


Figure A3.2a. Circuit diagram of a 2-to-1 line multiplexer

```
// DSCH 2.4c
// 10/03/2002 11:57:48
// D:\Documents and Settings\colin3\Desktop\New Folder\MUX.sch

module MUX( I0,I1,S,Z);
 input I0,I1,S;
 output Z;
 mux mux1(Z,I0,I1,S)
endmodule

// Simulation parameters
```

Figure A3.2b. Generated Verilog file

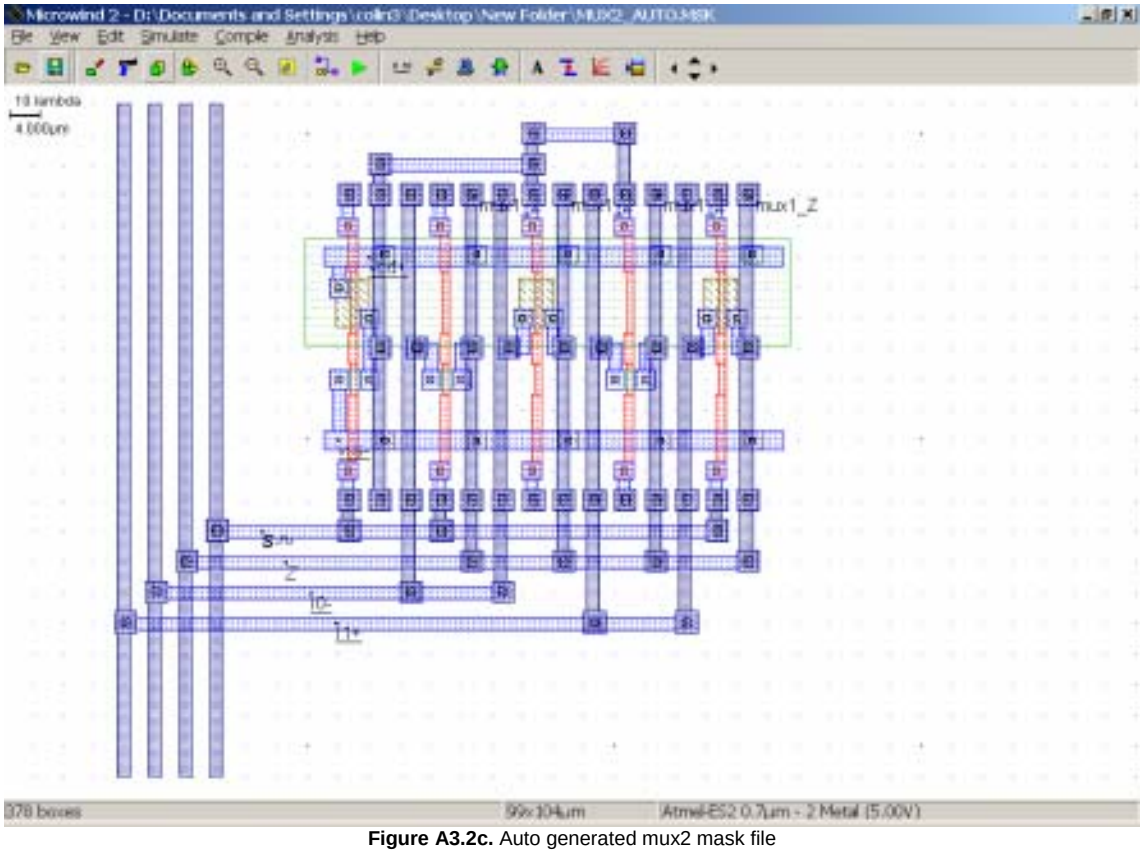


Figure A3.2c. Auto generated mux2 mask file

A3.3. Automatic Generation of the Mask File for a 4-to-1 line SSI Multiplexer

Draw the mux4 circuit in DSCH2 then generate the Verilog file and compile in MicroWind.

