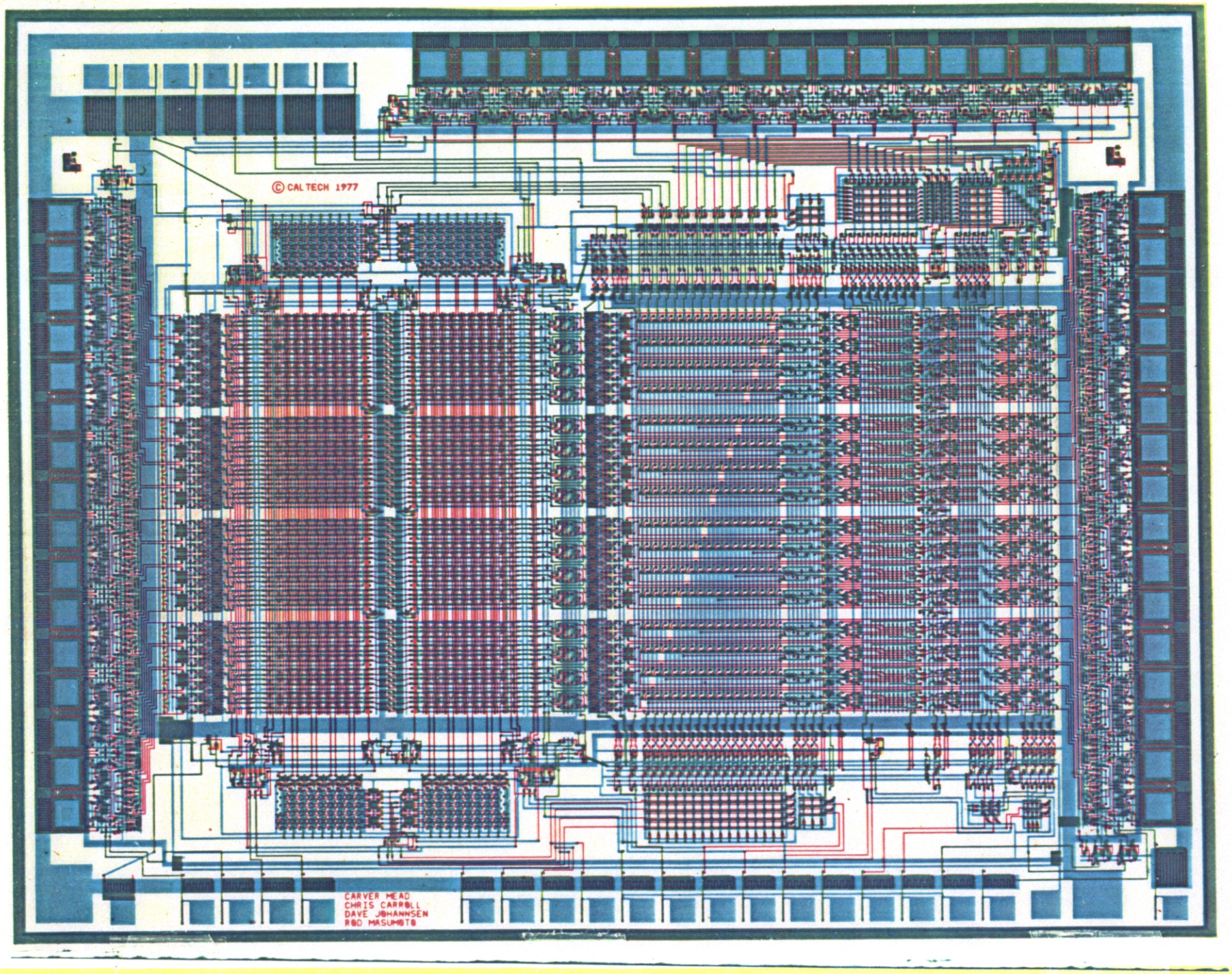
INTRODUCTION TO LSI SYSTEMS

CARVER A. MEAD

Professor of Computer Science,
Electrical Engineering,
and Applied Physics,
California Institute of Technology

LYNN A. CONWAY

Manager, LSI Systems Area,
Palo Alto Research Center,
Xerox Corporation


T MASK 2
POLYSILICON
METAL
© CALTECH 1977
CARVER MEAD
CHRIS CARROLL
DAVE JOHANNSEN
ROD MASUMOTO

PREFACE:

As a result of improvements in fabrication technology, Large Scale Integrated (LSI) electronic circuitry has become so dense that a single silicon LSI chip may contain tens of thousands of transistors. Many new LSI chips such as microprocessors consist of multiple, complex subsystems, and thus are really *integrated systems* rather than integrated circuits.

What we have seen so far is only the beginning. Achievable circuit density approximately doubles with each passing year. How long can this continue, and how small can the transistor be made? From the physics we find that the linear dimensions of transistors can be reduced to less than 1/10 of those in current LSI before they cease to function as the sort of switching elements from which we can easily build digital systems. It will eventually be possible to fabricate chips with hundreds of times as many components as today's LSI. The transistors in such very large scale integrated (VLSI) circuitry will have linear dimensions smaller than the wave-length of visible light. New, non-optical, high-resolution lithographic techniques are now being developed by many firms to enable fabrication of such circuitry.

The coming VLSI presents a challenge not only to those involved in development of appropriate fabrication technology, but also to computer scientists and computer architects. The ways in which digital systems are structured, the procedures used to design them, the tradeoffs between hardware and software, and the methodology and metrics of analysis of algorithms will all be greatly affected by the maturation of electronic technology toward VLSI. We believe this will be an important area of activity for computer science on through the 1980's.

Until recently, the design of integrated electronic circuitry has been largely the province of circuit and logic designers working within semiconductor firms. Computer system architects have traditionally built systems from standard integrated circuits designed and manufactured by these firms, but haven't often participated in the specification and design of these integrated circuits. Electrical Engineering and Computer Science (EE/CS) curricula have reflected this tradition, with courses in device physics and integrated circuit design aimed at and generally taken by different students than those interested in digital system architecture and computer science.

This text is written to fill the current gap by providing students of computer science and electrical engineering with an introduction to LSI system architecture and design. Combined with individual study in related research areas and participation in actual system design projects, the text could serve as a basis for a graduate course sequence in LSI systems. Portions could be used for an undergraduate text on the subject, or to augment a graduate course on computer architecture. It could also be used to extend, in the system direction, a classical electrical engineering course in integrated circuits. We assume the reader's background contains the equivalent of an introductory course sequence in computer science, and introductory courses in electronic circuits and digital design.

Up till now there have been major obstacles in the path of those attempting to gain an overall understanding of LSI systems. For one thing, it has been difficult to separate the forest from the trees. LSI electronics, developing in a heatedly competitive business environment, has proliferated into a large array of different device technologies, circuit design families, logic design techniques, maskmaking and wafer fabrication techniques, etc. The technologies have sprung up from the grass roots of "silicon valley" in California. Most participants in the industry have of necessity concentrated on rather narrow specialties. Texts on the subject have tended to give very detailed accounts of some very narrow horizontal segment of the overall subject, such as device physics or circuit design. We have chosen instead to provide the minimum information about devices, circuits, fabrication technology, logic design techniques, and system architecture, which is sufficient to enable the reader to fully span the entire range of abstractions, from the underlying physics to complete LSI digital computer systems. A rather small set of carefully selected key concepts is all that is necessary for this purpose. We believe that only by carrying along the least amount of unnecessary mental baggage at each step in such a study, will the student emerge with a good overall understanding of the subject. Once this range of abstractions is spanned, the sequence of concepts can then be mapped into the reader's own space of application and technology.

Another major obstacle has been the high rate of change of LSI electronics. The uninitiated could easily get the feeling that much energy could be invested in learning material which becomes obsolete as rapidly as it is assimilated. The major driving mechanism in all this change is the continual improvement in fabrication technology. This evolutionary process results in the feasibility of manufacturing smaller and smaller devices as time passes. By including the effects of scaling down device dimensions as an essential ingredient of all topics in this text, many of the important changes of the architectural parameters of the technology are predicted, expected, and indeed hoped for.

The key concepts of this text are illustrated by way of specific examples. In any given technology, form follows function in a particular way. The skill of mapping function into form, once acquired, can be readily applied to any technology. Because of its density, speed, topological properties, and general availability of wafer fabrication, nMOS has been chosen as the technology in which examples are implemented.

Energetic readers will soon discover their own examples. The territory is vast, and largely unexplored. The rewards are great for those who simply press forward.

Carver A. Mead
Pasadena, California

Lynn A. Conway
Belmont, California

TABLE OF CONTENTS

Chapter 1: MOS Devices and Circuits

Sections:

The MOS Transistor - - - The Basic Inverter - - - Inverter Delay - - - Parasitic Effects - - - Driving Large Capacitive Loads - - - Space vs Time - - - Basic NAND and NOR Logic Circuits - - - Super Buffers - - - A Closer Look at the Electrical Parameters - - - Depletion Mode vs Enhancement Mode Pullups - - - Delays in Another Form of Logic Circuitry - - - Pullup/Pulldown Ratios for Inverting Logic Coupled by Pass Transistors - - - Properties of Cross Coupled Circuits - - - Effects of Scaling Down the Dimensions of MOS Circuits and Systems

In this chapter we begin with a discussion of the basic properties of the n-channel MOS field effect transistor (MOSFET). We then describe and analyze a number of circuits composed of interconnected MOS transistors. The circuits described are typical of those we will commonly use in the design of LSI systems. The analysis, though highly condensed, is conceptually correct and provides a basis for the solution of most system problems typically encountered.

Integrated systems in nMOS technology contain three levels of conducting material separated by intervening layers of insulating material. Proceeding from top to bottom, these levels are termed the *metal*, the *polysilicon*, and the *diffusion* levels respectively. Patterns for paths on these three levels, and the locations of contact cuts through the insulating material to connect certain points between levels, are transferred into the levels during the fabrication process from *masks* similar to photographic negatives. The details of the fabrication process will be discussed in chapter 2.

In the absence of contact cuts through the insulating material, paths on the metal level may cross over paths on the polysilicon or diffusion levels with no significant functional effect. However, wherever a path on the polysilicon level crosses a path on the diffusion level, a transistor is created. Such a transistor has the characteristics of a simple switch, with a voltage on the polysilicon level path controlling the flow of current in the diffusion level path. Circuits composed of such transistors, interconnected by patterned paths on the three levels, form our basic building blocks. With these basic circuits, we will architect LSI systems, to be fabricated on the surface of monolithic crystalline chips of silicon.

The MOS Transistor

An MOS transistor will be produced on the integrated system chip wherever a polysilicon path crosses a diffusion path, as shown in figure 1a. The electrical symbol used to represent the MOS transistor in our circuit diagrams is shown in figure 1b, along with symbols and polarities of certain voltages of interest. Note that the source and drain terminals of the device are physically symmetrical. In practice these terminal labels are assigned such that V_{ds} is normally positive. A more detailed view of the rectangular region called the gate, where the polysilicon (poly) crosses the diffusion, is given in figure 1c. During fabrication the diffusion paths are formed after the poly paths are formed. The poly gate and thin layer of oxide under it masks the region under the gate during diffusion, and there is no direct diffusion connection between the source and drain terminals of the transistor.

In the absence of any charge on the gate, the drain to source path through the transistor is like an open switch. The gate, separated from the substrate by the layer of thin oxide, forms a capacitor. If sufficient positive charge is placed on the gate so that V_{gs} exceeds a threshold voltage V_{th}, electrons will be attracted to the region under the gate to form a conducting path between drain and source. Most of the transistors we will use in our systems have threshold voltages greater than zero. These are called enhancement mode MOSFETs, and their threshold voltage typically equals ~ 0.2(VDD), where VDD is the positive supply voltage for the particular technology.

The basic operation performed by the MOS transistor is to use charge on its gate to control the movement of negative charge between source and drain through the channel under the gate. The current from source to drain equals the charge induced in the channel divided by the transit time or average time that a packet of negative charge requires to move from source to drain. The transit time itself is the distance the charge has to move divided by its average velocity. In semiconductors under normal conditions, the velocity is proportional to the electric field driving the charge carriers. The relationship between drain to source current I_{ds}, drain to source voltage V_{ds}, and gate to source voltage V_{gs} is sketched in figure 1d. In a common mode of MOS transistor operation called saturation, the average electric field E in the channel is proportional to the difference between V_{gs} and V_{th}, and is not a function of V_{ds}. This is the region in fig.1d where the I_{ds} lines, plotted at constant V_{gs}, run horizontally. In saturation, the transit time is given by eq. 1.

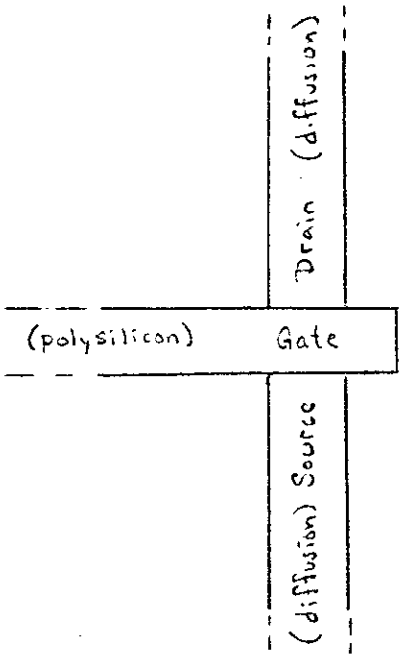


Fig.1a. MOS TRANSISTOR
(TOP VIEW)

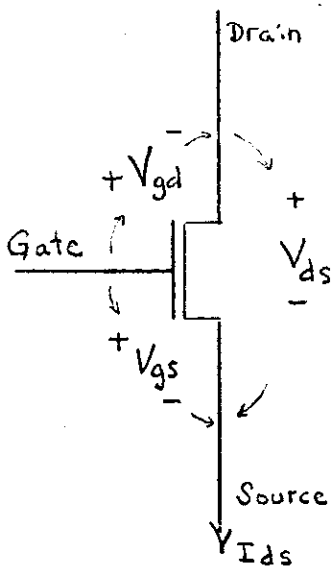


Fig.1b. MOS TRANSISTOR SYMBOL
(Subscripts in "plus to minus direction" sequence)

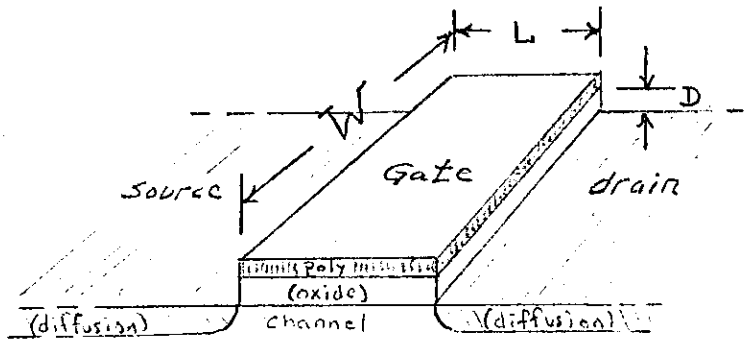


Fig.1c. MOSFET GATE DIMENSIONS

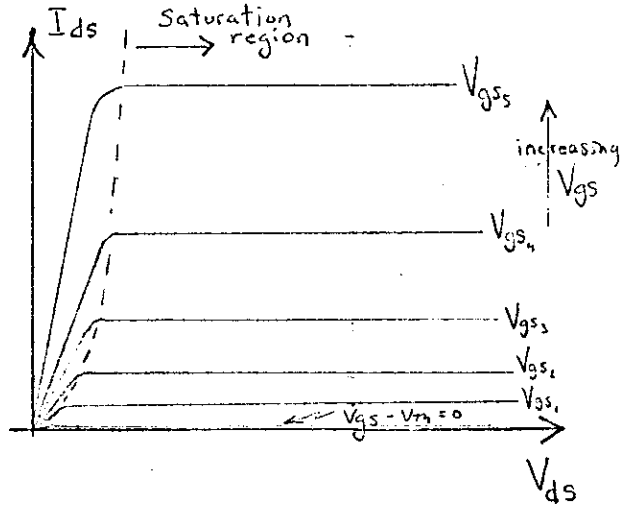


Fig.1d. CURRENT VS VOLTAGE

Transit time:

$$\tau = \text{L/velocity} = 2L/\mu E = 2L^2/[\mu(V_{gs} - V_{th})] \qquad \text{[eq.1]}$$

The proportionality constant μ is called the *mobility* of the charge carriers, in this case electrons, under the influence of an electric field in the conducting material of the channel region. It is a velocity per unit electric field (cm^2/volt-sec). The factor of 2 arises because of the non-uniformity of the electric field in the channel region[1]. We shall see that the transit time is the fundamental time unit of the entire integrated system.

The amount of negative charge in transit is just the gate capacitance times the voltage on the gate in excess of the threshold voltage. The capacitance of two parallel conductors of area A, separated by insulating material of thickness d, equals $\varepsilon A/d$. The proportionality constant ε is called the permittivity of the insulating material, and has a simple interpretation. It is the capacitance of parallel conductors of area A = 1 cm^2, separated by a thickness d = 1 cm of the insulator material, and is in the units farad/cm. Therefore, the gate capacitance equals $\varepsilon WL/D$. Thus the charge in transit is given by eq. 2, and the current is given by eq. 3.

Charge in transit:

$$Q = - C_g(V_{gs} - V_{th}) = - \frac{\varepsilon WL}{D}(V_{gs} - V_{th}) \qquad \text{[eq.2]}$$

Current:

$$I_{ds} = - I_{sd} = - \frac{\text{charge in transit}}{\text{transit time}} = \frac{\mu\varepsilon W}{2LD}(V_{gs} - V_{th})^2 \qquad \text{[eq.3]}$$

Note that in eq. 1, the entire drain to source voltage was not available for reducing the transit time. Drain voltage in excess of one threshold below the gate voltage creates a short, high electric field region adjacent to the drain which the carriers cross very quickly. However, the electric field in the major portion of the channel from the source up to this point is proportional to $V_{gs} - V_{th}$, as shown in figure 1e.

Although the above equations are greatly simplified, they provide sufficient information to make many design decisions with which we will be faced, and also give us insight into the scaling of devices to smaller sizes. In particular, the transit time τ can be viewed as the basic time unit of any system we shall build in the integrated technology. In almost all situations, the fastest operation which we can perform is to transfer a signal from the gate of one MOS transistor onto the gate of another. In the absence of stray capacitance, the transit time is the minimum time in which one transistor can transfer a charge from its gate onto the gate of the subsequent transistor. For example, to transfer a charge from one transistor onto two transistors identical to it requires a minimum of two transit times. Thus, the transit time of the basic transistor in an integrated system can be viewed as the unit of time in which all other times in the system are scaled.

The Basic Inverter

The first logic circuit we will describe is the basic digital inverter. Analysis of this circuit is then easily extended to analysis of basic NAND and NOR logic gates. The inverter's logic function is to produce an output which is the complement of its input. When describing the logic function of circuits in LSI systems, we assign the value logic-1 to voltages equaling or exceeding some defined logic threshold voltage, and logic-0 to voltages less than this threshold voltage.

Were there an efficient way to implement resistors in the MOS technology, we could build a basic digital inverter circuit using the configuration of figure 2a. Here, if the inverter input voltage V_{in} is less than the transistor threshold voltage V_{th}, then the transistor is switched off, and V_{out} is "pulled-up" to the positive supply voltage VDD. In this case the output is the complement of the input. If V_{in} is greater than V_{th}, the transistor is switched on and current flows from the VDD supply through the load resistor R to GND. If R were sufficiently large, V_{out} could be "pulled-down" well below V_{th}, thus again complementing the input. However, the resistance per unit length of minimum width lines of various available conducting elements is far less than the effective resistance of the switched on MOSFET. Implementing a sufficiently large inverter load using resistive lines would require a very large area compared to that occupied by the transistor itself.

To circumvent this problem a depletion mode MOSFET is used as a pullup for the basic inverter circuit, symbolized and configured as shown in figure 2b. In contrast to the usual

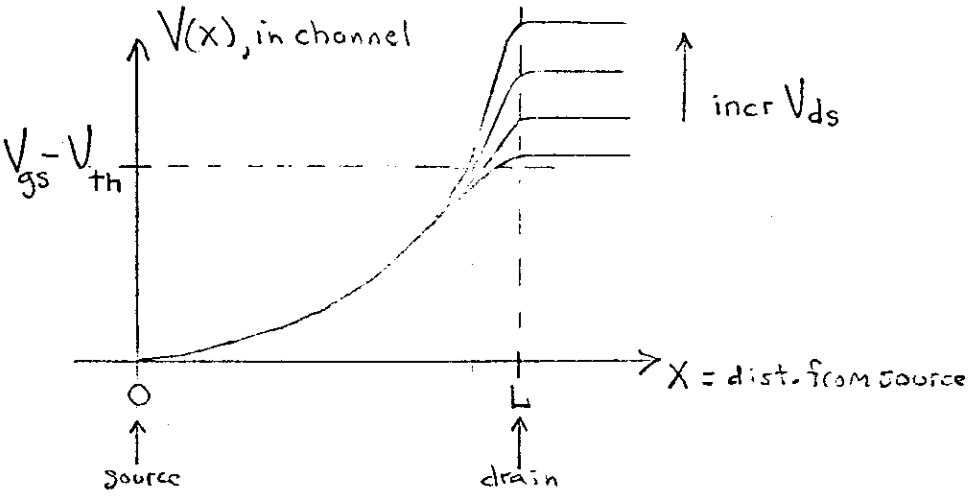


Fig. 1e. Voltage Profile Across Channel

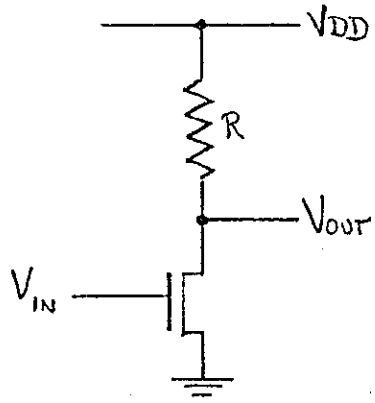
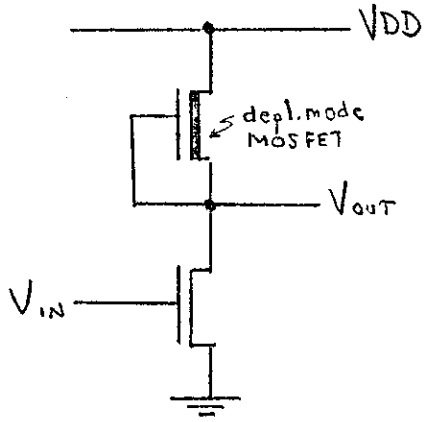
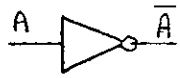


Fig. 2a. An Inverter





A	$\bar{A}$
0	1
1	0

Fig 2b. The Basic Inverter, Circuit Diagram, Logic Symbol, Logic Function

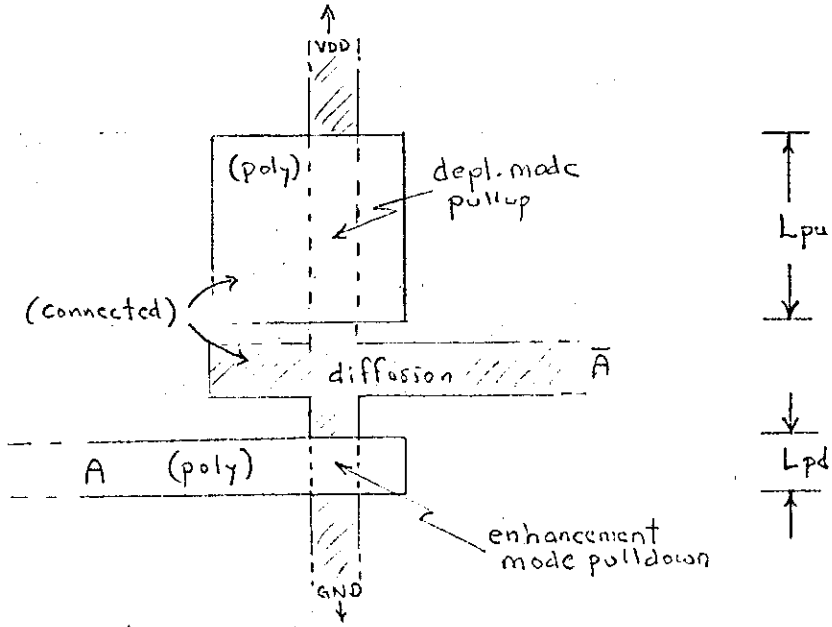


Fig 2c. Basic Inverter Layout

enhancement mode transistor, the depletion mode transistor has a threshold voltage, V_{dep}, that is less than zero. During fabrication, one of the masks is used to select any desired subset of transistors in the integrated system for processing as depletion mode transistors. For a depletion mode transistor to turn off, it requires a voltage on its gate relative to its source that is more negative than V_{dep}. But in this configuration its gate is connected to its source, and thus it is always turned on. Hence, when the enhancement mode transistor is turned off, for example by connecting zero voltage to its gate, the output of the inverter will be equal to VDD. We will find that for reasonable ratios of the gate geometries of the two transistors, input voltages above a defined logic threshold voltage, V_{inv}, will produce output voltages below that logic threshold voltage.

The top view of the layout of an inverter on the silicon surface is sketched in figure 2c. It consists of two polysilicon regions overhanging a path in the diffusion level which runs between VDD and GND. This arrangement forms the two MOS transistors of the inverter. The inverter input A is connected to the poly forming the gate of the lower of the two transistors. The pullup is formed by connecting the gate of the upper transistor to its source. The fabrication details of such connections will be described in a later chapter. The output of the inverter is shown leaving, on the diffusion level, from between the drain of the pulldown and the source of the pullup. The pullup is a depletion mode transistor, and it is usually several times longer than the pulldown, to insure proper inverter transfer characteristics.

Figures 3a and 3b show the I_{ds} versus V_{ds} and V_{gs} characteristics of a typical pair of MOS transistors used to implement an inverter. The characteristics of the pullup and pulldown differ only in their thresholds and, therefore, in the relative locations of their saturation regions.

We can now use a graphical construct to determine the actual transfer characteristic, V_{out} vs V_{in}, of the inverter circuit. From figures 2a and 2b we see that the V_{ds}(enh) of the enhancement mode transistor equals VDD minus V_{ds}(dep) of the depletion mode transistor. Also, V_{ds}(enh) equals V_{out}. In a steady state and with no current drawn from the output, the I_{ds} of the two transistors are equal. Since the pullup has its gate connected to its source, only one of its characteristic curves is relevant, namely the one for V_{gs}(dep) = 0. Taking these facts into account, we begin the graphical solution by superimposing plots of I_{ds}(enh) vs V_{ds}(enh), and the one plot of I_{ds}(dep) vs [VDD - V_{ds}(dep)]. Since the currents in both transistors must be equal, the intersections of these sets of curves yields V_{ds}(enh) = V_{out}

versus $V_{gs}(enh) = V_{in}$. The resulting transfer characteristic is plotted in figure 3d.

Studying figures 3c and 3d, consider the effect of starting with $V_{in} = 0$ and then gradually increasing V_{in} towards VDD. While the input voltage is below the threshold of the pulldown transistor, no current flows in that transistor, the output voltage is constant at VDD, and the drain to source voltage across the pullup transistor is equal to zero. When V_{in} is first increased above the enhancement mode threshold, current begins to flow in the pulldown transistor. The output voltage slightly decreases as the input voltage is first increased above V_{th}. Subsequent increases in the input voltage rapidly lower the pulldown's drain to source voltage, until the point is reached where the pulldown leaves the saturation region. Then as V_{in} continues to increase, the output voltage asymptotically approaches zero. Figure 3d also shows the effect of changes in the transistor length to width ratios on the transfer characteristics and on the logic threshold voltage. The resistive impedance of the MOS transistor is proportional to the length to width ratio Z of its gate region. Using the subscript pu for the pullup transistor and pd for the pulldown transistor: If $Z_{pu} = L_{pu}/W_{pu}$ is increased relative to $Z_{pd} = L_{pd}/W_{pd}$, then V_{inv} decreases, and vice-versa. The gain, or negative slope of the transfer characteristic near V_{inv}, increases as Z_{pu}/Z_{pd} increases. The gain G must be substantially greater than one for digital circuits to function properly.

Inverter Logic Threshold Voltage

The most fundamental property of the basic inverter circuit is its logic threshold voltage, V_{inv}. The logic threshold here is *not* the same as V_{th} of the enhancement mode transistor, but is that voltage on the input of the enhancement mode transistor which causes an equal output voltage. If V_{in} is increased above this logic threshold, V_{out} falls below it, and if V_{in} is decreased below V_{inv}, V_{out} rises above it. The following simple analysis assumes that both pullup and pulldown are in saturation, so that equation 3 applies. Usually the pullup is not quite in saturation, but the following is still nearly correct. V_{inv} is approximately that input voltage which would cause saturation current through the pulldown transistor to be equal to saturation current through the pullup transistor. Referring to eq.3, we find the condition for equality of the two currents given in eq.4.

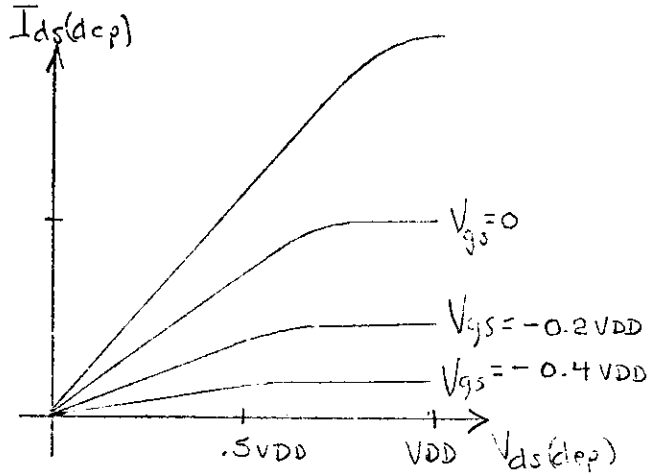


Fig 3a. Inverter Pullup Characteristics

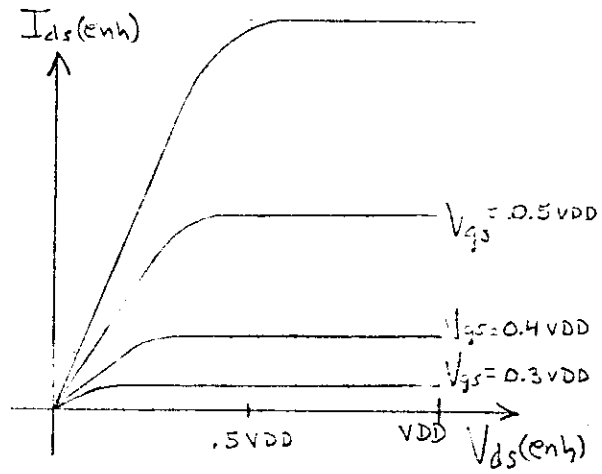


Fig 3b. Pulldown Characteristics

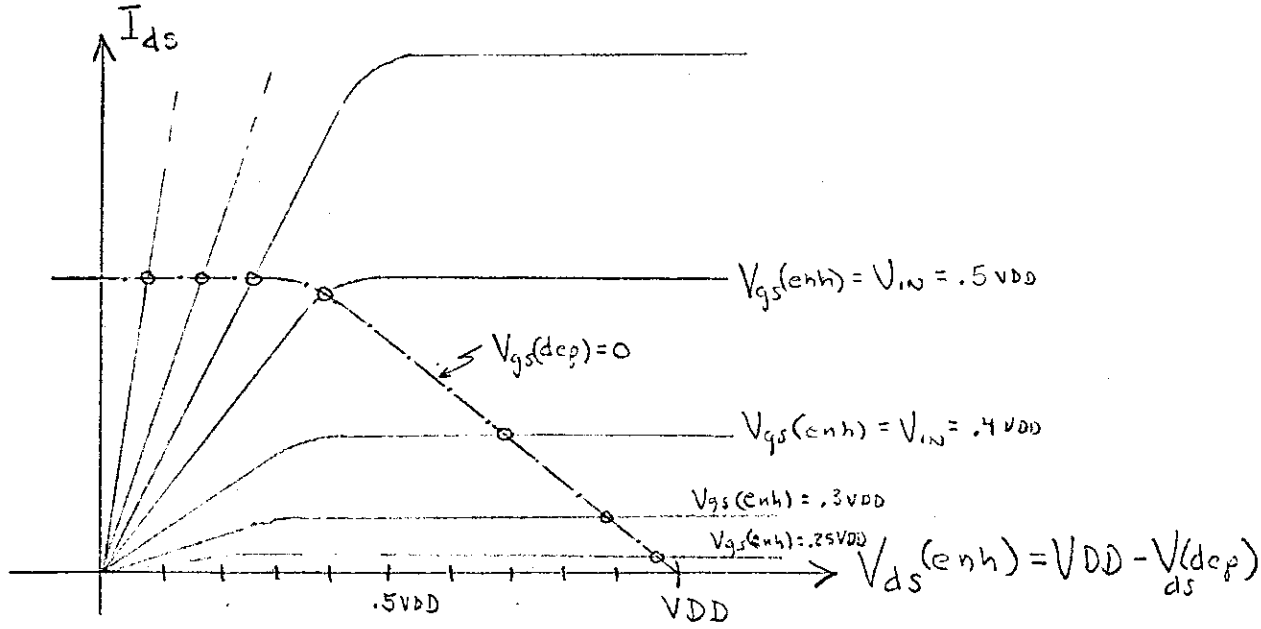


Fig 3c. $I_{ds(enh)}$ vs $V_{ds(enh)}$ & $I_{ds(dep)}$ vs $[VDD - V_{ds(dep)}]$

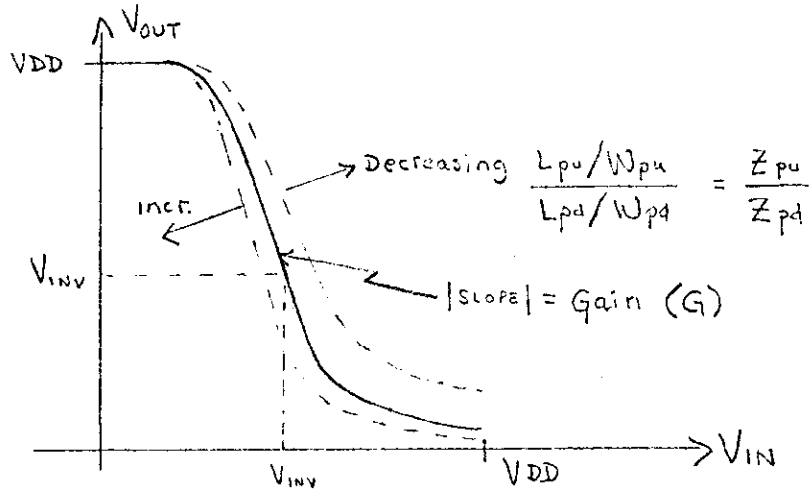


Fig. 3d Basic Inverter V_{OUT} vs V_{IN}

Currents equal when:

$$\frac{Wpd}{Lpd}(V_{inv} - V_{th})^2 = \frac{Wpu}{Lpu}(-V_{dep})^2 , \qquad \text{[eq.4]}$$

or thus when:

$$V_{inv} = V_{th} - V_{dep}/[Zpu/Zpd]^{1/2} \qquad \text{[eq.4a]}$$

Here we note that the current through the depletion mode transistor is dependant only on its geometry and threshold voltage V_{dep}, since its $V_{gs} = 0$. Note that V_{inv} is dependent upon the thresholds of both the enhancement and depletion mode transistors, and also the square root of the ratio of the $Z = L/W$ of the enhancement mode transistor to that of the depletion mode transistor.

Tradeoffs are possible between these system threshold voltages and the areas and current driving capability of transistors in the system's inverters. To maximize $(V_{gs} - V_{th})$ and increase the pulldowns' current driving capability for a given area, V_{th} should be as low as possible. However, if V_{th} is too low, inverter outputs won't be driveable below V_{th}, and inverters won't be able to turn off transistors used as simple switches. The original choice of $V_{th} \sim 0.2VDD$ is a reasonable compromise here.

Similarly, to maximize the current driving capability of pullups of given area, we might set the system's V_{dep} as far negative as possible. However, eq. 4a shows that for chosen V_{inv} and V_{th}, decreasing V_{dep} requires an increase in L_{pu}/W_{pu}, typically leading to an increase in pullup area. The compromise made in this case is usually as follows. The negative threshold of depletion mode transistors is set during fabrication such that with gate tied to source, they turn on approximately as strongly as would an enhancement mode transistor with VDD connected to its gate and its source grounded. In other words, depletion mode and enhancement mode transistors of equal gate dimensions would have equal drain to source currents under those conditions. Applying eq.3 in those conditions we find that:

$$(-V_{dep})^2 \sim (VDD - V_{th})^2.$$

Therefore, $-V_{dep} \sim (VDD - V_{th})$, and $V_{dep} \sim -0.8VDD$. While adjustments in the details of this choice are often made in the interest of optimization of processes for a particular product, we will assume here this approximate equality of turn-on voltages of the two transistor types for the sake of simplicity. Substituting this choice of V_{dep} into eq.4a,

we find that for V_{th} small compared to VDD:

$$V_{inv} \sim VDD/[Zpu/Zpd]^{1/2} \qquad \text{[eq.4b]}$$

In general it is desirable that the margins around the inverter threshold be approximately equal, i.e., that the inverter threshold, V_{inv}, lie approximately midway between VDD and ground. We see from eq.4b that this criterion is met by a ratio of pullup Z to pulldown Z of approximately 4:1. We will see later that the choice of $V_{dep} \sim VDD - V_{th}$, producing a ratio of 4:1 here, will lead to a balancing of performances in certain other important circuits.

Inverter Delay

A minimum requirement for an inverter is that it drive another identical to itself. Let us analyze the delay through a string of inverters of identical dimensions since this is the simplest case in which we can estimate the performance. Assume that the ratio of Z of the pullups to Z of the pulldowns equals k. Inverters connected in this way are shown in Fig. 4a. We will sometimes use the alternative pullup symbol shown ("resistor with gate"), to clarify its functional purpose.

Let us assume that prior to t = 0, the voltage at the input of the first inverter is zero, and hence, the voltage output of the second inverter will be low. At time t=0, let us place a voltage equal to VDD on the input of the first inverter and follow the sequence of events which follows. The output of the first inverter, which leads to the gate of the second inverter, will initially be at VDD. Within approximately one transit time, the pulldown transistor of the first inverter will remove from this node an amount of charge equal to VDD times the gate capacitance of the pulldown of the second inverter. The pullup transistor of the second inverter is now faced with the task of supplying a similar charge to the gate of the third inverter, to raise it to VDD. Since it can supply at most only 1/k'th of the current that can be supplied by the pulldown transistor, the delay in the second inverter stage is approximately k times that of the first.

It is thus convenient to speak of the inverter pair delay which includes the delay for one lowgoing transition and one highgoing transition. This inverter pair delay is approximately (1+k) times the transit time, as shown in figure 4. The fact that the rising transition is

slower than the falling transition by approximately the inverter transistors' geometry ratios is an inherent characteristic of any ratio type logic. It is not true of all logic families. For example, in families such as complementary MOS where there are both pMOS and nMOS devices on the same silicon chip and both types operate strictly as pulldown enhancement mode devices, any delay asymmetry is a function of the difference in mobilities of the p and n type charge carriers rather than of the transistor geometrical ratios.

Fig. 4b shows an inverter driving the inputs of several other inverters. In this case, for a fanout factor f, it is clear that in either the pullup or pulldown direction, the active device must supply f times as much charge as it did in the case of driving a single input. In this case, the delay both in the up and downgoing directions is increased by approximately the factor f. In the case of the downgoing transition, the delay is approximately f times the transit time of the pulldown transistor, and in the case of the upgoing transition, the delay is approximately the inverter ratio k times the fanout factor times the pulldown transit time.

In the discussions of transit time given earlier, it was assumed that both the depletion mode pullup device and the enhancement mode pulldown device were operating in the saturation region where they act like current sources. It was also assumed that all capacitances were constant, and not a function of voltage. This condition is not strictly met in the technology we are discussing. Delay calculations given in this text are based on a "switching model" where individual stages spend a small fraction of their time in the mid-range of voltages around V_{inv}. This assumption introduces a small error of the order of 1/G. Because of these and other second order effects, the switching times actually observed vary somewhat from those derived.

Parasitic Effects

In integrated systems, capacitances of circuit nodes are due not only to the capacitance of gates connected to the nodes, but also include capacitances to ground of signal paths connected to the nodes and other stray capacitances. These other capacitances, sometimes called parasitic or stray capacitances, are not negligible. While gate capacitances are typically an order of magnitude greater per unit area than the capacitances of the signal paths, the signal paths may sometimes be larger in area than the associated gate regions. Therefore, a substantial fraction of the delay encountered may be accounted for by stray capacitance rather than by the inherent properties of the active transistors. In the simplest

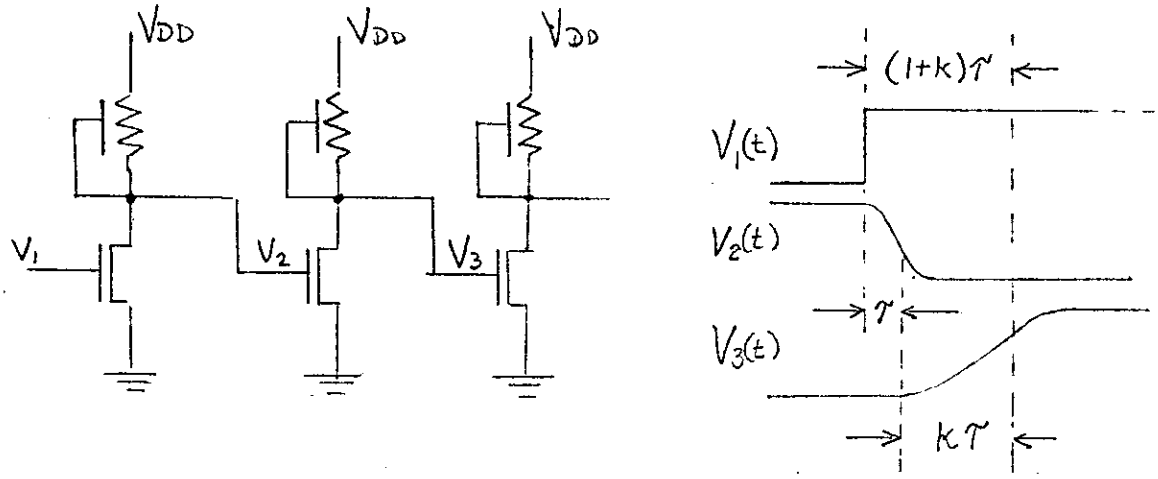
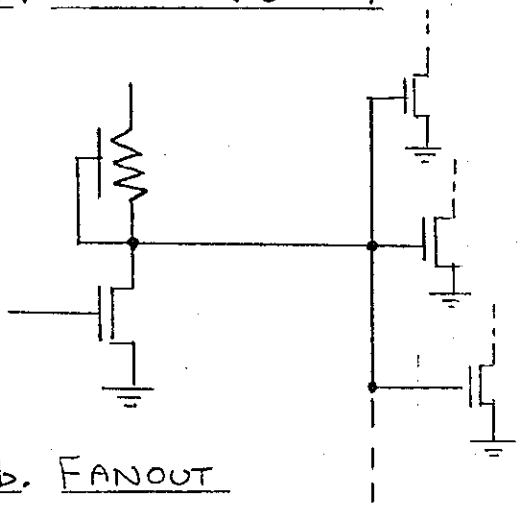


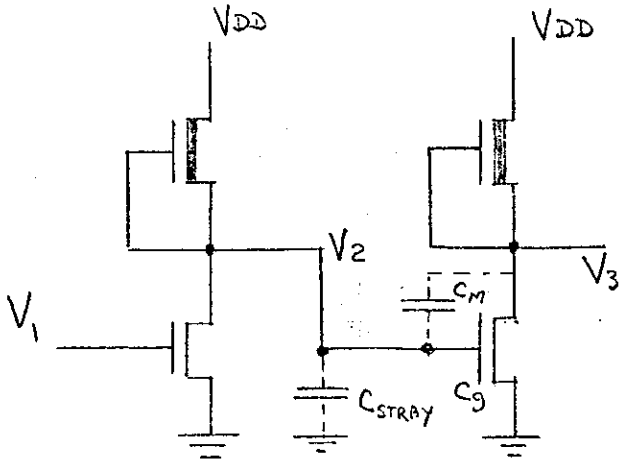
Fig. 4a. INVERTER DELAY



for fanout of f:

down delay ~ fτ
up delay ~ kfτ

Fig. 4b. FANOUT



START: $V_2 = 0$, $V_3 = 1$

$Q_g = 0$, $Q_M = -C_M$

FINISH: $V_2 = 1$, $V_3 = 0$

$Q_g = C_g$, $Q_M = +C_M$

Fig. 5. MILLER EFFECT Tot. Effect. Input Cap = $C_g + 2C_M + C_{STRAY}$

case where the capacitance of a node is increased by the presence of parasitic area attached to the node, the delays can be accounted for by simply increasing the transit time by the ratio of the total capacitance to that of the gate of the transistor being driven. We now require time to supply charge not only to the gate itself but also to the parasitic capacitance.

There is one type of parasitic, however, which is not accounted for so simply. All MOS transistors have a parasitic capacitance between the drain edge of the gate and the drain node. This effect is shown schematically in figure 5. In an inverter string such as that mentioned earlier, this capacitance will be charged in one direction for one polarity of input, and in the opposite direction for the opposite polarity input. Thus, on a gross scale its effect on the system is twice that of an equivalent parasitic capacitance to ground. Therefore, gate to drain capacitances should be approximately doubled, and added to the gate capacitance C_g and the stray capacitances, to account for the total capacitance of the node and thus for the effective delay time of the inverter. The effective inverter pair delay then = $\tau(1+k)C_{total}/C_g$.

Driving Large Capacitive Loads

As we have seen, the delay per inverter stage is multiplied by a fanout factor. The overall performance of a system may be seriously degraded if it contains any large fanouts, where one circuit within the system is required to drive a large number of circuits similar to itself. As we shall see, this situation often occurs in the case of control drivers which are required to drive a large number of inputs to memory cells or logic function blocks. A similar and even more serious problem is driving wires which go off the silicon chip to other chips or input/output devices. In such cases the ratio of the capacitance which must be driven to the inherent capacitance of a gate circuit on the chip is often many orders of magnitude, causing a serious delay and a degradation of system performance.

Consider how we may drive a capacitive load C_L in the minimum possible time given that we are starting with a signal on the gate of an MOS transistor of capacitance C_g. Define the ratio of the load capacitance to the gate capacitance, C_L/C_g, as Y. It seems intuitively clear that the optimum way to drive a large capacitance is to use our elementary inverter to drive a larger inverter and that larger inverter to drive a still larger inverter until at some point the larger inverter is able to drive the load capacitance directly. Using an argument similar to the fanout argument it is clear that for one inverter to drive another inverter, where the

second is larger in size by a factor of f, results in a delay f times the inherent inverter delay, τ. If n such stages are used, each larger than the previous by a factor f, then the total delay of the inverter chain is $nf\tau$, where f^n equals Y. Note that if we use a large factor f, we can get by with few stages, but each stage will have a long delay. If we use a smaller factor f, we can shorten the delay of each stage, but are required to use more stages. What value of n minimizes the overall delay for a given Y? We compute this value as follows:

Delay of one stage $= f\tau = (Y)^{1/n}\tau$,

Thus total delay is $= F(n) = n(Y)^{1/n}\tau$, and [eq.5]

$$dF(x)/dx = \tau(Y)^{1/x}[1 - \ln Y/x]$$

The function $dF(x)/dx$ equals 0, and thus F(x) is minimum, when $x = \ln Y$. Therefore the total delay is minimized when $n \sim \ln Y$. Total delay is minimized when each stage is larger than the previous one by a factor of e, the base of natural logarithms. Minimum total delay is the elementary inverter delay τ times e times the natural logarithm of the ratio of the load capacitance to the elementary inverter capacitance.

$$\text{Min. total delay} \sim \tau e[\ln(C_L/C_g)] \qquad \text{[eq.5a]}$$

Space vs Time

From the results of the sections on inverter delay, parasitic effects, and driving large capacitances, we see that areas and distances on the silicon surface trade off against delay times. For an inverter to drive another inverter some distance away, it must charge not only the gate capacitance of the succeeding inverter but also the capacitance to ground of the signal path connecting the two. Increasing the distance between the two inverters will therefore increase the inverter pair delay. This effect can be counterbalanced by increasing the area of the first inverter, so as to reduce the ratio of the load capacitance to the gate capacitance of the first inverter. But the delay of some previous driving stage is then increased. There is no way to get around the fact that transporting a signal from one node to another some distance away requires charging or discharging capacitance, and therefore

takes time. Note that this is not a velocity of light limitation as is often the case outside the chip. The times are typically several orders of magnitude longer than those required for light to traverse the distances involved. To minimize both the time and space required to implement system functions, we will tend to use the smallest possible circuits and locate them in ways which tend to minimize the interconnection distances.

The results of a previous section can be used here to illustrate another interesting space vs time effect. Suppose that the minimum size transistors of an integrated system have a transit time τ and gate capacitance C_g. A minimum size transistor within the system produces a signal which is then passed through successively larger inverting logic stages and eventually drives a large capacitance C_L with minimum total delay equal to t_{min}. With the passage of time, fabrication technology improves and we replace the system with one in which the circuits are all scaled down in size by dividing by a factor α (the motivation for this is clear: the new system may contain α^2 as many circuits). As described in a later section, we will find that the transit times of the smallest circuits will now be $\tau' = \tau/\alpha$, and their gate capacitance will be $C_g' = C_g/\alpha$. Referring to equation 5a., we find that the new minimum total delay, t_{min}', to drive C_L is:

$$t_{min}' = t_{min}[(1/\alpha)\ln(\alpha)]$$

Therefore, as the inverters scale down and τ gets smaller, more inverting logic stages are required to obtain the minimum offchip delay. Thus the relative delay to the outside world becomes larger. However, the absolute delay becomes smaller.

Basic NAND and NOR Logic Circuits

NAND and NOR logic circuits may be constructed in integrated systems as simple expansions of the basic inverter circuit. The analysis of the behavior of these circuits, including their logic threshold voltages, transistor geometry ratios and time delays, is also a direct extension of the analysis of the basic inverter.

The circuit layout diagram of a two input NAND gate is shown in figure 6a. The layout is that of a basic inverter with an additional enhancement mode transistor in series with the pulldown transistor. NAND gates with more inputs may be constructed by simply adding more transistors in series with the pulldown path. The electrical circuit diagram, truth table

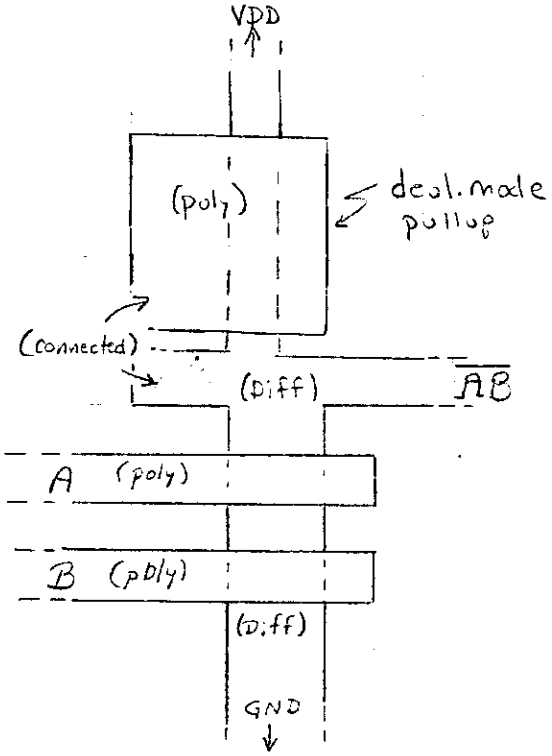
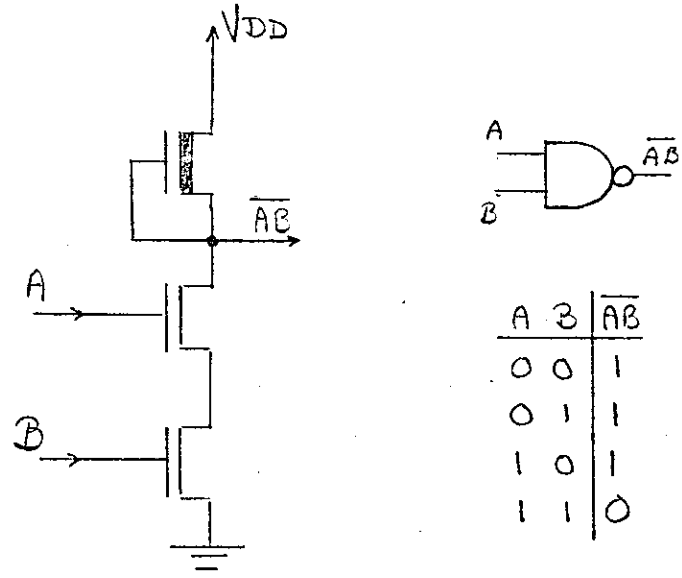


Fig. 6a. NAND GATE
(Top view)



A	B	$\overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

Fig. 6b. NAND GATE, CIRCUIT DIAGRAM, LOGIC SYMBOL, LOGIC FUNCTION

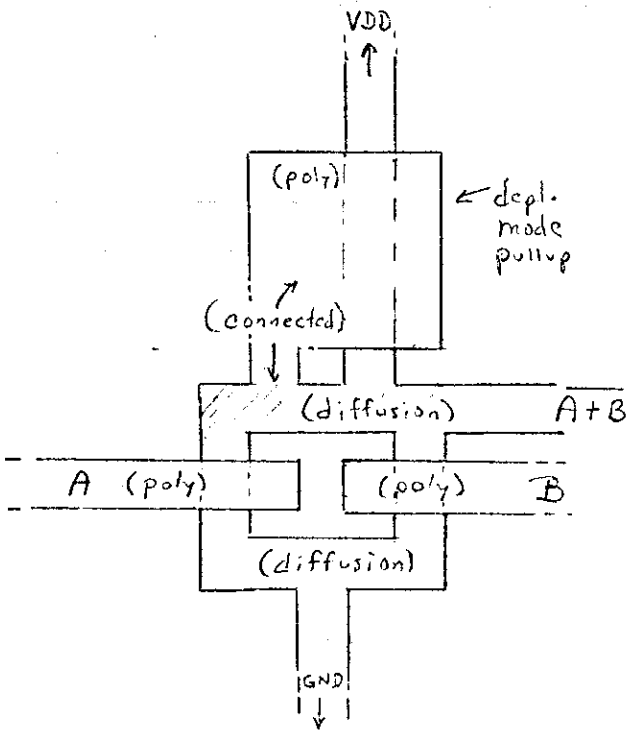
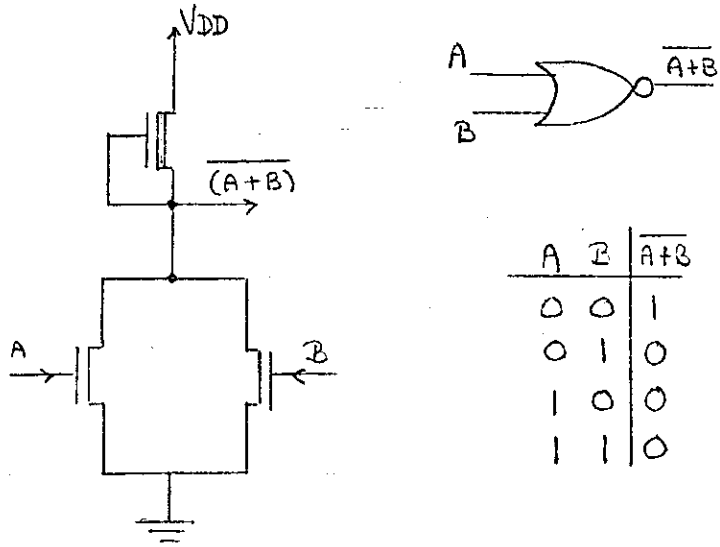


Fig. 6c. NOR GATE
(Top View)



A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Fig 6.d NOR GATE, CIRCUIT DIAGRAM LOGIC SYMBOL, LOGIC FUNCTION

and logic symbol for the two input NAND gate are shown in figure 6b. If either of the inputs A or B is a logic-0, the pulldown path is open and the output will be high, and therefore a logic-1. For the output to be driven low, to logic-0, both inputs must be high, at logic-1. The logic threshold voltage of this NAND gate is calculated in a similar manner to that of the basic inverter, except equation 4b is rewritten with the length of the pulldowns replaced with the sum of the lengths of the two pulldowns (assuming their widths are equal) as follows:

$$V_{thNAND} \sim VDD/[(Lpu/Wpu)/((Lpd_a+Lpd_b)/Wpd)]^{1/2}$$

This equation indicates that as pulldowns are added in series to form NAND gate inputs, the pullup length must be enlarged to hold the logic threshold voltage constant. The logic threshold voltage of an n-input NAND gate, assuming all the pulldowns have equal geometries, is:

$$V_{thNAND} \sim VDD/[(Lpu/Wpu)/(nLpd/Wpd)]^{1/2}$$

As inputs are added and pullup length is increased, the delay time of the NAND gate is also correspondingly increased, both for rising and falling transitions.

$$\tau_{NAND} \sim n\tau_{inv}$$

The circuit layout diagram of a two input NOR gate is shown in figure 6c. The layout is that of a basic inverter with an additional enhancement mode transistor in parallel with the pulldown transistor. Additional inputs may be constructed by simply placing more transistors in parallel with the pulldown path. The circuit diagram, truth table and logic symbol for the two input NOR gate are shown in figure 6d. If either of the inputs A or B is a logic-1, the pulldown path to ground is closed and the output will be low, and therefore a logic-0. For the output to be driven high, to logic-1, both inputs must be low, at logic-0. If one of its inputs is kept at logic-0, and the other swings between logic-0 and logic-1, the logic threshold voltage of the this NOR gate is the same as that of a basic inverter of equal pullup to pulldown ratio. If this ratio were 4:1 to provide equal margins, then $V_{thNOR} \sim$ VDD/2 with only one input active. However, if both pulldowns had equal geometries, and if both inputs were to move together between logic-0 and logic-1, V_{thNOR} would be reduced to $\sim VDD/(8)^{1/2}$. The logic threshold voltage of an n-input NOR circuit decreases as a function of the number of active inputs (inputs moving together from logic-0 to logic-1).

The delay time of the NOR gate with one input active is the same as that of an inverter of equal transistor geometries, except for added stray capacitance. Its delay time for falling transitions is decreased as more of its inputs are active.

Super Buffers

As we have noted, ratio type logic suffers from an asymmetry in its ability to drive capacitive loads. This asymmetry results from the fact that the pullup transistor has of necessity less driving capability than the pulldown transistor. There are, however, methods for avoiding this asymmetry. Shown in figures 7a and 7b are circuits for inverting and a non-inverting drivers, which are approximately symmetrical in their capability of sourcing or sinking charge into a capacitive load. Drivers of this type are called *super buffers.*

Both types of super buffer are built using a depletion mode pullup transistor and an enhancement mode pulldown transistor, with a ratio of Z's of approximately 4:1 as in the basic inverter. However, the gate of the pullup transistor, rather than being tied to its source, is tied to a signal which is the complement of that driving the pulldown transistor. When the pulldown transistor gate is at a high voltage, the pullup transistor gate will be approximately at ground, and the current through the super buffer will be similar to that through a standard inverter of the same size. However, when the gate of the pulldown transistor is put to zero, the gate of the pullup transistor will go rapidly to 5 volts since it is the only load on the output of the previous inverter, and the depletion mode transistor will be turned on at approximately twice the drive which it would experience if its gate were tied to its source. Since the current from a device in saturation goes approximately as the square of the gate voltage, the current sourcing capability of a super buffer is approximately four times that of a standard inverter. Hence, the current sourcing capability of its pullups is approximately equal to the current sinking capability of its pulldowns, and wave forms from super buffers driving capacitive loads are nearly symmetrical.

The effective delay time, τ, of super buffers is thus reduced to approximately the same value for highgoing and lowgoing wave forms. Needless to say, when large capacitive loads are to be driven, super buffers are universally used. The arguments used in the last section to determine how many stages are used to drive a large capacitive load from a small source apply directly to super buffers as well.

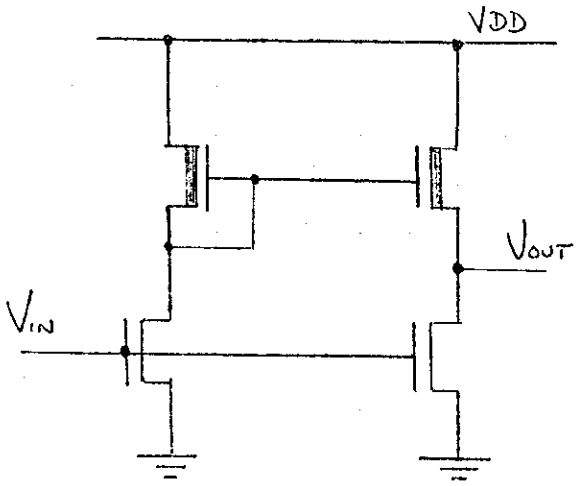


Fig 7a. INVERTING SUPER BUFFER

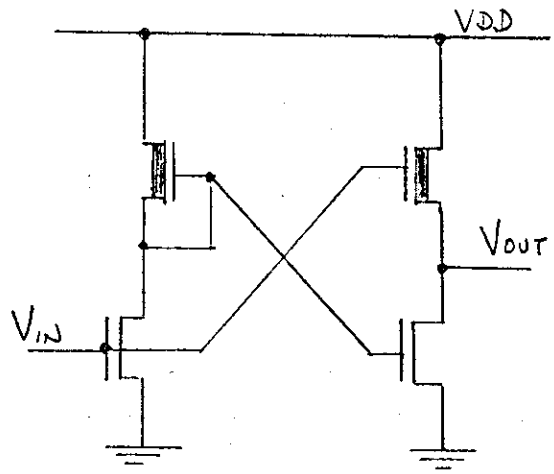


Fig 7b. NON-INVERTING SUPER BUFFER

A Closer Look at the Electrical Parameters

Up to this point we have talked in very simple terms about the properties of the MOS transistors. They have had a capacitance associated with their gate input and a transit time associated with current flowing from the source to the drain. In the saturation region where the drain to source voltage is high enough, the transistor acts as a current source. The current flowing from drain to source is a function only of the gate to source voltage as given by eq. 3. However, MOS transistors act as current sources only over a fraction of the range encountered in normal operation. Figure 9a summarizes the various regions of MOS transistor operation. Note that once the voltage from drain to source is smaller than the gate voltage minus the threshold voltage, the field in the channel is determined by the drain to source voltage rather than by the gate voltage. As the drain to source voltage decreases below this point, the transit time increases, since the electric field in the channel is lower. For low drain to source voltages, the characteristic of the device is given[1] in eq. 7a.

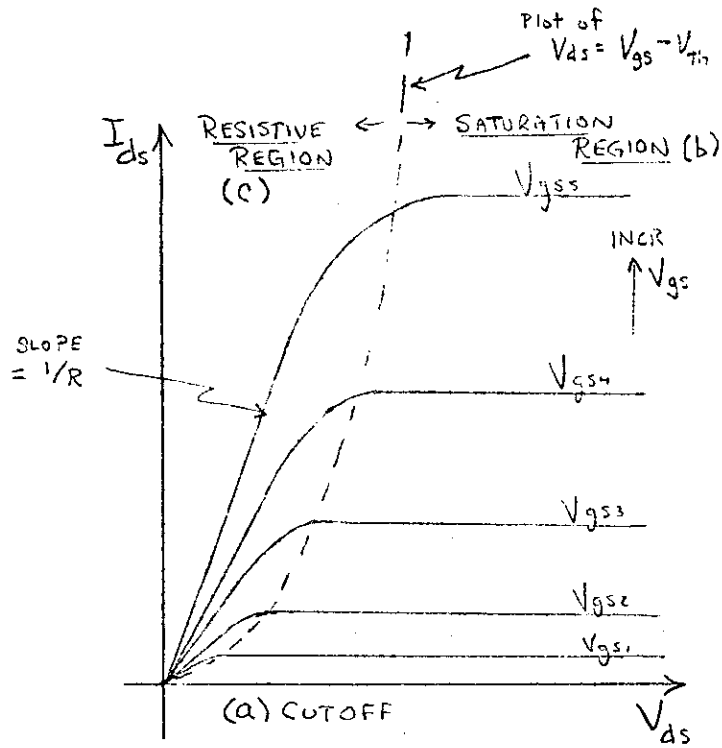
$$I_{ds} = Q/\tau = \mu C_g[(V_{gs} - V_{th})\, V_{ds} - (V_{ds}^2)/2]/L^2 \qquad \text{[eq.7a]}$$

For very low drain to source voltages, we may neglect the V_{ds}^2 term, and the characteristic of the device is given in eq. 7b.

$$I_{ds} = Q/\tau = \mu C_g[(V_{gs} - V_{th})\, V_{ds}]/L^2 \qquad \text{[eq.7b]}$$

Note that in this region, the drain current is proportional to the source-drain voltage and also to the gate voltage above threshold. The electric field along the channel is uniform at very low V_{ds}, and hence there is no factor of 2 as there was in equation 3. Any device with a current through it proportional to the voltage across it, may be viewed as a resistor, and in the case of an MOS device with *low* drain to source voltage, the resistance is given by eq. 8. Note, however, that the resistance increases as V_{ds} increases towards V_{gs}-V_{th}, as shown by the decreasing slopes of the transfer characteristics in figure 9a as the device nears saturation.

$$V_{ds}/I_{ds} = R = L^2/[\mu C_g(V_{gs} - V_{th})] \qquad \text{[eq.8]}$$



(a) CUTOFF:

$V_{gs} < V_{th}$, $I_{ds} = 0$

(b) SATURATION REGION:

$V_{gs} \geq V_{th}$, V_{ds} sufficiently high so $V_{gd} < V_{th}$. (i.e. $V_{ds} \geq V_{gs} - V_{th}$)

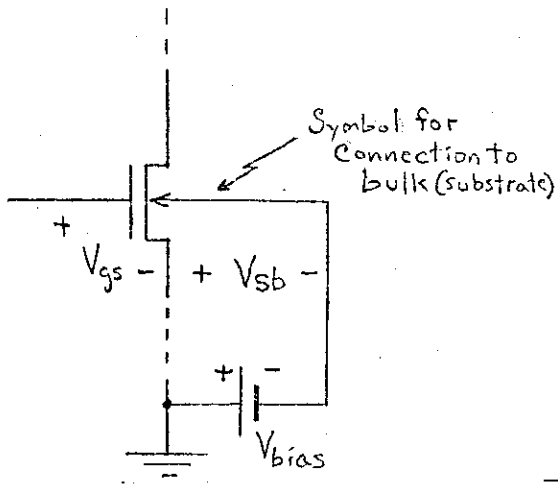
MOSFET ACTS AS CURRENT SOURCE WITH I_{ds} PROPORTIONAL TO $(V_{gs} - V_{th})^2$

(c) RESISTIVE REGION:

$V_{gs} \geq V_{th}$, V_{ds} sufficiently low so $V_{gd} \geq V_{th}$ (i.e. $V_{ds} < V_{gs} - V_{th}$)

MOSFET ACTS AS RESISTOR WITH RESISTANCE INV. PROP. TO $(V_{gs} - V_{th})$

Fig 9a. SUMMARY OF MOS TRANSISTOR CHARACTERISTICS



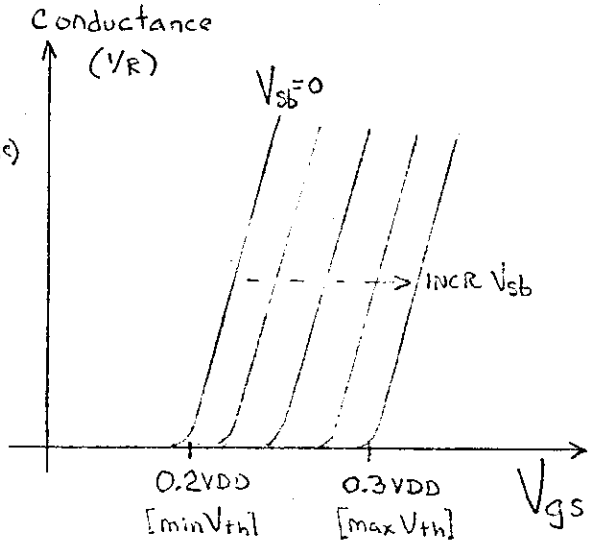


Fig. 9b THE BODY EFFECT

In both equations 7 and 8, the capacitance is the gate to channel capacitance of the turned on transistor. In the simple case where this transistor is driving the gate of another one identical to it, the time response of the system will be an exponential with a time constant RC, given in eq. 9. This time constant is similar to the transit time τ given in eq. 1.

$$RC_g = L^2/[\mu(V_{gs} - V_{th})] = \tau/2 \qquad \text{[eq.9]}$$

There is another electrical characteristic we may occasionally have to take into account. It turns out that the threshold voltage of an MOS transistor is not a constant, but varies slightly as a function of the voltage between the source terminal of the transistor and the silicon substrate, which is usually ground. This is the so called *body effect* which is illustrated in figure 9b. If the source to bulk (substrate) voltage, V_{sb} , equals zero, then V_{th} is at its minimum value of approximately 0.2 VDD. As V_{sb} is increased, V_{th} increases slightly. For typical processes, V_{th} reaches a maximum value of about 0.3 VDD for V_{sb} ~ VDD. V_{dep} is similarly affected, ranging from about -0.8 VDD to -0.7 VDD as the V_{sb} of a depletion mode MOSFET is raised from zero to VDD volts. As shown in figure 9b, it is possible to insert a fixed bias voltage between the circuit ground and the substrate (rather than just have these be identical). Such a *substrate bias* provides an electrical mechanism for setting the threshold to an appropriate value.

Depletion Mode vs Enhancement Mode Pullups

With its gate tied to VDD, a standard enhancement mode transistor would always be on, and thus could be used for a load device in inverting logic circuits. Early MOS processes used pullup devices of exactly this type.

In this section we will make a comparison of the rising transients of the two types of pullup circuits. As noted earlier, rising transients in ratio type logic are usually slower than falling transients, and thus rising transients generally have greater impact on system performance. In the simplest cases, this asymmetry in the transients results from the current sourcing capability of the pullup transistor being less than that of its pulldown counterpart. The simple intuitive time arguments given earlier are quite adequate for making estimates of system performance in most of these cases. However, there are situations in which the

transient time may be much longer than a naive estimate would indicate. The rising transient of the enhancement mode pullup is one of these.

A depletion mode pullup transistor feeding a capacitive load is shown schematically in figure 10a. Since $V_{gs} \geq V_{th}$ and $V_{gd} \geq V_{th}$, the pullup transistor is in the resistive region. The final stages of the rising transient are given by the following exponential:

$$V(t) = VDD[1 - e^{-t/(RC_L)}]$$

For this case of pullup transistor, V_{gs} is equal to zero, the threshold voltage is a negative V_{dep}, and the time-constant of the rising transient, derived from eq. 9, is given by eq. 10. Note, by the way, that RC_L is inversely proportional to $-V_{dep}$. As expected, the presence of a larger load capacitance has the effect of simply lengthening the transit time of the circuit by the ratio of the load capacitance to the gate capacitance of the pullup.

$$RC_L = L^2C_L/\mu C_g\,(-V_{dep}) = \tau\, C_L/2C_g \qquad \text{[eq.10]}$$

A somewhat more complicated situation is presented by an enhancement mode transistor sourcing charge into a capacitive load. This situation is shown schematically in Fig. 10b. Note that since $V_{gd} = 0$, the transistor is in saturation whenever $V_{gs} > V_{th}$. The problem with sourcing charge from the enhancement mode transistor is that as the voltage at the output node gets closer and closer to one threshold below VDD, the amount of current provided by the enhancement mode transistor goes down rapidly. The current dependence upon the output voltage V is given in eq. 11.

$$Q = -\frac{\varepsilon WL}{D}[(VDD - V_{th}) - V]$$

$$\tau = 2L^2/\mu[(VDD - V_{th}) - V]$$

$$I_{ds} = -Q/\tau = \frac{\mu\varepsilon W}{2LD}[(VDD - V_{th}) - V]^2 \qquad \text{[eq.11]}$$

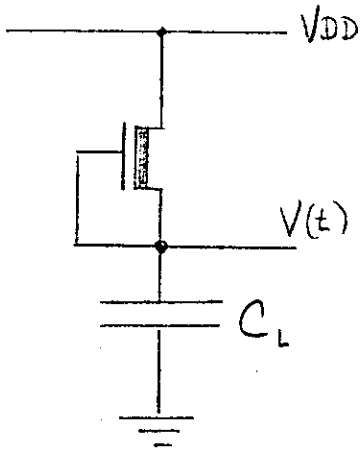


Fig 10a. DEPLETION MODE MOSFET PULLING UP CAPACITIVE LOAD

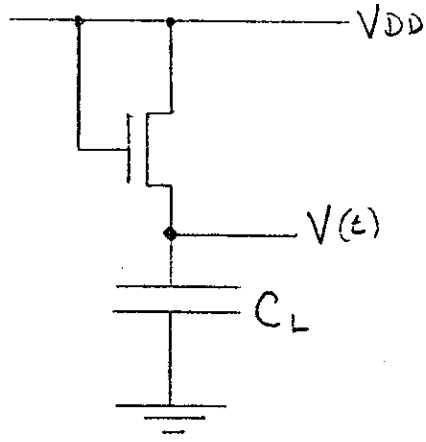


Fig 10b. ENHANCEMENT MODE MOSFET PULLING UP CAPACITIVE LOAD

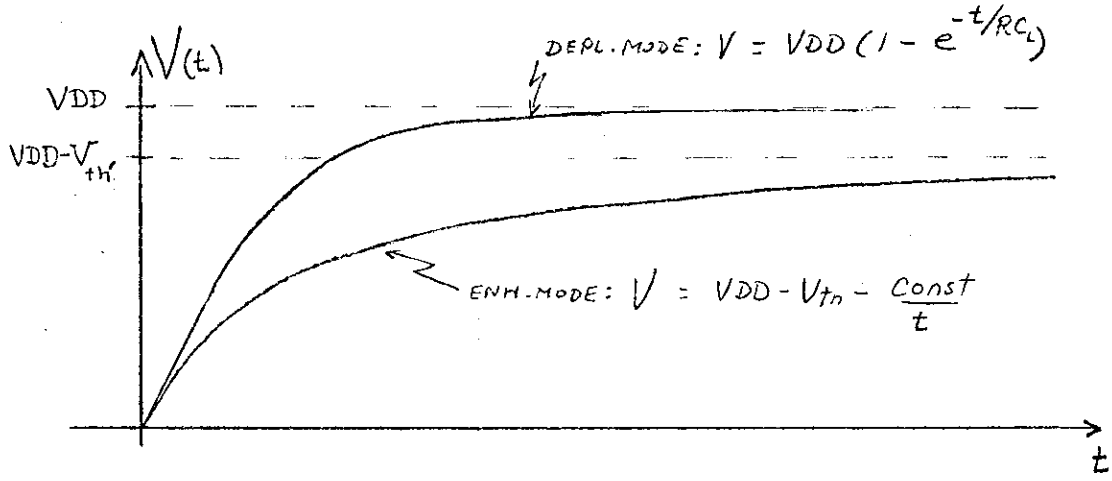


Fig. 10c. COMPARISON OF RISING TRANSIENTS FOR THE TWO TYPES OF LOADS.