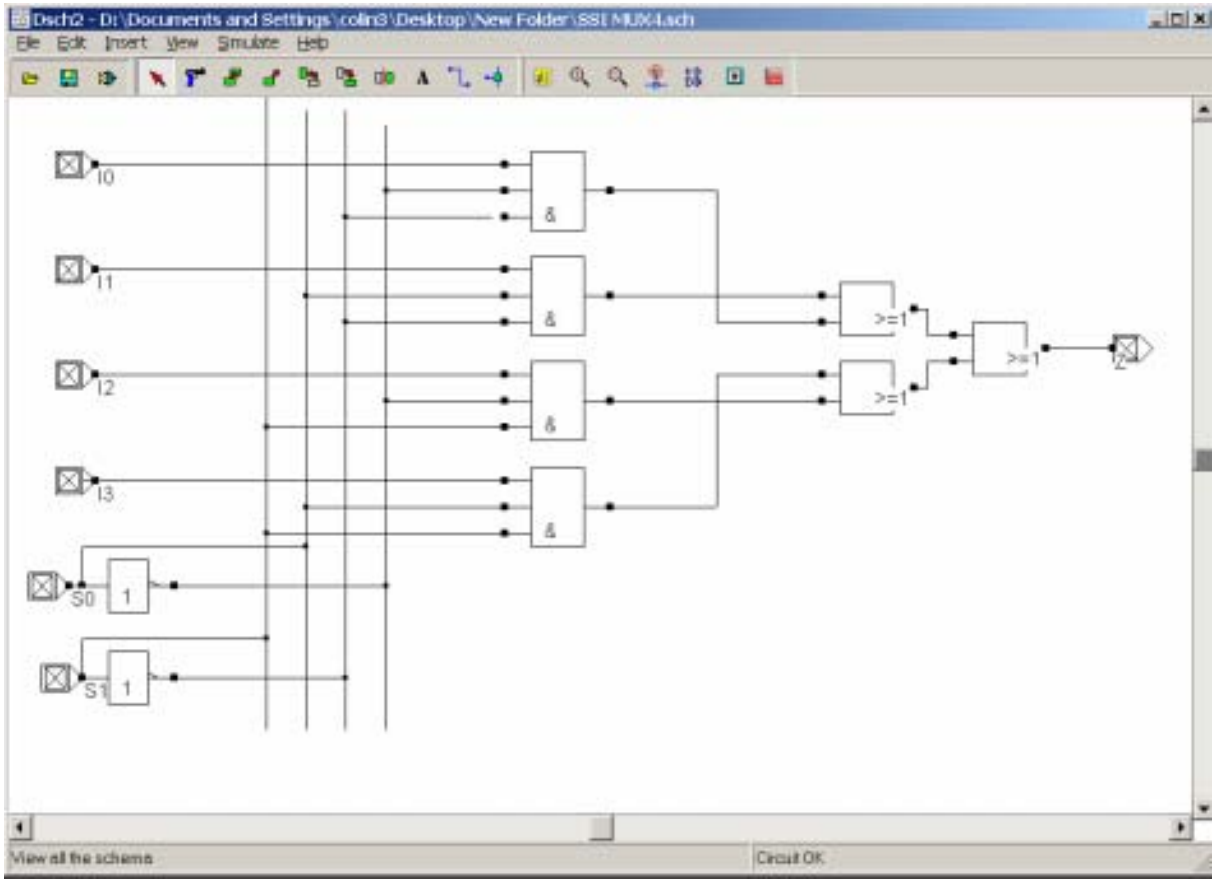


Figure A3.3a. Circuit diagram of a 4-to-1 line SSI multiplexer

```
// DSCH 2.4c
// 10/03/2002 11:44:24
// D:\Documents and Settings\colin3\Desktop\SSI MUX4.sch

module SSI MUX4( I0,I1,I2,I3,S0,S1,Z);
 input I0,I1,I2,I3,S0,S1;
 output Z;
 not not1(w2,S1)
 and and2(w5,I3,S0,S1)
 and and3(w8,I2,w7,S1)
 and and4(w10,I1,S0,w2)
 and and5(w13,I0,w7,w12)
 or or6(w14,w10,w13)
 or or7(w15,w5,w8)
 or or8(Z,w14,w15)
 not not9(w7,S0)
endmodule

// Simulation parameters
```

Figure A3.3b. Generated Verilog file

Just look at figure A3.3c it looks extremely complicated, but it does work. Clearly if this was designed manually using NMOS pass transistors as shown in figure 3.2b (or CMOS transmission gate shown in figure 3.3b), the design would be much simpler. But the design shown in figure 3.2b took about an hour to draw, while the auto generated design shown in figure A3.3c took only 5 minutes. Clearly the 4-to-1 multiplexer is a small simple design, for larger more complex designs auto generation is the only option with perhaps critical areas being design manually. For example think how difficult it would be to design an entire microprocessor manually (latest AMD processor has 37.5 million transistors, see appendix 4).