The fact that the pullup current decreases as the output voltage nears its maximum value causes the rising transient from such a circuit to be of qualitatively different form than that of a depletion mode pullup. Equating $I_{ds} = C_L dV/dt$ with the expression in equation 11, and then solving for V(t), yields the rising voltage transient in equation 12:

$$V(t) = VDD - V_{th'} - C_L \frac{LD}{\mu\varepsilon Wt} \qquad \text{[eq.12]}$$

Note that in this configuration, the threshold voltage Vth' of the pullup is near its maximum value as V(t) rises towards VDD, due to the body effect. A comparison of the rising transients of the preceding two circuits, assuming the same load capacitance and the same pullup source current at zero output voltage, is shown in Fig. 10c. The rising transient for the depletion mode pullup transistor is crisp and converges rapidly towards VDD. However, the rising transient for the enhancement mode pullup transistor, while starting rapidly, lags far behind and within the expected time response of the system, never even comes close to one threshold below VDD. Even for very large t, $V(t) < VDD - V_{th'}$. The practical effect of this property of enhancement mode transistors is that circuits designed to work from a voltage derived from the output of such a circuit should be designed with an inverter threshold V_{inv} at a considerably lower value than those designed to work with the output of a depletion mode pullup circuit. In order to obtain equal inverter margins without sacrificing performance, we will normally use depletion mode pullups.

Delays in Another Form of Logic Circuitry

Enhancement mode transistors, when used in small numbers and driving small capacitive loads, may often be used as switches in circuits of simple topology to provide logic signal steering functions of much greater complexity than could be easily achieved in ratio type inverting logic. These circuits are reminiscent of relay switching logic, and transistors used in this way are referred to as "pass transistors" or "transmission gates". Example circuits using this type of design are given in Chapter 3. A particularly interesting example is the Manchester carry chain[5], used for propagating carry signals in parallel adders. In each stage of the adder a carry propagate signal is derived from the two input variables to the adder, and if it is desired to propagate the carry, this propagate signal is applied to the gate of an

enhancement mode pass transistor. The source of the transistor is carry-in to the present stage, and the drain of the transistor is carry-out to the next stage. In this way, a carry can be propagated from less to more significant stages of the adder without inserting a full inverter delay between stages. The circuit is shown schematically in Fig. 11a.

The delay through such a circuit does not involve inverter delays but is of an entirely different sort. A voltage along the chain divides into V_{ds} across each pass transistor. Thus V_{ds} is usually low, and the pass transistors operate primarily in the resistive region. We can think of each transistor as a series resistance in the carry path, and a capacitance to ground formed by the gate to channel capacitance of each transistor, and the strays associated with the source, drain, and connections with the following stage. An abstraction of the electrical representation is shown in Fig. 11b. The minimum value of R is the turned on resistance of each enhancement mode pass transistor, while the minimum value of C is the capacitance from gate to channel of the pass transistor. Strays will increase both values, in particular that of C. The response at the node labelled V_2 with respect to time is given in eq. 13. In the limit as the number of sections in the network becomes large, eq. 13 reduces to the differential form shown in eq. 14 where R and C are now the resistance and capacitance per unit length, respectively.

$$C\, dV_2/dt = [(V_1 - V_2) - (V_2 - V_3]/R \qquad \text{[eq.13]}$$

$$RC\, dV/dt = d^2V/dx^2 \qquad \text{[eq.14]}$$

Equation 14 is the well-known diffusion equation, and while its solutions are complex, in general the time required for a transient to propagate a distance x in such a system is proportional to x^2. One can see qualitatively that this might be so. Doubling the number of sections in such a network doubles both the resistance and the capacitance, and therefore causes the time required for the system to respond to increase by a factor of approximately four. The response of a system of n stages to a step function input is shown in Fig. 11c.

If we add one more pass transistor to such a chain of n pass transistors, the added delay through the chain is small for small n, but very large for large n. Therefore, it is highly desirable to group the pass transistors used for steering, multiplexing, and carry-chain type logic into short sections and interpose inverting logic between these sections. This approach applied to the carry chain is shown in figure 11d. The delay through a section of n pass

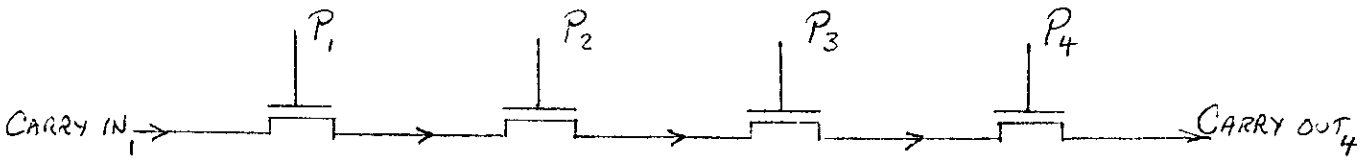


Fig. 11a. PASS TRANSISTOR CHAIN PROPOGATING A CARRY SIGNAL

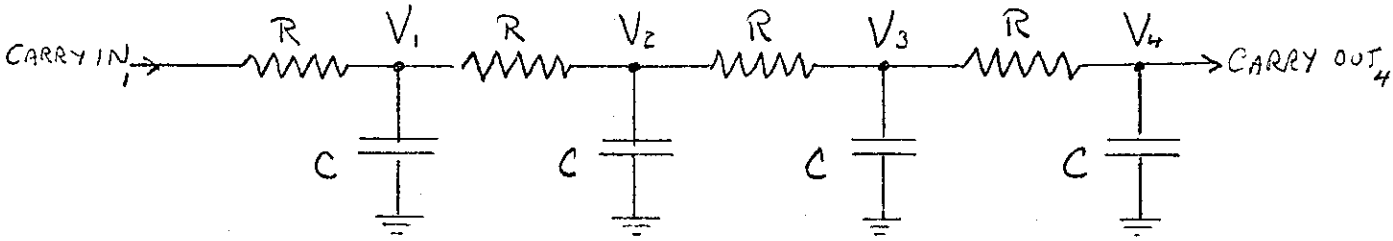


Fig 11b. EQUIVALENT CIRCUIT

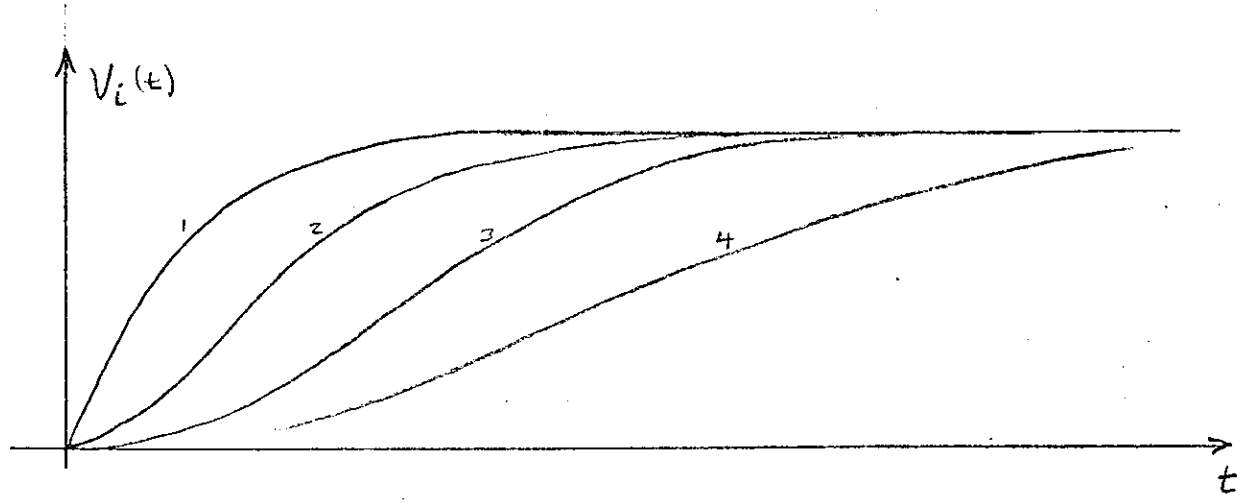


Fig. 11c. RESPONSE TO STEP FUNCTION INPUT

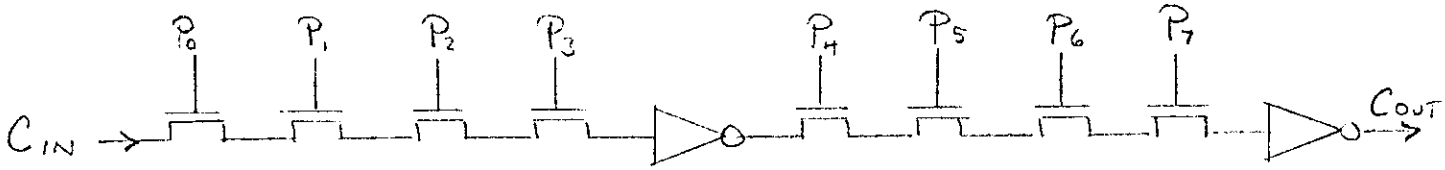


Fig 11d. MINIMIZING DELAY BY INTERPOSING INVERTERS

transistors is proportional to RCn^2. Thus the total delay is $kRCn^2$ plus the delay through the inverter τ_{inv}. The average delay per stage is given in eq. 15. To minimize the delay per stage, n should be chosen such that the delay through the n pass transistors is just equal to the inverter delay.

$$\text{Total delay} = kRCn^2 + \tau_{inv},$$

$$\text{Average delay/stage} = kRCn + \tau_{inv}/n \qquad \text{[eq.15]}$$

$$\text{Min. delay when: } kRCn^2 = \tau_{inv}$$

Since logic done by steering signals with pass transistors does not require static power dissipation, a generalization of this result may be formulated. It pays to put as much logic into steering type circuits as possible until there are enough pass transistors to delay the signal by approximately one inverting logic delay. At this point, the level of the signal can be restored by an inverting logic stage and the processing proceed from there.

The pass transistor has another important advantage over an inverting logic stage. When used to control or steer a logic signal, the pass transistor has only an input, control, and output connections. A NAND or NOR logic gate implementing the same function, in addition to containing two more transistors and thus occupying more area, also requires VDD and GND connections. As a result, the topology of interconnection of pass transistor circuits is far simpler than that of inverting logic circuits. This topological simplicity of pass transistor control gates is an important factor in the system design concepts developed in later chapters.

Pullup/Pulldown Ratios for Inverting Logic Coupled by Pass Transistors

Early in this chapter we found that when an inverting logic stage directly drives another such stage, a pullup to pulldown ratio $Z_{pu}/Z_{pd} = (L_{pu}/W_{pu})/(L_{pd}/W_{pd})$ of 4:1 yields equal inverter margins and also provides an output sufficiently less than V_{th} for an input equal to VDD. Rather than coupling inverting logic stages directly, we often couple them with pass transistors for the reasons developed in the preceding section, thus affecting the required pullup to pulldown ratio.

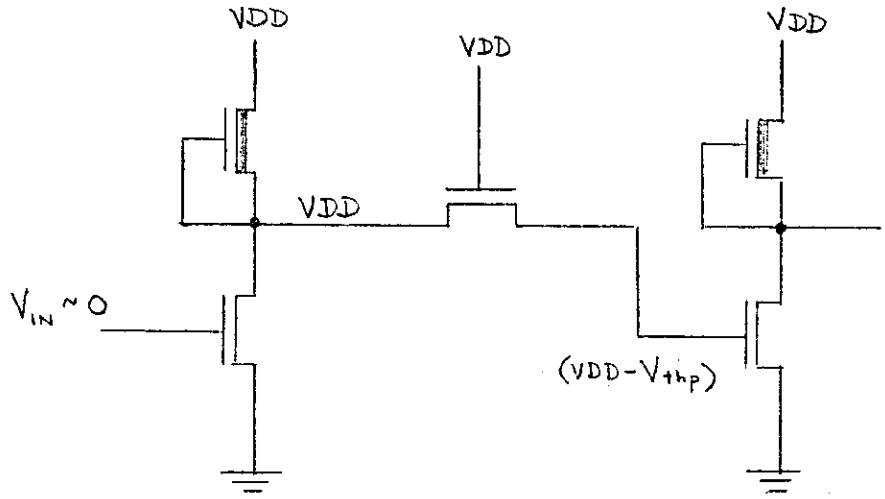


Figure 12a. Inverters Coupled by Pass Transistor

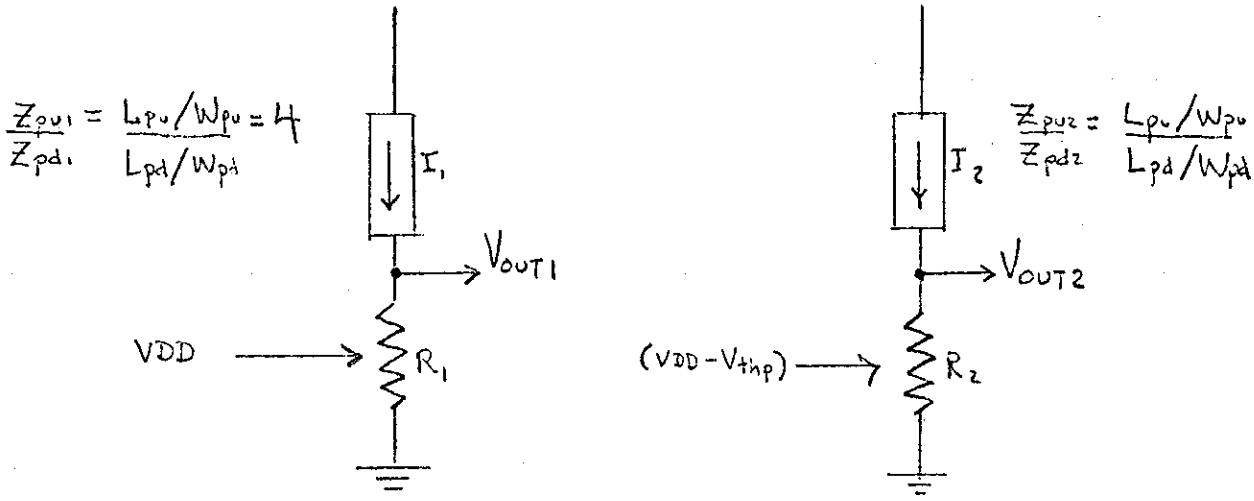


Figure 12b. For $V_{OUT2} = V_{OUT1}$, $Z_{pu2}/Z_{pd2} = 8$.

Figure 12a shows two inverters connected by a pass transistor. If the output of the first inverter nears VDD, the input of the second inverter can at most rise to (VDD - V_{thp}), where V_{thp} is the threshold of the pass transistor. Even with VDD on its gate, the pass transistor opens the connecting path once its input side rises above (VDD - V_{thp}). For the second inverter to have its output driven as low with an input of (VDD - V_{thp}) as would a standard inverter with an input of VDD, it must have a larger pullup to pulldown ratio, calculated as follows.

With inputs near VDD, the pullups of inverters are in saturation, and the pulldowns are in the resistive region. Figure 12b shows equivalent circuits for two inverters, one with VDD as input, the other with (VDD - V_{thp}). For their output voltages to be equal under these conditions, I_1R_1 must equal I_2R_2, and substituting from eq.3 and eq.8 we find:

$$(Z_{pu1}/Z_{pd1})(VDD - V_{th}) = (Z_{pu2}/Z_{pd2})(VDD - V_{th} - V_{thp})$$

Since V_{th} of the pulldowns ~ 0.2VDD, and V_{thp} of the pass transistor ~ 0.3VDD due to the body effect, then $Z_{pu2}/Z_{pd2} \sim 2Z_{pu1}/Z_{pd1}$. Thus a ratio $(L_{pu}/W_{pu})/(L_{pd}/W_{pd}) = 8$ is usually used for inverting logic stages placed as level restorers between sections of pass transistor logic.

Properties of Cross-Coupled Circuits

In many control sequencing and data storage applications, memory cells and registers are built using two inverters driving each other, as shown in figure 13a. This circuit can be set in either the state where V_1 is high and V_2 is low, or in the state where V_2 is low and V_1 is high. In either case, the condition is stable and will not change to the other condition unless it is forced there through some external means. The detailed methods of setting such cross-coupled circuits into one state or another will be discussed in detail later. However, it is important at the present time to understand the time evolution of signals impressed upon cross-coupled circuits, since they exhibit properties different from circuits not having a feedback path from their output to an input.

We have seen that there exists a voltage at which the output of an inverter is approximately equal to its input voltage. If a cross-coupled circuit is inadvertently placed in a situation where its input voltage is equal to this value, then an unstable equilibrium condition is

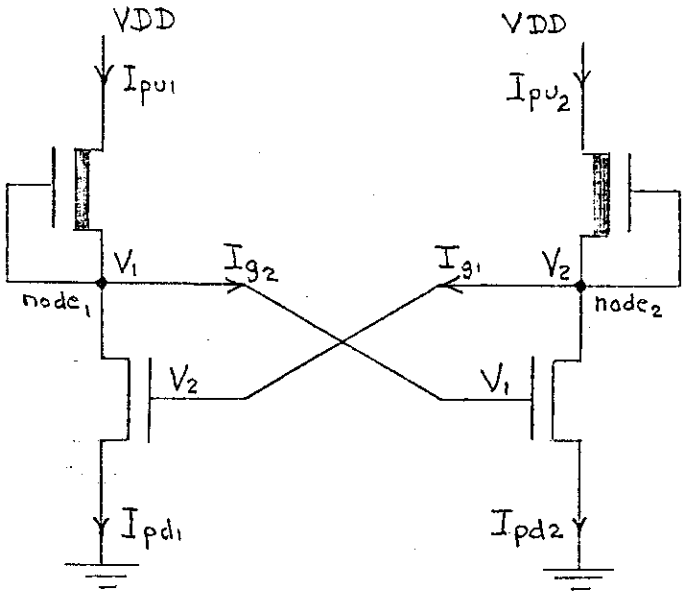


Fig 13a. CROSS COUPLED INVERTERS

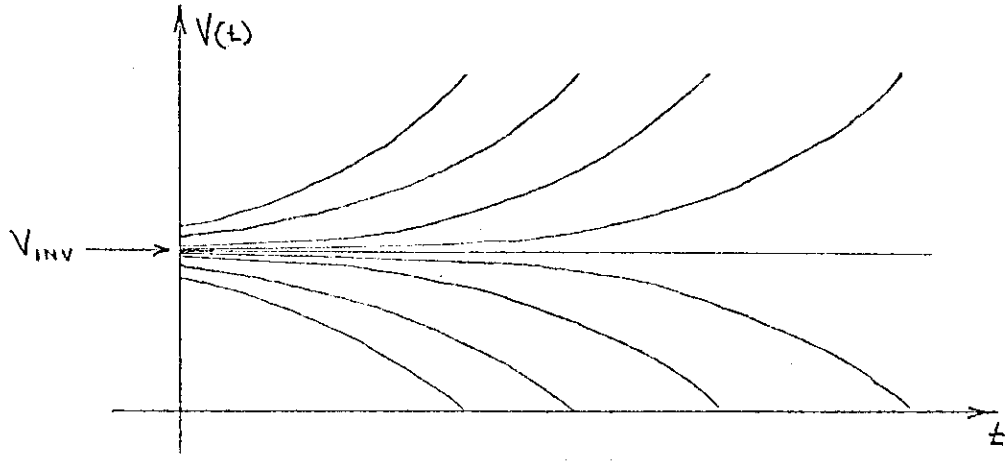


Fig 13b. V(t) FOR CROSS COUPLED INVERTERS

created where voltages V_1 and V_2 are equal. Since the net current flowing onto either gate is now zero, there is no forcing function driving the system to any voltage other than this equilibrium one, and the circuit can stay in this condition for an indefinite period. However, if either voltage changes, even very slightly, the circuit will leave this unstable equilibrium. For example, if the voltage V_1 is increased from its unstable equilibrium value V_{inv} by a slight amount, this will in time cause a lowering of voltage V_2 , as net current flows from gate 1. This lowering of V_2 will at some later time cause V_1 to increase further. As time goes on, the circuit will feedback on itself until it rests in a stable equilibrium state.

The possibility of such unstable equilibria in cross coupled circuits has important system implications[2], as we will later see. For this reason, we will make a fairly detailed analysis of this circuit's behavior near the metastable state. While it is not essential that the reader follow all the details of the analysis, the final result should be carefully studied. The time constant of the final result depends in detail on the regions of operation of the transistors near the metastable state, as given in the following analysis. However, the exponential form of the result follows simply from the fact that the forcing function pushing the voltage away from the metastable point is proportional to the voltage's distance away from that point. This behavior is characteristic of bistable storage elements in any technology.

The time evolution of this process can be traced as follows. At the unstable equilibrium, the current in the pullups equals that in the pulldowns, and is a constant times $(V_{inv}-V_{th})^2$. If V_1 is then changed by some small $\Delta V_1$to V_{init}, I_{pu2} remains constant but I_{pd2} changes immediately, producing a non-zero I_{g1}:

$$I_{g1} = I_{pu2} - I_{pd2} = k[(V_{inv}-V_{th})^2 - (V_{inv}+\Delta V_1-V_{th}^2)]$$

For small ΔV_1, $I_{g1} = -2k(V_{inv}-V_{th})\Delta V_1$. More precisely, since I_{g1} = function(V_1,V_2), then near V_{inv}:

$$\partial I_{g1}/\partial V_1 = -2k(V_{inv}-V_{th})$$

Noting that the pullups are not quite in saturation, but are in the resistive region, and:

$$\partial I_{g1}/\partial V_2 = -1/R_{pu} ,$$

where R_{pu} = effective resistance of the pullup near V_{inv}. Noting that $I_{g1} = C_g dV_{g2}/dt$, we find that:

$$dI_{g1}/dt = -2k(V_{inv}-V_{th})[dV_1/dt] - (1/R_{pu})[dV_2/dt] = C_g[d^2V_2/dt^2]$$

Evaluating the constants in this equation yields $-2k(V_{inv}-V_{th}) = C_g/\tau_o$, where τ_o is the transit time of the pulldowns for t near zero. Evaluating the effective resistance of the pullups in terms of the parameters of the pulldowns yields $1/R_{pu} \sim C_g/\tau_o$. Therefore:

$$-(2/\tau_o)[dV_1/dt] - (1/\tau_o)[dV_2/dt] = d^2V_2/dt^2$$

Similarly:
$$-(2/\tau_o)[dV_2/dt] - (1/\tau_o)[dV_1/dt] = d^2V_1/dt^2$$

Near t = 0, dV_1/dt approximately equals $-dV_2/dt$, and therefore:

$$d^2V_1/dt^2 = -(1/\tau_o)\, dV_2/dt = (1/\tau_o)^2\, V_1 + \text{const.} \qquad \text{[eq.16a]}$$

The solution to eq. 16a is an exponential diverging from the equilibrium voltage V_{inv} with time, with a time constant $\tau_o/2$ equal to one half the pulldown delay time. Note that the solution given in eq.16b satisfies the conditions that $V(0) = V_{init}$, and $V(t)$ = constant if $V_{init} = V_{inv}$:

$$V_1(t) = V_{inv} + (V_{init} - V_{inv})\, e^{t/\tau_o} \qquad \text{[eq.16b]}$$

If a circuit has truly been balanced at the equilibrium voltage, it will only be able to move from this value by virtue of noise or external stimuli. However, in real systems such stimuli are always present and the length of time which it will take for the inverter to settle into one of its stable states is dependent upon how far the initial voltage is from the unstable equilibrium. Various models have been made of this process, making certain assumptions about the statistics of the stimuli tending to unbalance the balanced device. The details of these models are unimportant. What is important for the present purpose is that it is possible to balance any cross-coupled memory device in such a way that the time required for it to recover is arbitrarily long. The time evolution of such a system is shown in Fig. 13b, for several initial voltages near V_{inv}. The time for the cross-coupled system to reach one of its equilibria is thus logarithmic in the displacement from V_{inv}, and is given approximately by eq. 16c:

$$t \sim \tau_o \ln[V_{inv}/(V_{init}-V_{inv})] \qquad \text{[eq.16c]}$$

Effects of Scaling Down the Dimensions of MOS Circuits and Systems

This section examines the effects on important system parameters resulting from scaling down all dimensions of an integrated system, including those vertical to the surface, by dividing them by a constant factor α. The voltage is likewise scaled down by dividing by the same constant factor α. Using this convention, all electric fields in the circuit will remain constant and hence many non-linear factors affecting performance will not change as they would if a more complex scaling were used.

Figure 14a. shows a MOSFET of dimensions L, W, D, with a (Vgs - Vth) = V. Figure 14b. shows a MOSFET similar to that in figure 14a., but of dimensions $L' = L/\alpha$, $W' = W/\alpha$, $D' = D/\alpha$, and $V' = V/\alpha$. Refer to equations 1., 2., and 3. From these equations we will find that as the scale down factor α is increased, the transit time, the gate capacitance, and drain to source current of every individual transistor in the system scale down proportionally, as follows:

$$\tau \propto L^2/V, \quad \tau'/\tau = [(L/\alpha)^2/(V/\alpha)]/[L^2/V], \quad \therefore \quad \tau' = \tau/\alpha$$

$$C \propto LW/D, \quad C'/C = (L/\alpha)(W/\alpha)/(D/\alpha)]/[LW/D], \quad \therefore \quad C' = C/\alpha$$

$$I \propto WV^2/LD, \quad I'/I = [(WV^2/\alpha^3)/(LD/\alpha^2)]/[WV^2/LD], \quad \therefore \quad I' = I/\alpha$$

Switching power, P_{sw}, is the energy stored on the capacitance of a given device divided by the *clock period*, or time between successive charging and discharging of the capacitance. A system's clock period is proportional to the τ of its smallest devices. As devices are made smaller and faster, the clock period is proportionally shortened. Also, the dc power, P_{dc}, dissipated by any static circuit equals I times V. Therefore,

$$P_{sw} \propto CV^2/\tau \propto WV^3/DL, \quad \therefore \quad P_{sw}' = P_{sw}/\alpha^2$$

$$P_{dc} = IV, \quad \therefore \quad P_{dc}' = P_{dc}/\alpha^2$$

Both the switching power and static power per device scale down as $1/\alpha^2$. The average dc power for most systems can be approximated by adding the total P_{sw} to one-half the total dc power which would result if all level restoring logic pulldowns were turned on. The contribution of pass transistor logic to the average dc power drawn by the system is due to

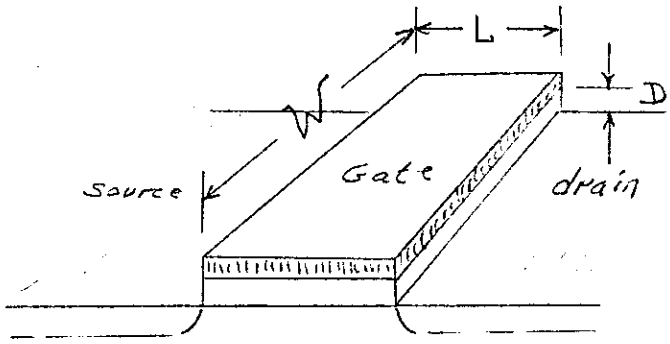


Fig.14a. MOSFET, 1977

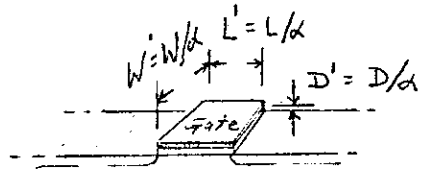


Fig.14b. MOSFET SCALED DOWN by α, 19XX

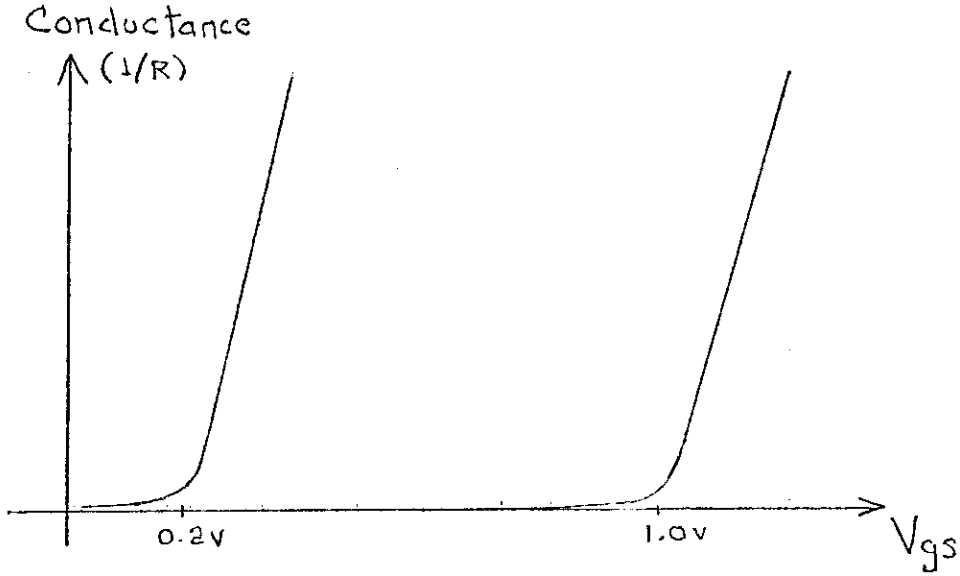


Fig.15 Conductance as a Function of Threshold Voltage

the switching power consumed by the driving circuits which charge and discharge the pass transistor control gates.

The power-delay product at maximum clock frequency, an important metric of device performance, equals the switching energy per device E_{sw} and scales down as follows:

$$E_{sw} \propto CV^2 , \qquad \therefore \quad E_{sw}' = E_{sw}/\alpha^3$$

A more detailed plot of the channel conductance of an MOS transistor near the threshold voltage is shown in figure 15. Below the nominal threshold, the conductance (1/R) is not in reality zero, but depends on gate voltage and temperature as follows:

$$1/R \propto e^{(V_{gs}-V_{th})/(kT/q)},$$

where T is the absolute temperature, q is the charge on the electron, and k is Boltzmann's constant. At room temperature, kT/q ~ 0.025 volts. At present threshold voltages, as in the right curve in figure 15., an off device is below threshold by perhaps 20 kT/q, i.e. by about 0.5 volts, and its conductance is decreased by a factor of the order of ten million. Said another way, if the device is used as a pass transistor, a quantity of charge which takes a time τ to pass through the on device, will take a time on the order of $10^7\tau$ to "leak" through the off device. The use of pass transistors switches to isolate and "dynamically store" charge on circuit nodes is common in many memory applications using 1977 transistor dimensions. However, if the threshold voltage is scaled down by a factor of perhaps 5, as shown in the left plot in figure 15., then an off transistor is only 4kT/q below threshold. Therefore its conductance when "off" is only a factor of 100 or so less than when it is "on". Charge stored dynamicaly on a circuit node by the transistor when "on", will safely remain on that node for only a few typical system clock periods. The charge will not remain on the node for a very large number of periods as in present memory devices using this technique.

Now, suppose we scale down an entire LSI system by a scale down factor of ten. The resulting system will have one hundred times the number of circuits per unit area. The total power per unit area remains constant. All voltages are reduced by the factor of ten. The current per unit area is increased by a factor of ten. The time delay per stage is decreased by a factor of ten. The power-delay product decreases by a factor of one thousand.

This is a rather attractive scaling in all ways except for the current density. The delivery of

the required average dc current presents an important obstacle to scaling. This current must be carried to the various circuits in the system on metal level paths, in order that the voltage drop from the offchip source to the onchip subsystems will not be excessive. Metal paths have an upper current density limit imposed by a phenomenon called metal migration, discussed further in chapter 2. Many metal paths in today's integrated circuits are already operated near their current density limit. As the above type of scaling is applied to a system, the conductors get narrower, but still deliver the same current on the average to the circuits supplied by them.

Therefore, it will be necessary to find ways of decreasing system current requirements to approximately a constant current per unit area relative to the present current densities. In n-channel silicon gate technology, this objective can be partially achieved by using pass transistor logic in as many places as possible and avoiding restoring logic except where it is absolutely necessary. Numerous examples of this sort of design are given in this text. This design approach also has the advantages of tending to minimize delay per unit function and to maximize logic functions per unit area. However, when scaled down to submicron size, the pass transistors will suffer from the subthreshold current problem. It is possible that when the fabrication technologies have been developed to enable scaling down to sub-micron devices, a technology such as complementary MOS, which does not draw any dc current, may be preferable to the nMOS technology used to illustrate this monograph. However, even if this occurs, the methodology developed in the text can still be applied in the design of integrated systems in that technology.

The limit to this kind of scaling occurs when the devices created are no longer able to perform the switching function. To perform the switching function, the ratio of transistor on to off conductance must be $>>1$, and therefore the voltage operating the circuit must be many kT/q. For this reason, even circuits optimized for operation at the lowest possible supply voltages still require a VDD of ~ 0.5 volts. 1977 devices operate with a VDD of approximately five volts and minimum channel lengths of approximately five microns. Therefore, the kind of scaling we have envisioned here will take us to devices with approximately one half micron channel lengths and current densities approximately ten times what they are today. Power per unit area will remain constant over that range. Smaller devices might be built but must be used without lowering the voltage any further. Consequently the power per unit area will increase. Finally, there appears to be a fundamental limit[3] of approximately one quarter micron channel length, where certain physical effects such as the tunneling through the gate oxide, and fluctuations in the

positions of impurities in the depletion layers, begin to make the devices of smaller dimension unworkable.

References

1. W. M. Penney, L. Lau, Eds., "MOS Integrated Circuits", Van Nostrand Reinhold, 1972, pp.60-85.

2. T. J. Chaney, C .E. Molnar, "Anomalous Behavior of Synchronizer and Arbiter Circuits", *IEEE Transactions of Computers*, April 1973, pp. 421-422.

3. B. Hoeneisen, C. A. Mead, "Fundamental Limitations in Micro-electronics--I. MOS Technology", *Solid-State Electronics*, Vol.15, 1972, pp. 819-829.

4. - - - charge control reference - - -

5. *- - - Manchester carry chain ref - - -*

Reading References

R1. J. F. Gibbons, "Semiconductor Electronics", McGraw-Hill, 1966, is a classic text containing a good introduction to basic semiconductor theory.

R2. W.M. Penney, L. Lau, Eds., "MOS Integrated Circuits", Van Nostrand Reinhold,1972, is a good, general text on MOS devices and circuits.

R3. R. W. Keyes, "Physical Limits in Digital Electronics", *Proceedings of the IEEE*, Vol. 63, No. 5, May 1975, pp. 740-767, is an excellent invited survey paper on this topic.

Chapter 2: LSI Fabrication

Sections:

Patterning - - - Scaling of Patterning Technology - - - The Silicon Gate n-Channel Process - - - Yield Statistics - - - Scaling of the Processing Technology - - - Design Rules - - - Formal Description of Design Rules - - - Electrical Parameters - - - Dependence of Capacitance on Voltage - - - Current Limitations in Conductors - - - Choice of Technology

The series of steps by which a geometric pattern or set of geometric patterns is transformed into an operating integrated system is called a *wafer fabrication process*, or simply a *process*. An integrated system in MOS technology consists of a number of superimposed layers of conducting, insulating, and transistor forming materials. By arranging predetermined geometric shapes in each of these layers, a system of the required function may be constructed. The task of the integrated system designer is to devise the geometric shapes and their locations in each of the various layers of the system. The task of the process itself is to create the layers and transfer into each of them the geometric shapes determined by the system design. In this chapter we describe the patterning sequence and how it is applied in a simple, specific LSI process: nMOS. A number of other topics are covered which are related to the processing technology, or are closely tied to the properties of the underlying materials. The final section of the chapter discusses the factors affecting the choice of a particular process technology.

Patterning

The overall fabrication process consists of the successive *patterning* of a particular *sequence of layers*. The sequence of steps, called *patterning*, by which geometrical shapes are transferred into a layer of the final system, is very similar for each of the layers. The overall process is more easily visualized if we first describe the details of the patterning of one layer. We can then describe the particular sequence of layers used in the process to build up an integrated system, without repeating the details of patterning for each of the layers.

A common step in many processes is the creation of a silicon dioxide insulating layer on the surface of a silicon wafer, and the selective removal of sections of the insulating layer

exposing the underlying silicon. We will use this step for our patterning example. The step begins with a bare polished silicon wafer, shown in cross section in figure 1. The wafer is exposed to oxygen in a high temperature furnace to grow a uniform layer of silicon dioxide on its surface, as shown in figure 2. After the wafer is cooled it is coated with a thin film of organic resist material as shown in figure 3. The resist is thoroughly dried and baked to insure its integrity. The wafer is now ready to begin the patterning.

Whatever the origin of the pattern to be transferred to the wafer surface, by the time of actual wafer fabrication it almost universally exists as a *mask*. A mask is merely a transparent support material coated with a thin layer of opaque material. Certain portions of the opaque material are removed, leaving opaque material on the mask in the precise pattern required on the silicon surface. Such a mask with the desired pattern engraved upon it is brought face down into close proximity with the wafer surface as shown in figure 4. The dark areas of opaque material on the surface of the mask are located where it is desired to leave silicon dioxide on the surface of the silicon. Openings in the mask correspond to areas where it is desired to remove silicon dioxide from the silicon surface. When the mask has been brought firmly into proximity with the wafer itself, its back surface is flooded with an intense source of ionizing radiation such as ultraviolet light or low energy x-rays. The radiation is stopped in areas where the mask has opaque material on its surface. Where there is no opaque material on the mask surface, the ionizing radiation radiation passes on through and into the resist, the silicon dioxide, and silicon. While the ionizing radiation has little effect on the silicon dioxide and the silicon, it breaks down the molecular structure of the resist into considerably smaller molecules.

After exposure to the ionizing radiation, the wafer has the characteristics shown in figure 5. In areas exposed to the radiation, the resist molecules have been broken down to much lighter molecular weight than that of unexposed resist molecules. The solubility of organic molecules in various organic solvents is a very steep function of the molecular weight of the molecules. Hence, it is possible to dissolve exposed resist material in solvents which will not dissolve the unexposed resist material. In this way the resist can be "developed" as shown in figure 6 by merely immersing the silicon wafer in a suitable solvent.

Thus far, the pattern originally existing as a set of opaque geometries on the mask surface has been transferred into a corresponding pattern in the organic resist material on the surface of the silicon dioxide. This same pattern can now be transferred to the silicon dioxide itself by exposing the wafer to a material which will etch silicon dioxide but not

Figure 1.

BARE WAFER

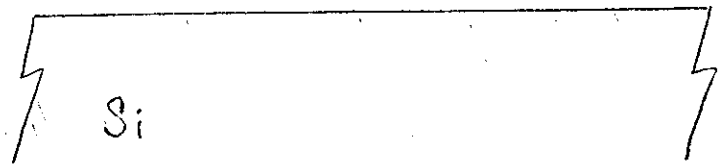


Figure 2.

OXIDATION

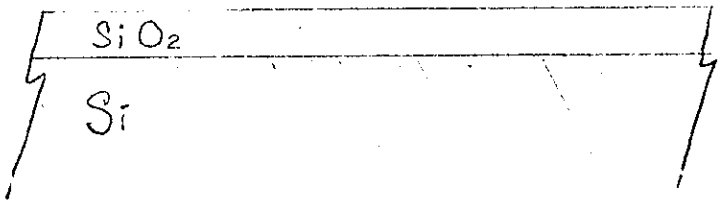


Figure 3.

COAT W/ RESIST

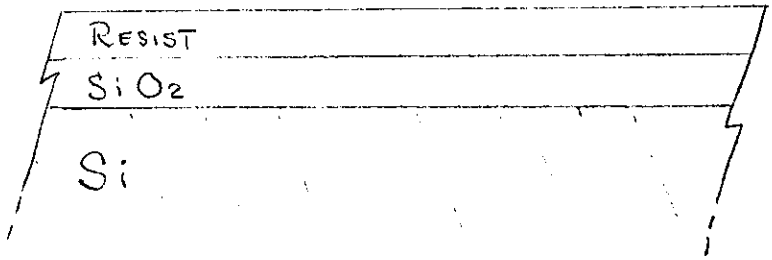


Figure 4.

MASK & EXPOSE

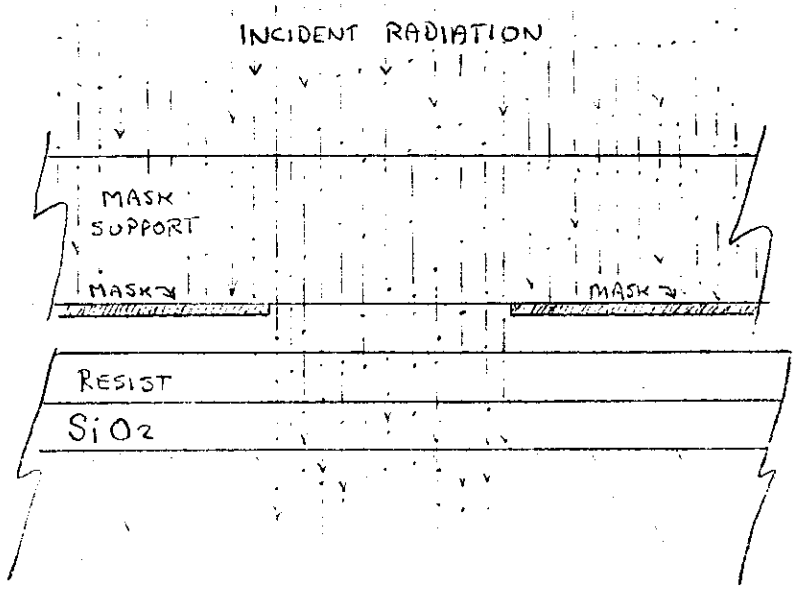


Figure 5.

EXPOSED RESIST

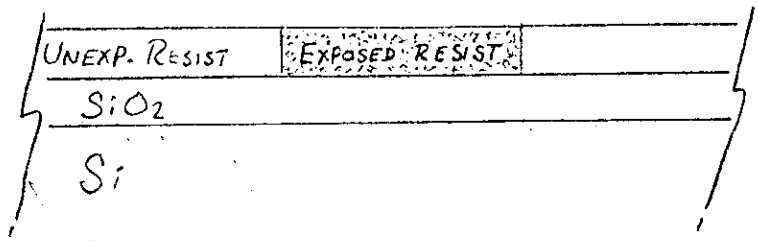


Figure 6.

DEVELOP RESIST

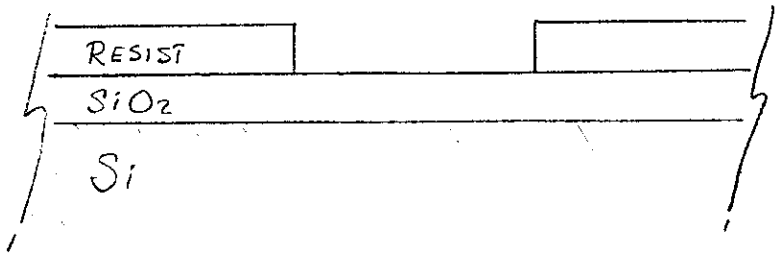


Figure 7.

ETCH

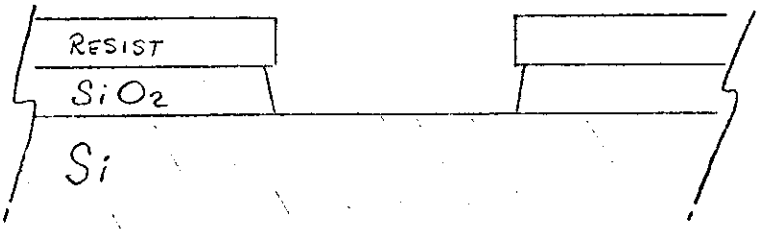
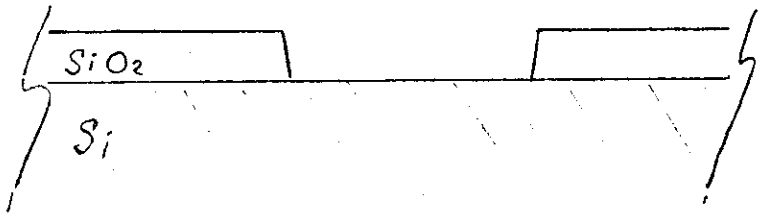


Figure 8.

REMOVE RESIST



attack either the organic resist material or the silicon wafer surface. This etching step is usually done with hydrofluoric acid, which easily dissolves silicon dioxide. However, organic materials are very resistant to hydrofluoric acid, and it is incapable of etching the surface of silicon. The result of this etching step is shown in figure 7.

The final step in the patterning is the removal of the organic resist material. This can be done by using strong organic solvents which will dissolve even unexposed resist material, by using strong acids such as chromic acid which actively attack organics, or by exposing the wafer to atomic oxygen which will oxidize away any organic materials present on its surface. All three techniques have been used to remove the resist materials. Once the resist material is removed, the finished pattern on the wafer surface is as shown in figure 8. Notice that we have transferred the geometric pattern which originally existed on the surface of the mask directly into the silicon dioxide on the wafer surface. While a foreign material was present on the wafer surface during the patterning process it has now disappeared and the only materials present are those which will in fact be part of the finished wafer.

A similar sequence of steps is used to selectively pattern each of the layers of the integrated system. These differ only in the details of the etchants used, etc. Thus as we study the processing of the various layers, the reader need not visualize all the details of the patterning sequence for each layer, but only recognize that a mask pattern for a layer can be transferred into a pattern in the material of that layer.

Scaling of Patterning Technology

As discussed in chapter 1., semiconductor devices could be at least an order of magnitude smaller in linear dimension than those typically manufactured in 1977 and still function correctly. The fundamental dimensional limitation is approximately a one quarter micron channel length, corresponding to a length unit λ (to be discussed under design rules) of approximately 0.1 micron. This limitation appears to apply to both bipolar and MOS technologies. It has been possible for several years to create sub-micron lines using electron beam and x-ray techniques, and there is considerable research and development under way to bring these patterning technologies into general manufacturing use. At present the lithography system appearing most promising for the manufacture of sub-micron devices uses masks made of very thin supporting materials and a heavy metal such as gold as the opaque material. There are resist materials having resolutions of small fractions of a

micron and which are sensitive to electron beams. The opaque material on the thin support may thus be patterned by using direct writing of a scanned electron beam to expose resist on its surface, followed by etching of the resulting selected areas of material. This mask can then be positioned above the silicon wafer and soft x-rays used to expose resist material on the wafer surface. The most desireable source for such soft x-rays appears to be synchrotron radiation from an electron storage ring, operating at approximately 500MEV. Electron storage rings are exceedingly efficient sources of soft x-rays in the one to two hundred volt quantum energy range. This range is optimum for exposure of resists of the sort which have inherent resolutions in the sub-micron region. In addition, sychrotron radiation can be extracted from a storage ring in such a way that it is exceedingly highly collimated. Exposure times for such soft x-rays used with the kind of resists useful at sub-micron resolution are not appreciably longer than those required today with ultra-violet light and standard photo resists. X-rays have sufficiently short wavelength that they exhibit the very desireable property of creating essentially straight sides on the exposed resist material, making them ideal in a patterning process.

The Silicon Gate n-Channel MOS Process

We now describe the particular sequence of patterned layers used to build up nMOS integrated circuits and systems. Figures 9 through 14 illustrate a simple but complete sequence of patterning and processing steps which are sufficient to fabricate a complete, integrated system. The example follows the fabrication of one simple circuit within a system, but all other circuits are simultaneously implemented by the same process. The example used is the basic inverter circuit. The top illustration in figures 9 through 14 shows the top view of the layers of the circuit layout. The lower illustration in each of those figures shows the cross section through the cut indicated by the downward arrows. The vertical scale in these cross sections has been greatly exaggerated for illustrative purposes.

The opening in the opaque material of the *first* mask is shown by the green outline in the top portion of figure 9. This opening exposes all areas that will be eventually be the diffusion level. It includes the sources and drains of all transistors in the circuit, together with the transistor gate areas, and any diffusion level circuit interconnection paths. This mask is used for the first step in the process, the patterning of silicon dioxide on silicon as described in the previous section. The resulting cross section is shown in the lower portion of figure 9.

The second step in the process is to differentiate transistors which are normally "on" (depletion mode) from those which are normally "off" (enhancement mode). This is done by overcoating the wafer with resist material, exposing the resist material through openings in a *second* mask, and developing it in the manner shown in figure 10. This patterning step leaves an opening in the resist material over the area to be selectively turned into normally on transistors. The actual conversion of the underlying silicon is then done by implanting ions of arsenic or antimony into the silicon surface. The resist material, where present, acts to prevent the ions from reaching the silicon surface. Therefore, ions are only implanted in the silicon area free of resist. The implanted layer, which causes a slight n-type conductivity in the underlying silicon, is shown by the yellow box in figure 10. Once the depletion areas are defined, the resist material is removed from the surface of the wafer.

The wafer is then heated while exposed to oxygen, to grow a very thin layer of silicon dioxide over its entire surface. It is then entirely coated with a thin layer of polycrystalline silicon, usually called *polysilicon* or *poly* for short. Note that this polysilicon layer is everywhere insulated from the underlying materials by the layer of thin oxide, and additionally by thicker oxide in some areas. The polysilicon will form the gates of all the transistors in the circuit and will also serve as a second layer for circuit interconnections. A *third* mask is used to pattern the polysilicon by steps similar to those previously described, with the result shown in red in figure 11. The left-most polysilicon area will function as the gate of the pull down transistor of the inverter we are constructing, while the square to the right will function as the gate of the depletion mode pull up transistor.

Once the polysilicon areas have been defined, n-type regions can be diffused into the p-type silicon substrate, forming the sources and drains of the transistors and the first level of interconnections. This step is done by first removing the thin gate oxide in all areas not covered by the (red) polysilicon. The wafer is then exposed to n-type impurities such as arsenic, antimony or phosphorus at high temperature for a sufficient period of time to allow these impurities to convert the exposed underlying silicon to n-type material. The areas of resulting n-type material are shown in green. Notice, in the *cross section* of figure 12., that the red polysilicon area and the thin oxide under it act to prevent impurities from diffusing into the underlying silicon. Therefore the impurities only reach the silicon substrate in areas not covered by the polysilicon and not overlain by the thick original oxide. In this way the active transistor area is formed in all places where the patterned polysilicon overlies the thin oxide area defined in the previous step. The diffusion level

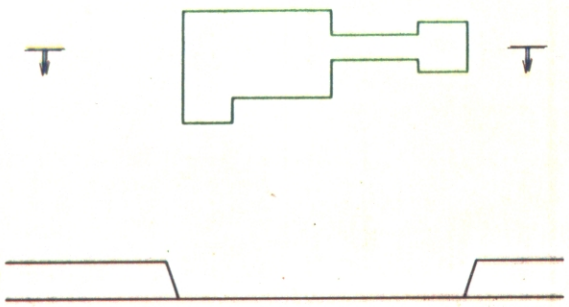


Fig.9. Patterning SiO_2

Fig.10. Ion Implantation

Fig.11. Patterning Polysilicon

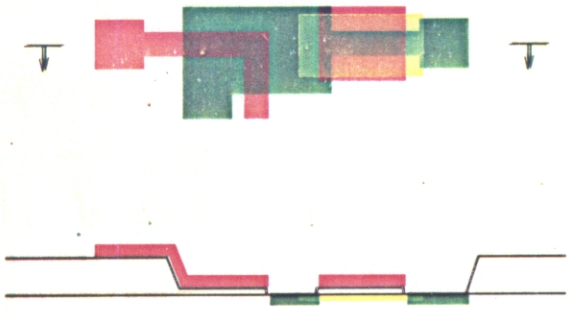


Fig.12. Patterning Diffused Region

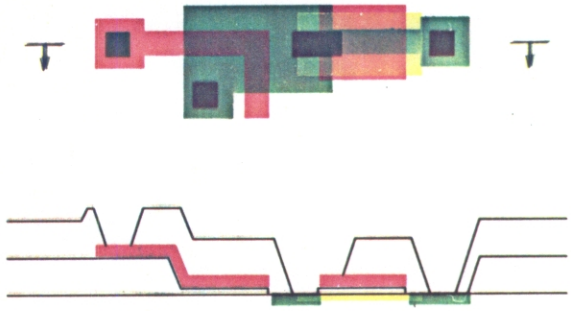


Fig.13. Placing Contact Cuts

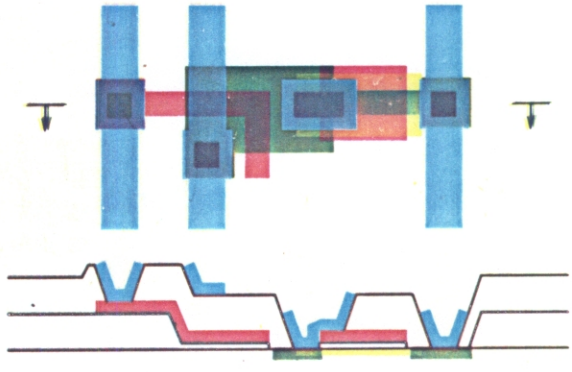


Fig.14. Patterning the Metal Layer

sources and drains of the transistors are automatically filled in between the polysilicon areas and extend up to the edges of the thick oxide areas. The major advantage of the silicon gate process is that it does not require a critical alignment between a mask which defines the green source and drain areas and a separate mask which defines the gate areas. Rather, the transistors are formed by the intersection of the two masks, and the conducting n-type diffused regions are formed in all areas where the green mask is not covered by the red mask.

We now have all the transistors of the basic inverter circuit defined. There remains only the task of electrically connecting them. Required are a connection from some other circuit in to the input gate, a connection between the gate and source of the pullup, and appropriate connections to VDD and GND. These interconnections will be made with a metal layer that can make contact to both the diffused areas and the polycrystalline areas. However, in order to ensure that the metal does not make contact to underlying areas except where intended, another layer of insulating oxide is coated over the entire circuit. At the places where the overlying metal is to make contact to either the polysilicon or the diffused areas, the overlying oxide is selectively removed by the patterning process as previously described. The result of coating the wafer with the overlying oxide and removing this oxide in places where it is desired to make contacts, is shown in figure 13. In the top view, the black areas are those defined by openings in the contact mask, the *fourth* in the process's sequence of mask patterns. In cross section notice that in the contact areas all oxide has been removed down to either the polycrystalline silicon or the diffused area.

Once the overlying oxide has been patterned in this way, the entire wafer is coated with metal, usually aluminum, and the metal is patterned with a *fifth* mask to form the conducting areas required by the circuit. The top view in figure 14 shows three metal lines running vertically, the left most connecting to the input gate of the inverter, the center one being ground, and the right-most one forming the VDD connection to the inverter. The peculiar structure* formed by the metal square slightly to the right of center connects the polysilicon gate of the depletion mode pull up transistor to its source and to the drain of the pull down transistor. Rather than making two separate contacts from the metal line to the pullup's polysilicon gate region and to the adjacent diffusion region, area can be conserved by coalescing the contacts into the compact arrangement shown. This geometrical arrangement is known as a *butting contact* and will be used extensively throughout the text.

*In general, it is good practice to avoid placing contacts over active transistor area whenever possible. However, butting contacts in the location shown here reduce the area and simplify the geometry of the basic inverter and

many other circuits, and have been so placed by the author's in many systems run on a number of commercial fabrication processes without ill effects. A more conservative approach would be to place the butting contact adjacent to, rather than over, the active pullup area.

The inherent properties of the silicon gate process allow the blue metal layer to cross over either the red polysilicon layer or the green diffused areas, without making contact unless one is specifically provided. The red polysilicon areas, however, cannot cross the green diffused areas without forming a transistor. The transistors formed by the intersection of these two masks can be either enhancement mode if no yellow implantation is provided, or depletion mode if such an implantation is provided. Hence, the enhancement mode transistors are defined by the intersection of the green and red masks while the depletion mode transistors are defined by the intersection of the green, red and yellow masks.

If we wish to fabricate only a small number of prototype system chips and to have access to the metal level for the probing of test points, this completes the wafer fabrication sequence. However, when fabricating large numbers of chips of a debugged design, the wafer surface is usually coated with another layer of oxide. This step, called overglassing, provides physical protection for the devices in the system. A *sixth* mask is then used to pattern contact cuts in the overglassing at the locations of relatively large metal contact pads.

Each wafer contains many individual chips. The chips are separated by scribing the wafer surface with a diamond scribe, and then fracturing the wafer along the scribe lines. Each individual chip is then cemented in place in a package, and fine metal wire leads are bonded to the metal contact pads on the chip and to pads in the package which connect with its external pins. A cover is then cemented over the recess in the package which contains the silicon chip, and the completed system is ready for testing and use.

Yield Statistics

Of the large number of individual integrated system chips fabricated on a single silicon wafer, only a fraction will be completely functional. Flaws in the masks, random particles on the wafer surface, defects in the underlying silicon, etc., all cause certain devices to be less than perfect. With present design techniques, any single flaw of sufficient size will kill an entire chip.

The simplest model for the *yield*, or the fraction of the chips fabricated which do not contain fatal flaws, assumes (naively) that the flaws are randomly distributed over the wafer, and that one or more flaws anywhere on a chip will cause it to be non-operative. If there are N fatal flaws per unit area, and the area of an individual chip is A, the probability that a chip has n flaws is in the simplest case just given by the Poisson distribution, $P_n(NA)$. The probability of a good chip is:

$$P_0(NA) = e^{-NA} \qquad \text{[eq.1]}$$

While this equation does not accurately represent the detailed behavior of real fabrication processes, it is a good approximate model for estimating the yield of alternative designs at any given point in time. The exponential is such a steep function that a very simple rule is possible: chips with areas many times 1/N will simply never be found without flaws. Areas must be kept less than a few times 1/N if one flaw will kill a system. There may be design forms which will permit systems to work even in the presence of flaws. If such forms can be developed, the entire notion of yield will be completely changed and much larger chips will be possible.

Once a wafer has been fabricated, each chip must be tested to determine if it is functional. Testing of simple combinatorial logic networks is straightforward and may be done completely, even for very complex networks. - - -*reference?* - - - Complete testing of complex systems with internal sequencing is not in general possible, and most LSI chips manufactured, even at 1977 levels of complexity, are not economically testable even for a small fraction of their possible internal states.

As time passes and the number of devices per chip continues to increase, it becomes important to consider including special functions in the design of LSI systems to improve their testability. The basic problem is to linearize an otherwise combinatorial problem. One approach to this is:

(i) Define the entire system as a set of register to register transfer blocks, i.e. successive stages of storage registers with combinational logic between them.
(ii) Provide for reading and writing from the external world to/from each of the storage registers.

The storage locations are first tested independantly for their ability to reproducibly store

data or control information. If all storage locations pass this test, each combinatorial logic block can be tested separately, by use of its input and output storage locations. Such a test becomes essentially linear in the number of components, and may be accomplished in an acceptable time period, even for extremely complex systems. However, without access to the individual storage locations, testing rapidly becomes hopeless. For this reason even present day microprocessors are very incompletely tested. When one is used for a while, an apparantly new and sudden malfunction may simply be the first occurrance of a particular state of control and data in the system, and thus may represent the first time the device had been "tested "under those conditions.

From experience gained in testing memory parts, it is known that the behavior of one circuit can be influenced by the state of a nearby circuit. For example, a memory cell may be able to remember both a logic-1 and a logic-0 if its neighbor is at a logic-0, but may be able to retain only a logic-0 if its neighbor is at a logic-1. Failures of this type are dependent upon the data patterns present in the system, and are known as *pattern sensitive* failures. In a reasonable (or even an unreasonable) time, it is not possible to exercise even a minute fraction of all the combinations of bit patterns of many integrated systems. What is done instead is to apply our knowledge of the physics of such failures, and construct a *model* for possible failure modes. In the memory example, we may conclude that any flaw not visible optically will be unable to reach beyond the immediate locality of the cell involved. Hence, pattern sensitivity in the behavior of a particular cell may be introduced by other cells in the same row or column of an array of memory cells, or by nearest diagonal neighbors. A test for pattern sensitivity under this model is quite fast, being only slightly worse than a linear function of the number of devices on the chip.

In order to test for pattern sensitive failures, we must construct a physical model for the possible failure mechanisims. This model will inevitably include the physical proximity of other signals. For this reason, any practical test for pattern sensitive failures must be based on a knowledge of the physical location of the various elements of the subsystem being tested. The task of preparing such tests is thus greatly eased by regularity in the design and physical layout of a system.

Scaling of the Processing Technology

In order to have a complete process for sub-micron transistors, it is necessary not only to

make patterns in the resist material but to transfer these patterns to the underlying layers in the silicon and silicon dioxide. This has traditionally been done by wet etching processes. However, wet etching processes do not scale well into the sub-micron range. Alternatives are currently being developed which appear workable. Etching with plasmas (i.e. glow discharges of gaseous materials resulting in free ions of great chemical activity) is already used in a number of advanced processing laboratories. It is known that very controllable etching can be achieved in this way and it seems likely that essentially no wet processing will be used in the construction of sub-micron devices. Ion implantation, an ideal method for achieving controled doses of impurity ions in the silicon surface, is already a common production technique in essentially all MOS processing facilities. Metal layers for sub-micron lithographic processes must be thicker in relationship to their width than today's commercial processing technology allows. A possible solution to this problem may be the use in metal patterning of a process known as ion milling. In this process ions of modest energy sputter away any metal not covered with resist material, yielding much steeper sides on the metal thus patterned than do current wet etching processes.

It thus appears that the basic technological pieces exist to enable development of a complete patterning and wafer fabrication process at sub-micron dimensions. In reality, the ultimate submicron process will not emerge full-blown, but dimensions will gradually be reduced. As one after another of the myriad of technological difficulties are surmounted. The sketch we have given is rather an artists conception of the possibility of such an ultimate process. We do believe, however, that the evolution of this process is of fundamental importance to the entire electronics industry.

Design Rules

Perhaps the most powerful attribute of modern wafer fabrication processes is that they are *pattern independent.* That is, there is a clean separation between the processing done during wafer fabrication and the design effort which creates the patterns to be implemented. This clean separation requires a clean definition to the designer of the capabilities of the processing line. This specification usually takes the form of a set of permissible geometries which may be used by the designer with the knowledge that they are within the resolution of the process itself and that they do not violate the device physics required for proper operation of transistors and interconnections formed by the process. When reduced to their simplest form, such geometrical constraints are called *design rules.* The constraints are of the form of minimum allowable values for certain *widths, separations, extensions,* and *overlaps* of geometrical objects patterned in the various levels of a system.

As processes have improved over the years, the absolute values of the permissible sizes and spacings of various layers have become progressively smaller. There is no evidence that this trend is abating, in fact there is every reason to believe that at least another order of magnitude of linear dimensional shrinkage is possible. For this reason we present a set of design rules in dimensionless form, as constraints on the allowable ratios of certain distances to a basic length unit. The basic unit of length measurement used is equal to the fundamental resolution of the process itself. This is the distance by which a geometrical feature on any one layer may stray from another geometrical feature on the same or on another layer, all processing factors considered and an appropriate safety factor added. It is set by phenoma such as overetching, misalignment between mask levels, distortion of the silicon wafer ("runout") due to high temperature processing, over or underexposure of resist, etc. All dimensions are given in terms of this elementary distance unit. We refer to this unit as the *length-unit* λ. In 1977 the length-unit $\lambda \sim 3$ microns.

The rules given below have been abstracted from a number of processes over a range of values of λ, corresponding to different points in time at different fabrication areas. They represent somewhat of a "least common denominator " likely to be representative of nMOS design rules for a reasonable period of time, as the value of λ decreases in the future.

A typical minimum for the line width W_d of the diffused regions is 2λ, as shown in figure

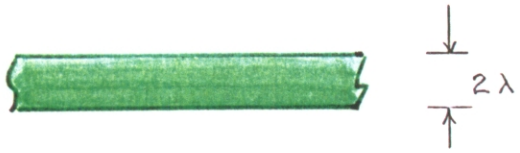


Fig.15. $W_d/\lambda \geq 2$

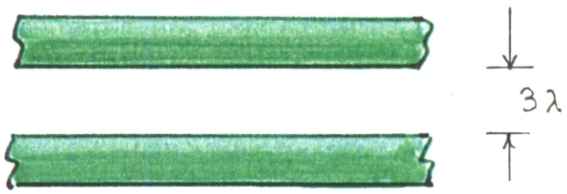


Fig.16. $S_{dd}/\lambda \geq 3$

Fig.17. $W_p/\lambda \geq 2$

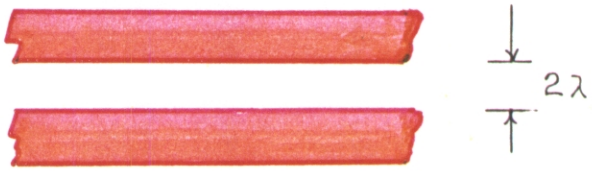


Fig.18. $S_{pp}/\lambda \geq 2$

Fig.19. $S_{pd}/\lambda \geq 1$

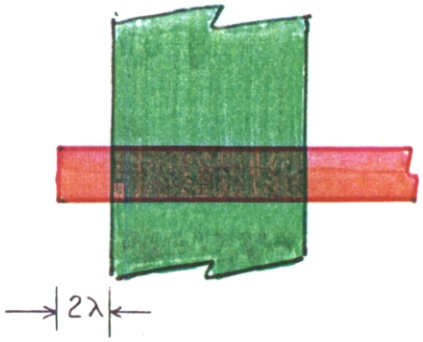


Fig.20. $E_{pd}/\lambda \geq 2$

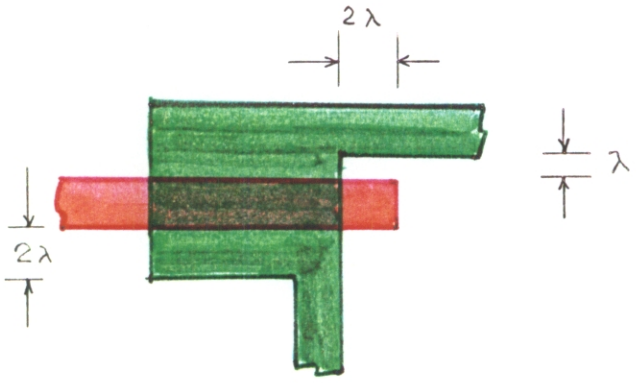


Fig.21. Example of Several Rules

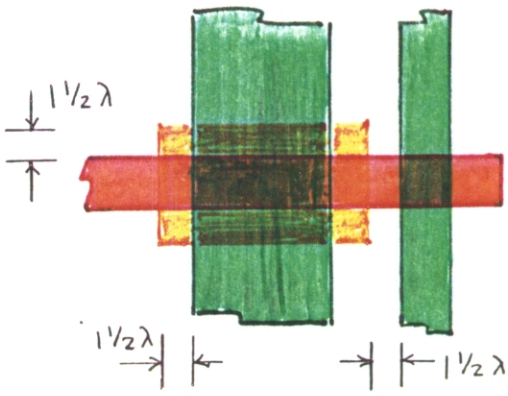


Fig.22. $S_{ig}/\lambda \geq 1\frac{1}{2}$ $E_{ig}/\lambda \geq 1\frac{1}{2}$

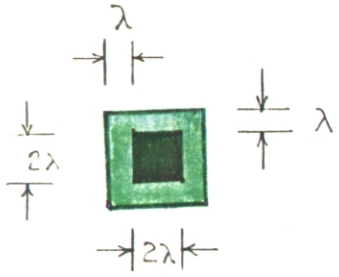


Fig.23. $W_c/\lambda \geq 2$, $E_{dc}/\lambda \geq 1$

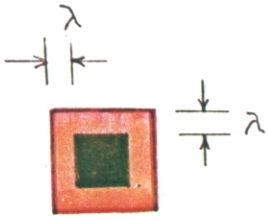


Fig.24. $E_{pc}/\lambda \geq 1$

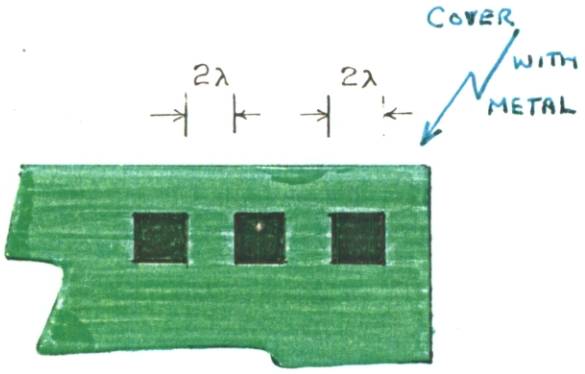


Fig.25. $S_{cc}/\lambda \geq 2$

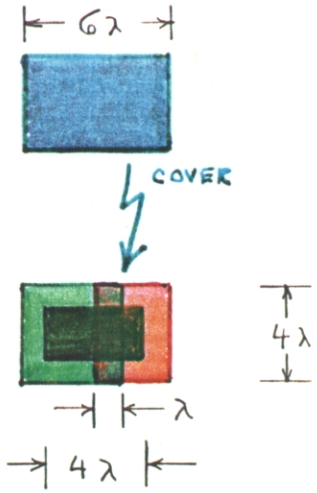


Fig.26. $O_{pd}/\lambda = 1$, & DETAILS OF BUTTING CONTACT

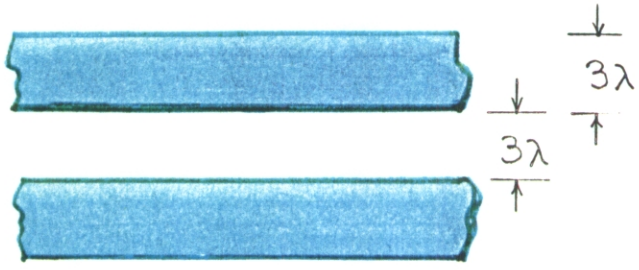


Fig.27. $W_m/\lambda \geq 3$, $S_{mm}/\lambda \geq 3$

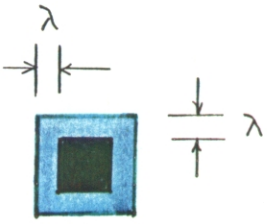


Fig.28. $E_{mc}/\lambda \geq 1$

15. The spacing required between two electrically separate diffused regions is a parameter which depends not merely upon the geometric resolution of the process, but also upon the physics of the devices formed. If two diffused regions pass too close together, the depletion layers associated with the junctions formed by these regions may overlap and result in a current flowing between the two regions when none was intended. Also, the actual geometrical boundary of the diffused region is in fact somewhat larger than the geometrical boundary of the mask due to the fact that diffusion occurs sideways as well as downwards into the silicon. To this dimension must be added the width of the depletion* layers associated with two adjacent diffused areas. * See later section on voltage variation with capacitance. In typical processes a safe rule of thumb is to allow 3λ of separation, S_{dd}, between any two diffused regions which are unconnected, as shown in figure 16. The width of a depletion layer associated with any diffused region depends upon the voltage on the region. If one of the regions is at ground potential, its depletion layer will of necessity be quite thin. In addition some processes provide a heavier doping level at the surface of the wafer between the diffused areas in order to alleviate the problem of overlap of depletion layers. In cases where either very low voltage exists on both diffused regions or where a heavily doped region has been implanted in the surface between the diffused areas, it is often possible to space diffused areas 2λ apart. However, this should not be done without carefully checking the actual process by which the design is to be fabricated.

The minimum for the width W_p of polysilicon lines is similarly 2λ. No depletion layers are associated with polysilicon lines, and therefore the separation, S_{pp}, of two such lines may be as litle as 2λ. These are illustrated in figures 17 and 18.

We have so far considered the diffused and polysilicon layers separately. Another type of design rule concerns how the two layers interact with each other. Figure 19 shows a situation where a diffused line is running parallel to an independent polysilicon line, to which it is not anywhere connected. The only consideration here is that the two unconnected lines not overlap. If they did they would form an unwanted capacitor. Avoidance of this overlap requires a separation S_{pd} of only λ between the two regions as shown in figure 19. A slightly more complex situation is shown in figure 20, where a polysilicon gate area intentionally crosses a diffused area, thereby forming a transistor. In order to make absolutely sure that the diffused region does not reach around the end of the gate and short out the drain to source path of the transistor with a thin diffused area, it is necessary for the polysilicon gate to extend a distance E_{pd} of at least 2λ beyond the nominal boundary of the diffused area, as shown in figure 20.

A composite of several of these design rules is shown in figure 21. Note that the minimum width for a diffused region applies to diffused regions formed between a normal boundary of the diffused region and an edge of a transistor as well as to a diffused line formed by two normal boundaries. This situation is illustrated in the lower left corner of the figure.

As we have seen in figure 10, ion implantation in the region which becomes the gate of a transistor will convert the resulting transistor into the depletion mode type. It is important that the implanted region extend outward beyond all four boundaries of the gate region, as shown in figure 22. To avoid any possibility that some small fraction of the transistor might remain enhancement mode, the yellow ion implantation region should extend a distance E_{ig} of at least 1½ λ beyond each edge of the gate region. The separation S_{ig} between an ion implantation region and an adjacent enhancement mode transistor gate region should also be at least 1½ λ. Both situations and their design rules are illustrated in the figure.