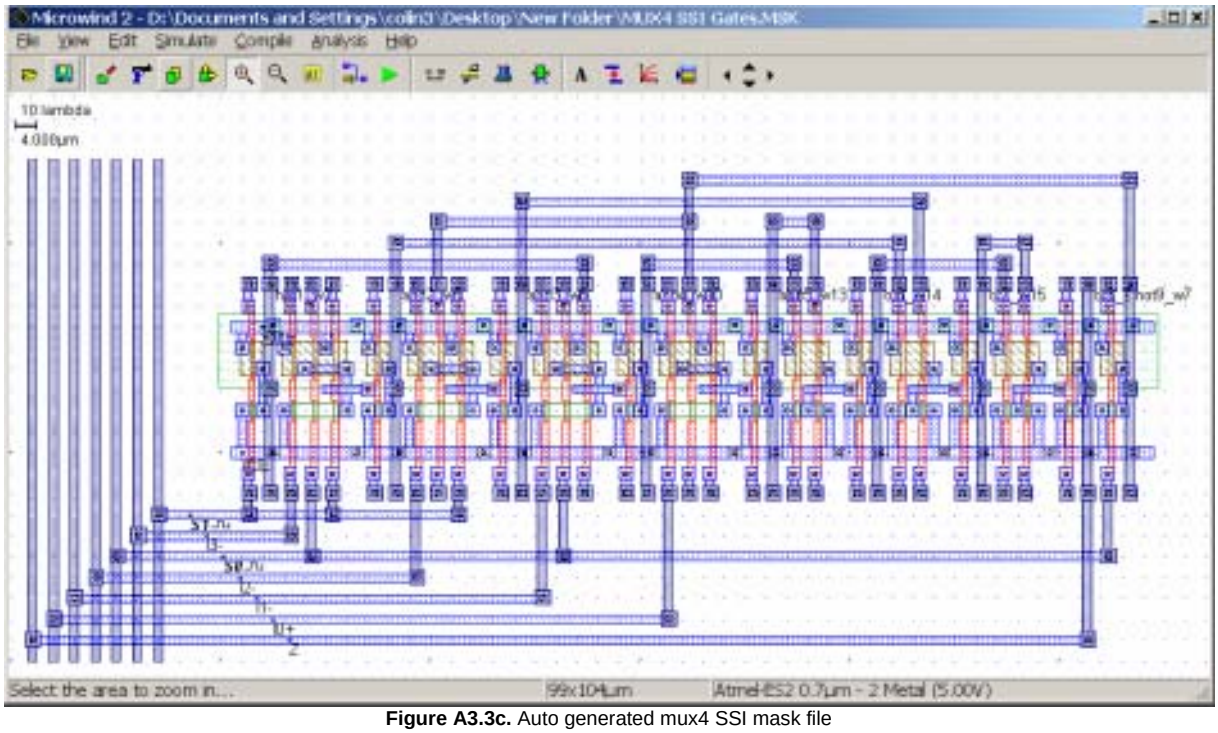


Figure A3.3c. Auto generated mux4 SSI mask file

A4. PHYSICAL ASPECTS OF THE ATHLON XP MICROPROCESSOR

"The new Athlon XP series core looks very similar to the Athlon MP and mobile Athlon 4, with the same square (128mm^2) shaped core. The transistor count remains the same as the Athlon MP at 37.5 million, a boost from the 37 million found on the older Athlon cores.

The Athlon XP will only be available in 266MHz bus versions to be paired with DDR ram. AMD also claims a 20% power reduction with the new Athlon XP cores, but it is hard for us to verify that. One note though, the same 1.75 core voltage is still used, as well as the addition of half a million transistors, but AMD claims the new core is more efficient in using these new transistors and thus allows for a decrease in power consumption. Also, the Athlon XP has moved to a new organic packaging just like the Pentium 3 and 4. However, it still retains the same appearance of the older Athlon's, but with a new brownish-red colour, which can be seen below." [W5]