A contact may be formed between the metal layer and either the diffused level or the polysilicon level by means of the contact mask. A set of rules apply to the amount by which each layer must provide an area surrounding any contact to it, so that the contact opening not find its way around the layer to something unintended below it. Since no physical factors apply here other than the relative registration of two levels, a very simple set of design rules results. Each level involved in a given contact must extend beyond the outer boundary of the contact cut by λ at all points, as illustrated by extension distances E_{dc}, E_{pc}, and E_{mc}, in figures 23, 24, and 26. The contacts themselves, like the minimum width lines in the other levels, must be at least 2λ long and 2λ wide (W_c). This situation is illustrated for the diffusion and polysilicon levels in figures 23 and 24. When making contact between a large metal region and a large diffused region, many small contacts spaced 2 λ apart should be used, as shown in figure 25.

A special case arises when it is desired to connect a polysilicon layer directly to a diffused layer, using the butting contact. The detailed geometric layout of the butting contact is shown in figure 26. In its minimum sized configuration, it is composed of a square region of diffision 4 λ on a side, overlapped by a 3 λ by 4 λ rectangle of polysilicon. A rectangular contact cut, 2 λ by 4 λ in size, is made in the center of this structure. The structure is then overlayed with metal, thus connecting the polysilicon to the diffusion. The rules involved in figure 26 are identical to those given so far, with the addition of a

minimum of one λ overlap, O_{pd}, of the diffused and polysilicon layers in the center area of the contact.

In considering the design rules for the metal layer, notice that this layer in general runs over much more rugged terrain than any other level, as can be seen by referring to the cross section of figure 14. For this reason it is generally accepted practice to allow somewhat wider minimum lines and spaces for the metal layer than for the other layers. As a good working rule 3λ widths (W_m), and 3λ separations (S_{mm}) between independent metal lines should be provided, as shown in figure 27. The metal layer must surround the contact layer in much the same way that the diffused and polysilicon layers did. Since the resist material used for patterning the metal generally accumulates in the low areas of the wafer, it tends to be thicker in the neighborhood of contact than elsewhere. For this reason metal tends to be slightly larger after patterning in the vicinity of a contact than elsewhere. Thus it is generally sufficient to allow only one λ of space around the contact region for the metal, as for the other two layers. The rule for metal surrounding contacts is shown in figure 28.

The above design rules are likely to remain valid as the length-unit λ scales down in size with the passage of time. Occasionally, for specific commercial fabrication processes, some one or more of these rules may be relaxed or replaced by more complex rules, enabling slight reductions in the area of a system. While this may be important for certain competitive products such as memory systems, it has the disadvantgage of making the system design a captive of the process specific design rules. Extensive redesign and checking is required to scale down such a design as the length-unit scales down. For this reason, we recommend use of the dimensionless rules given, especially for prototype systems. Designs implemented according to these rules are easily scaled, and may have reasonable longevity.

Formal Description of Design Rules

- - - someone needs to figure this out, so we can put something in here - - -

Electrical Parameters

The design rules permit the designer to create any pattern, with the knowledge that the appropriate transistors, lines, etc., will be created by the process from his original patterns.

To complete a design it is necessary to also know the electrical parameters of the transistors, diffused layers, polysilicon layers, etc., so that the performance of the circuit can be evaluated. The resistances per square of the various layers and the capacitance per square micron with respect to underlying substrate are shown in Table 1. Note that the resistance of a square of material contacted along two opposite sides is independant of the size of the square, and equals the resistivity of the material divided by its thickness. The tabulated values are typical of processes running in 1977. As the circuit dimensions are scaled *down* by dividing by a factor α, the parameters scale approximately as shown in the table.

Resistances:	Metal	~ 0.1 ohms/□	Resistances/square scale
	Diffusion	~ 10 ohms/□	*up* by α., as dimensions
	Poly	~ 15-100 ohms/□	scale *down* by α
	Transistor	~ 10^4 ohms/□	
Capacitances:	Gate-channel	~ 4×10^{-4} pf/μ^2	Capacitances/micron^2 scale
	Diffusion	~ 0.8×10^{-4} pf/μ^2	*up* by α., as dimensions
	Poly	~ 0.4×10^{-4} pf/μ^2	scale *down* by α
	Metal	~ 0.3×10^{-4} pf/μ^2	

Table 1. Typical MOS Electrical Parameters (1977).

The relative values of resistance of metal, diffusion, poly, and drain to source paths of transistors are quite different. Diffusion and good polysilicon layers have approximately one hundred times the resistance per square area of the metal layer. A fully turned on transistor has approximately one thousand times the resistance of the diffused and polysilicon layers. The capacitances are not as wildly different as the resistances of the various layers. Compare the capacitances in Table 1 to the gate to channel capacitance, as a reference. The diffused areas typically have one fifth the capacitance per square micron. Polysilicon on thick oxide has approximately one tenth, and the metal layer slightly less than one tenth, of the gate-channel capacitance per square micron.

The relative values of the resistances and capacitances are expected to not vary dramatically as the processes evolve towards smaller dimensions. One note of warning: There is a wide range of possible values of polysilicon resistance for different commercial processes. Polycrystalline silicon suffers from inordinately high resistances at the crystal grain boundaries if the doping level in the polysilicon itself is not held quite high. This disease

does not affect the diffused layers themselves. For this reason, any processing which tends to degrade the doping levels in the diffused and polysilicon layers, affects the polysilicon resistance much more dramatically than the resistance of the diffused area. It is in general difficult to design circuits which are optimum over the entire range of polysilicon resistivity. If the designer's circuit is to be run on a variety of fabrication lines, it is desireable for the circuit to be designed in such a way that no appreciable current is drawn through a long thin line of polysilicon. In an important example in Chapter 5., polysilicon lines are used as busses along which information flows. The timing of these busses can be dramatically affected by the resistance of the polysilicon. However, the protocol used on these busses has the polysilicon lines precharged during one period of a clock and then pulled low by the appropriate bus source during a following clock period. In this way the circuit is guaranteed to work independent of the resistance of the poly. However, it may be considerable slower in processes of high poly resistivity.

Dependence of Capacitance on Voltage

In Table 1 we gave typical capacitance for the various layers to the substrate. These capacitances are those which would be measured if the voltage on the particular layer were zero (relative to the substrate). The dependence upon voltage of the capacitance of the different layers may sometimes be important and we will now discuss how this dependence arises. References R1 and R2 are good sources for those wishing more background information on the concepts of device physics used in this section and the later sections of this chapter.

When a negative voltage is applied to an n-type diffused region relative to the p-type bulk silicon, the negative electrons are pushed out of the n-type layer into the bulk and a current flows. In integrated systems we are careful to never allow the voltage on the n-type diffused regions to be more negative than the p-type bulk. Diffused regions are biased positively with respect to the p-type bulk, resulting in a reversed biased p/n junction. With the exception of a small leakage current, the reverse biased p/n junction acts merely to isolate one diffused region from another. The p-type bulk of our integrated system has a small number (typically 10^{15}-10^{16} per cubic centimeter) of impurity atoms. When a voltage is applied to an n-type diffused region, it's influence is felt well out into the p-type bulk. Positive charge carriers in the p-type bulk and repelled from the positively charged n-type layer, thereby exposing negatively charged impurity ions. The region surrounding

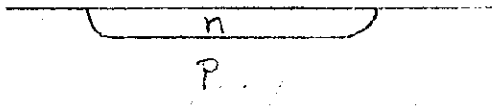


Fig. 29a. n-Type Diffusion in p-Type Bulk Silicon

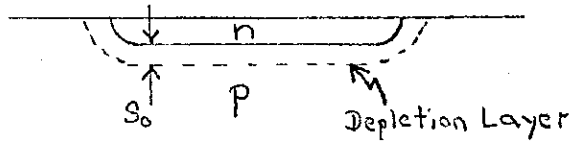


Fig. 29b. Depletion Layer

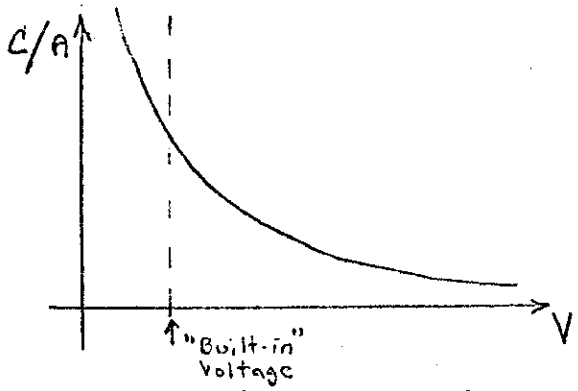


Fig. 30. C/A as fcn(V)

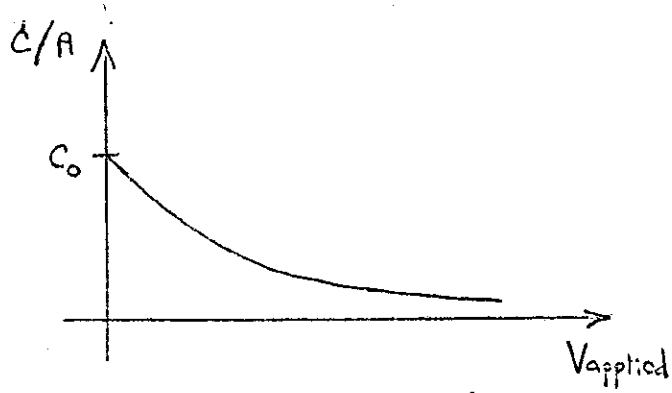


Fig. 31. C/A as fcn($V_{applied}$)

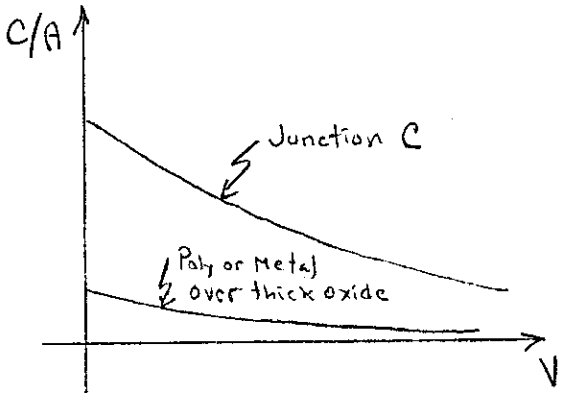


Fig. 32. Capacitance of Poly or Metal over Thick Oxide

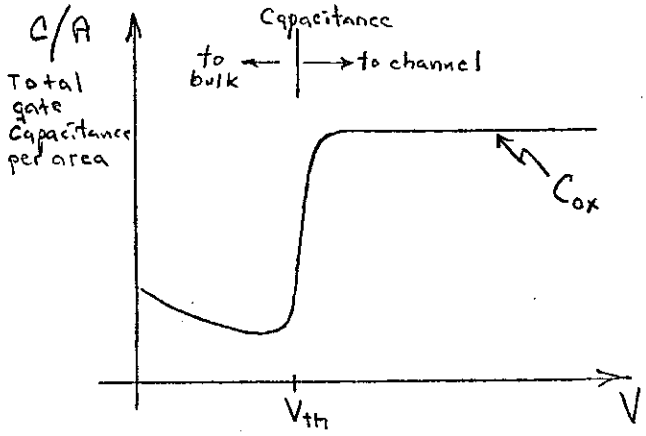


Fig. 33. MOS Gate Capacitance as fcn V

the n-type diffused layer which has been depleted of positive charge carriers is referred to as a depletion layer and is shown schematically in Figure 29b. As the voltage on the n-type layer is increased, charge carriers are pushed further back from the junction between the n-type layer into the p-type bulk, widening the depletion layer and exposing more charged impurity ions. The charge thus induced in the depletion layer as the voltage on the n-type diffused region is increased is responsible for the capacitance of the n-type diffused region relative to the substrate.

We will now consider a unit area of the junction. The total charge in the depletion layer per unit area is proportional to the number per unit volume of impurity ions in the bulk (N), and the width, s_0, of the depletion layer.

$$\text{Total charge/area} \propto Ns_0$$

The electric field in the region is proportional to the charge per unit area.

$$\text{Electric field} \propto \text{charge/area} \propto Ns_0$$

The voltage between the n-type diffused layer and the p-type bulk on the far side of the depletion layer is proportional to the electric field times thickness of the depletion layer, and therefore to the density of negatively charged ions in the depletion layer times the square of the width of the depletion layer.

$$\text{Voltage} \propto \text{electric field} \times s_0 \propto Ns_0^2$$

The capacitance per unit area is just the charge per unit area divided by the voltage across the depletion layer. From the above equations the capacitance is proportional to the square root of the density of impurity atoms in the p-type bulk divided by the voltage on the across the junction.

$$\text{Capacitance/area} = Q/V \propto 1/s_0 \propto (N/V)^{1/2}$$

This relationship is plotted in Figure 30. Notice that the capacitance tends towards infinity as the voltage across the junction tends to zero. It would seem that this large capacitance would be disastrous for the performance of our integrated systems. However, this proves not to be the case. When the p/n junction was formed the n-type region had an excess of

negative charge carriers while the p-type bulk had an excess of positive charge carriers. When the two were brought together to form the junction, there was no voltage to prevent charge carriers of either type from flowing over into the opposite region. This initial flow caused the n-type layer to become more positive than the p-type layer. This flow ceased when just enough voltage built up to stop. In silicon the voltage required to prevent the flow of charge carriers in such a situation is approximately 0.7 volt. Thus the true voltage across the junction is this initial "built-in" voltage plus the voltage we apply in our circuit. The variation of the capacitance per unit area with applied voltage is shown in Figure 31. An approximate equation which can be used to calculate the junction capacitance C_j per unit area of diffused layers as a function of the applied voltage is given by.

$$C_j = 4.5\times10^{-12}[N/(V+0.7)]^{1/2} \ pf/\mu^2$$

In this equation, N, the density of impurity ions in the p-type bulk, should be given in number per cubic centimeter. The voltage is in volts and the capacitance per unit area is evaluated in picofarads per square micron. This equation is adequate for most design purposes.

Aside from the diffused regions, there are two other situations where the capacitance is of interest. The first is poly or metal over thick oxide and the second is the gate of an MOS transistor. We will discuss poly or metal over oxide first. Figure 32 illustrates once more the capacitance per unit area of a junction over the p-type bulk. If the poly or metal layer was lain on an oxide much thinner than the depletion layer, its capacitance would be nearly the same as that of the corresponding p/n junction. However, if an oxide is interposed whose thickness is of the order of the depletion layer thickness, the capacitance of the poly or metal line will be decreased. The formula which applies in this case is given by:

$$1/C_{total} = 1/C_j + 1/C_{ox}$$

A typical dependence is shown in Figure 32. For an oxide thickness d, $C_{ox} = 3.5\times10^{-2}/d$, where the thickness d is given in angstrom units (10^{-4} microns), and the result is in picofarads per square micron as before.

The most spectacular voltage dependence of a capacitance in the technology we will be using is that of the gate of an MOS transistor. When the gate voltage V_{gs} is less than the threshold voltage V_{th}, the capacitance of the gate to the bulk is just that given above for

metal or poly over oxide since the voltage on the gate merely depletes positive charge carriers back from the channel area. However, when the voltage on the gate reaches the threshold voltage of the transistor, negative charge is brought in under the gate oxide from the source of the transistor and the capacitance changes abruptly from the small valve associated with depleting charges in the bulk to the much larger oxide capacitance between the gate and the channel region. Further increase in voltage on the gate merely increases the amount of mobile charge under the gate oxide with no change in the width of the depletion layer underneath the channel. Hence, the character of the gate capacitance changes abruptly as the gate voltage passes through the threshold voltage.

The dependence of the total gate capacitance on gate voltage is shown in Figure 33. The capacitance from channel to bulk is completely separate from the gate to channel capacitance. It is associated with the depletion layer underneath the channel region, and is almost identical to that of a diffused region of the same area. When the gate voltage is below the threshold voltage, the gate to channel capacitance disappears altogether and we are left only with the small parasitic overlap capacitances between the gate and the source and drain regions.

Current Limitations in Conductors

One limit which is not covered in either the design rules or the electrical parameters section is that associated with the maximum currents through metal conductors. There is a physical process called *metal migration* whereby a current flux through a metal conductor, exceeding a certain limit, causes the metal atoms to move slowly in the direction of the current. If there is a small constriction in the metal, the current density will be higher and therefore more metal atoms will be carried forward from that point, narrowing the point still more. Hence, metal migration is a destructive mechanism causing open circuits in the metal layer carrying heavy currents.

For metals like aluminum this limit is a few times 10^5 amperes per square centimeter. This limit does not interfere too drastically with the design of integrated systems in current MOS technologies. However, many metal conductors in present integrated systems are operated near their current limit, and currents do not scale well as the individual elements are made smaller. Applying the scaling rules developed earlier, we found that the power per unit area is independent of the scale down ratio. However, the supply voltage decreases and therefore

the current per unit area increases as the devices are scaled down. For this reason it will not be possible to use processes for very large scale integrated systems where the metal thickness scales in the same way as other dimensions in the circuit. Much work will likely be done to develop processes enabling fabrication of metal lines of greater height relative to width than is presently possible.

Short pulses of current are known to contribute much less to metal migration than steady direct current. Nanosecond pulses of currents two orders of magnitude higher than the dc limit mentioned above may be carried in metal conductors without apparent damage. Therefore, switching current may not be as damaging to metal conductors as a steady current.

These effects strongly favor processes like CMOS which do not require static dc current, and favor design methodologies which maximize system function per unit dc current.

Choice of Technology

Before proceeding to the chapters on system design, let us briefly examine some alternative technologies. Then, using the knowledge developed in these first two chapters, we will discuss the reasons for selecting nMOS as the single technology used to illustrate LSI systems in this text. Some of the factors which must be considered in choosing a technology include circuit density, richness of available circuit functions, performance per unit power, the topological properties of circuit interconnection paths, suitablility for total system implementation, and general availability of processing facilities.

As the technology advances, more system modules can be placed on the same sized chip. An ultimate goal is the fabrication of large scale systems on single chips of silicon. For this goal to be attained, it is necessary that any signal which is required in the system other than inputs, outputs, VDD, and GND, must be generated in the technology on the chip. In other words no subsystem can require a different technology for the generation of its internal signals. Thus such technologies as magnetic bubbles are ruled out for full integrated systems because they are not able to create the signals required for all operations in the on-chip medium.

We believe that for any silicon technology to implement practical large scale systems, it

must provide two kinds of transistor. The rationale for this observation is as follows. In order to provide some kind of nonlinear threshold phenomenon there needs to be a transistor which is normally off at the lowest voltage used in the system. Bipolar technologies use NPN transistors for this purpose. nMOS technology uses n-channel enhancement mode devices. In addition to the transistor which is normally off when its control element is at the lowest voltage, a separate type of transistor needs to be supplied to allow the output of a driver device to reach the highest voltage in the circuit (VDD). In the bipolar technologies PNP lateral devices are used to supply this function, in the n-channel technology a depletion mode device is used, and in complementary MOS technology a p-channel enhancement mode device is used. Any of these three choices allow output voltages of drivers to reach VDD and thus meet the criterion mentioned above.

To date there have emerged three technologies which are reasonably high in density and scale to submicron dimensions without an explosion in the power per unit area required for their operation. These are the n-channel silicon gate process, the complementary MOS silicon gate process, and the integrated injection logic, or I^2L, process[2]. Although present forms of the I^2L technology lack an additional level of interconnect which is available in the silicon gate technologies, there is no inherent reason that such a level could not be provided. It is important to note that the types ot array functions which can be created in a technology are greatly enhanced by increased flexibility of interconnect. I^2L has the advantage over nMOS that the power per unit area (and hence the effective τ of its elementary logic functions) can be controlled by an off chip voltage. The decision concerning what point on the speed vs power curve to operate may thus be postponed until the time of application (or even changed dynamically).

The nMOS scaling has been described previously. Any technology in which a capacitive layer on the surface unduces a charge in transit under it to form the current control "transistor" will scale in the same way. Examples include Schottky Barrier Gate FET's (MESFETs), Junction FET's, and CMOS.

There are certain MOS processes (VMOS, DMOS) of an intermediate form in which the channel length is determined by diffusion profiles. While competitive at present feature sizes, these likely are interim technologies which will present no particular advantage at submicron feature sizes.

The scaling of the bipolar technology[1] is quite different from that of the MOS technologies.

For completeness, we include here a discussion of the scaling of bipolar devices, which may be of interest to those familiar with those technologies.

Traditionally, bipolar circuits have been "fast" because their transit time was determined by the narrow base width of the bipolar devices. In the 1950's, technologists learned how to construct narrow bipolar transistor base regions by forming them as the difference between two impurity diffusion profiles. This technique allowed very precise control of the distance perpendicular to the silicon surface, and therefore permitted the construction of very thin base regions with correspondingly short transit times. Since current in a bipolar device flows perpendicular to the surface, both the current and the capacitance of such devices are decreased by the same factor as the device surface dimensions are scaled down, resulting in no change in delay time performance. The base widths of high performance bipolar devices is already as thin as device physics allows. For this reason, the delay times of bipolar devices is expected to remain approximately constant as their surface dimensions are scaled down.

The properties of bipolar devices may be analyzed as follows. The collector current is due to the diffusion of electrons from emitter to collector. For a minority carrier density N(x) varying linearly with distance x, from N_0 at the emitter to zero at the collector (where x=d), the current per unit area is:

$$I/A = q(2D)dN/dx = q(2D)N_0/d = q(2kT/q)\mu N_0/d \qquad \text{[eq.2]}$$

where the diffusion constant $D = \mu kT/q$. The factor of two multiplies the diffusion constant in eq.2 because high performance bipolar devices operate at high injection level (injected minority carrier density much greater than equilibrium majority carrier density). Now, the inherent stored charge in the base region is:

$$Q/A = N_0 d/2 \qquad \text{[eq.3]}$$

Therefore, the transit time is

$$\tau = Q/I = d^2/[4\mu kT/q] \qquad \text{[eq.4]}$$

The form of equation 4 is exactly the same as that for MOS devices (eq.1., chapter 1.), with the voltage in the bipolar case being equal to 4kT/q (at room temperature kT/q = 0.025

volts). A direct comparison of the transit times is shown in Table 2.

Table 2. Transit Time:
τ = (Distance)^2/(Mobility×Voltage)

	MOSFET:	MESFET, JFET:	Bipolar:
Distance:	channel length	channel length	base width
Voltage:	~ VDD/2, many kT/q	~ VDD/2, many kT/q	4kT/q
Mobility: cm^2/v-sec, (Si)	surface mobility, ~800	bulk mobility, ~1300	bulk mobility, ~1300

At the smallest dimensions to which devices can be scaled, the base width of bipolar devices and the channel length of FET devices are limited by the same basic set of physical constraints, and are therefore similar in dimension. The voltage on the FET devices must be many times kT/q to achieve the required nonlinearity. Hence at the ultimate limiting small dimensions the two types of devices have roughly equivalent transit times. At these limiting dimensions, choices between competing technologies will be made primarily on the grounds of the topological properties of their interconnects, the functional richness of their basic circuits, simplicity of process, and ability to control dc current per unit area. As supply voltages are scaled down to the 1 volt range, MOS devices become similar in most respects to other FET type devices, and it is possible that mixed forms (MOS-JFET, MOS-MESFET, Bipolar-MESFET, etc.) may emerge as the ultimate integrated system technologies.

We have chosen to illustrate this monograph with examples drawn from the n-channel silicon gate depletion mode load technology. The reasons for this choice in 1977 are quite clear. In addition to meeting the required technical criteria we have described, this technology provides some important practical advantages to the student and to the teacher. It is the only high density technology which has achieved universal acceptance across company and product boundaries. The reader who wishes to implement his designs can go to essentially any company involved in fabricating silicon wafers and run his parts without fear that slight changes in the process or the vagaries of relationships with a particular firm will cut off his source of supply. It is also presently the highest density process available. This certainty of access to fabrication lines, the more generally widespread knowledge of

nMOS technology among members of the technical community, its density, and its performance similarity with bipolar technology in its ultimate scaling, are all important factors supporting its choice for this text on LSI Systems. However, the principles and techniques developed in this text can be applied to essentially any technology.

References

1. B. Hoeneisen, C. A. Mead, "Fundamental Limits in Micro-electronics--II. Bipolar Technology", *Solid-State Electronics*, Vol.15, 1972, pp. 891-897.

2. F. M. Klaassen, "Device Physics of Integrated Injection Logic", *IEEE Transactions on Electron Devices*, March 1975, pp. 145- , and cited papers by Hart & Slob, and by Berger & Weidmann.

Reading References

R1. A. S. Grove, "Physics and Technology of Semiconductor Devices", J. Wiley and Sons, 1967, is the early classic text on process technology and device physics.

R2. W. G. Oldham, "The Fabrication of Microelectronic Circuits", *Scientific American*, September, 1977, provides an excellent overview of the fabrication process.

R3. I. E. Sutherland, C. A. Mead, T. E. Everhart, "Basic Limitations in Microcircuit Fabrication Technology", ARPA Report R-1956-ARPA, November, 1976, contains a quantitative discussion of the many limiting factors in fabrication.

R4. R. S. Muller, T. I. Kamin, "Device Electronics for Integrated Circuits", Wiley, 1977, provides insight into the device physics relevant to current integrated circuit practice.

Chapter 3: Data and Control Flow in Systematic Structures

Sections:

Notation - - - Two Phase Clocks - - - The Shift Register - - - Relating Different Levels of Abstraction - - - Implementing Dynamic Registers - - - Implementing a Stack - - - Register to Register Transfer - - - Combinational Logic - - - The Programmable Logic Array - - - Finite State Machines - - - Towards a Structured Design Methodology

The process of designing a large-scale integrated system is sufficiently complex that only by adopting some type of regular, structured design methodology can one have hope that the resulting system will function correctly, and not require a large number of redesign iterations. However, the methodology used should allow the designer to take full advantage of the architectural possibilities offered by the underlying technology.

In this chapter we present a number of examples of data and control flow in regularized structures, and discuss the way in which a hierarchy of these structures can be assembled into larger groups to form subsystems, and then finally the overall LSI system. The design methodology suggested in this chapter is but one of many ways in which LSI system design might be structured. The particular circuit form presented does tend to produce systems of very simple and regular interconnect topology, and thus tends to minimize the areas required to implement system functions. Arrays of pass transistor logic in register to register transfer paths are used wherever possible to implement system functions. This approach tends to minimize power dissipated per unit area, and, with level restoration at appropriate intervals, tends to minimize the time delay per function. The methodology developed here is applied in later chapters to the architecture and design of a data processing path and its controller, which together form a microprogrammed digital computer.

Computer architects, who usually design systems in a rather structured way using commercially available MSI and LSI circuit modules, are often surprised to discover how unstructured is the design within those modules. In principle one can use the basic NAND and NOR logic gates described in Chapter 1 to implement combinational logic, build latches from these gates to implement data storage registers, and then proceed to design LSI systems using traditional logic design methodology as applied to discrete devices. LSI systems are often designed this way at the present time. However, it is unlikely that such unstructured approaches to system design can survive as the technology scales down towards VLSI.

There are historical reasons for the extensive use of random logic within LSI systems. The first microprocessors produced by the semiconductor industry were fairly direct mappings into LSI of early generation central processor architectures. A block diagram of the Intel 4004, the earliest microprocessor to see widespread commercial application, is illustrated in figure 1a. The actual chip layout of the 4004 shown in Figure 1b indicates the complexity of the LSI implementation of this simple central processing unit. Such LSI systems, directly mapping data paths and control functions appropriate in earlier component technologies, of necessity contained a great deal of random logic. However, the extensive use of random logic results in chip designs of very great geometrical and topological complexity, relative to their logical processing power.

To deal with such complexity, system design groups have often stratified the design problem into architecture, logic design, circuit design, and finally circuit layout, with specialists performing each of these levels of the design. However, such stratification may result in missed opportunities for important simplifications in the realization of system functions.

Switching theory provides formal methods for minimizing the number of gates required to implement logic functions. Unfortunately, such methods are of little value in LSI systems, since the area occupied on the silicon surface by circuitry is far more a function of the topological properties of the circuit interconnections than it is of the number of logic gates implemented. The minimum gate implementation of a function often requires much more surface area for its layout than does an alternative design using more transistors but having simpler interconnect topology.

There are known ways of structuring system designs implemented using traditional logic design methods (notably the *polycell*[1] technique), but these usually pay a heavy penalty in area, power, and delay time. The polycell technique provides the logic designer having limited knowledge of LSI systems with a means of implementing modest integrated circuit designs directly from logic equations. Such techniques, while being very valuable expedients, do not take advantage of the true architectural potential of the technology, and do not provide the student with insight which might lead to directions for further progress.

Switching theory not only yields the minimum number of gates to implement a logic function, but it also directly synthesizes the logic circuit design. Unfortunately, at the present time there is no formal theory which provides us even with the limiting minima of areas, power, and delay times for the implementation of logic functions in integrated

The Intel 4004 Microprocessor: An Early LSI System

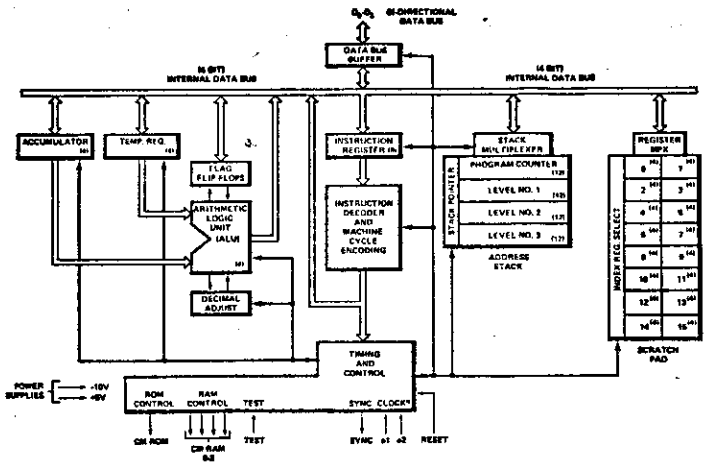


Fig. 1a. 4004 Block Diagram

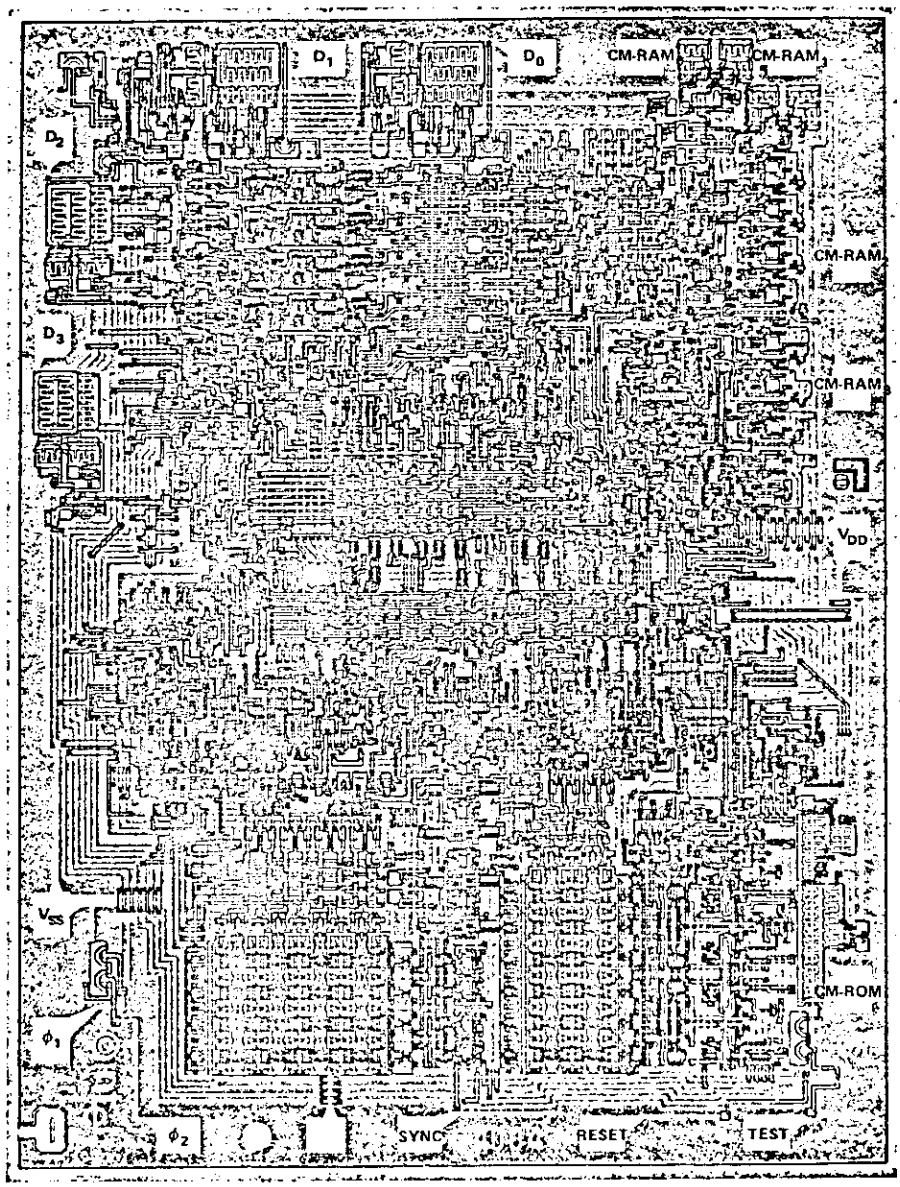


Fig. 1b. 4004 Chip Photomicrograph with Pin Designations

[reprinted with permission of G. Moore, Intel Corp.]

systems.

In the absence of a formal theory which provides us with the minima of area, power, and delay time to implement logic functions, we can at best develop and illustrate alternative design methodologies which tend to minimize these physical parameters. Proposed design methodologies should in addition provide means of structuring system designs so as to constrain complexity as circuit density increases towards VLSI. We hope that the examples and techniques presented in this text will serve to clarify these issues and stimulate others to join in the search for the answers[2].

Notation

There are a number of different levels of symbolic representation for MOS circuits and subsystems used in this text. Figures 2a., 2b., 2c., and 2d., illustrate a NAND gate at several such levels. At times it may be necessary to show all the details of a circuit's *layout geometry* in order to make some particular point. For example, a clever variation in some detail of a circuit's layout geometry may lead to a significant compaction of the circuit's area without violating the design rules.

Often, however, a diagram of just the topology of the circuit conveys almost as much information as a detailed layout. Such *stick diagrams* may be annotated with important circuit parameters if needed, such as the L/W ratios shown in figure 2b. Many of the important system architectural parameters of circuits and subsystems are a reflection of their interconnect topologies.

Alternative topologies often lead to very different layout areas after compaction. Discovering a clever starting topological structure for a design usually yields far better results than does applying brute force to the compression of final layout geometries. For this reason, many of the important structural concepts in this chapter and throughout the text will be represented (and, hopefully, more easily conveyed to the reader) by use of colored stick diagrams. The color coding of the stick diagrams is the same as for layout geometries, and is as follows: *green* symbolizes *diffusion*; *yellow* symbolizes *ion implantation* for depletion mode transistors; *red* symbolizes *polysilicon*; *blue* symbolizes *metal*; *black* symbolizes a *contact*.

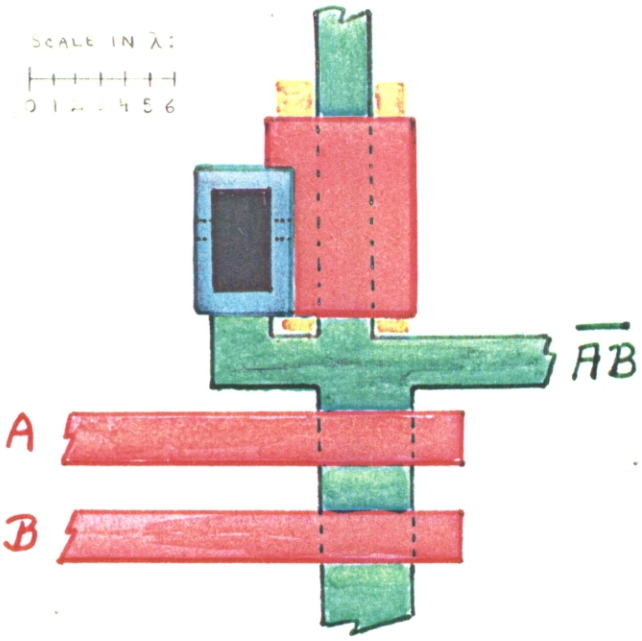


Fig.2a. NAND Gate: Layout Geometry

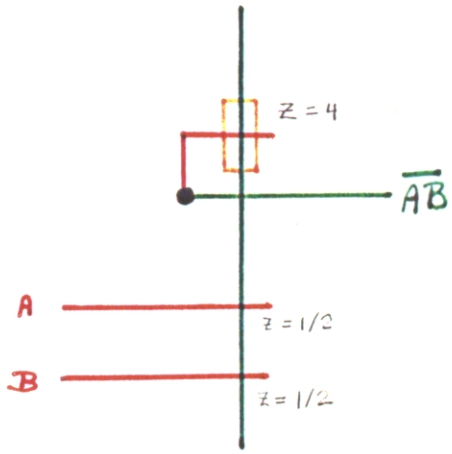


Fig.2b. NAND Gate: Topology (Stick Diagram)

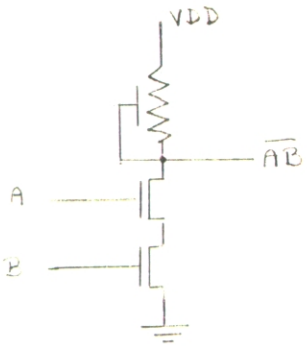


Fig.2c. NAND Gate: Circuit Diagram

Fig.2d. NAND Gate: Logic Symbol

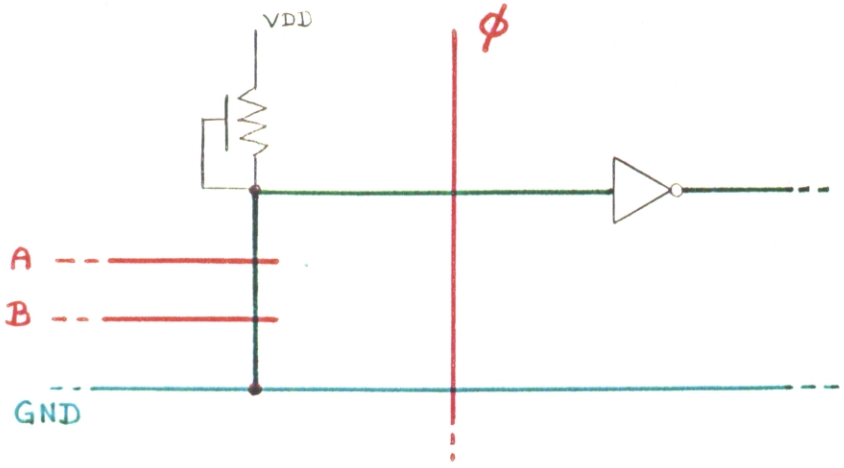


Fig. 2e. Example of Mixed Notation

Later, through a number of examples in chapter 4, we will present the details of procedures by which the stick diagrams are transformed into circuit layouts, and then digitized for maskmaking. Note that if this topological form of representation were formalized, one might consider "compiling" such descriptions by implementing algorithms which "flesh out and compress" the stick diagrams into the final layout geometries[3], according to the constraints imposed by the design rules.

When the details of neither geometry nor topology are needed in the representation, we may revert to the familiar *circuit diagrams* and *logic symbols*. At times we may find it convenient to *mix* several levels in one diagram, as shown in figure 2e. A commonly used mixture is: (i) stick diagrams in portions where topological properties are to be illustrated, (ii) circuit symbols for pullups, and (iii) logic symbols, or defined higher level symbols, for the remaining portions of the circuit or system.

We will define logic variables in such a way that a *high voltage* on a signal path representing that variable corresponds to that variable being *true* (logic-1). Conversely, a *low voltage* on a signal path representing that logic variable corresponds to the variable being *false* (logic-0). Here *high voltage* and *low voltage* mean well above and well below the logic threshold of any logic gates into which the signal is an input. This convention simplifies certain discussions of logic variables and the voltages on the signal paths representing them. Thus when we refer to the logic variable β being *high*, we indicate simultaneously that β is *true* (logic-1) and is represented on the so-named signal path by a *high voltage* (well above the logic threshold). In boolean equations and logic truth tables we use the common notation of 1 and 0 to represent *true* and *false* respectively, and by implication *high* and *low voltages* on corresponding signal paths.

Two Phase Clocks

We will often make use of a particular form of "clocking" scheme to control the movement of data through MOS circuit and subsystem structures. By clocking scheme we mean a strategy for controlling the times at which data is allowed to move into and through successive processing stages in a system, while controlling the isolation of one stage from another during a particular cycle of processing. Many alternative clocking schemes are possible, and a variety are in current use in different LSI systems[4]. The clocking scheme used in an LSI system is closely coupled with the basic circuit and subsystem structuring,

and has major architectural implications. For clarity and simplicity we have selected one clocking scheme, namely the use of *two-phase, non-overlapping clock signals*. This scheme is used consistently throughout the text, and is well matched to the type of basic structures possible in MOS technology.

The two clock signals φ_1 and φ_2 are plotted as a function of time in figure 3. The signals both switch between zero volts (logic-0) and a voltage near VDD (logic-1), and both have the same period, T. Note that both signals are non-symmetric, and have non-overlapping *high* times. The *high* times are somewhat shorter than the *low* times. Thus φ_2 is *low* all during each of those time intervals from when φ_1 rises, nears VDD, and then falls back to zero. Symbolizing the *rise* of a signal φ as $\uparrow\varphi$, and the *fall* as $\downarrow\varphi$, we also have a similar rule for φ_1, namely $\varphi_1 = 0$ all during each time interval from $\uparrow\varphi_2$ to $\downarrow\varphi_2$. Therefore, at all times the logic AND of the two signals equals zero: $[\varphi_1(t)] \cdot [\varphi_2(t)] = 0$, for all t. For convenience, we will often use the following equivalence in our descriptions: *"during the time period when φ_i is high"* $\equiv$ *"during φ_i"*. In the next section we will illustrate the use of these two clocking signals to move data through some simple MOS circuit structures.

The Shift Register

Perhaps the most basic circuit structure for movement of a sequence of data bits is the *serial shift register*, shown in circuit diagram form in figure 4a. The shift register consists simply of level restoring inverters coupled by pass transistors, with the movement of data controlled by applying the clock signals φ_1 and φ_2 to the gates of alternate pass transistors in the sequence.

Data is shifted from left to right as follows. Suppose a logic signal X is present on the leftmost input to the shift regiter when clock signal φ_1 rises. Then, during the time when φ_1 is *high*, this signal will propagate through the pass transistor and be stored as charge on the input capacitance of the first inverter stage. For example, if the signal X is *low*, then the inverter input gate capacitance will be discharged towards zero volts during the time when φ_1 is *high*. On the other hand, if X is *high*, the inverter input capacitance will be charged up towards VDD - V_{th} during φ_1.

When the clock signal φ_1 falls, the pass transistor becomes an open circuit, isolating the charge on the input of the inverter. The second clock phase is now initiated by the rise of

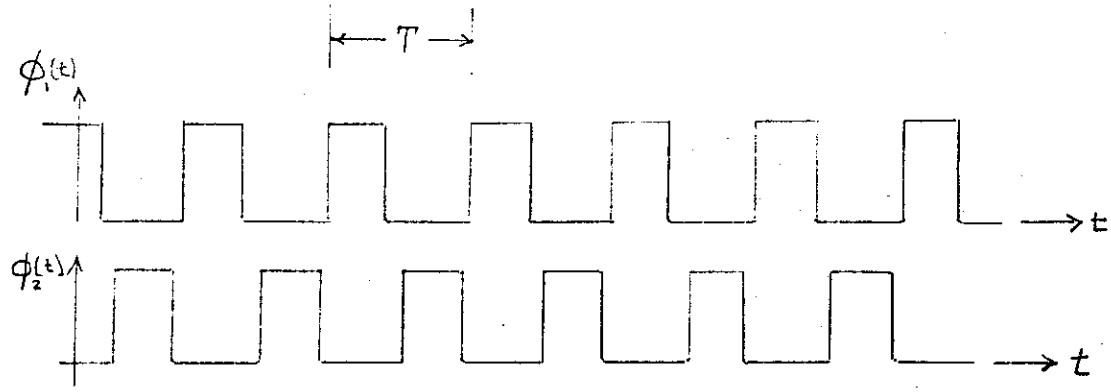


Fig.3. Two Phase Non-Overlapping Clock Signals

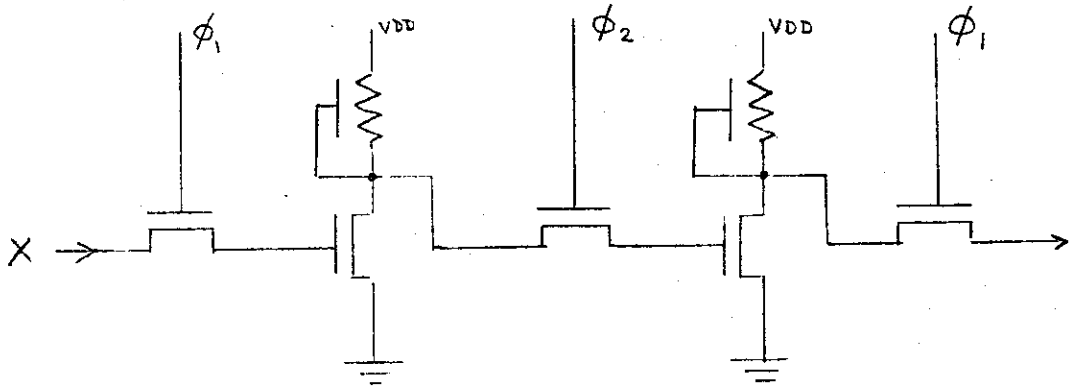


Fig.4a. Shift Register: Circuit Diagram

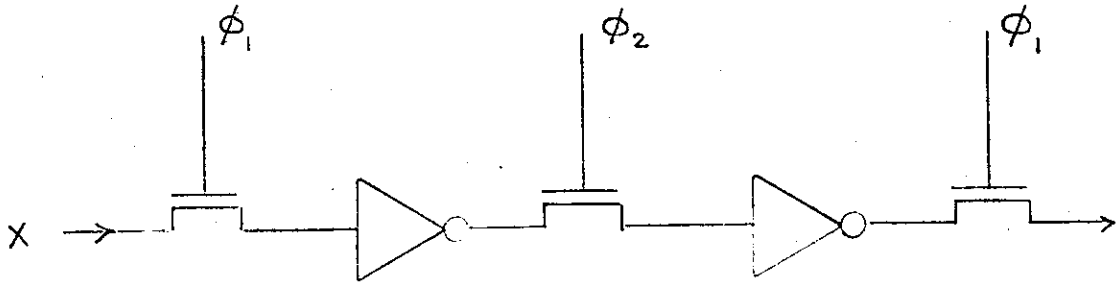


Fig.4b. Shift Register: In Mixed Notation

φ_2. During the time interval when φ_2 is *high* the logic signal X, now inverted, will flow through the second pass transistor onto the gate of the second inverter. This pattern can be repeated an arbitrary number of times to produce a shift register of any length.

It is important to note that since the clock signals do not overlap, the successive pairs of stages of the shift register are effectively isolated from one another during the transfer of data between inverter pairs. For example, when φ_1 is *low*, and φ_2 is *high*, all adjacent inverters connected by the φ_2 controlled pass transistors are in the process of transferring data from the left to the right members of the pairs. All these pairs of inverters are isolated from each other by the intervening φ_1 controlled pass transistors which are all open circuits when φ_1 is *low*.

It is also important to note that the shortest period, T, we can use for clocks controlling such data transfers is determined by the time required to adequately charge or discharge the inverter input gate capacitance through the pass transistor and the preceding stage pullup or pulldown in the worst case stage of the shift register. To this time must then be added an increment of time sufficient to insure that the clocks do not overlap. This concept of minimum clock period as a function of basic circuit parameters will be extended to the more general case of a microprogrammed computer system in later chapters.

Figures 4b and 4c illustrate the serial shift register using mixed notations. In figure 4b, each inverter circuit diagram has been replaced by its logic symbol. In figure 4c, the pass transistor circuit symbols have been replaced by their stick diagrams. When visualizing the inverter, as represented by its logic symbol, in a circuit structure containing mainly stick diagrams, two points should be kept in mind:

(i) The input to the inverter leads directly to the gate, and thus the gate capacitance, of the inverter's pulldown transistor. This input may be used to store a data bit by isolating the charge representing the bit with a pass transistor. Note that the input path will end up on the poly level within the inverter. A contact cut may thus be required to connect the poly gate and the metal or diffusion path on which the signal enters the inverter.

(ii) Since the connection between the source and gate of the inverter pullup transistor requires a connection of all three conducting levels, the inverter output signal may easily be routed out on any one of the three levels.

Identical serial shift registers can be stacked next to each other and used to move a sequence

Fig.4c. Shift Register: More Mixed Notation

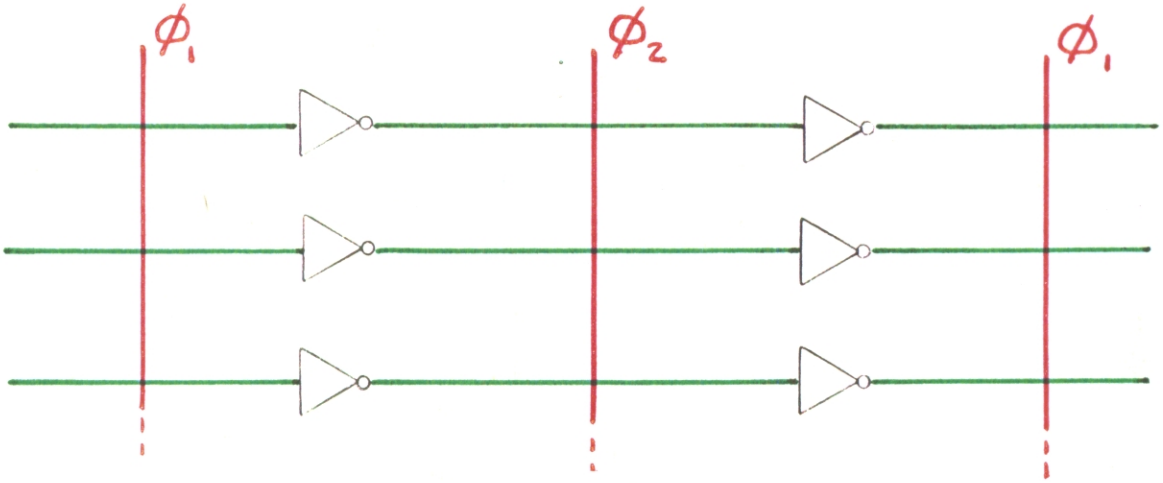


Fig.5a. Array of Shift Registers

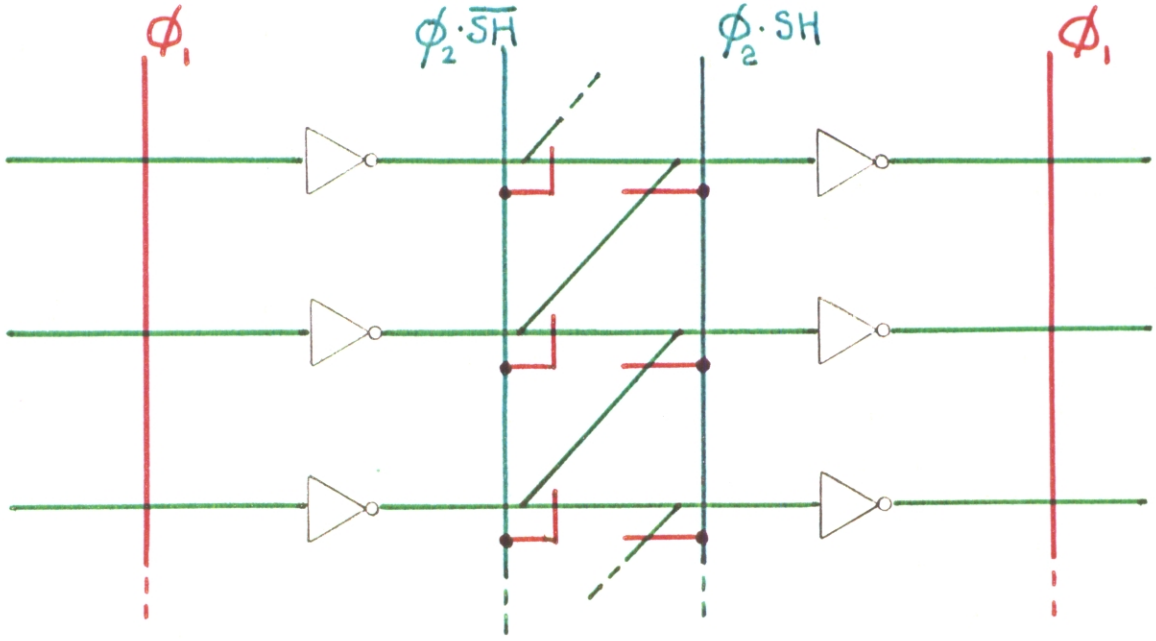


Fig.5b. Shift-Up Register Array

of data *words*, as shown in figure 5a. The simple structure in figure 5a anticipates the elegant simplicity of the topology of many important MOS LSI system functions. By connecting the successive inverter stages with diffusion paths, the pass transistors controlled by the clock signals are formed by simply running vertical clock lines in poly. The structure in figure 5a also anticipates another important point: topological simplification often results when control signals flow on lines that are at right angles to the data flow lines. In this way as many bits as necessary can be processed in parallel with the same control signals.

The example in figure 5a is so rudimentary it is perhaps difficult to visualize the two clock signals as actually containing control information. Let us consider a slightly more complex example, the *shift-up register array* shown in figure 5b. In this structure, each data bit moving from left to right during φ_2 has two alternative pass transistor paths through which it can proceed to the next stage: a straight through path, and a path which shifts it up to the next higher row. If the shift control signal SH is *low*, then $[\varphi_2 \cdot SH']$ is *high*, and the straight through pass transistor paths are used during φ_2. At the same time, $[\varphi_2 \cdot SH]$ is *low*, thus preventing data flow through the shift-up pass transistor paths. On the other hand, if SH is *high*, the straight through pass transistors become open circuits and the shift-up pass transistor paths are used during φ_2, resulting in the entire data word being shifted vertically as well as horizontally. Here the vertical control lines are run in metal, and the pass transistors are selectively formed by crossing the appropriate diffusion paths with short poly lines.

Relating Different Levels of Abstraction

In the discussions in this chapter, we will not have to make extensive calculations of the detailed electrical behavior of the devices and circuits involved in order to analyze the general behavior of digital logic constructed with these devices and circuits. Most of the examples presented in this chapter, and throughout the text, build upon the use of pass transistors coupling inverting logic stages as a means of structuring designs. The general results of chapter one provide the solutions to most device and circuit problems encountered, such as ratio and delay calculations, etc. In most cases, design concepts can be worked out using stick diagrams, and only at the stage of transforming the circuit topology into the detailed circuit layout geometry will these calculations need to be worked out, either by hand or with circuit simulation programs.

This indicates that we can greatly simplify our mental model of the circuits involved, so as to more quickly and easily analyze or explain to others the function of a given circuit, and more easily visualize and invent new circuit structures without drifting too far away from physically realizable and workable solutions. Of course, it is in general a dangerous practice to oversimplify our abstractions of electronic circuit behavior, and there are some nMOS circuits of deceptively simple appearance which have exceedingly complex behavior. However, throughout large portions of digital LSI systems, if the circuit and subsystem design is carefully structured as suggested in this text, an extremely simple mental model of device and circuit behavior will prove adequate to predict circuit and subsystem behavior.

Figure 6a illustrates a simple way of visualizing the operation of successive inverting logic stages coupled by pass transistors. Assume for the moment that any pass transistors in the paths between stages are *on*. To visualize the time behavior of an inverter, and the effect of the pullup L/W to pulldown L/W ratio, imagine the flow of current from VDD to GND as the flow of a fluid, and the inverter's two transistors as valves. The pullup transistor is always *on*, and so the upper "valve" is always open. However, the "valve" corresponding to the pulldown transistor may be either open or closed, depending on the amount of charge on its gate.

In figure 6a, the input to inverter-A is a logic-0, so the pulldown of inverter-A is *off*, and the lower valve is closed. This results in a diversion of current to the large charge storage site corresponding to the gate of the pulldown of inverter-B. If sufficient charge has flowed onto this gate, corresponding to a high level of fluid in the tank representing the gate capacitance, then the pulldown of inverter-B is turned *on*, and thus the lower valve of inverter-B is open. If the lower valve in inverter-B is much larger than the upper one, corresponding to a practical pullup to pulldown size ratio, then the pulldown of inverter-B can sink all the source current provided by the pullup, and in addition drain off any charge remaining on the next gate(s), given sufficient time. Thus we can visualize the sequence of inversions of a logic signal propagating through successive inverter stages as an alternation between high and low levels of fluid in the storage tanks. In addition, we can visualize some of the time behavior of the signal's propagation: the larger the gate capacitances (storage tanks), the longer it takes to build up enough charge tc open the next stage, and similarly the longer it takes to drain charge off the next stage to turn it off.

Figure 6b represents the same physical circuitry modelled in figure 6a, but on successively higher levels of abstraction. When analyzing circuit or logic diagrams showing successive

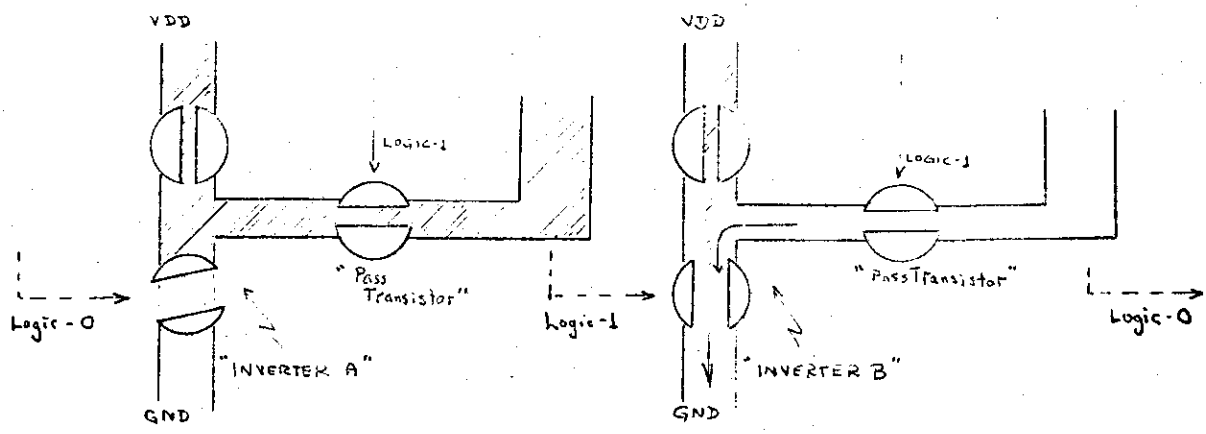


Fig. 6a. A Way of Visualizing the Operation of Successive Inverter Stages Coupled by Pass Transistors

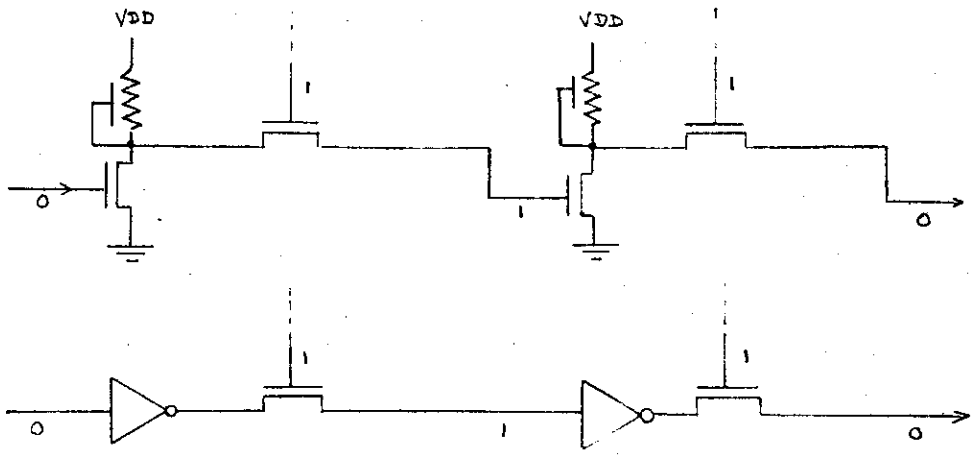
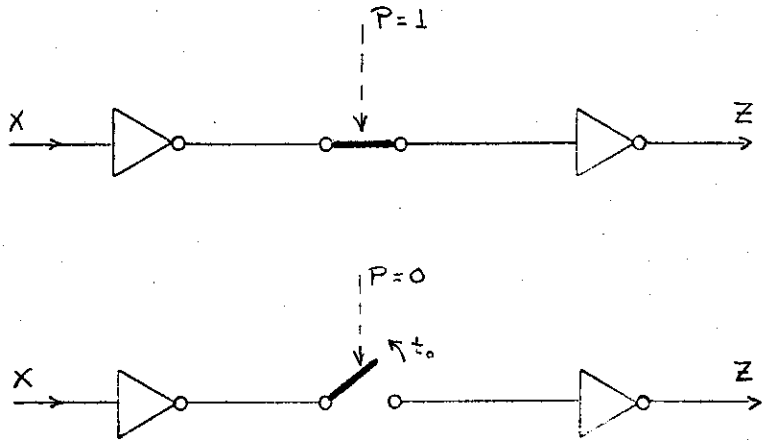


Fig. 6b. Circuit and Logic Diagrams of Successive Inverter Stages Coupled by Pass Transistors



While P = 1,

$Z(t) = X(t - \Delta t)$

If $P \rightarrow 0$, at $t = t_0$,

$Z(t) = Z(t_0)$, for $t > t_0$

Fig. 6c. The Effect of the Pass Transistor "Switch"

inverting logic stages, as in figure 6b, one should keep the model of figure 6a in mind. Whether one is a novice or an expert in LSI system design, it is very helpful to compress the details of any given lower level of abstraction, so as to reduce the complexity of the problems presented at the next higher level, and enable the mind to span problems of larger scope.

We are now able to visualize a very simple model for the pass transistor: it is in fact like a valve, or "switch" in the path between an inverter and the next charge storage site, i.e. the input gate of the next inverter. Figure 6c shows two inverters coupled by a pass transistor, with the pass transistor informally symbolized as a "switch". In the upper diagram of figure 6c, the pass transistor input is a logic-1, and so the "switch" is in the *on* position, resulting in the output Z being equal to the input X, after a suitable delay time Δt. Thus during the time the pass transistor gate input P = 1, the output $Z(t) = X(t - \Delta t)$.

In the lower diagram of figure 6c, the pass transistor "switch" is moved to the *off* position due to P going to logic-0. Therefore, according to our model, the valve in the path is shut, and the charge, or lack of charge, is isolated in the storage site. Thus once the pass transistor valve is shut, Z remains at a constant value, independant of changes in X: i.e., if $P \rightarrow 0$, at $t = t_0$, then $Z(t) = Z(t_0)$, for $t > t_0$.

These simple visualizations of the inverter and the pass transistor will carry us fairly far into LSI subsystem design. Several logic circuits in this chapter are drawn first in stick diagram form, and then informally sketched with pass transistors replaced with "switches", both to clarify the behavior of the circuits involved, and to further demonstrate the applicability of the model.

Implementing Dynamic Registers

Registers for the storage of data play a key role in digital system design. It is interesting to note that a group of adjacent inverters, with their gates isolatable by preceding pass transistors, can be considered a form of temporary storage register. This arrangement is illustrated in figure 7, which shows several levels of symbolism for this *dynamic register*. Such a register is very simple in structure. It consists of only three transistors per bit position: the pass transistor and the two transistors of the inverter. However, this dynamic form of register will preserve data only as long as charge can be retained on the inverter

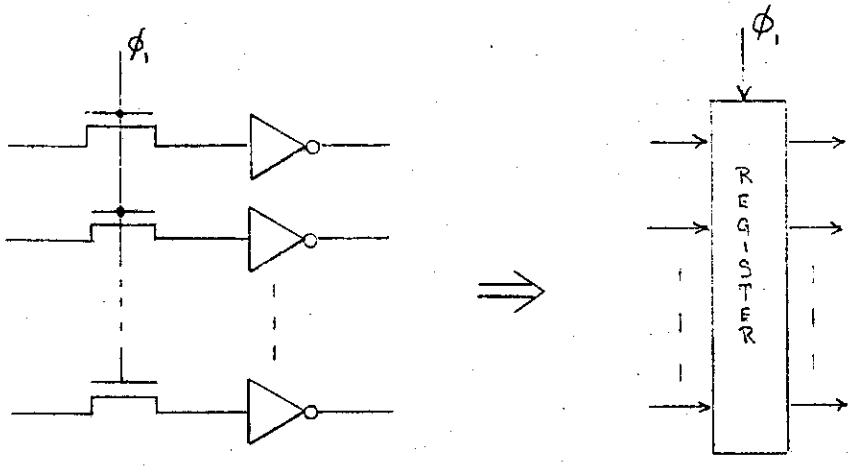


Fig.7. A Dynamic Register

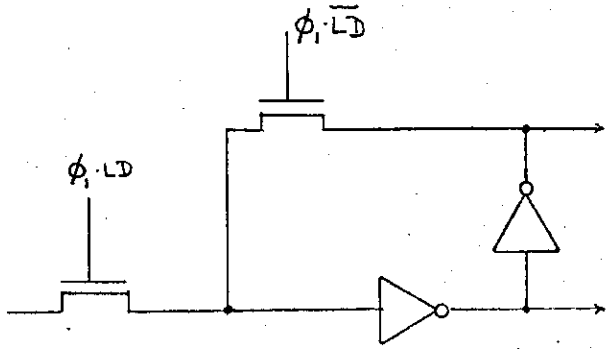


Fig.8. A Selectively Loadable Dynamic Register Cell

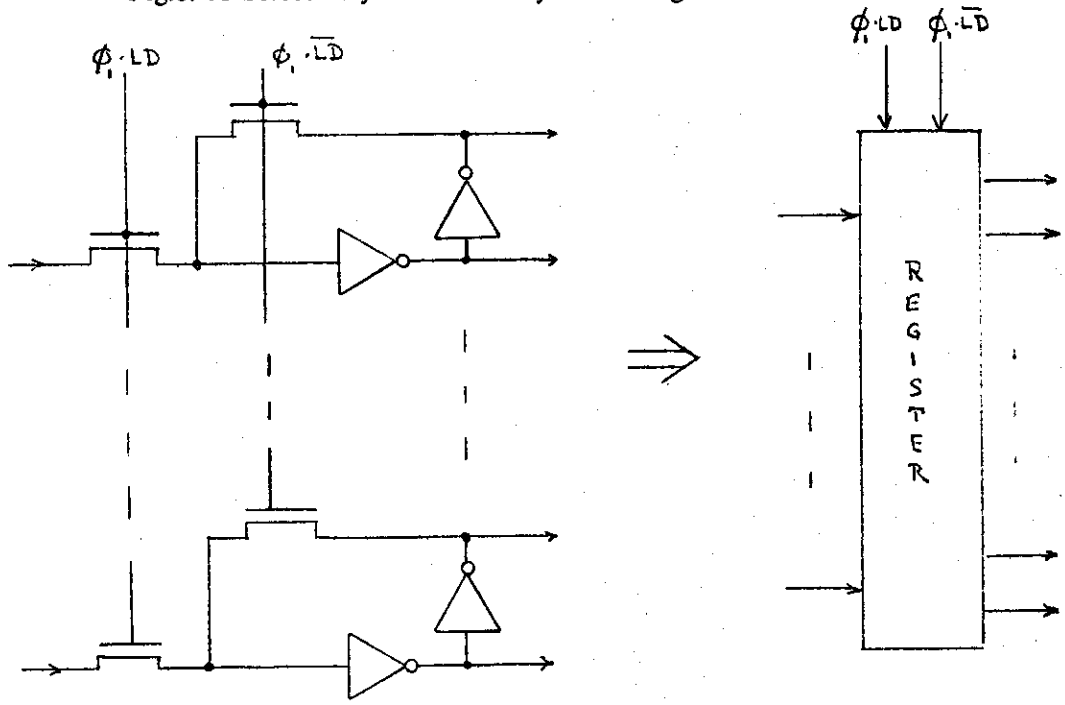


Fig.9. A Selectively Loadable Dynamic Register

input gates. Typically dynamic registers are used in situations where the input gate updating control signals are applied frequently. In particular, it is ideal in a clocked system in which it is reloaded every clock cycle, as in the shift register.

Suppose we wish to construct a simple register which can be loaded during the appropriate clock phase under the control of a *load* signal, and which will retain its information through an indefinite number of successive clock periods until it is reloaded using the *load* signal. A one bit cell for such a register may be constructed using cross coupled inverters in the configuration shown in figure 8. This register cell is still dynamic in form, since it uses charge storage on the gate of the first inverter to preserve its state. However, it need not be updated on every successive φ_1 as was the simple register in figure 7. The pass transistor leading to it from the preceding stage is switched on only when *both* φ_1 and LD are *high.* On any following φ_1 when LD is *low*, the cell updates itself by the feedback path through the second pass transistor. Figure 9 illustrates a selectively loadable register composed of such cells. One important feature of this type of register is that it provides as output both the true and complemented forms of the stored data. This feature is often useful when the data is to be processed by a following network of combinational logic.

There are many other, more elaborate, forms of dynamic and static registers, and flip-flops, some of which we will discuss in specific examples in later chapters. However, the above two simple forms of register are sufficient for many of the required data storage applications within LSI systems.

Implementing a Stack

The ideas used to construct simple dynamic registers in the preceding section may be applied to the construction of more sophisticated and interesting subsystems. In this section we will describe the implementation of a *stack*. This type of subsystem is commonly called a *last-in, first-out* stack (LIFO), or *pushdown* stack, although we will diagram it horizontally rather than vertically. It is a shift register array with three basic operations: during each full clock period (1) we can *push* in a new data word at one end of the array, pushing all previously entered words one word position further into the array, or (2) we can leave all words in their current position, or (3) we can *pop* out a word from the end of the array, pulling all previously entered words back out by one word position.

Figure 10a shows the structure of one horizontal row of the stack. Here we have implemented a left-right shift register which can perform the three operations: shift data left to right, hold data in place, or shift data right to left. There are four control signals used, two of them being active during φ_1 and two of them being active during φ_2. The signals φ_1 and φ_2 are our familiar two phase, non-overlapping clock signals.

In order for data to be shifted from left to right, the shift right control line (SHR) is driven *high* during φ_1, followed by driving the transfer right control line (TRR) *high* during φ_2. The bit of data appearing at the left is thus transferred by this operation onto the gate of the first inverter during φ_1, and thence to the gate of the second inverter during φ_2. In order for data to be held in place, the signal transfer left (TRL) is driven *high* during φ_1 and transfer right (TRR) is driven *high* during φ_2, causing the data to recirculate upon itself without shifting. Note that the data can be obtained at any time from the output of the first inverter. However, since new data may come to the gate of the first inverter during φ_1, the only safe time to take data out to the left is during φ_2. The transfer of data from right to left is caused by driving the shift left control (SHL) line *high* during φ_2, followed by driving transfer left (TRL) *high* during φ_1.

Figure 10b illustrates a possible topological structure of one horizontal row of the stack. The two inverters for one stage of the row are nested between two horizontal pathways on the diffusion level for shifting bits right or left. VDD, GND, and the four control lines run vertically in metal. The four pass transistors required for controlling the movement of data are conveniently implemented by short poly lines which cross the horizontal diffusion tracks at appropriate positions. Note that the entire row is composed of 180° rotations and repetitions of a basic cell containing one inverter.

In a typical implementation of the complete LIFO stack, a number of such rows would run parallel to each other in the horizontal direction. The number would equal the width in bits of the data words involved. The control lines would run across the entire stack, perpendicular to the direction of data flow. For data words of any substantial width, the capacitive loading on the control signals would be sufficient to warrant use of super-buffer drivers.

The stack as a whole may be controlled with only two logic signals: one signalling *push*, and the other signalling *pop*. The activation of neither of these two signals would cause the data to merely recirculate in place, awaiting the next active instruction.