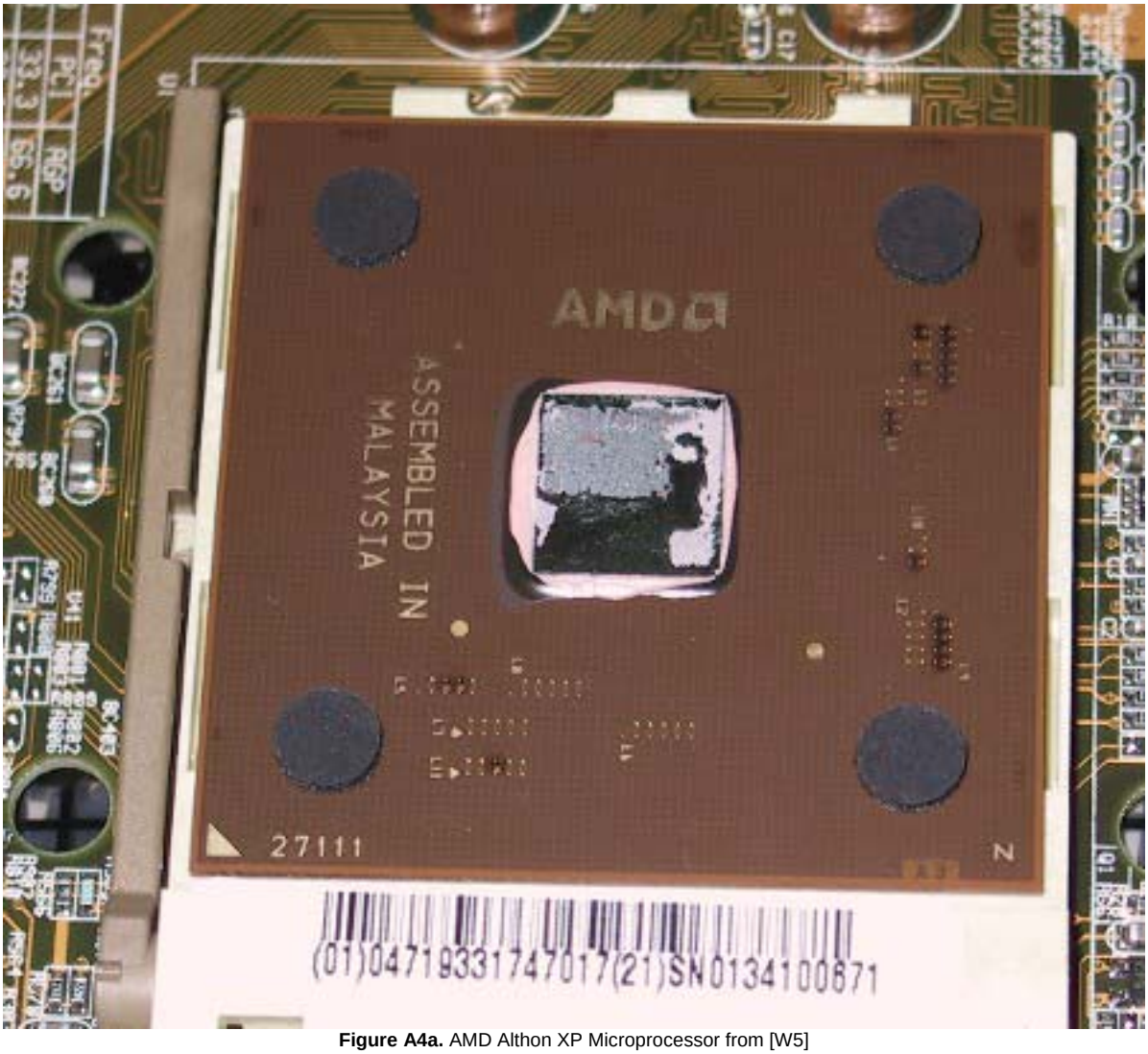


Figure A4a. AMD Althon XP Microprocessor from [W5]

AMD are one of the worlds leading microprocessor manufactures (founded in 1969, and have shipped more than 175 million PC processors worldwide). Their XP range out performs Intel's Pentium 4 range in every way (manly because there processors do more per clock pulse [QuantiSpeed Technology] and have more L1 cache) and are generally more reasonability priced. Technical details about AMD processors and a direct comparison between Intel and AMD processors (not found on Intel's website) can be found on the AMD official website [W6]. Clearly modern microprocessor design is at the leading edge of VLSI design.

A5. CD ROM: CONTAINING MICROWIND DESIGN FILES

This CD ROM contains all of the design files used in this report.