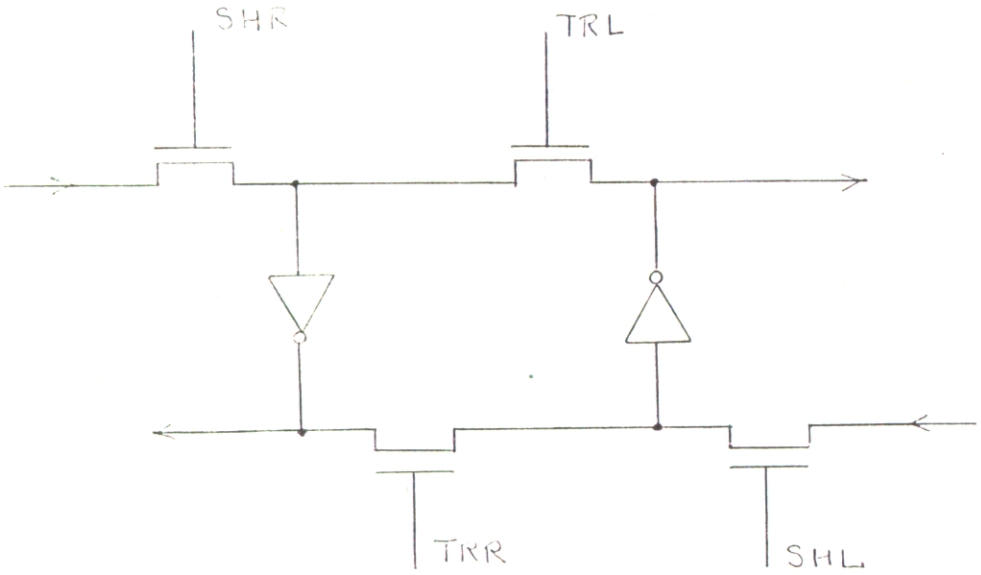


Fig.10a. One Horizontal Row of the Stack

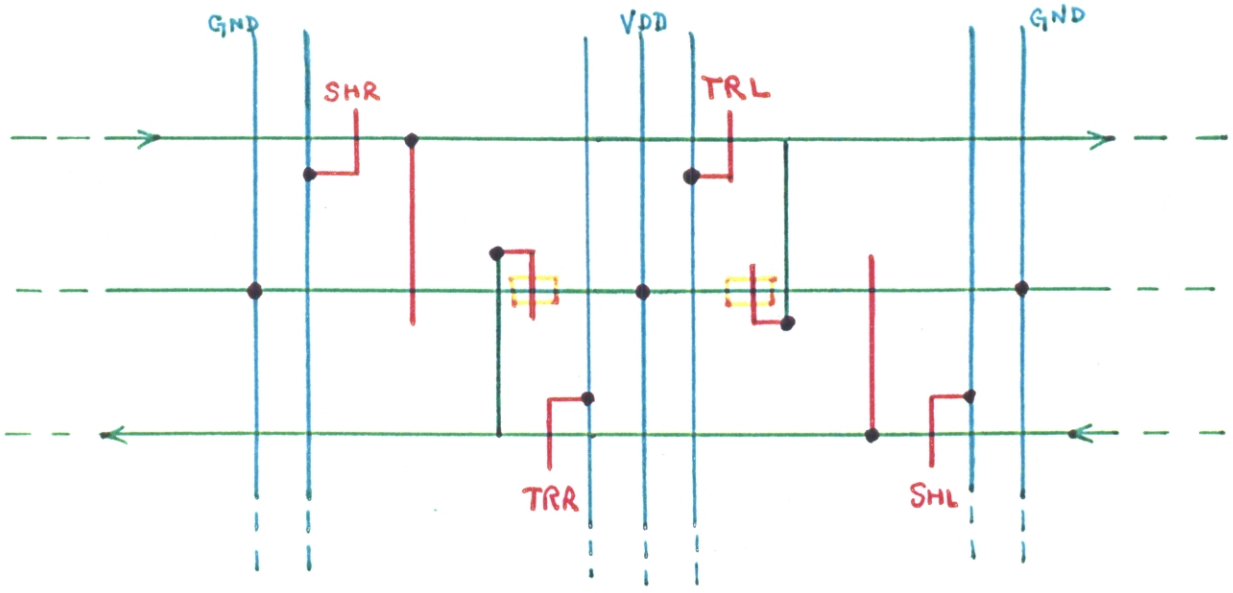


Fig.10b. Topology of One Horizontal Stack Row

Let us consider how to derive, from *push* and *pop*, the control signals for driving the four control lines SHR, TRR, SHL, TRL. A possible scheme is shown in Fig. 10c. We use random logic for this purpose since only a few gates are required to control the large, regular array of circuit cells in the stack. The operation which determines what the stack will do during the subsequent clock phase is brought in on the path labeled OP. It is important to note in the following that only one signal path (OP) is required to bring in both *push* and *pop* logic signals, since these are active on mutually exclusive clock phases.

During the time period when φ_1 is *high*, the signal OP is fed through the upper pass transistor into the inputs of the two NOR gates g_1 and g_2. The outputs for these two NOR gates are *low* during this period, since φ_1 is *high*.

If the incoming OP signal is *high* while φ_1 is *high*, then the lower input of NOR gate g_2 will be *low*. Thus when φ_1 falls *low*, the output of g_2 will go *high*, thereby driving SHL *high*. If the OP signal is instead kept *low* while φ_1 is *high*, then the output of the NOR gate g_1 will go *high* on the fall of φ_1, thereby driving TRR *high*.

During the period when φ_2 is *high* and either the shift left (SHL) or the transfer right (TRR) operation is being executed, the signal on the OP line is being stored on the corresponding input gates of the lower two NOR gates, g_3 and g_4. Thus, if OP is *high* while φ_2 is *high*, a logic-0 is stored on the input of the NOR gate g_4, and during the subsequent φ_1 *high* period, SHR will be driven *high*. Conversely, if OP is *low* while φ_2 is *high*, TRL will be driven *high* during the following φ_1 *high* period.

This control scheme is summarized in the timing diagrams in figure 10e. Here we see that holding OP *high* during φ_2, followed by *low* during φ_1, implements *push*. Holding OP *low* during both φ_1 and φ_2 causes the data to recirculate in place. Holding OP *high* during φ_1, followed by *low* during φ_2, implements *pop*. Thus, the single signal path, OP, is sufficient to carry the stack control signals into the stack.

This kind of control scheme recognizes that between any operation and its next occurrence, there must be a lull period where control information is taken in and set up for the subsequent operation. The scheme takes advantage of these lull periods, when possible, to perform other operations which can be done without conflict. It is an example of a fundamental design technique which can be extended to larger system structures.

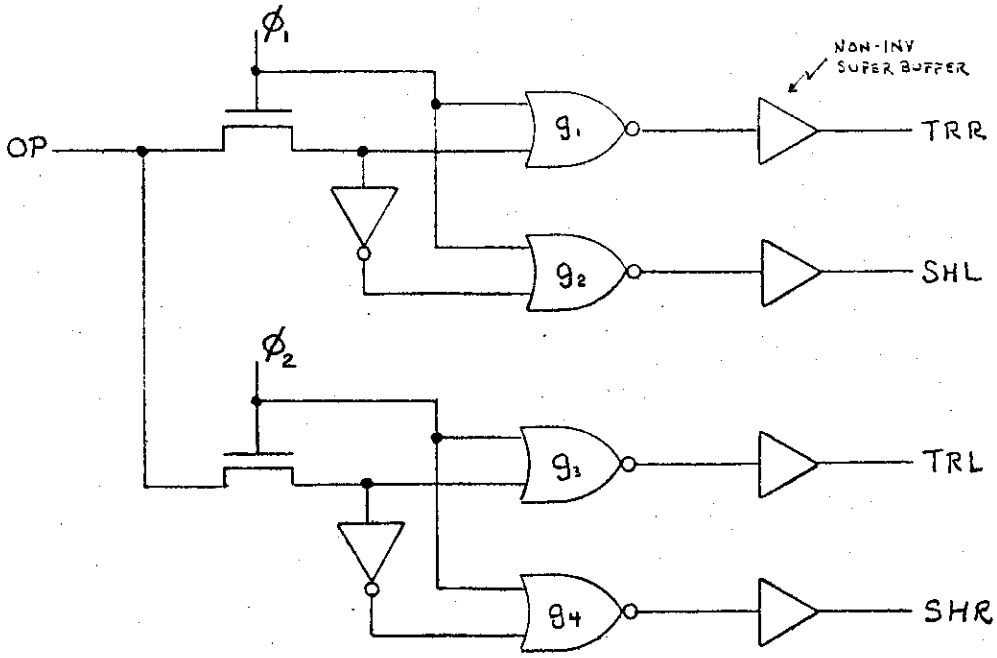


Fig.10c. Generating the Stack Control Signals

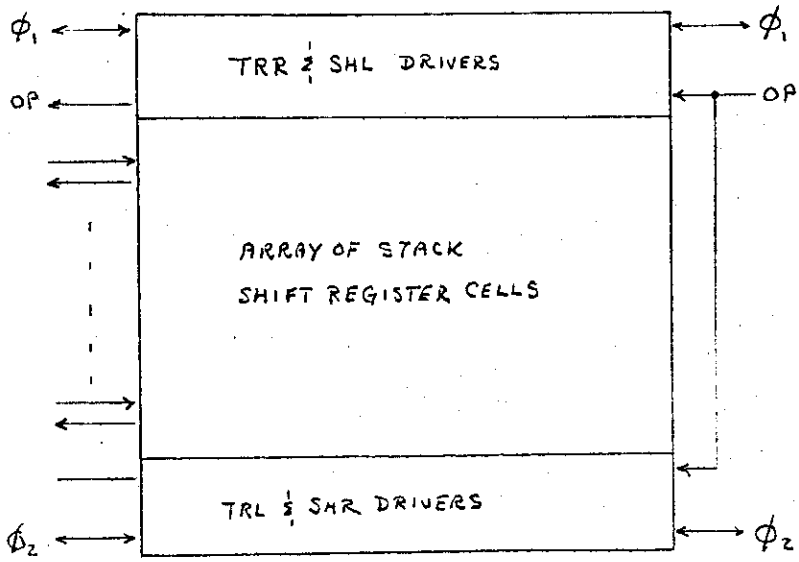


Fig.10d. Stack Geometry and Interconnect Topology

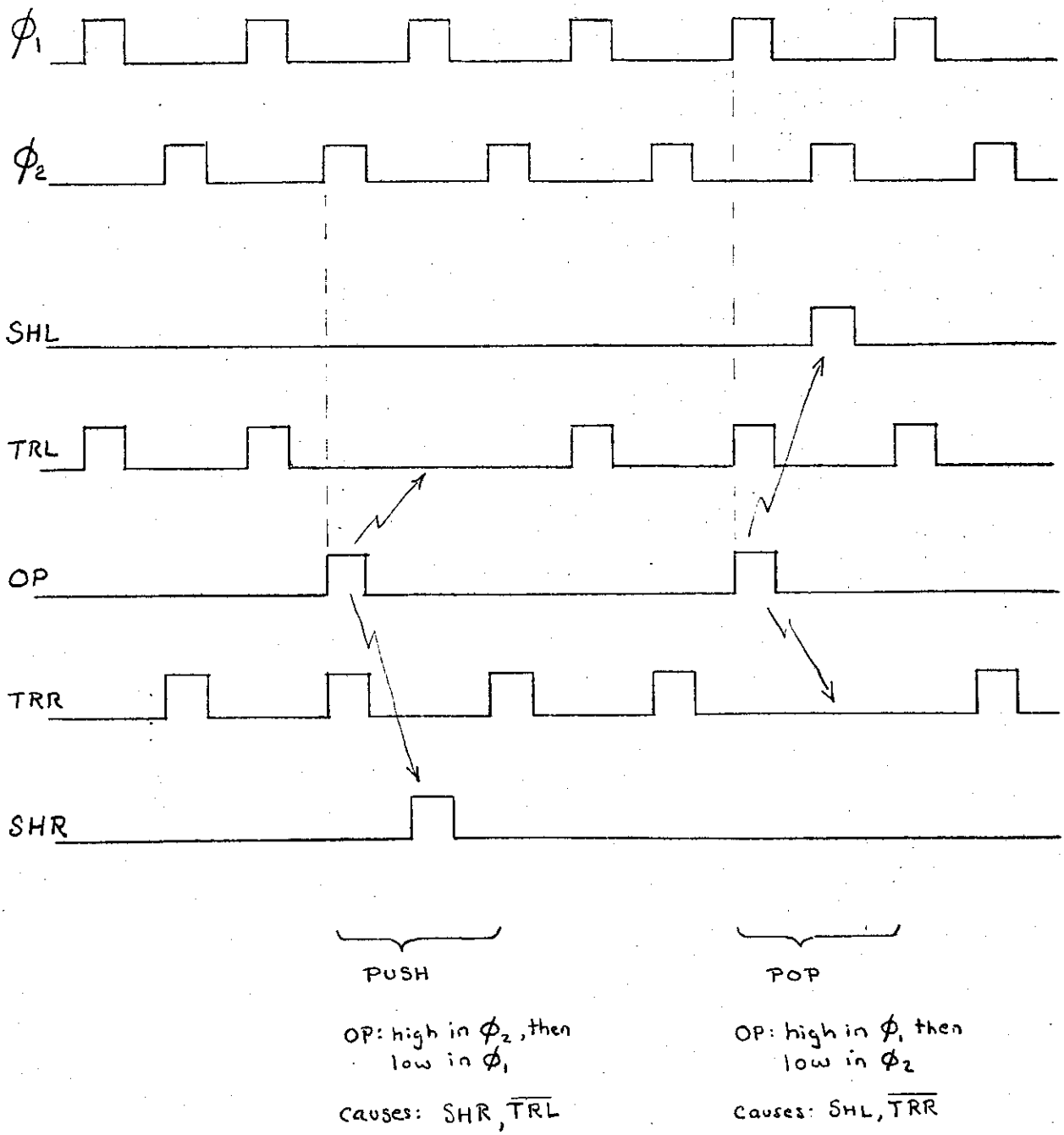


Fig.10e. Stack Control Signal Timing Diagrams

When planning the overall architecture of a larger system, it is often useful to represent subsystems, such as the stack, using a higher level of symbolism. To be truly useful, such representations should, in addition to a functional definition, include the *topological* factors associated with the interconnection points of the subsystem and the *geometrical* factors of its shape and relative physical dimensions.

A system level sketch of one particular implementation of the stack is shown in figure 10d. Identical driver circuitry is placed along the top and bottom edges of the shift register array. The transfer right and shift left drivers which are set up using φ_1 (and active during φ_2) are placed along the top of the shift register array. The transfer left and shift right drivers which are set up using φ_2 (and active during φ_1) are placed along the bottom of the array. This choice has made the two timing signals local to the drivers which they control. The only unfortunate part of the interface is that the OP bit is required on both the top and the bottom of the shift register array.

The integration of this subsystem into a larger LSI system design will require that the data in and out paths be matched to those of subsystems to which the array is connected, and that the φ_1, φ_2, and OP signals be available at either the left or right side of the array. By using system level representations that reflect as closely as possible the dimensions and locations of critical signals in all major subsystems, the interactions between topologies and dimensions of the subsystems can be assessed. The feasibility of an overall system architecture can thus be ensured prior to detailed design and layout.

Register to Register Transfer

From an implementation point of view it is often desirable to combine logic steering functions with the clocking of data into registers, since both require pass transistors as their elementary functional building block. An example is the shift-up register array shown in figure 6. From the next higher level system view, however, it is desirable to separate the two functions conceptually. In Fig. 11a we have shown some combination of inputs, X0 through Xn going through some combination of pass transistors, *which may or may not have logic functions attached*, into the input gates of some inverting logic elements. This combination of function is then abstracted into a register clocked on the phase during which the input pass transistors are turned on. Any logic function associated with the input pass transistors is considered part of the preceding combinational logic module. This extends the applicability of the concept of dynamic register previously developed in figure 7.

Using this notation, any processing function can be built up using blocks of the form shown in Fig. 11b. Here we have a clocked input register, a block of strictly combinational logic *with no timing attached*, and an output register clocked on the opposite phase. In this case the inputs are stored in the input register during φ_1. They then propagate into and through the combinational logic (C/L), with the resulting outputs stored in the output register during φ_2. Any single data processing step can be viewed as a transfer from the first register to the second through the combinational logic block.

A sequence of such operations can be performed on a data stream by a series of such combinational blocks separated by registers as shown in Fig. 11c. Since different sets of data words in the stream may be operated upon at the same time, but at different locations, this data path is a type of pipelined processing structure. Such pipelined processing structures offer the opportunity for improved processing bandwidth by performing many different operations concurrently. Notice that the throughput rate of such a pipeline system of register to register transfer operations is limited by the delay time through the slowest of the combinational logic blocks. If no registers had been interposed between the function blocks, and each operand set separately run through the entire sequence of combinational logic modules, the throughput rate would be much lower.

In line with the ideas developed earlier in this chapter, control over the detailed functions

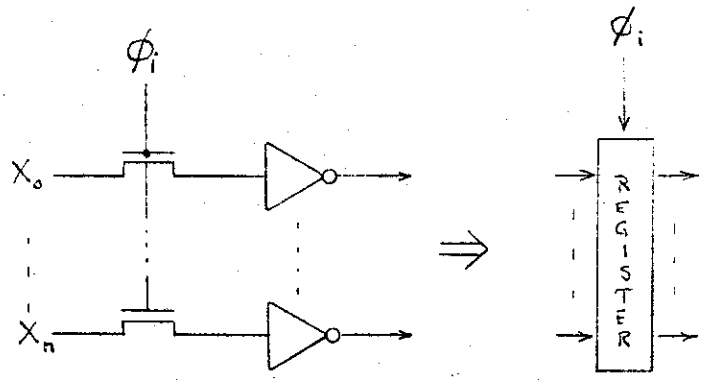


Fig.11a. A Register

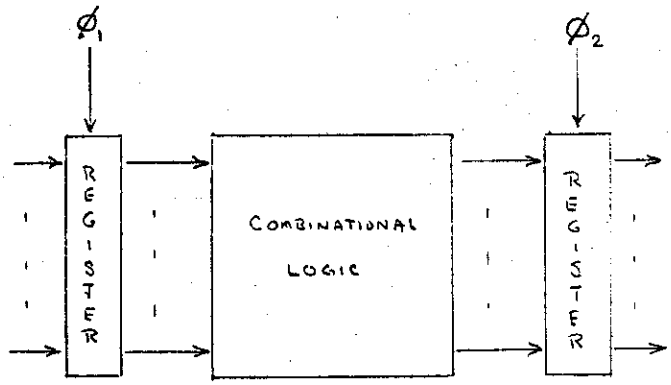


Fig.11b. A Section of a Data Path

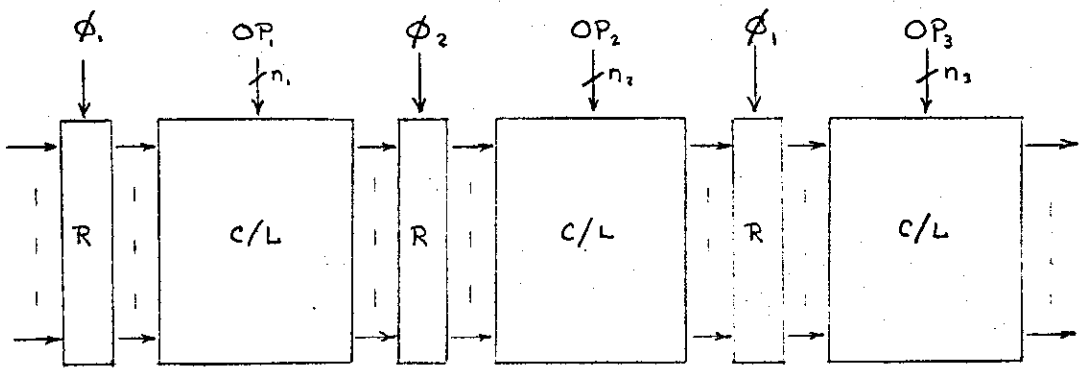


Fig.11c. General Form for a Data Path

of the combinational logic modules is often implemented in circuit structures of very simple and regular topology, if the control signals enter the data path at right angles to the data flow direction. Figure 11c illustrates sets of such control inputs as n_1 lines carrying the control function OP_1 into the first C/L module, n_2 lines carrying OP_2 into the second, etc.

The idea of data being processed while passing through combinational logic interspersed between register stages in a sequence of register to register transfers is a basic and important concept in the hierarchy of digital system architecture. We already have described the implementation of registers in LSI. The next sections will describe some ways to implement combinational logic functions.

Combinational Logic

Combinational logic modules contain no data storage circuitry. The outputs of a combinational logic module are functions only of the inputs to that module, provided that a sufficient delay time has been allowed for those inputs to propagate through the module's circuitry.

In LSI systems, combinational logic design problems will typically fall within one of three general classes. The first is when a small amount of simple logic is required, for example to derive control signals at the periphery of a system module (as in the stack control signal generation) or to implement a simple function within a single circuit cell (which may then be replicated in a regular array). In these cases, traditional logic design procedures using static NAND and NOR gates can be applied. Such designs involving a few gates are usually rather simple, however, and can be produced by inspection rather than by use of formal minimization and synthesis procedures.

Even in these simple cases, the minimum static logic gate implementation does not necessarily result in either the most regular, the minimum area, the minimum delay, or the minimum power design. In fact, we often find alternative techniques to the use of static logic gates, which in specific instances lead to "better" designs by one of these measures than would minimum gate implementations. For example, figure 12a shows a *selector* logic circuit (I. Sutherland), in which one of the inputs S_1, S_2, S_3, S_4 is selected for output by the control variables A, and B according to the function:

$$Z = S_1A'B' + S_2AB' + S_3A'B + S_4AB$$

This selector circuit is composed simply of poly paths crossing diffusion paths. Where depletion mode transistors are placed, the diffusion level path is always connected, thus placing control in the selectively located enhancement mode pass transistors, which function as simple switches. Figure 12c shows the circuit's paths from inputs to outputs using the "switch" abstraction for each of the pass transistors. For each possible combination of values of A and B, there is a path through the selector to Z from only one of the inputs S_i. For the specific inputs shown in the example in figure 12c, the signal S_2 propagates through to Z since both A and B' are *high*. Note that no static power is consumed by the circuit, and the area occupied by the circuit is minimal since no contact cuts are required within it.

The second general class of combinational logic design problems are those rather complex functions for which clever ways of structuring topologically regular implementations have been discovered.

As an example, consider the implementation of a *tally* function. This function has n inputs and n+1 outputs. The k^{th} output is to be *high*, and all other outputs *low*, if k of the inputs are *high*. The boolean equations representing this function for the simple case of three inputs are:

$$Z_0 = X_1'X_2'X_3'$$

$$Z_1 = X_1X_2'X_3' + X_1'X_2X_3' + X_1'X_2'X_3$$

$$Z_2 = X_1X_2X_3' + X_1X_2'X_3 + X_1'X_2X_3$$

$$Z_3 = X_1X_2X_3$$

If this function were designed with random logic consisting of active pullup, static logic gates, it would result in a topological kludge. Figure 12b shows a topologically regular implementation of the tally function. A major portion of the logic function is implemented using a regular array of identical cells each containing only two pass transistors. The design is based on the shift-up register idea presented earlier. A *high* signal propagates through the array from the pullup at the lower left. Whenever one of the variables X_i is *high*, the propagating *high* signal moves up to the next higher horizontal diffusion level path. Thus the number of paths it moves up equals the number of inputs X_i which are *high*. Logic-0 signals propagate through the array from the ground points to all other outputs.

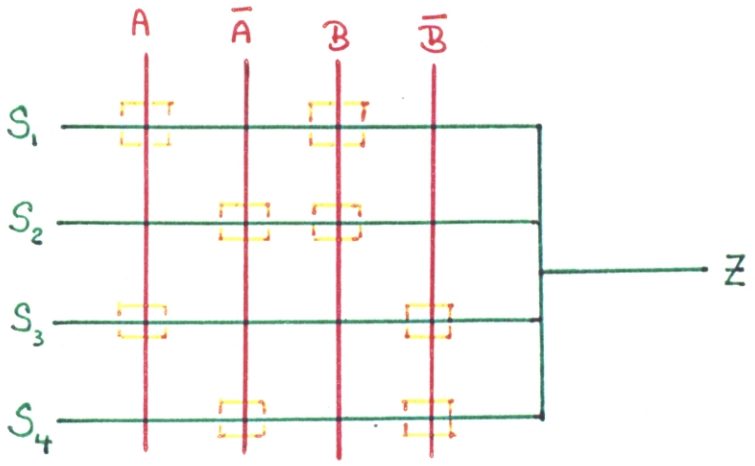


Fig.12a. Selector Logic Circuit

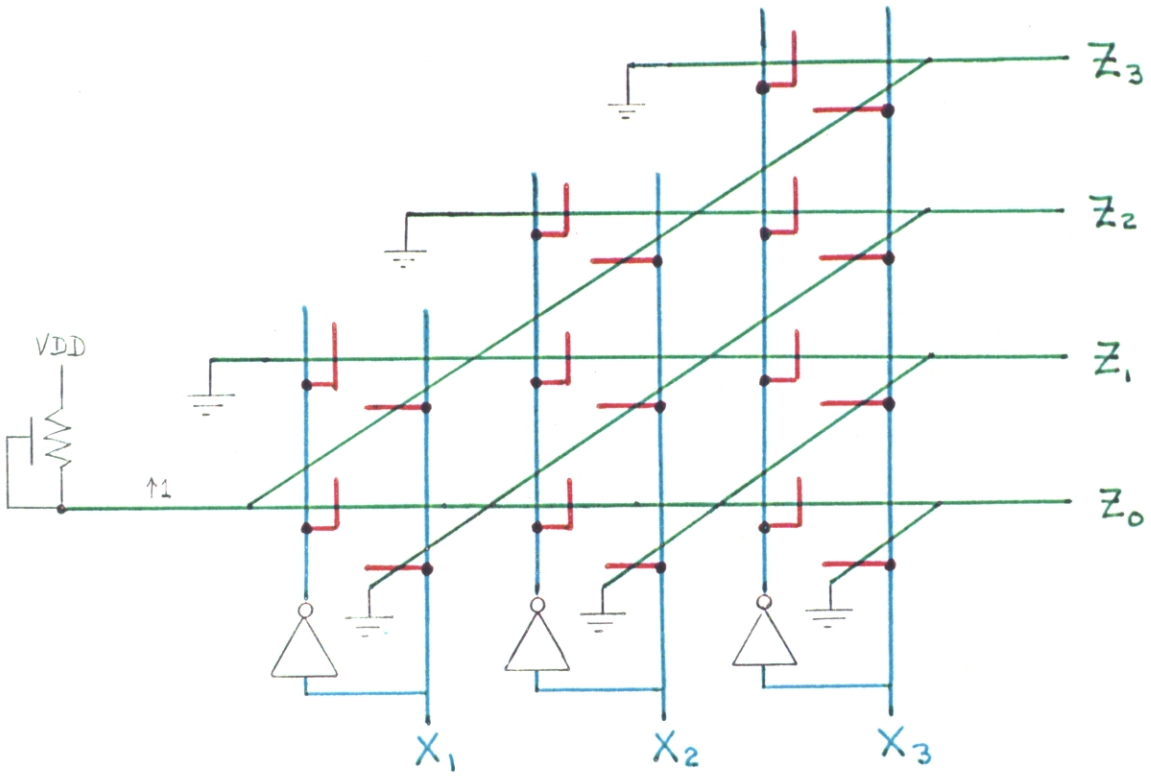


Fig.12b. A Tally Circuit

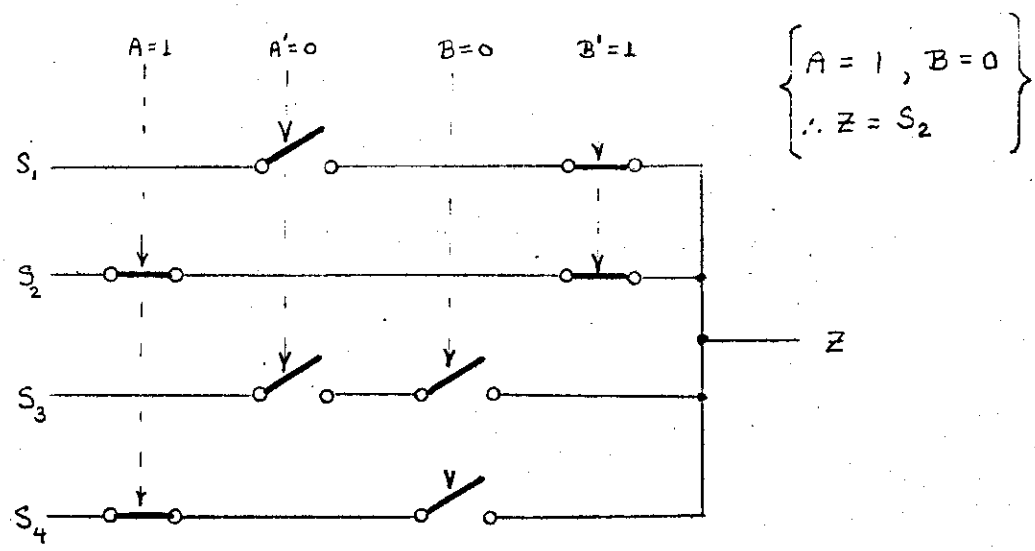


Fig.12c. Example of Operation of Selector Logic Circuit

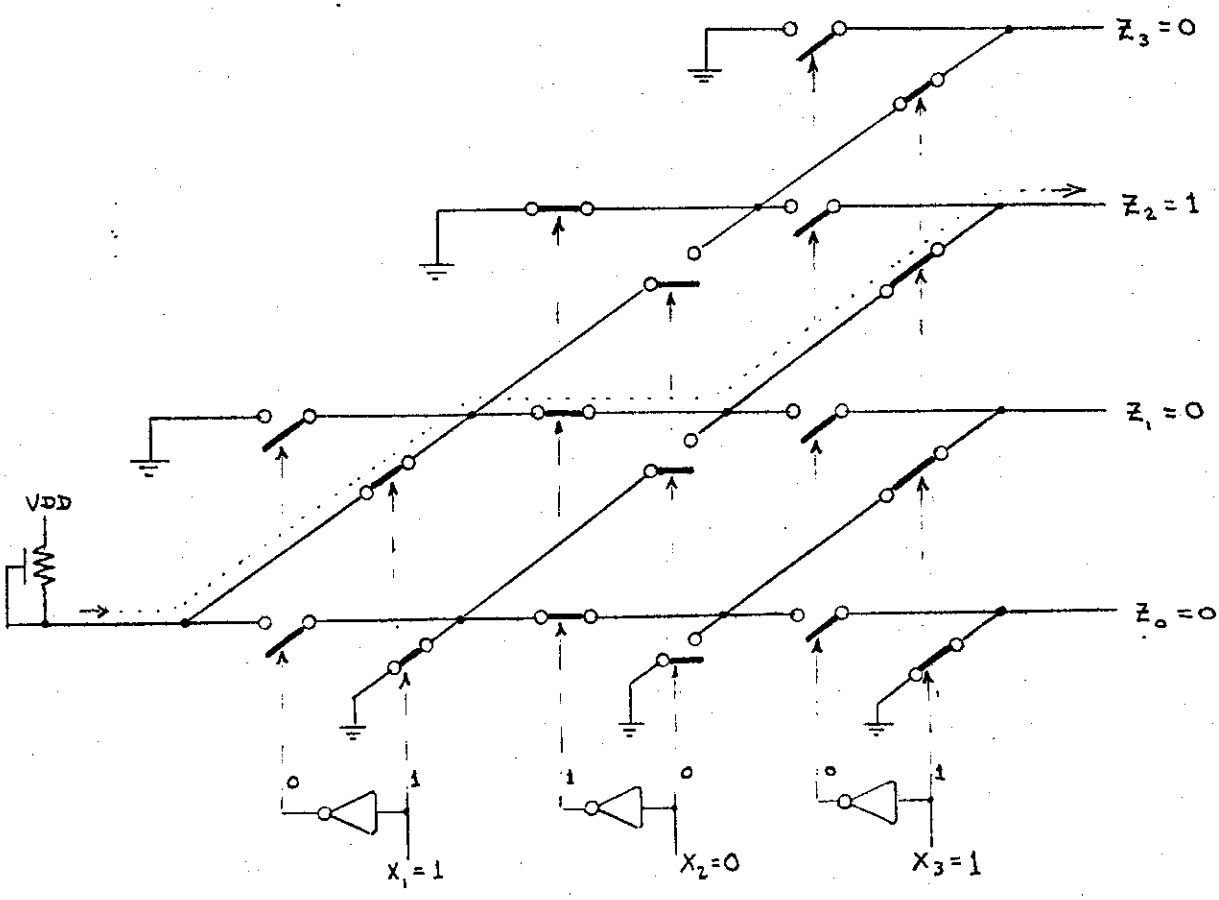


Fig.12d. Example of Operation of Tally Circuit

Figure 12d shows the paths from inputs to outputs for this tally circuit, using the "switch" abstraction for the pass transistors. The figure shows a specific example of a set of inputs controlling the pass transistors of the circuit. Since two of the inputs are *high*, the logic-1 signal is shifted up two rows and emerges at Z_2.

This tally function design can be easily expanded to handle more than three inputs by simply extending the array structure upwards and to the right. However, remember that the delay through n pass transistors is proportional to n^2. Thus it may be necessary to insert level restoration prior to such extension. Similar comments apply to the extension of the selector circuit previously shown, and to extension of other pass transistor logic arrays one might invent.

The electronic logic gates traditionally used in digital design are unilateral elements: they allow a logic signal to propagate in one direction only. It should be noted that the pass transistor is a bilateral circuit element. It permits the flow of current, and thus the passage of a logic signal, in either direction when its gate is *high*. While this property of the pass transistor is not necessarily of fundamental importance in LSI systems, it is an interesting and occasionally useful one.

Early relay switching logic used switching contacts which were bilateral elements. Interesting discussions of relay switching logic are contained in both references R4 and R5. The tally array example just given is a basic *symmetric network* mapped directly into nMOS from relay switching logic (see R5, p.241). The mathematics of switching universally used in digital systems today was proposed by Claude Shannon (R6) in 1938. Shannon demonstrated that the calculus of propositions, based on the algebra of logic developed by Boole (R7), was directly applicable to relay switching circuits.

The third and final combinational logic design situation occurs when a complex function must be implemented for which no direct mapping into a regular structure is known. This is the subject of the next section.

The reader should note that, in the design methodology developed in this text, the combinational logic between stages in the register to register transfer paths is typically done by operations on the *charge* moving between the stages, using pass transistors to perform these operations. Researchers at the present time are searching for alternative structures and

techniques for performing these functions, including the use of charge transfer devices[5].

The Programmable Logic Array

On many occassions it is convenient to implement the combinational logic interspersed between register stages with regular structures of pass transistors. However, we will often encounter important combinational logic functions which do not map well into such regular structures. In particular, combinational logic used in the feedback paths of finite state machines is often highly complex and inherently irregular. Also, we may wish to delay binding the details of the logic functions used to determine finite state machine sequencing until most of the design is complete. If the combinational logic were implemented in an irregular structure, such changes could require a major redesign.

Fortunately, there is a way to map irregular combinational functions onto regular structures, using *programmable logic arrays* as described in this section. This technique of implementing combinational functions has a great advantage: functions may be significantly changed without requiring any major design or layout changes of the PLA structure.

One very general and regular way to implement a combinatorial logic function of n-inputs and m-outputs is to use a memory of 2^n words of m-bits each. The n-inputs form an address into the memory, and the m-outputs are the data contained in that address. Such a memory implements the full truth table for the output functions. Many systems are in fact built using memories as combinational logic elements. A common form of memory for this purpose is the *read-only memory* (ROM) where the data is permanently placed in the memory by a mask pattern. There is one major difficulty with this approach: it is often the case that most of the possible input combinations cannot occur, due to the nature of the specific problem. Stated another way, many combinational logic functions require only a small fraction of all 2^n product minterms for a canonical sum of products implementation. In such cases, a ROM is very wasteful of area.

The *programmable logic array* (PLA) is a structure which has all the generality of a memory for implementing combinational logic functions. However, any specific PLA structure need contain a row of circuit elements only for each of those product terms (the essential prime implicants; R4, Ch.4) that are actually required to implement a given logic function. Since it does not contain entries for all possible minterms, it is usually far more

compact than a ROM implementation of the same function. To achieve full compaction, the various output functions must be jointly minimized before the PLA layout pattern can be defined. However, such minimization is not essential. Less than full compaction increases the independance of the different entries, so that changes in function may require only local changes in the PLA.

An illustration of the overall structure of a PLA is shown in figure 13a. The diagram includes the preceding and following clocked registers, in order to show how easily these are implemented adjacent to the PLA. The inputs, stored during φ_1 in the input register, are run vertically through a matrix of circuit elements called the AND-plane. The AND-plane generates specific logic combinations of the inputs and their complements. The outputs of the AND-plan leave at right angles to its inputs and run horizontally through another matrix called the OR-plane. The outputs of the OR-plane then run vertically and are stored in the output register during φ_2.

The circuit diagram of a specific programmable logic array is shown in figure 13b. This diagram will help to clarify the structure and function of the AND and OR-planes of the PLA. The input register bit for each input path is formed by a pass transistor clocked on φ_1 leading to both inverting and non-inverting super buffers. These buffers drive two lines running vertically across the AND-plane, one for the input term and one for its complement. The outputs of the AND-plane are formed by horizontal lines with pull-up transistors at their leftmost end. The function of the PLA's AND-plane is then determined by the locations and gate connections of pull-down transistors connecting the horizontal lines to ground.

Each horizontally running output from the AND-plane carries the NOR combination of all input signals which lead to the gates of transistors attached to it. For example, the horizontal row labelled R_3 has three transistors attached to it in the AND-plane, one controlled by A, one by B and one by C'. If any of these inputs is *high*, then R_3 will be pulled down towards ground and will be *low*.

Thus, $R_3 = (A + B + C')' = A'B'C$. Similarly, $R_4 = (A + B' + C)' = A'BC'$.

The OR-plane matrix of circuit elements is identical in format to the AND-plane matrix, but rotated 90 degrees. Once again, each of its outputs is the NOR of the signals leading to the gates of all transistors attached to it. In figure 13b for example, both R_3 and R_4 lead to

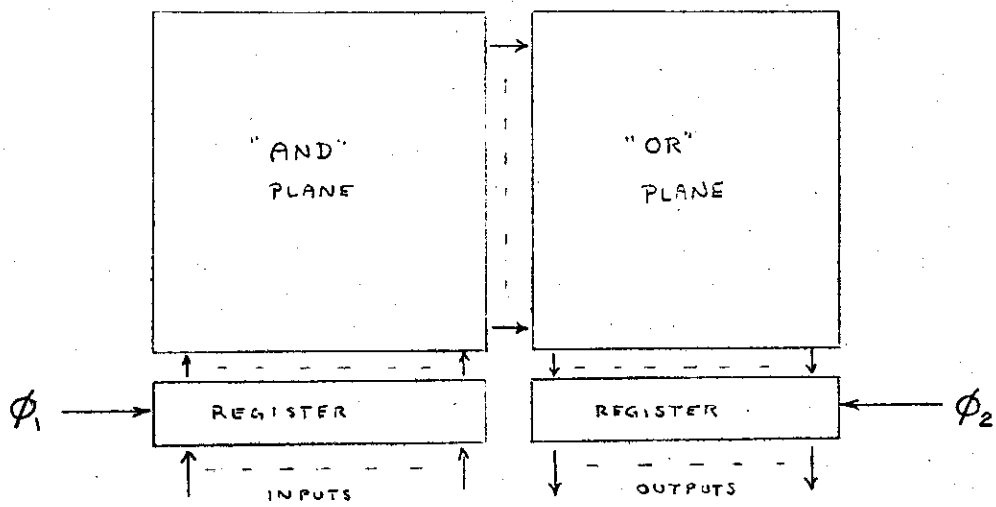


Fig.13a. Overall Structure of the PLA

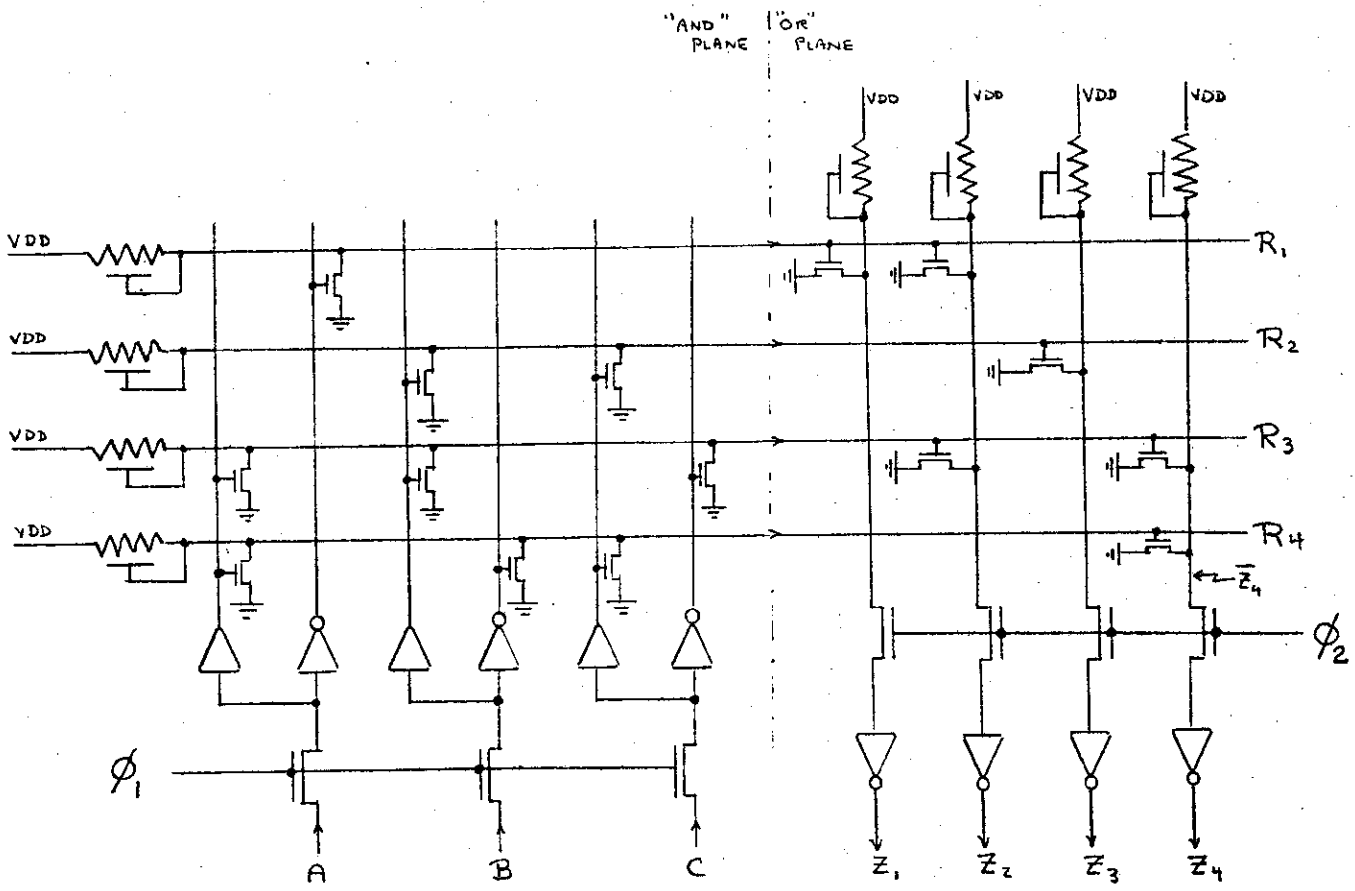


Fig.13b. Circuit Diagram of PLA Example

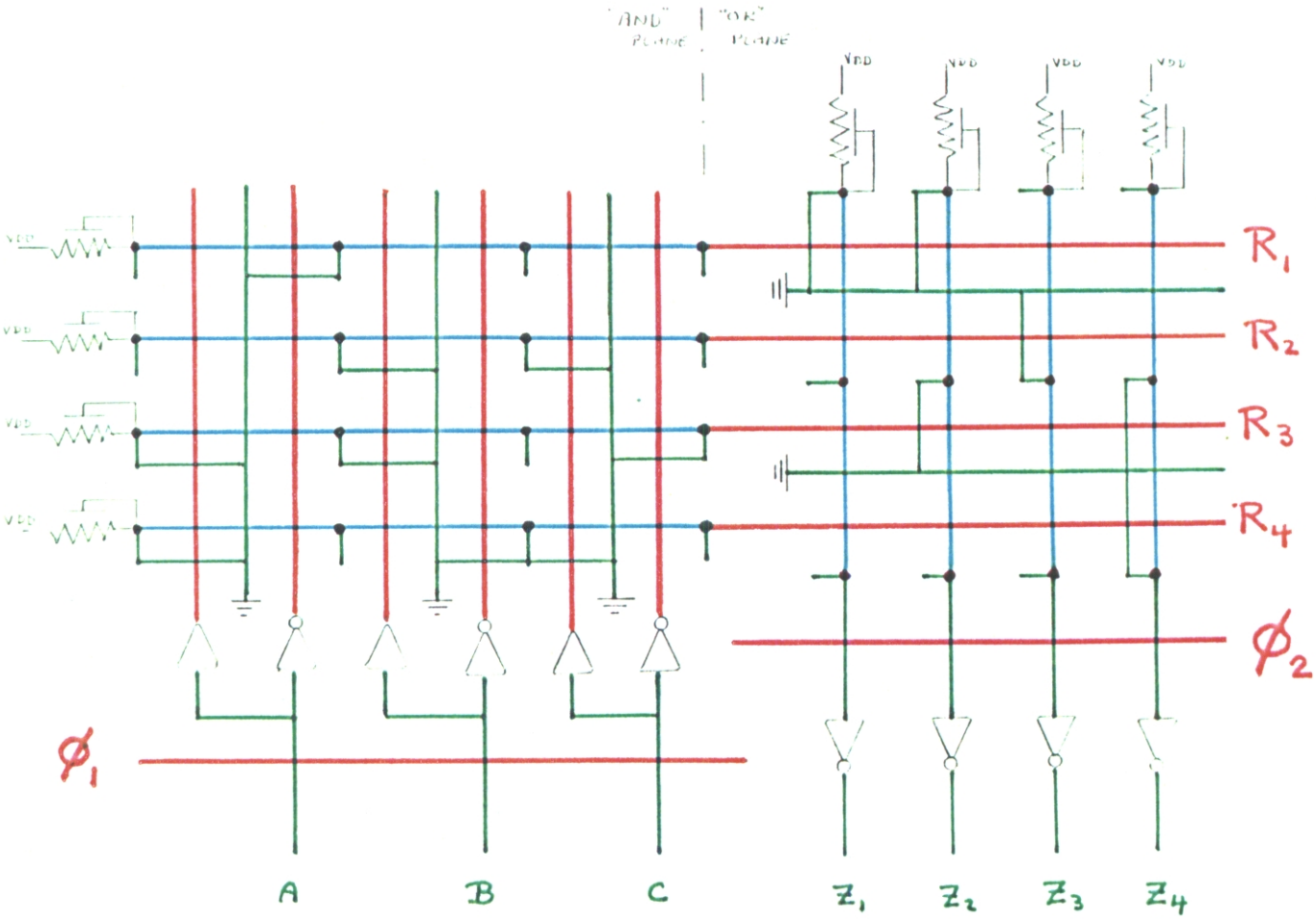


Fig.13c. Stick Diagram of PLA Example

Product Terms:

$R_1 = (A')' = A$

$R_2 = (B+C)' = B'C'$

$R_3 = (A+B+C')' = A'B'C$

$R_4 = (A+B'+C)' = A'BC'$

Outputs:

$Z_1 = A$

$Z_2 = A + A'B'C$

$Z_3 = B'C'$

$Z_4 = A'B'C + A'BC'$

the gates of transistors leading from the output line Z_4' to ground. If either R_3 or R_4 is *high,* Z_4' will be *low.* Thus, $Z_4' = NOR(R_3,R_4) = (A'B'C + A'BC')'$. Up to this point the PLA implements the *NOR-NOR canonical form* of boolean function of its inputs.

The output lines of the OR-plane matrix are run into an output register formed by pass transistors (clocked on φ_2) leading into inverting drivers. Note that the output Z_4 at this point is: $Z_4 = A'B'C + A'BC$. This expression illustrates why the two PLA planes, each implementing the NOR function, are usually referred to as the AND and OR-planes. Following the output register, the outputs are fairly easily described as the *sum of products canonical form* of boolean functions of the PLA inputs, that is, the OR of AND terms. Each horizontal line of the PLA carries one *product term.* The number of horizontal lines is minimized when the product terms they carry are the essential prime implicants of the output functions.

Figure 13c shows one possible layout topology for implementing the PLA in nMOS circuitry. The example is the same circuit illustrated in figure 13b. The input lines crossing each plane are run in poly. The output lines from each plane are run in metal. Paths running to ground are placed between alternate poly lines, on the diffusion level. It is then a simple matter to form, and selectively locate as diffusion lines under the appropriate input poly lines, the pulldown transistors connecting the metal output lines to ground.

Although the PLA may implement a very irregular combinational function, the irregularity is confined to the irregular locations of pulldown transistors which "program" the functions. The overall structure and topology of the PLA are very regular. Note that its overall shape and size is a function of the parameters: (i) the number of inputs, (ii) the number of product terms, (iii) the number of outputs, and (iv) the length unit λ.

Finite State Machines

In many cases in the processing of data, it is necessary to know the outcome of the current processing step before proceeding with the next. Results of the current step may be used as inputs in the next step. The configuration shown in figure 14a can be used to implement a processing stage having this requirement. In figure 14a, a typical register to register transfer stage is modified by simply feeding back some of its outputs to some of its inputs. This implements a form of sequential machine known as a *finite state machine.*

The feedback signals form a binary number which may be regarded as identifying the *state* of the machine. The value of this number is stored, along with the external inputs, in the first register during φ_1. These combined inputs then propogate through the combinational logic. The resulting outputs are stored in the second register during φ_2. The falling edge of φ_2 must occur a sufficient time later to insure that all signals have propagated throught the combinational logic. Each φ_1 - φ_2 cycle results in two new sets of outputs: (i) the external outputs which are typically used for controlling other units of the system, and (ii) a new feedback number, which defines the *next state* of the machine. This process repeats during each clock period. The number of possible states is determined by the number of bits in the feedback path, and is *finite*.

There are a number of ways of abstractly representing the states, the required state transitions, and the outputs of sequential machines under given input sequences. Possible representations include state diagrams, transition tables, boolean difference equations, etc. A large body of theory has been developed concerning the synthesis and analysis of sequential machines. The serious reader will benefit from a further study of the results of switching theory on this subject (R3, R4).

Implementations of simple finite state machines are used to produce the very lowest level of system control sequencing, since they have the property that they can autonomously generate control sequences. The sequential machine having a finite number of states is a very important element in the hierarchy of fundamental concepts used in LSI system architecture.

The configuration shown in figure 14a implements a *synchronous* machine, since the feedback loop is only activated at times determined by the clock signals. In a synchronous machine the external outputs Z_j and the next state terms Y_f, at clock period k, are not just functions of the present inputs X_i to its combinational logic, but are also functions of the present state. The present state is thus a function of past input sequences applied to past states.

Outputs:

$$Z_1(k) = f_1[X_1(k), \ldots, X_m(k); Y_1(k), \ldots, Y_n(k)],$$
$$Z_2(k) = f_2[X_1(k), \ldots, X_m(k); Y_1(k), \ldots, Y_n(k)], \quad \text{etc.}$$

State:

$$Y_1(k+1) = g_1[X_1(k), \ldots, X_m(k); Y_1(k), \ldots, Y_n(k)],$$
$$Y_2(k+1) = g_2[X_1(k), \ldots, X_m(k); Y_1(k), \ldots, Y_n(k)], \quad \text{etc.}$$

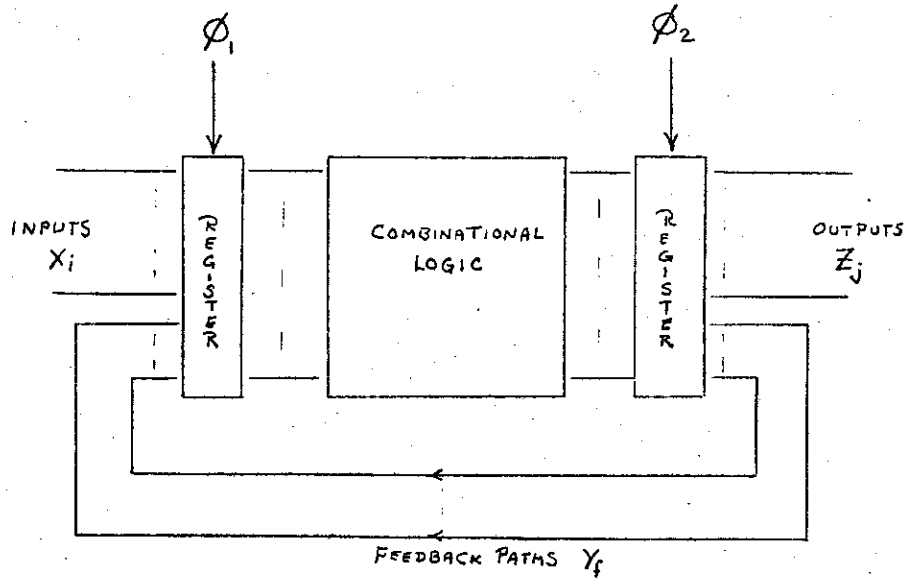


Fig.14a. Feedback in Register to Register Transfer Path, Implementing a Finite State Machine

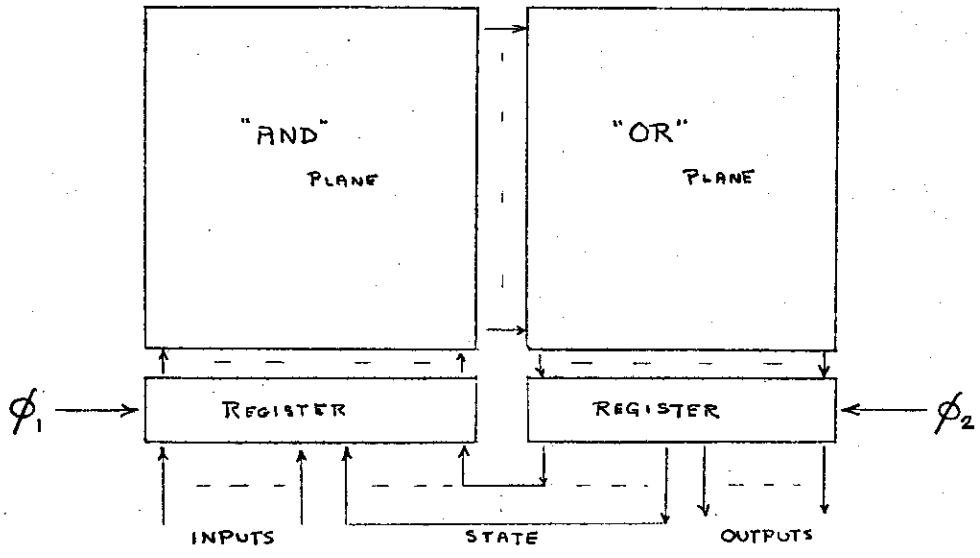


Fig.14b. PLA Implementation of a Finite State Machine

If a sequential machine contains a feedback loop which is continuously active, then it may begin a response to a change in inputs or state at any time, rather than just at fixed clock times. Such a sequential machine is referred to as an *asynchronous sequential machine.* The analysis of asynchronous sequential machines and their implementation is far more complex than that of synchronous ones. Great care must be exercised to avoid any difference in state sequencing and outputs under arbitrary differential delays of signals through the circuit paths of such machines (R3. Ch.5). There will be only a few special cases where we use the asynchronous form of sequentiai machine (Chapter 7), and these will be subject to detailed analysis.

Where sequential machines are required within our LSI systems, we will generally implement them in synchronous form. Synchronous machines are rather easy to implement correctly, and fit naturally into the two phase clocking scheme used for moving data around within our systems.

However, the reader should carefully note that an implementation of a synchronous sequential machine functions correctly only if the delays in the circuit paths are sufficiently short compared to the clock period. If we were to implement many copies of a particular machine, the probability of correct function for any given copy is thus a function of (i) the clock period used, and (ii) the distribution of differential delays in that copy's signal paths. Our estimate that a particular copy will function correctly is thus based in part on assumptions about the ratio of likely deviations in circuit delays to the clock period.

There is a very straightforward way to implement simple finite state machines in LSI systems: we use the PLA form of combinational logic and just feed back some of the outputs to the inputs, as illustrated in figure 14b. The circuit's structure is topologically regular, has a reasonable topological interface as a subsystem, and is of a shape and size that are functions of the appropriate parameters. The function of this circuit is determined by the "programming" of its PLA logic. If, for example, early in a design cycle there is some uncertainty in the details of the desired sequencing of such a circuit, it is easy to provide layout space for extra, unused inputs, minterms, or outputs as contingencies.

The following simple example will help illustrate the basic concepts of finite state machines and their implementation in nMOS circuitry. A busy highway is intersected by a little used farmroad, as shown in figure 15a. Detectors are installed which cause the signal C to go

high in the presence of a car or cars on the farmroad at the positions labelled C. We wish to control traffic lights at the intersection, so that in the absence of any cars waiting to cross or turn left on the highway from the farmroad, the highway lights will remain green. If any cars are detected at either position C, we wish the highway lights to cycle through caution to red, and the farmroad lights then to turn green. The farmroad light is to remain green only while the detectors signal the presence of a car or cars, but never longer than some fraction of a minute. The farmroad light is then to cycle through caution to red, and the highway light then to turn green. The highway light is not to be interruptible again by the farmroad traffic until some fraction of a minute has passed.

A state diagram model of a finite state machine to control the lights is sketched in figure 15b. This diagram identifies four possible states of the machine, and indicates the input conditions which cause all possible state transitions. A block diagram of the PLA circuit implementing the machine is shown in figure 15c. The circuit uses the signal C as an input, and provides outputs HL and FL which encode the colors of the highway and farmroad lights it controls. Note that a timer is used to provide, as controller inputs, the short and long timeout signals (TS, and TL), at appropriate times following a start timer (ST) signal output from the controller. This timer could be implemented as a digital counter in the same nMOS circuitry. Another abstract model describing the desired function of the controller is given in the state transition table in figure 15d, which contains similar information to that in the state diagram.

The detailed sequencing of the machine under various input sequences is described by both the state diagram and transition table models of the controller. Consider starting in the state HG, where the highway lights are green. The machine remains in state HG as long as either no cars are detected or the long timeout has not occurrred, in other words as long as (C)AND(TL) = 0. After the long timeout occurs, if any cars are detected, the machine restarts the timer and changes state to HY, where the highway lights are yellow. It remains in state HY only until the short timeout occurs, and then restarts the timer and changes to state FG, where the farmroad lights are green. It remains in state FG until either no cars are detected or the long timeout occurs, i.e. (C)'OR(TL) = 1. Then it restarts the timer and changes to state FY, where the farmroad light is yellow. It remains in state HY only until the short timeout occurs. It then restarts the timer and changes to state HG, the starting state.

The locations of transistors in the PLA light controller circuit can be determined by "hand

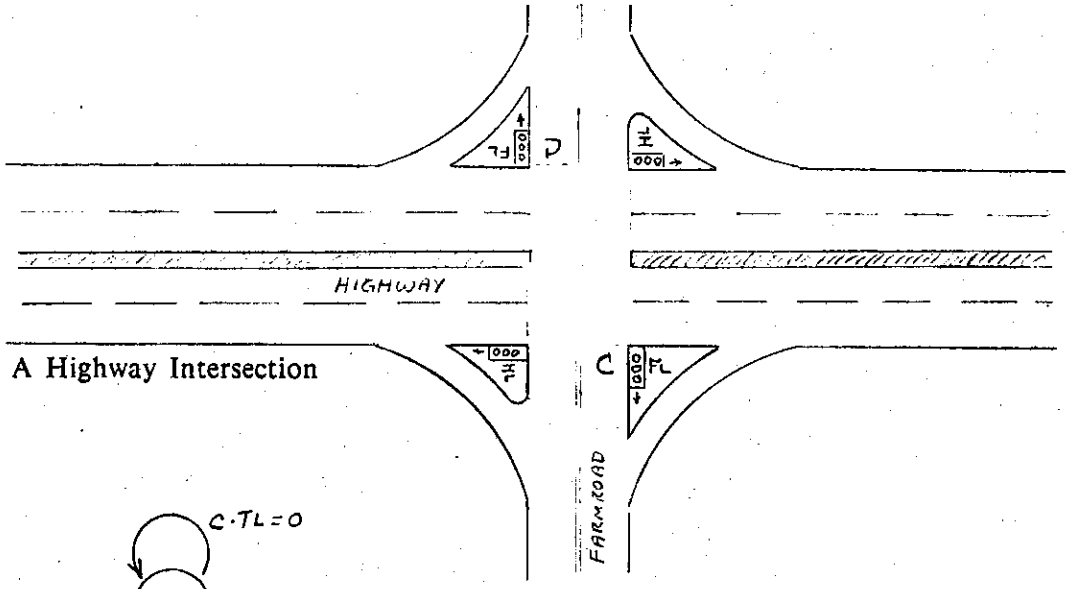


Fig.15a. A Highway Intersection

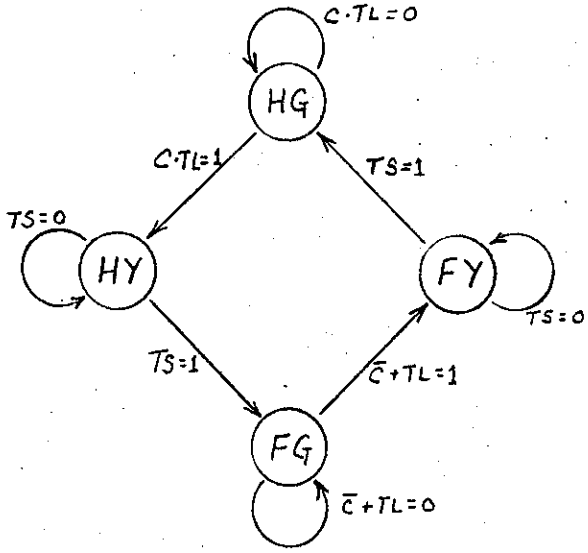


Fig.15b. Light Controller State Diagram

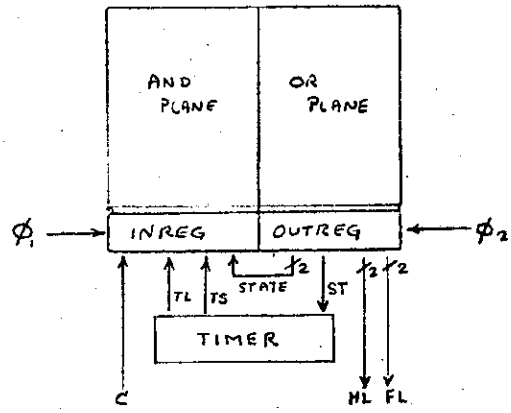


Fig.15c. Controller Block Diagram

In Present State:	*If Inputs Are:* (inputs not listed = don't cares)	*Then Next State Will Be:*	*Ouputs Are:* HL	FL	ST
Highway Green	(Cars)AND(TimeoutL) = 0	Highway Green	Green	Red	No
	(Cars)AND(TimeoutL) = 1	Highway Yellow	Green	Red	Yes
Highway Yellow	TimeoutS = 0	Highway Yellow	Yellow	Red	No
	TimeoutS = 1	Farmroad Green	Yellow	Red	Yes
Farmroad Green	(Cars)'OR(TimeoutL) = 0	Farmroad Green	Red	Green	No
	(Cars)'OR(TimeoutL) = 1	Farmroad Yellow	Red	Green	Yes
Farmroad Yellow	TimeoutS = 0	Farmroad Yellow	Red	Yellow	No
	TimeoutS = 1	Highway Green	Red	Yellow	Yes

Fig.15d. Transition Table for Light Controller

assembling" the "program" specified in the "symbolic" transition table in figure 15d, resulting in the encoded state transition of figure 15e. First we assign codes to the states. In the example: state HG is encoded as (Y_0,Y_1) = (0,0); HY as (0,1); FG as (1,1); and FY as (1,0). Next, we assign codes to the output light control signals: In the example, Green is encoded as (0,0), Yellow as (0,1), and Red as (1,0). We now form the encoded state transition table by constructing one row for each product term implied by the original symbolic table of figure 15d. A row in table 15d specifying a state transition as a function of a single input variable or single product term of input variables produces a single row in table 15e. A row in table 15d specifying a state transition as a function of a sum or sum of products of input variables, leads to a corresponding number of rows in table 15e.

The placement of the transistors within the PLA matrices follows directly from the encoded state transition table:

(i) For each logic-1 in the next state and output columns in the table, we run a diffusion path *from* the corresponding next state or output line in the PLA OR-plane, *under* the corresponding product term line, *to* ground. This creates a transistor controlled by the product term line. Then, if that controlling product term line is ever *high*, the path to the output inverter will be *low*, and the output will be *high*. The output line will be *low* unless some product term line controlling it is *high*.

(ii) For each logic-1 in the input and present state columns in the table, we run a diffusion path *from* the corresponding product term line, *under* the corresponding *inverted* input or next state line in the PLA AND-plane, *to* ground. The transistor thus created is controlled by the *inverted* input or next state line. Unless that controlling line is *high*, the product term line will be *low*.

(iii) For each logic-0 in the input and present state columns in the table, we run a diffusion path *from* the corresponding product term line, *under* the corresponding *non-inverted* input or next state line in the PLA AND-plane, *to* ground. The transistor thus created is controlled by the *non-inverted* input or next state line. Unless that controlling line is *high*, the product term line will be *low*.

Note that if all lines are *low* which control the transistors connecting a given product term line to ground, then that product term line will be *high*. Otherwise it will be *low*.

Stored during φ_1 in INREG				*Stored during φ_2 in OUTREG*						*Product terms:*
Inputs:			*Present State:*	*Next State:*	*Outputs:*					
C	TL	TS	Y_{p0}, Y_{p1}	Y_{n0}, Y_{n1}	ST	HL_0	HL_1	FL_0	FL_1	
0	X	X	0, 0 (HG)	0, 0 (HG)	0	0	0	1	0	R1
X	0	X	0, 0 (HG)	0, 0 (HG)	0	0	0	1	0	R2
1	1	X	0, 0 (HG)	0, 1 (HY)	1	0	0	1	0	R3
X	X	0	0, 1 (HY)	0, 1 (HY)	0	0	1	1	0	R4
X	X	1	0, 1 (HY)	1, 1 (FG)	1	0	1	1	0	R5
1	0	X	1, 1 (FG)	1, 1 (FG)	0	1	0	0	0	R6
0	X	X	1, 1 (FG)	1, 0 (FY)	1	1	0	0	0	R7
X	1	X	1, 1 (FG)	1, 0 (FY)	1	1	0	0	0	R8
X	X	0	1, 0 (FY)	1, 0 (FY)	0	1	0	0	1	R9
X	X	1	1, 0 (FY)	0, 0 (HG)	1	1	0	0	1	R10

Fig.15e. Encoded State Transition Table for the Light Controller

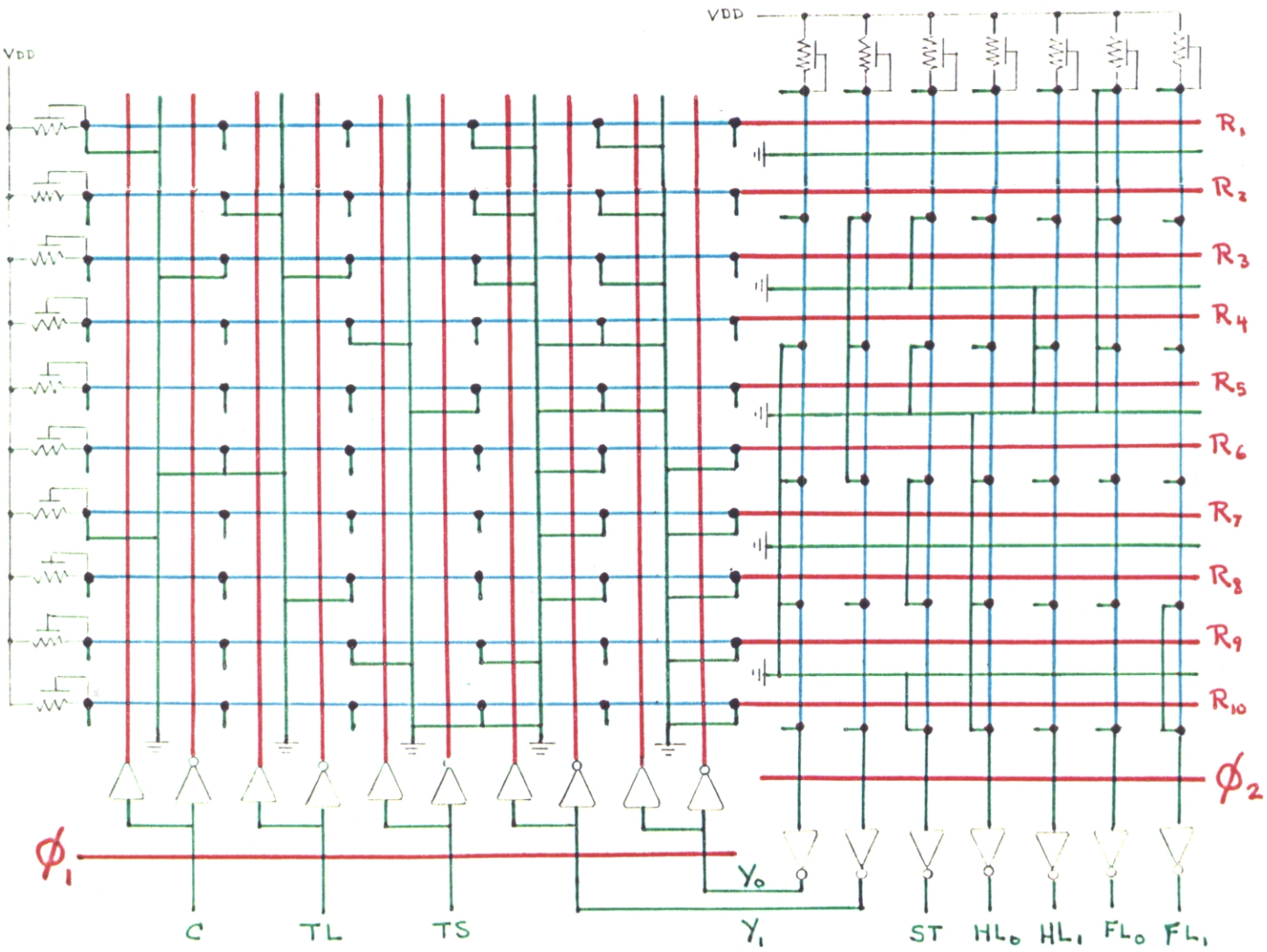


Fig.15f. PLA Sequential Circuit Implementing the Light Controller

The PLA circuit in figure 15f is programmed from the transition table in figure 15e, according to the rules above, and implements the traffic light controller. Note that this LSI implementation does not exactly strain itself to meet the time response requirements of the control problem: it can run at a clock rate at least 10^7 times as fast as required. Also, note that the PLA controller is roughly $(110\lambda)^2$ in area. Using the 1977 value of $\lambda = 3\mu$, this controller is $(330\mu)^2 \sim 0.001$ cm^2 in area. A PLA controller this size may contain over 150 transistors, but occupies only $1/250^{th}$ of the area of a typical 0.25 cm^2 silicon chip in 1977. By the middle to late-80's, as λ scales down towards its ultimate limits, such a controller will require only ~ $1/25{,}000^{th}$ of the area of such a chip.

As we will see in later chapters, a data processing machine of any desired complexity can be created by interconnecting register to register data processing paths constructed along the lines of that shown in figure 11c, such paths being controlled by finite state machines implemented as shown in figure 14b. The data paths form the "highways" for the movement of data, under control of the finite state machine "traffic controllers".

Towards a Structured Design Methodology

The task of designing very complex systems involves managing, in some highly structured way, the space and time relationships between the various levels of system building blocks so that the entire system will function as intended when it is finished. The beginnings of a structured design methodology for LSI systems can be produced by merging together in a hierarchy the concepts presented in this chapter. Designs are then done in a "top down" manner, but with a full understanding by the architect of the successive, lower levels of the hierarchy.

To begin, we plan our digital processing systems as combinations of register to register data transfer paths, controlled by finite state machines. Then the geometric shapes, relative sizes, and interconnection topologies of all subsystem modules are collectively planned so all modules will merge together snugly, with a minimum of space and time wasted by random interconnect wiring. Storage registers are typically constructed by using charge stored on input gates of inverting logic. The combinational logic in the data paths is typically implemented using steering logic composed of regular structures of pass transistors. Most of the combinational logic in the finite state machines is typically implemented using PLA's. All functioning is sequenced using a two-phase, non-overlapping clock scheme.

When viewed in its entirety, a system designed in this manner is seen as a hierarchy of building blocks, from the very lowest level device and circuit constructs, on up to and including the high level system software and application programs in which the intended functions of the system are finally expressed. Individuals with an understanding of the key concepts of each of the levels in this hierarchy will recognize that the boundaries between the levels are rather elastic ones. Each level of activity might best be optimized not on its own as a specialty, but as it fits into an overall systems picture. Viewed in this way, for example, the activity "logic design" in LSI might best be conceptualized as the search for techniques and inventions which best couple the physical, topological, and geometric properties of LSI devices and circuits with the desired properties of digital LSI systems. The search for alternative components for any given design hierarchy, and the search for alternative hierarchies, will be done best by those who span more than one specialty.

In chapters 5 and 6 we will apply this methodology to the design of a digital computer system. A one chip implementation of the data path for this computer system is illustrated in the frontispiece. Consistent use of the described design methodology resulted in a design of great regularity, short delay times, low power consumption, and high logical processing capability. As we will see in chapter 4, regular designs, with small numbers of basic circuit cell types replicated in two dimensions to form subsystems, also have some significant implementation advantages over less structured designs.

References

1. - - - *polycell ref* - - -

2. I. E. Sutherland, C. A. Mead, "Microelectronics and Computer Science", *Scientific American*, September 1977, pp. 210-228.

3. J. D. Williams, "Sticks -- A New Approach to LSI Design", M.S.E.E. Thesis, Dept. of Electrical Engineering, M. I. T., June, 1977.

4. W. M. Penney, L. Lau, Eds., "MOS Integrated Circuits", Van Nostrand Reinhold Co., 1972, Chapter 5.

5. C. H. Sequin, M. F. Tompsett, "Charge Transfer Devices", Academic Press, 1975, Ch. VIII.

Reading References

R1. C. G. Bell, A. Newell, "Computer Structures: Readings and Examples", McGraw-Hill, 1971, contains an excellent discussion of the levels in the hierarchy of computer architecture, and many specific examples of computer structures.

R2. B. Soucek, "Microprocessors and Microcomputers", John Wiley, 1976, is a good introductory reference containing sections on basic digital design, and on the interfacing and programming of a number of current microprocessors.

R3. D. L. Dietmeyer, "Logic Design of Digital Systems", Allyn and Bacon, 1971, is a comprehensive text on switching theory and logic design containing excellent material on synchronous and asynchronous sequential machines.

R4. Z. Kohavi, "Switching and Finite Automata Theory", McGraw-Hill, 1970, is another good text on switching theory.

R5. S. H. Caldwell, "Switching Circuits and Logical Design", John Wiley, 1958, is an early text containing interesting material on relay contact networks.

R6. C. E. Shannon, "Symbolic Analysis of Relay and Switching Circuits", *Trans. of AIEE*, Vol. 57, 1938, pp. 713-723, is the classic paper proposing a method for the mathematical treatment of switching circuits.

R7. G. Boole, "An Investigation of the Laws of Thought", London, 1854, reprinted by Dover Publications, contains a presentation of the algebra of logic on which Shannon based his switching algebra.